UHH Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Methods for Downstream Continual Learning with Foundation Models

Dissertation

submitted in partial fulfilment of
the requirements for the degree of
Doctor rerum naturalium
(Dr. rer. nat.)

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics

Kyra Ahrens

Hamburg, 2025

Day of submission:
May 2, 2025

Day of oral defense:
July 18, 2025

Dissertation Committee:

Prof. Dr. Stefan Wermter (reviewer, advisor)
Dept. of Computer Science
University of Hamburg, Germany

Prof. Dr. Loo Chu Kiong (reviewer)
Dept. of Artificial Intelligence
Universiti Malaya, Malaysia

Prof. Dr. Chris Biemann (chair)
Dept. of Computer Science
University of Hamburg, Germany

Prof. Dr. Anne Lauscher (deputy chair)
University of Hamburg Business School
University of Hamburg, Germany

Abstract — *English*

The advent of self-supervised pretraining on large-scale internet data has produced powerful generic representations, giving rise to pretrained *foundation models* that can be fine-tuned for a wide range of downstream tasks more efficiently and effectively than models trained from scratch. Typically, downstream adaptation assumes that all relevant fine-tuning data are available from the outset. However, in dynamic environments where new data arrive continuously, the model's predictive performance on previously learned tasks degrades as new tasks are introduced, a problem commonly referred to as *catastrophic forgetting*. A naive solution is to retrain the model whenever new data arrive, but this is prohibitively expensive for real-time processing of large data streams. Moreover, this approach requires storing all past data, which can directly conflict with legislative and privacy constraints governing the handling of sensitive information.

Continual Learning (CL) has emerged as a subfield of machine learning dedicated to mitigating catastrophic forgetting by enabling neural networks to learn from non-stationary data—practically represented as sequence of datasets with disjoint probability distributions—while preserving previously acquired knowledge. However, most existing CL approaches are designed for relatively small networks trained from scratch on simple tasks, which limits their scalability and practical utility for large foundation models.

In this thesis, we address the challenge of downstream CL with foundation models. Ideally, such approaches leverage the broad generalizability of pretrained representations and introduce cost-effective, scalable mechanisms for incrementally adapting to a variety of downstream tasks. To this end, we compare and contrast different paradigms for CL, including methods that regularize crucial model parameters, replay a selected subset of historical data, or introduce new parameters as data distributions shift. We clarify which contexts favor (or hinder) each paradigm's applicability to foundation models. Beyond the widely adopted focus on unimodal tasks such as image *or* text classification, we also explore more demanding multimodal tasks that require integrating pretrained representations from multiple sources, for instance combining images *and* text. This broader scope mirrors practical deployments in which heterogeneous modalities must be fused and diverse tasks must be solved within a single, unified representation space.

We introduce *four* novel strategies for downstream CL with foundation models: two dedicated to *unimodal* incremental classification and two focused on *multimodal* reasoning. For unimodal CL, our first contribution is a dual-memory approach that uses self-organizing networks to map foundation model features onto a topology-preserving manifold, thereby retaining discriminative structure and increasing robustness to representational drift. The second approach aggregates low- and mid-level activations from multiple foundation model layers to construct high-fidelity class prototypes, markedly improving class separability and generalization under severe domain shift and data scarcity. Shifting to multimodal CL, the third proposed method incrementally trains a lightweight modality interaction network that fuses multimodal representations by selectively specializing a compact subset of interaction parameters for each task while keeping the remainder shared. The fourth and final method optimizes the extraction of knowledge from stored multimodal features by injecting targeted spatially aware noise, ultimately improving the reconstruction of prior data distributions.

Our results demonstrate that all four methods introduced in this thesis greatly enhance the ability of foundation models to manage non-stationary data during fine-tuning, while imposing minimal assumptions about data type, size, or presentation order. By strategically augmenting pretrained representations, safeguarding crucial parameters that capture generalized knowledge, and recovering past distributions from a minimal set of stored representations, this thesis provides practical and scalable solutions for a broad spectrum of CL problems involving large pretrained models. Collectively, these advances empower foundation models to meet the demands of dynamic and evolving downstream data in real-world settings.

Zusammenfassung — *Deutsch*

Die zunehmende Verbreitung selbstüberwachter Vortrainingsverfahren auf großskaligen Internetdaten hat leistungsfähige generische Repräsentationen hervorgebracht. Daraus resultierende vortrainierte *Basismodelle* können durch nachgelagerte Feinabstimmung weitaus effizienter und wirkungsvoller an ein breites Spektrum von *Downstream*-Aufgaben adaptiert werden als Modelle, die vollständig neu trainiert werden. In der Regel wird dabei angenommen, dass sämtliche relevanten Feinabstimmungsdaten zu Beginn vollständig verfügbar sind. In dynamischen Einsatzszenarien jedoch, in denen fortlaufend neue Daten eintreffen, verschlechtert sich die Vorhersageleistung des Modells auf zuvor gelernte Aufgaben, sobald neue Aufgaben hinzukommen – ein Phänomen, das gemeinhin als *katastrophales Vergessen* bezeichnet wird. Eine naive Gegenmaßnahme bestünde darin, das Modell bei jedem Dateneingang vollständig neu zu trainieren; für die Echtzeitverarbeitung umfangreicher Datenströme ist dies jedoch prohibitiv kostspielig und setzt zudem die Speicherung sämtlicher historischer Daten voraus, was häufig gegen gesetzliche und datenschutzrechtliche Vorgaben verstößt.

Das Forschungsgebiet des *kontinuierlichen Lernens* versucht, dieses katastrophale Vergessen zu mindern, indem es neuronalen Netzen ermöglicht, aus nicht-stationären Daten – in der Praxis repräsentiert durch Sequenzen von Datensätzen mit disjunkten Wahrscheinlichkeitsverteilungen – zu lernen und dabei zuvor erworbenes Wissen zu bewahren. Die meisten etablierten Methoden zum kontinuierlichen Lernen wurden jedoch für relativ kleine Netze entworfen, die von Grund auf und auf vergleichsweise einfachen Aufgaben trainiert werden; dies limitiert ihre Skalierbarkeit und praktische Nutzbarkeit für große Basismodelle.

Die vorliegende Dissertation stellt sich der Herausforderung der kontinuierlichen Feinabstimmung von Basismodellen. Idealerweise nutzen entsprechende Verfahren die weitreichende Generalisierungsfähigkeit vortrainierter Repräsentationen und führen kosteneffiziente, skalierbare Mechanismen ein, um sich inkrementell an vielfältige Downstream-Aufgaben anzupassen. Zu diesem Zweck vergleichen und kontrastieren wir verschiedene Paradigmen des kontinuierlichen Lernens, darunter Verfahren, die kritische Modellparameter regularisieren, eine ausgewählte Teilmenge historischer Daten wiedergeben oder bei Verteilungsverschiebungen neue Parameter einführen. Wir zeigen auf, in welchen Kontexten welches Paradigma die Anwendbarkeit auf Basismodelle begünstigt beziehungsweise einschränkt. Über den traditionellen Fokus auf unimodale Aufgaben wie Bild- *oder* Textklassifikation hinaus untersuchen wir zudem anspruchsvollere multimodale Aufgaben, die vortrainierte Repräsentationen aus mehreren Quellen kombinieren, etwa aus Bildern *und* Text. Dieser erweiterte Fokus reflektiert reale Szenarien, in denen heterogene Modalitäten zusammengeführt und diverse Aufgaben in einem einheitlichen Repräsentationsraum gelöst werden müssen.

Wir präsentieren *vier* neuartige Strategien für das kontinuierliche Downstream-Lernen mit Basismodellen: zwei für die *unimodale* inkrementelle Klassifikation und zwei für *multimodales* Schlussfolgern. Für unimodales kontinuierliches Lernen besteht unser erster Beitrag in einem Dualen-Gedächtnis-Ansatz, der selbstorganisierende Netze nutzt, um Vektorrepräsentationen eines Basismodells auf eine topologiebewahrende Mannigfaltigkeit abzubilden; dadurch bleibt die diskriminative Struktur erhalten und die Robustheit gegenüber Repräsentationsverschiebungen erhöht sich. Der zweite Ansatz aggregiert niedrige und mittlere Aktivierungen aus mehreren Schichten des Basismodells, um hochauflösende Klassenprototypen zu konstruieren, was die Klassentrennschärfe und Generalisierung unter starkem Domänenshift und Datenknappheit deutlich verbessert. Im Bereich multimodales kontinuierliches Lernen trainiert die dritte Methode inkrementell ein leichtgewichtiges Modalitäts-Interaktions-Netzwerk, das multimodale Repräsentationen fusioniert, indem es für jede Aufgabe selektiv einen kompakten Teil der Interaktionsparameter spezialisiert und den Rest über alle Aufgaben hinweg teilt. Die vierte Methode optimiert schließlich die Wissensextraktion aus gespeicherten multimodalen Vektorrepräsentationen, indem sie zielgerichtetes, räumlich bewusstes Rauschen injiziert und so die Rekonstruktion früherer Datenverteilungen verbessert.

Unsere Ergebnisse zeigen, dass alle vier in dieser Arbeit vorgestellten Methoden die Fähigkeit von Basismodellen erheblich steigern, nicht-stationäre Daten während der Feinabstimmung sequenziell zu erlernen, wobei sie nur minimale Annahmen hinsichtlich Datentyp, -umfang oder Präsentationsreihenfolge treffen. Durch das strategische Ergänzen vortrainierter Repräsentationen, das Schützen entscheidender Parameter, die generalisiertes Wissen kodieren, und das Wiederherstellen vergangener statistischer Verteilungen aus einem minimalen Satz gespeicherter Repräsentationen liefert diese Arbeit praktische, skalierbare Lösungen für ein breites Spektrum kontinuierlicher Lernprobleme mit großen vortrainierten Modellen. Zusammengenommen versetzen diese Fortschritte Basismodelle in die Lage, den Anforderungen dynamischer und sich fortlaufend wandelnder Downstream-Daten in realen Einsatzumgebungen gerecht zu werden.

Contents

I Background and Motivation

II Unimodal Representation Augmentation

III Multimodal Representation Integration

IV Closing

Appendices

List of Figures

List of Tables

Part I

Background and Motivation

Chapter 1

Introduction

"It is not the most intellectual of the species that survives; it is not the strongest that survives; but the species that survives is the one that is able best to adapt and adjust to the changing environment in which it finds itself."

Darwin's Theory paraphrased by Leon C. Megginson

1.1 Motivation

Imagine being the head chef of a bustling culinary school. Your kitchen is legendary—a state-of-the-art *foundation* capable of producing nearly any dish from all over the world. From day one, you have meticulously trained your staff in hundreds of recipes: soufflés from France, dumplings from China, intricate pastries from Vienna. This training took months, but it was worth it. Your kitchen staff—the *foundation model*—now possesses an impressive breadth of culinary skills.

However, your school faces a constant stream of new requests. One day, a guest wants a gluten-free twist on a traditional baguette; the next, a celebrity chef challenges you to replicate a 200-year-old Italian dessert. These updates may seem like minor tweaks—just fine-tuning an already expert team—but here's the catch: each time you teach your crew a new recipe, they risk forgetting key details of the old ones. Perhaps your sous-chefs start mixing up the measurements for your famous soufflé or forget the subtle folding technique for dumpling wrappers. You could retrain everyone from scratch with each new request, but that would be expensive, time-consuming, and wasteful. Clearly, you need a way to continuously incorporate new recipes without discarding your staff's hard-earned knowledge.

This scenario mirrors how contemporary foundation models in machine learning operate. Like your kitchen staff, foundation models start with an extensive repertoire of capabilities, developed through large-scale pretraining on diverse data sampled from a joint probability distribution. However, when fine-tuning them sequentially for new downstream tasks, unrestricted parameter updates can lead to *catastrophic* *forgetting*, where previously learned knowledge is overwritten. In dynamic environments where data evolve and new tasks gradually emerge, we cannot afford to start from scratch each time. Instead, we need models that can update incrementally in a way that preserves old skills while effectively learning new ones.

In this thesis, we explore methods for downstream *Continual Learning* (CL) with foundation models. Rather than retraining on all data at once, we aim to exploit their existing knowledge, efficiently adapt them to incrementally available tasks, and prevent them from forgetting previously learned ones. In short, we strive to develop artificial systems that learn like real-world chefs: building on what they already know, selectively refining their expertise, and remaining versatile in the face of ever-changing culinary—or computational—challenges.

1.2 Research Objectives

CL research in the context of foundation models aims at equipping these large-scale pretrained neural networks with the ability to adapt to emerging tasks or shifting data distributions while preserving previously acquired capabilities. Although foundation models often excel under the assumption that training samples originate from a fixed—albeit unknown—probability distribution, real-world applications such as autonomous driving or embodied robotics commonly involve evolving, complex, and multimodal data streams. In these dynamic settings, foundation models are susceptible to catastrophic forgetting, whereby earlier learned skills deteriorate when the model is fine-tuned on new tasks. A naive solution might involve retaining all historical data and re-initiating the fine-tuning process whenever the training data distribution shifts; however, such a strategy is computationally prohibitive, memory-intensive, and incompatible with real-time constraints. Moreover, strict privacy or regulatory restrictions on data storage can render such a solution entirely impractical, underscoring the need for both versatile and efficient strategies for downstream continual fine-tuning.

A CL method is considered *versatile* if it makes minimal assumptions about both (i) the types of data it learns and (ii) how those data are presented. The first point highlights the necessity of accommodating unimodal as well as multimodal tasks, whereas the second addresses assumptions regarding the number of passes over each task's data stream, the availability of historical data, the presence of or information about task boundaries, and other forms of prior knowledge that can simplify the CL problem. In this context, we define *unimodal* tasks as those requiring a single input source (*e.g.*, text), while *multimodal* tasks involve the integration of multiple data sources (*e.g.*, text and images). Beyond these considerations of versatility, an *efficient* CL method should introduce only minimal additional computational and memory overhead beyond the base cost of the foundation model itself. CL methods that excel in both versatility and efficiency thus lay the groundwork for scalable, real-time adaptive systems capable of robust performance in diverse, complex, and continuously evolving environments.

Accordingly, the primary goal of this thesis is to **develop efficient and versatile CL methods that enable foundation models to be incrementally fine-tuned on sequences of diverse and complex unimodal and multimodal tasks, all while preserving previously acquired capabilities.** Guided by this overarching goal, we define the following research objectives:

- **Objective 1:** Advance the robustness of downstream continual fine-tuning on unimodal tasks with respect to both the sequence order in which tasks are learned and variations in the number of training samples used for fine-tuning.

- **Objective 2:** Develop a CL approach that integrates seamlessly with any foundation model pretrained on unimodal tasks, while making minimal assumptions about data presentation and requiring low runtime and memory overhead.

- **Objective 3:** Generate diagnostic multimodal CL datasets to analyze the role of specific components in modality interaction networks and use these insights to design a resource-efficient CL method that preserves critical model parameters to prevent forgetting during continual fine-tuning on multimodal data.

- **Objective 4:** Enable scalable continual downstream fine-tuning of foundation models on open-domain real-world multimodal data without forgetting, under minimal assumptions about data presentation and with low computational and memory costs.

The first two objectives address CL for unimodal tasks while progressively reducing the assumptions underlying the continual fine-tuning process of a foundation model. **Objective 1** focuses on mitigating the adverse effects of task ordering and limited training samples, whereas **Objective 2** further emphasizes resource efficiency and minimizes reliance on task-specific information or stored data. Building on these insights, the subsequent objectives extend to multimodal tasks. **Objective 3** is to investigate the behavior of modality interaction networks during CL and leverage these findings to improve the resource efficiency of methods that protect key network parameters. Finally, **Objective 4** extends these capabilities to open-domain real-world data, allowing foundation models to learn continuously with minimal constraints on data diversity, presentation, and resource consumption. Achieving this final objective means that a CL method is truly scalable, versatile, and efficient. Collectively, these objectives contribute to a robust and scalable framework for continual downstream fine-tuning of foundation models, advancing CL methods that effectively tackle a wide range of challenging tasks and scenarios.

1.3 Main Approaches and Novel Contributions

In this thesis, we propose novel strategies and design methodologies for CL in the context of downstream fine-tuning of foundation models. Specifically, we introduce four innovative CL approaches: two designed to enhance representations in unimodal learning settings and two aimed at fusing representations across multiple modalities. Through comprehensive experiments, we not only evaluate our methods and their variations but also provide new insights into which CL strategies and paradigms that were originally developed for models trained from scratch remain effective in the era of foundation models. In summary, our contributions are as follows:

Chapter 4 (Unsupervised Representation Modeling via Self-organization): We present **DRILL**, a dual-memory CL method that integrates self-organizing network principles with foundation models to preserve topological relationships in the latent feature space and mitigate representational drift. Our approach combats catastrophic forgetting by augmenting features extracted from a foundation model with prototypical class vectors, thereby stabilizing the training process and improving retention of previously learned tasks. We evaluate DRILL under challenging scenarios that involve imbalanced data and varying task orders, demonstrating reduced variability in outcomes and strong classification performance—particularly when underrepresented classes appear early in the learning sequence. By consistently achieving high classification accuracy regardless of task order or fluctuations in training sample sizes, DRILL directly addresses **Objective 1** of this thesis.

Chapter 5 (Prototyping with Intermediate Features): We introduce **LayUP**, a novel prototype-based CL method that leverages first- and second-order feature statistics extracted from multiple intermediate layers of a foundation model. By exploiting low- and mid-level representations, LayUP captures richer, more invariant features that enhance classification robustness—especially under significant domain shifts and in low-data scenarios. Our method is conceptually straightforward and demonstrates notable improvements in both performance and resource efficiency compared to strong baselines across diverse image classification benchmarks and CL settings. Crucially, LayUP realizes **Objective 2** by seamlessly integrating with any unimodal foundation model under minimal data presentation assumptions, while requiring substantially lower memory and computational overhead than prior CL methods for pretrained models.

Chapter 6 (Multimodal Fusion with Selective Specialization): We propose **SMS**, a novel multimodal CL approach applicable to a wide variety of foundation models. In SMS, a selected subset of network modules is dedicated to each task, while the remaining modality interaction parameters are shared. These task-specific and shared modules are trained through an alternating adaptation-consolidation scheme inspired by principles of cognitive development. To systematically evaluate different strategies for module specialization and balance task-specific adaptation with shared knowledge retention, we introduce two diagnostic **LILAC** datasets. These datasets facilitate controlled experiments aimed at revealing the distinct contributions of various layers and components within modality interaction networks in a CL setting. Our results demonstrate that dedicating even a small portion of parameters within the modality interaction network to task-specific specialization consistently yields superior performance compared to existing baselines. This clearly highlights the pivotal role that targeted parameter specialization plays in effectively mitigating catastrophic forgetting. Thus, by offering the LILAC datasets alongside an efficient SMS method designed to strategically preserve crucial parameters, this work meets **Objective 3** of this thesis.

Chapter 7 (Noise-augmented Multimodal Latent Replay): We introduce **NLR**, a novel latent replay method for multimodal CL that augments buffer-stored features—extracted by a foundation model—with spatially aware noise. NLR dynamically identifies regions of interest, such as the location of a target object, within the latent space and applies minimal perturbations to these salient foreground features, while imposing a higher degree of controlled statistical noise on the surrounding background features. This mechanism expands the diversity of the feature space without significantly distorting the encoded representation of the foreground, thereby alleviating the background bias that is often observed when regions of interest occupy a limited spatial extent. Our experimental evaluations on continual referring expression segmentation tasks indicate that this noise augmentation strategy can reduce the dependency on stored latent representations by up to 80%. Furthermore, NLR imposes minimal assumptions regarding data presentation, *e.g.*, by operating effectively without explicit task boundaries and being both exemplar-free and fully compatible with online learning paradigms. In summary, NLR successfully addresses **Objective 4** and demonstrates a straightforward and effective approach to mitigating catastrophic forgetting in multimodal CL with foundation models.

1.4 Thesis Outline

This thesis is organized into four distinct parts. The introductory and concluding remarks are presented in Parts I and IV, respectively, while the core research is detailed in Parts II and III. In each of these core parts, two novel methodologies are introduced for modeling or integrating data representations encoded by foundation models within CL frameworks.

Part I (Background and Motivation): Following the introduction in Chapter 1, Chapter 2 outlines two major paradigm shifts that have propelled recent advances in deep learning. First, we trace the journey from medium-scale models—originally trained from scratch on human-annotated datasets—to foundation models that leverage large-scale, upstream pretraining on unlabeled, web-crawled data. Second, we analyze the rapid advancements in multimodal perception and reasoning, positing that models which integrate heterogeneous signals (*e.g.*, from vision and language) yield enhanced contextual representations and superior inferential capabilities compared to those trained on unimodal inputs. In Chapter 3, we introduce CL within the context of these paradigm shifts. We contend that large-scale pretraining, although transformative, is insufficient on its own to mitigate the phenomenon of catastrophic forgetting during downstream fine-tuning on sequences of tasks or datasets. To establish a robust theoretical foundation, we first review the emergence of CL as a paradigm in both biological and artificial systems. We then propose a framework for integrating CL strategies with foundation models and offer a concise review of related methodologies. This synthesis not only elucidates the current state of the art but also identifies the open challenges that this thesis seeks to address.

Part II (Unimodal Representation Augmentation): This part addresses the challenges of fine-tuning unimodal transformer-based foundation models in non-stationary environments. In Chapter 4, we show that capturing both short- and long-range dependencies in latent representations via self-organized topology mapping stabilizes the fine-tuning trajectory of pretrained transformer encoders. Within this framework, we introduce DRILL, a novel CL method, and benchmark its performance against several baselines using streams from five established text classification datasets under two imbalanced sampling strategies. Because DRILL requires fine-tuning all parameters of the foundation model and historical data storage, Chapter 5 explores alternative approaches to improve data efficiency in downstream continual fine-tuning without full model adaptation or storing historical data. Specifically, we propose LayUP, a prototype-based CL approach that leverages pretrained representations from multiple intermediate layers of a foundation model. We rigorously evaluate LayUP on ten diverse image recognition benchmarks, covering three challenging CL scenarios. Finally, we establish best practices for selecting the maximum layer depth for direct feature extraction to construct prototypes that align with the characteristics of the downstream fine-tuning data.

Part III (Multimodal Representation Integration): As a follow-up to the previous part, we tackle more challenging CL scenarios in which pretrained representations of multiple sources have to be continuously integrated. We perform a post hoc analysis in Chapter 6 to investigate which parts of a modality interaction network learn task-specific, specialized representations and which learn task-agnostic representations for knowledge transfer during CL. Based on this analysis, we describe the conceptually simple SMS method, which we subsequently evaluate on two novel datasets and for three different model configurations. Since SMS requires explicit task-specific information during both training and testing—and necessitates storing full network layers for each task—we explore alternative strategies in Chapter 7 to store only a small set of latent representations for replay during multimodal CL. This approach improves scalability, reduces storage overhead, relaxes data presentation assumptions, and preserves privacy by storing extracted features rather than raw data. Initially, we assess how buffer size and replay frequency affect latent replay and empirically demonstrate that replay is susceptible to background bias when only a limited number of latent representations are stored. Building on these findings, we propose our multimodal CL method NLR and perform extensive evaluations on three split datasets for continual referring expression segmentation.

Part IV (Closing): In Chapter 8, we provide a summary of the key contributions and findings presented in this thesis and explore open questions along with potential avenues for future research.

Chapter 2

Representation Learning in the Foundation Model Era

This chapter reviews advancements and methodologies in representation learning during the foundation model era, a period inaugurated by the advent of the transformer architecture (Sutskever et al., 2014; Vaswani et al., 2017). The *foundation model era* is characterized by the development and deployment of large-scale Pretrained Models (PTMs) that serve as versatile bases for a broad spectrum of deep learning applications. A defining shift in this era is the transition from training models entirely from scratch to adopting a two-phase approach: an *upstream* pretraining phase followed by *downstream* adaptation. In the pretraining phase, models are exposed to extensive datasets using self-supervised objectives—such as masked language modeling, contrastive learning, or similar techniques—to learn robust general-purpose representations. Subsequently, these models are fine-tuned in a supervised manner on considerably smaller, task-specific datasets.[1] This training paradigm has driven significant progress across multiple Artificial Intelligence (AI) domains, including Natural Language Processing (NLP) and Computer Vision (CV).

Section 2.1 discusses the architectural paradigm shift introduced by the transformer model and provides a brief overview of its applications in NLP and CV. Section 2.2 outlines transformer architectures for multimodal representation learning, with a particular focus on methodologies for learning, aligning, and integrating heterogeneous representations across different modalities.

2.1 Self-supervised Pretraining at Scale

The transformer model (Vaswani et al., 2017) has fundamentally reshaped the landscape of AI by reducing the dependence on recurrence and convolution mech-

[1] Throughout this thesis, *training from scratch* means optimizing a model whose parameters are randomly initialized, whereas *fine-tuning* means further training a model that starts from an existing checkpoint with pretrained weights.

anisms that were once standard in neural network architectures. In contrast to Recurrent Neural Networks (RNNs; Rumelhart et al., 1986) and Convolutional Neural Networks (CNNs; LeCun et al., 1989), which had long been the foundational paradigms in deep learning research, the transformer employs self-attention mechanisms exclusively to encode arbitrary sequences of inputs. This design enables parallel processing and facilitates the capture of long-range dependencies more effectively, while also achieving greater computational efficiency.

Although the original transformer model introduced by Vaswani et al. (2017) was trained from scratch for specific tasks, the subsequent development of Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2019) demonstrated that self-supervised pretraining on extensive corpora can markedly enhance a transformer's ability to generate high-quality context-aware representations. In *self-supervised learning*, models automatically derive supervisory signals from unlabeled data by constructing predictive tasks—*e.g.*, predicting masked tokens in the case of BERT—thus eliminating the need for manually annotated datasets. This approach differs from traditional supervised learning, which relies on external labels, and from certain unsupervised methods that do not explicitly formulate predictive tasks. Using self-supervised objectives during pretraining, transformer-based models acquire dense and informative representations of input sequences, often outperforming CNNs and RNNs on tasks that require complex contextual understanding.

2.1.1 Sequence Modeling with Transformers

Prior to the advent of the transformer architecture, sequential encoding in NLP applications was predominantly achieved using RNNs, including variants such as Gated Recurrent Unit (GRU; Cho et al., 2014) and Long Short-Term Memory (LSTM; Hochreiter and Schmidhuber, 1997). These models process input sequences one step at a time, each step depending on the previous state, which inherently restricts parallel processing. In contrast, transformers employ self-attention mechanisms that enable the simultaneous processing of entire sequences. This design not only facilitates parallel computation, but also enhances the model's ability to capture and maintain long-range dependencies. As a result, transformers have largely supplanted RNNs in many deep learning applications, thanks to their reduced training times and improved scalability.[2]

The original transformer—often termed *vanilla transformer* and depicted in Figure 2.1—was introduced as a sequence-to-sequence (Sutskever et al., 2014) model for machine translation. Its architecture comprises two main components: an encoder and a decoder, each built by stacking L identical layers. The encoder converts an input sequence of embedded tokens into a set of continuous representations, while the decoder generates the output sequence in an autoregressive fashion. At

[2] For a comprehensive overview of transformer architectures and their variants, see Lin et al. (2022b).

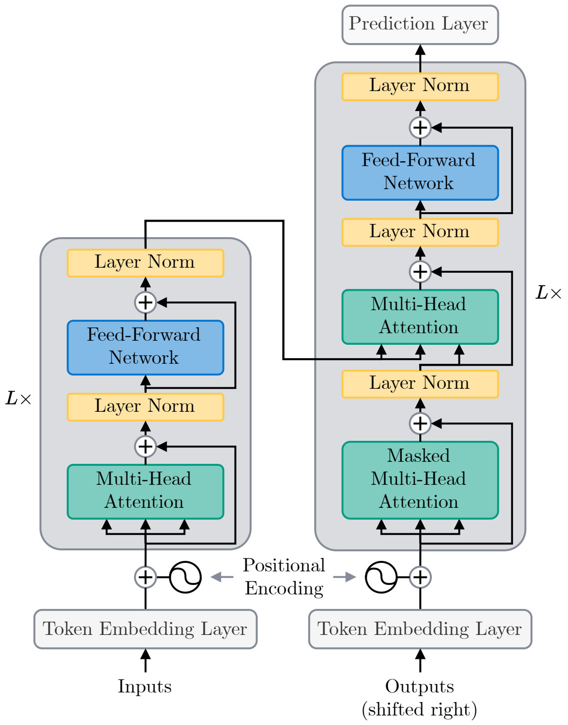


Figure 2.1: Overview of the transformer encoder-decoder architecture. The model employs multi-head attention, feed-forward networks, layer normalization, and skip connections as its main operations. Separate attention layers are used for encoder inputs and decoder outputs. Positional encoding, implemented through a sinusoidal function, enables the model to capture word order. Masked attention allows the decoder to generate token sequences autoregressively, producing one token at a time. Adapted from Vaswani et al. (2017).

each step, the decoder produces one token by conditioning on both the previously generated tokens and the encoder's representations.

Each encoder block in the transformer architecture comprises two primary sublayers: a multi-head self-attention mechanism and a position-wise Feed-Forward Network (FFN). The FFN typically consists of two fully connected layers, with a nonlinear activation function applied after the first layer. Each sublayer is equipped with a residual connection (He et al., 2016), and the combined output is normalized using layer normalization (Ba et al., 2016) to stabilize training and promote efficient gradient flow. In contrast, each decoder block incorporates an additional cross-attention mechanism—essentially a second multi-head attention sublayer—situated between the decoder's masked multi-head self-attention and its FFN. This cross-attention mechanism allows the decoder to attend to the encoder's output representations. Moreover, the self-attention in the decoder is modified via a masking strategy that prevents any position from accessing information from subsequent positions, thereby enforcing the autoregressive property during generation. Finally, the decoder output is projected through a linear transformation and converted to probability distributions over the vocabulary using a softmax function.

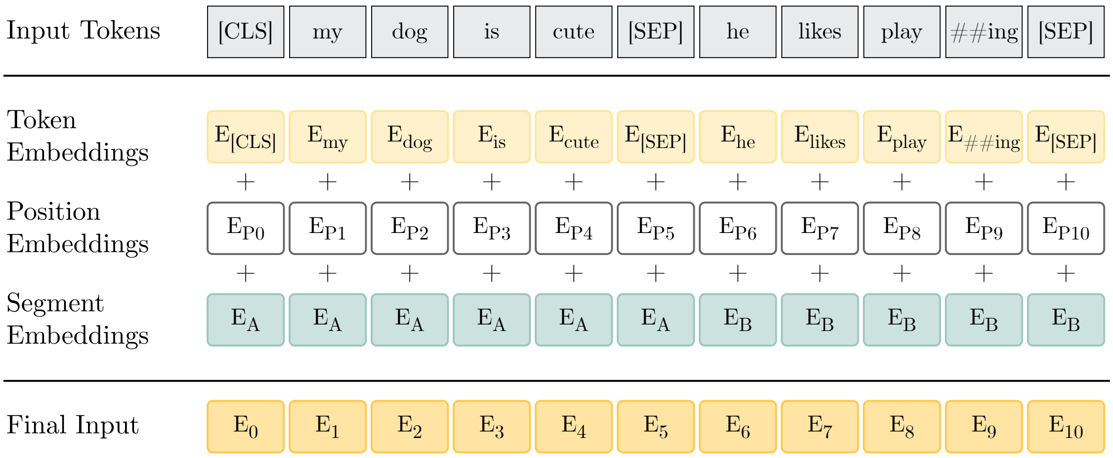
In order to provide information about the order of tokens in a sequence, positional encoding adds unique, fixed positional information to the input embeddings using sinusoidal functions to generate position-dependent vectors. Subsequent work explores learnable positional embeddings (Gehring et al., 2017; Devlin et al., 2019) as an alternative to fixed sinusoidal encodings, the former offering greater flexibility and adaptability to training data, and the latter potentially providing superior generalization to sequence lengths beyond those encountered during training (Liu et al., 2020b).

Many early modifications to the vanilla transformer architecture, some of which will be discussed in this chapter, focused exclusively on the encoder blocks. These adaptations were devised to showcase the model's efficacy across a broad spectrum of downstream tasks in both NLP and CV applications. In contrast, other studies have employed both the encoder and decoder for sequence generation, as evidenced by general-purpose Large Language Models (LLMs) such as OpenAI's GPT series (Radford et al., 2018, 2019; Brown et al., 2020; OpenAI et al., 2024). Although encoder-decoder models are highly effective—particularly for sequence-to-sequence learning—they tend to incur significant computational overhead due to their increased complexity and longer training times. Since this thesis does not address sequence generation and considering the substantial structural overlap between encoder and decoder blocks, we concentrate on encoder-only transformer models to establish a streamlined framework for CL tasks in downstream applications.

2.1.2 Transformers in Natural Language Processing

Building on the paradigm shift initiated by the vanilla transformer, subsequent research has harnessed its performance and computational efficiencies to advance a variety of NLP tasks. This progress has culminated in models that are pretrained on vast text corpora, with BERT (Devlin et al., 2019) representing a pivotal breakthrough. In contrast to earlier pretrained language models—such as the RNN-based ELMo (Peters et al., 2018) and the unidirectional GPT (Radford et al., 2018)—BERT utilizes a transformer encoder to capture bidirectional context simultaneously. Its self-supervised pretraining on extensive unlabeled data enables the model to learn deep *contextualized* representations, where each token embedding incorporates information from both preceding and succeeding contexts. This paradigm of large-scale pretraining followed by task-specific fine-tuning yields contextually rich representations that significantly enhance generalization and performance across diverse NLP applications.

BERT builds on the encoder architecture of the vanilla transformer, but introduces a unique input representation constructed by summing token embeddings, segment embeddings, and learnable positional embeddings, as depicted in Figure 2.2. This composite embedding enables BERT to capture the semantic meaning of tokens, their positional information within a sentence, and distinctions between sentences within the input sequence. Special tokens, specifically the classification ([CLS]) and separation ([SEP]) tokens, are incorporated to enable the model to handle vari-



Input Tokens	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[SEP]}$	E_{he}	E_{likes}	E_{play}	$E_{\#\#ing}$	$E_{[SEP]}$
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_{P0}	E_{P1}	E_{P2}	E_{P3}	E_{P4}	E_{P5}	E_{P6}	E_{P7}	E_{P8}	E_{P9}	E_{P10}
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
Final Input	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

Figure 2.2: Input representations for BERT. Each token is represented by the sum of three embeddings: token embeddings (representing each word or special token like [CLS] and [SEP]), position embeddings (indicating word order), and segment embeddings (distinguishing between sentences). This combined representation enables BERT to capture contextual relationships across the entire input sequence. Adapted from Devlin et al. (2019).

ous NLP tasks, including sentiment analysis and question answering, by explicitly marking the input data structure. The [CLS] token is prepended to each input sequence, with its embedding serving as an aggregate representation of the entire sequence, which is particularly useful for classification tasks. The [SEP] token is used to separate sentences or segments within the input, which allows BERT to model relationships between sentences.

BERT's self-supervised pretraining is built upon two primary objectives: *masked language modeling* and *next sentence prediction* (*e.g.*, as used in Logeswaran and Lee, 2018). In the masked language modeling task, a fraction of tokens within the input sequence is randomly masked, and the model is trained to predict these masked tokens based solely on their surrounding context. This process enables the model to acquire rich, bidirectional contextual representations. The next sentence prediction task, on the other hand, is designed to capture sentence-level coherence by training the model to distinguish whether a given pair of sentences are consecutive segments of the original text. Both objectives exploit unlabeled text corpora, thereby eliminating the need for costly manual annotations. In practice, BERT is trained on large-scale, diverse datasets, specifically the English Wikipedia (∼2.5 billion words) and the BooksCorpus (∼800 million words; Zhu et al., 2015).

Since its introduction, BERT has inspired a wide array of successor models that build upon its core architecture to address various aspects of language representation. Some approaches focus on refining and expanding pretraining objectives (*e.g.*, RoBERTa, Liu et al., 2019; XLNet, Yang et al., 2019), while others prioritize greater computational efficiency (*e.g.*, DistilBERT, Sanh et al., 2019; TinyBERT, Jiao et al., 2020). In addition, specialized variants have been developed for domain-specific tasks such as biomedical texts (*e.g.*, BioBERT, Lee et al., 2020) or scientific literature (*e.g.*, SciBERT, Beltagy et al., 2019) and for multilingual contexts (*e.g.*, mBERT, Devlin et al., 2019; XLM-R, Conneau et al., 2020). Despite these advances, the original BERT model continues to serve as a remarkably robust and widely adopted baseline in contemporary NLP research, and it is used as the foundation model in Chapter 4 of this thesis.

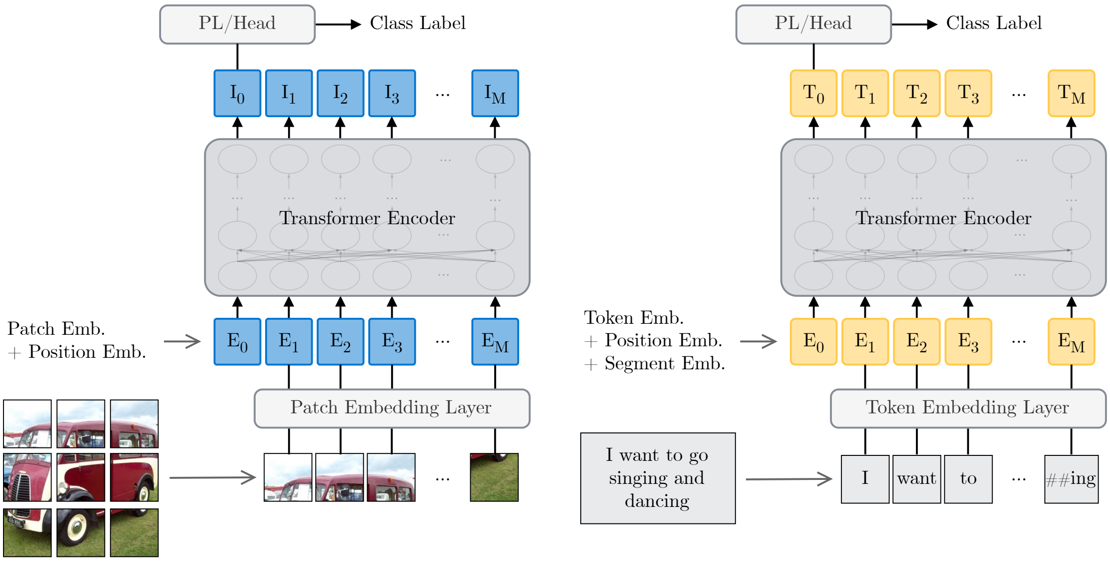
2.1.3 Transformers in Computer Vision

The transition from transformer architectures in NLP to their application in CV was relatively rapid. CNNs leverage strong inductive biases—such as local connectivity, spatial hierarchies, and translation invariance—to efficiently process images. In contrast, transformers inherently lack these biases, instead processing image patches in parallel as unordered sets, which enables them to capture global relationships across the entire image. However, deploying transformers in CV applications has presented significant challenges. Unlike NLP, where transformers handle sequences of discrete tokens, images are composed of high-dimensional pixel arrays, resulting in greater computational and memory demands. This challenge is compounded by the quadratic complexity of the self-attention mechanism,[3] which makes processing high-resolution images particularly resource intensive. Moreover, while the minimal inductive biases of transformers offer flexibility, they also necessitate substantial architectural modifications and access to large-scale training datasets to effectively learn spatial hierarchies and capture local patterns in image data.

The Vision Transformer (ViT; Dosovitskiy et al., 2021) is a seminal work demonstrating that transformers can fully replace traditional convolutions in deep neural networks while outperforming large pretrained CNN-based models despite using significantly fewer computational resources. ViT adapts the transformer architecture to image processing by partitioning images into fixed-size patches and treating each patch as an individual token, analogous to words in NLP. These patches are flattened and linearly projected into a sequence of embeddings, allowing the model to apply self-attention to capture global relationships across the image. This patch-based tokenization significantly reduces the sequence length compared to a pixel-wise representation, thereby mitigating the computational challenges associated with high-resolution inputs.

In addition, ViT is pretrained on large-scale datasets such as ImageNet-21K (∼14 million images; Deng et al., 2009) and JFT-300M (∼300 million images; Sun et al., 2017). This extensive pretraining enables the model to learn robust, generic representations despite the absence of the strong inductive biases inherent in CNNs. Furthermore, positional embeddings—similar to those used in NLP transformers—are incorporated to encode spatial information, thereby compensating for the lack of convolutional structure. Figure 2.3 illustrates the differences between image and

[3] A per-layer complexity analysis of the vanilla transformer is provided in Vaswani et al. (2017).



(a) Image processing with ViT

(b) Text processing with BERT

Figure 2.3: Input encoding with transformers for CV and NLP. (a) ViT divides images into fixed-size patches and embeds them via linear projection. **(b)** BERT tokenizes text sequences into words or subwords and embeds them accordingly. Both models add positional embeddings to their patch or token embeddings to retain spatial and order information. In BERT, segment embeddings are also added to differentiate between sentence pairs. For simplicity, we omit special tokens beyond the classification token. **PL** = Prediction Layer.

text processing in ViT and BERT, respectively, highlighting how each architecture adapts the transformer framework to its respective modality.

Hybrid models that combine CNNs and transformers aim to harness the spatial locality and parameter efficiency of CNNs alongside the global context awareness and attention mechanisms of transformers. These models typically integrate CNN layers to extract local features and reduce computational complexity, followed by transformer layers to capture long-range dependencies and complex interactions. For example, the Convolutional Vision Transformer (CvT; Wu et al., 2021a) integrates convolutional layers directly into the transformer's embedding process to enhance the quality of extracted features at the input stage. In contrast, the Swin Transformer (Liu et al., 2021) employs a hierarchical structure in which convolutional operations are incorporated through shifted windows in multiple stages, allowing efficient and scalable feature extraction in different resolutions. These hybrids demonstrate enhanced performance on various vision tasks compared with convolution-free architectures such as ViT.

Despite the advantages of hybrid models, the original ViT remains a widely recognized baseline in contemporary research due to its simplicity, strong performance, and its foundational role in inspiring subsequent transformer-based architectures in CV. As such, the ViT model will serve as the base model for Chapter 5 in this thesis.

2.1.4 Parameter-efficient Fine-tuning

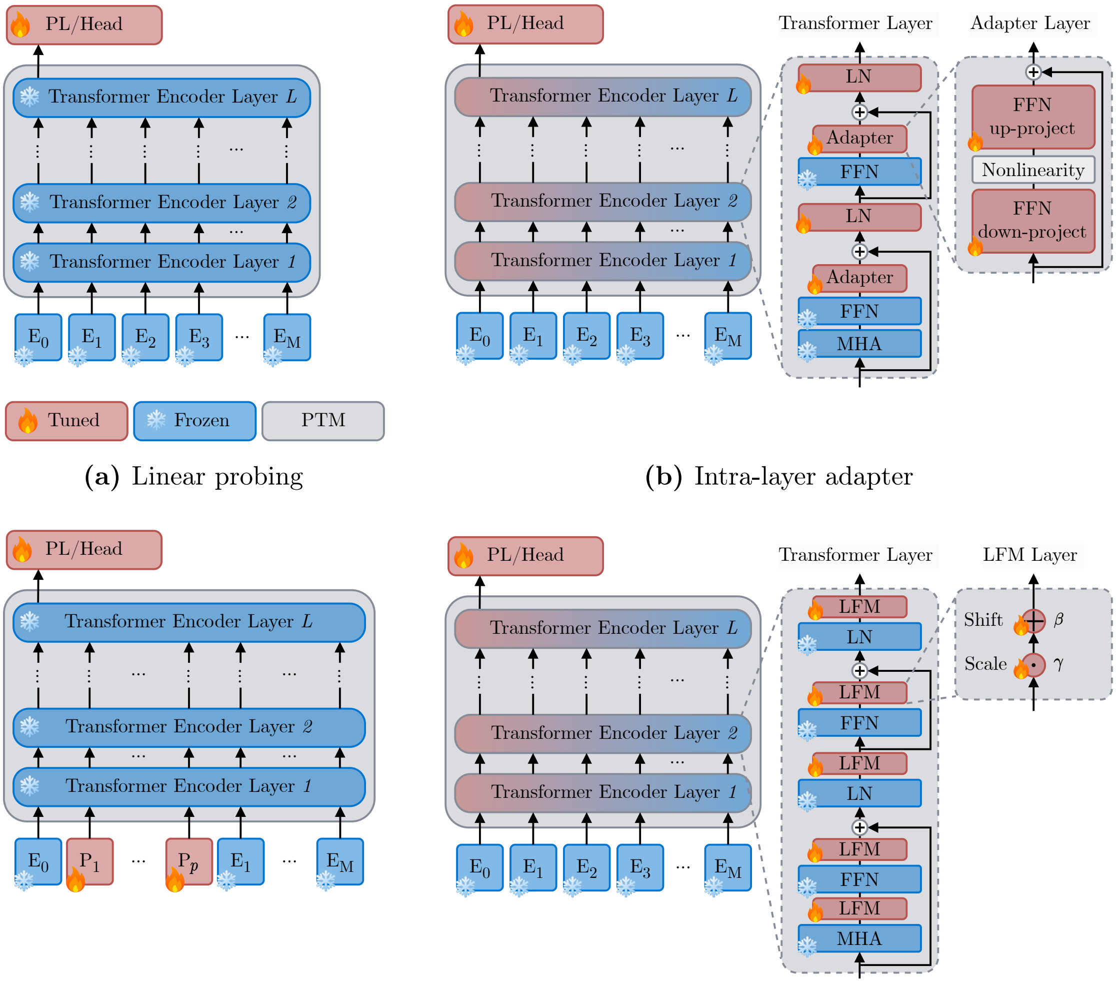
As transformer models continue to scale, fully fine-tuning all their parameters for every new task becomes prohibitively expensive in terms of computation and memory. Parameter-Efficient Fine-Tuning (PEFT) addresses this challenge by introducing a relatively small set of additional parameters—often only a fraction of the original model's size—while keeping the bulk of the pretrained parameters frozen. This approach significantly reduces resource requirements and can achieve performance that matches or even exceeds that of full fine-tuning, particularly in scenarios with limited training data.

This section presents four prominent PEFT strategies: linear probing, intra-layer adapters, prompt (or prefix) tuning, and linear feature modulation. An overview of these approaches is depicted in Figure 2.4. Given that (i) pretrained representations are generally more robust to distributional shifts in input data, and (ii) PEFT methods can perform on par with or even exceed the effectiveness of full backbone fine-tuning (Hu et al., 2022), the application of PEFT techniques to pretrained models has become a cornerstone of CL research.[4]

Linear probing. A straightforward approach to training task-specific parameters while keeping the pretrained backbone fixed is to introduce a separate Prediction Layer (PL), commonly referred to as a *linear probe* for classification and recognition tasks, for each downstream task, as illustrated in Figure 2.4a. This linear probe typically consists of a fully connected classification layer that linearly projects the [CLS] token embedding from the final (L^{th}) transformer encoder layer, which has dimensionality d_L, onto the set of target classes. As indicated in Table 2.1, linear probing is the most memory-efficient among PEFT methods due to its minimal parameter footprint. However, this efficiency is achieved at the expense of performance, as linear probing generally underperforms compared to fully fine-tuning the entire pretrained backbone.

Intra-layer adapters. *Adapters* are lightweight modules that are integrated within the layers of a foundation model to facilitate task-specific fine-tuning without altering the original model parameters, which remain frozen. Typically, intra-layer adapters are trained jointly with a task-specific linear probe, allowing for efficient adaptation to new tasks with a minimal increase in parameter count. The concept was initially introduced by Rebuffi et al. (2017a) in the context of CNNs and later refined and popularized by Houlsby et al. (2019) for use in pretrained transformers, particularly in NLP tasks.

[4] More details on PEFT methods for CL are provided in Section 3.3.2.



(a) Linear probing

(b) Intra-layer adapter

(c) Prompt tuning

(d) Linear feature modulation

Figure 2.4: Parameter-efficient fine-tuning methods for foundation models. (a) Linear probing adjusts only the output head parameters to the fine-tuning data. **(b)** Intra-layer adapters are bottleneck-structured multi-layer perceptrons inserted twice in each transformer encoder layer. **(c)** Prompt tuning learns p prompt tokens alongside the output head parameters. **(d)** Linear feature modulation learns scaling and shifting parameters, γ and β, applied to the token embeddings after each transformer block. **PTM** = Pretrained Model, **MHA** = Multi-Head Attention, **FFN** = Feed-Forward Network, **LN** = Layer Normalization, **LFM** = Linear Feature Modulation.

As depicted in Figure 2.4b, adapters in transformer models are implemented as lightweight, bottleneck-structured multi-layer perceptrons that are typically inserted after the multi-head attention mechanism and the FFN within each encoder block. This design has proven particularly effective for scenarios that involve incremental or sequential task adaptation, as adapters can be easily swapped or updated to accommodate new tasks. Innovations such as AdapterFusion (Pfeiffer et al., 2021) extend this paradigm by allowing the model to integrate multiple

Method	**Params. ($\times 10^6$)**	**Accuracy (%)**
Full	85.88	93.82
Linear probing	0.08	88.70
Adapter (Houlsby et al., 2019)	0.31	93.34
VPT (Jia et al., 2022)	0.54	93.17
SSF (Lian et al., 2022)	0.28	93.99

Table 2.1: Comparison of performance and resource demands of different fine-tuning strategies. The pretrained model is ViT-B/16, and the fine-tuning dataset is CIFAR-100 (Krizhevsky, 2009). *Params.* refers to the total number of parameters updated during the fine-tuning stage. *Full* refers to the joint fine-tuning of all ViT model parameters and is, therefore, not considered a PEFT method. The performance differences between Adapter, VPT, and SSF are minimal, offering no clear indication of the superiority of any PEFT method they represent. Reported values are taken from Lian et al. (2022). For further details, refer to the cited work.

task-specific adapters, thus facilitating cross-task knowledge transfer and improving parameter efficiency through the reuse of previously acquired task representations. Additionally, AdaptFormer (Chen et al., 2022) offers a more lightweight alternative by incorporating a bottleneck-shaped network as a parallel processing pathway—typically positioned after the FFN within each transformer layer—to reduce computational overhead while preserving performance.

Prompt or prefix tuning. Another PEFT strategy, illustrated in Figure 2.4c, involves appending learnable, task-adaptive vectors—commonly known as *prompts* or *prefixes*—to the input data. Unlike methods that modify parameters within the pretrained backbone, prompt and prefix tuning strategies extend the embedding sequence processed by the transformer encoder to steer the model's output toward task-specific predictions. The key difference between the two lies in the insertion point of the learnable vectors: prompt tuning appends these vectors only at the input layer, whereas prefix tuning introduces them at every transformer layer. Although prefix tuning requires more additional parameters, it has the potential to yield improved performance over prompt tuning. Similarly to adapter-based approaches, both methods typically employ a task-specific linear probe to further refine the model predictions. As demonstrated by Liu et al. (2022) and supported by the findings in Table 2.1, these techniques can achieve performance comparable to full backbone fine-tuning.

Originally introduced in the domain of NLP (Lester et al., 2021), prompt and prefix tuning have since demonstrated their adaptability in various fields. For instance, Visual Prompt Tuning (VPT; Jia et al., 2022) has been successfully applied to transformer architectures in CV tasks. Similarly, Context Optimization (CoOp; Zhou et al., 2022b) and Conditional Context Optimization (CoCoOp; Zhou et al., 2022a) have been developed specifically for Vision-Language (VL) tasks, thereby broadening the scope and applicability of this PEFT strategy.

Linear feature modulation. Another method for achieving task-specific adaptation with minimal parameter modifications is through *linear feature modulation*, which applies a linear transformation to the extracted features of a deep neural network to induce specific effects in the network's output, as depicted in Figure 2.4d. Let d denote the dimensionality of the feature embedding and M the number of input tokens; consider an input $x \in \mathbb{R}^{(M+1)\times d}$. The transformed output $y \in \mathbb{R}^{(M+1)\times d}$ is computed as

$$y = \gamma \odot x + \beta, \tag{2.1}$$

where $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ are learned scale and shift parameters, respectively, and $\odot$ denotes the dot product. This strategy is inspired by normalization techniques such as batch normalization (Ioffe and Szegedy, 2015) and layer normalization (Ba et al., 2016), which first normalize features and then apply learned scaling and shifting to modulate the feature distribution. These methods are widely adopted to stabilize training and enhance generalization.

Feature-wise Linear Modulation (FiLM; Perez et al., 2018) generalizes this concept to multimodal settings by using one modality (*e.g.*, text) to modulate another (*e.g.*, images). Given a conditioning input x_1 and target input x_2, FiLM is formulated as

$$y = \gamma(x_1) \otimes x_2 + \beta(x_1), \tag{2.2}$$

where $\gamma(x_1)$ and $\beta(x_1)$ are functions that generate scale and shift parameters conditioned on x_1 and $\otimes$ denotes the Hadamard product. This input-dependent modulation allows the model to adaptively adjust feature representations based on contextual cues from the conditioning input.

A more recent technique, Scale and Shift deep Features (SSF; Lian et al., 2022), refines this idea by focusing solely on scaling and shifting the intermediate features extracted by specific layers of a PTM. SSF incorporates scale and shift parameters during training to adjust feature distributions, thereby addressing distributional shifts between upstream and downstream tasks. Unlike normalization strategies or FiLM, SSF operates independently of individual input features, as its scale and shift parameters model the distribution of the entire downstream dataset. This characteristic allows the learned parameters to be reparameterized into the original model weights, preserving a unified parameter space and obviating the need for additional input-conditioned computations during inference. The resulting minimal computational overhead makes SSF especially advantageous in resource-constrained environments.

2.2 Multimodal Representation Learning

In Section 2.1, we reviewed several transformer architectures and PEFT strategies that have enabled large-scale PTMs to perform exceptionally well on *unimodal* tasks, such as text or image classification. Nonetheless, many real-world applications require a more comprehensive approach, wherein artificial systems must concurrently process and interpret information from multiple sensory modalities. For example, a robot navigating in an environment must be capable of processing verbal instructions while simultaneously analyzing visual cues. Models designed solely for unimodal processing are inherently limited as they lack the mechanisms needed to effectively fuse and interpret heterogeneous data sources. This shortcoming underscores the need for *multimodal perception* systems that can seamlessly integrate diverse sensory inputs. Consequently, advancing foundation model research towards handling multimodal data is imperative for addressing the complex challenges posed by real-world tasks.

Multimodal, or *crossmodal*, representation learning seeks to integrate and process heterogeneous data sources—such as text, images, and audio—into unified, coherent representations. By fusing diverse modalities, models can capture richer semantic contexts and improve their ability to interpret and interact with complex data. In the remainder of this thesis, we will focus on Vision-Language Models (VLMs), a specialized subset of multimodal architectures that jointly learn from visual and linguistic information. VLMs are designed to bridge the semantic gap between vision and language by aligning these modalities within a shared feature space. This alignment has proven crucial for advancing applications such as image captioning, visual question answering, and crossmodal retrieval. The subsequent sections will explore various strategies and paradigms for the large-scale pretraining of VLMs, with a focus on techniques that enhance the effective learning, alignment, and integration of multimodal information.

2.2.1 Bridging the Gap between Text and Images

A key challenge in pretraining VLMs is devising effective encoding strategies for heterogeneous modalities—namely, images and text—given their intrinsic differences. Visual data is characterized by dense, continuous pixel arrays that capture spatial relationships and context, whereas textual data is discrete and sequential, conveying semantic meaning through linguistic structure. Although the text encoders in VLMs are typically transformer-based, the approaches to image encoding diverge considerably.

Early attempts to adapt BERT for multimodal pretraining—exemplified by models such as VilBERT (Lu et al., 2019), LXMERT (Tan and Bansal, 2019), VL-BERT (Su et al., 2020), and UNITER (Chen et al., 2020)—rely on Faster R-CNN (Ren et al., 2015) to extract salient visual features. These methods generate sequences of region-of-interest features using object detection frameworks. In contrast, models such as Pixel-BERT (Huang et al., 2020) and SOHO (Huang et al., 2021) adopt model architectures such as ResNet (He et al., 2016) to process images at the pixel level. This approach enables the capture of finer-grained visual details and mitigates some limitations of Faster R-CNN associated with reliance on predefined region proposals.

More recent strategies, as seen in ALBEF (Li et al., 2021) and SimVLM (Wang et al., 2022c), eschew convolution-based methods entirely in favor of ViT. As discussed in Section 2.1.3, ViTs partition images into fixed-size patches and process them using self-attention mechanisms, thereby establishing a unified, transformer-based framework that naturally aligns with the text encoding pipeline.

Du et al. (2022) categorize all aforementioned VLMs as *fusion encoders*, designed to integrate text embeddings and image features through various fusion strategies that model the interaction between vision and language. By applying self-attention or cross-attention operations, the final-layer hidden states provide a unified representation of the two modalities. These models mainly differ in (1) their choice of text and image encoders, (2) whether they adopt a *single-stream* architecture (concatenating multimodal inputs at the beginning and processing them jointly within the same transformer layers, *e.g.*, UNITER, VL-BERT) or a *dual-stream* architecture (separately encoding multimodal signals and merging them later in the network, *e.g.*, VilBERT, LXMERT), and (3) the specific image-text datasets used for pretraining (Bugliarello et al., 2021).

2.2.2 Representation Alignment

A notable limitation of the fusion encoders presented in Section 2.2.1 is their dependence on deep transformer architectures to model complex VL interactions. In contrast, *dual encoder* architectures employ separate unimodal encoders for images and text, projecting their respective embeddings into a shared semantic space. The similarity between the modalities is then computed using lightweight pooling operations, such as a shallow attention layer (Lee et al., 2018) or a simple dot product (Faghri et al., 2018). This *shallow modality interaction* strategy significantly enhances computational efficiency and scalability by reducing the complexity and overhead associated with deeper fusion networks.

One notable implementation of the dual encoder paradigm is the Contrastive Language-Image Pretraining (CLIP; Radford et al., 2021) model introduced by OpenAI, as outlined in Figure 2.5. CLIP's image encoder is typically either a ResNet or a ViT, while the text encoder is a variant of the vanilla transformer encoder. CLIP leverages a large-scale dataset of web-crawled image-caption pairs to learn a joint embedding space where both modalities are represented. This *representation alignment* is achieved through a contrastive learning objective that maximizes the cosine similarity between embeddings of matching image-text pairs and minimizes it for non-matching pairs. Mapping both images and text into a shared space via CLIP facilitates zero-shot image classification or retrieval through *prompting*. Prompting, which was notably advanced by Brown et al. (2020), in-

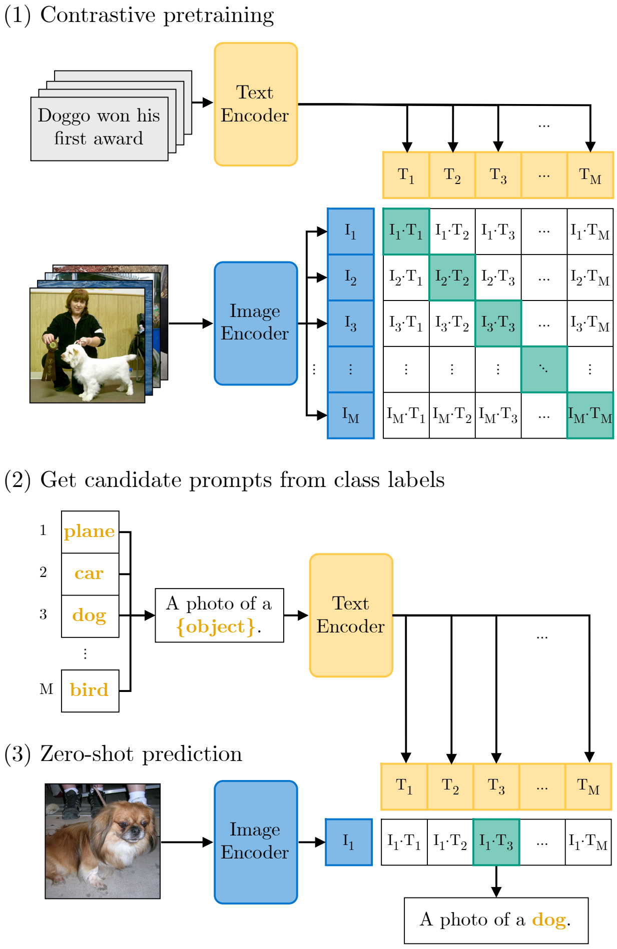

(1) Contrastive pretraining
Doggo won his first award
Text Encoder
T_1 T_2 T_3 ... T_M
Image Encoder
I_1 I_2 I_3 ⋮ I_M
$I_1{\cdot}T_1$ $I_1{\cdot}T_2$ $I_1{\cdot}T_3$ ... $I_1{\cdot}T_M$
$I_2{\cdot}T_1$ $I_2{\cdot}T_2$ $I_2{\cdot}T_3$ ... $I_2{\cdot}T_M$
$I_3{\cdot}T_1$ $I_3{\cdot}T_2$ $I_3{\cdot}T_3$ ... $I_3{\cdot}T_M$
$I_M{\cdot}T_1$ $I_M{\cdot}T_2$ $I_M{\cdot}T_3$ ... $I_M{\cdot}T_M$
(2) Get candidate prompts from class labels
1 plane
2 car
3 dog
M bird
A photo of a {object}.
Text Encoder
(3) Zero-shot prediction
Image Encoder
I_1
T_1 T_2 T_3 ... T_M
$I_1{\cdot}T_1$ $I_1{\cdot}T_2$ $I_1{\cdot}T_3$ ... $I_1{\cdot}T_M$
A photo of a dog.

Figure 2.5: CLIP pretraining and inference pipelines. CLIP jointly trains an image encoder and a text encoder by optimizing them to produce similar embeddings for paired image-text data. During inference, CLIP uses these embeddings to assess the alignment between a given image and multiple text prompts, typically structured as prompt templates with inserted class labels. This alignment is quantified through cosine similarity, allowing CLIP to rank text prompts by their relevance to the image or, conversely, to rank images by their relevance to a specific text prompt. Adapted from Radford et al. (2021).

volves using textual descriptions or queries to guide the model in identifying or retrieving relevant images without prior specific training on those categories.

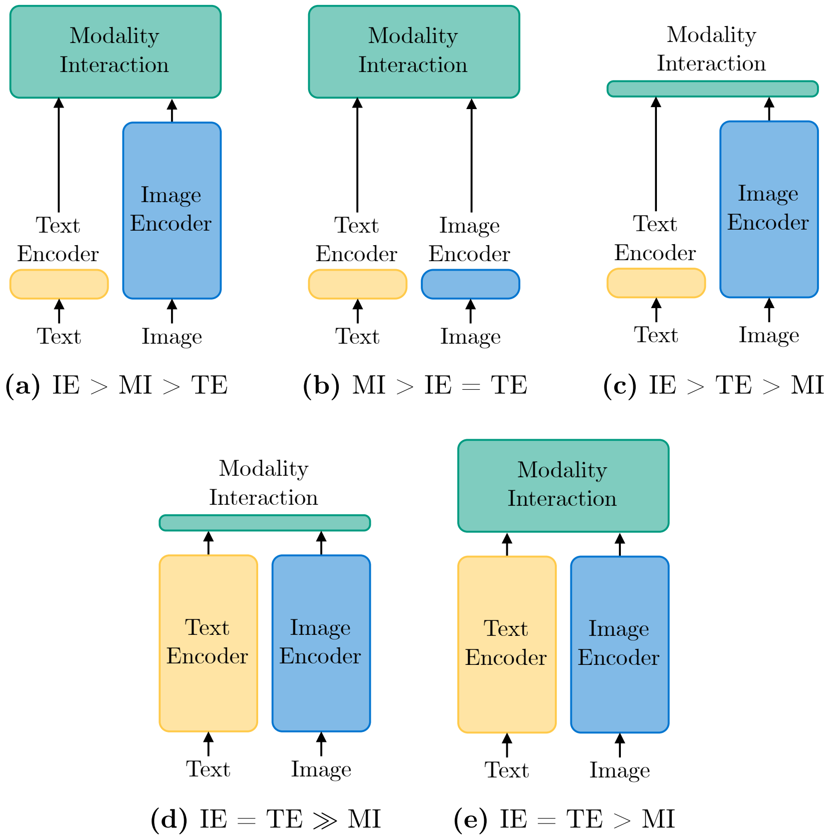
Despite the robust zero-shot capabilities derived from the alignment of image and text representations in CLIP, its shallow modality interaction mechanism constrains its performance on more complex VL grounding tasks beyond image classification and retrieval (Kim et al., 2021). To overcome this limitation while maintaining the advantages of pretrained, aligned image-text representations, several studies have proposed substituting CLIP's shallow interaction mechanism with a trainable, mid-sized modality interaction network (as opposed to the large fusion encoders outlined in Section 2.2.1). This modification extends CLIP's utility to a broader range of downstream VL tasks. For example, CLIPCap (Mokady et al., 2021) introduces a transformer-based projection that maps CLIP embeddings to a pretrained GPT-2 (Radford et al., 2019) decoder for image captioning. Similarly, CLIPSeg (Lüddecke and Ecker, 2022) employs FiLM on CLIP embeddings and trains a fusion decoder for referring expression segmentation. In the realm of robotic object manipulation, CLIPort (Shridhar et al., 2022) learns FiLM parameters to modulate CLIP embeddings within a semantic reasoning framework. Finally, our recently proposed NOVIC model (Allgeuer et al., 2025) trains a decoder-only transformer that generates open-vocabulary image labels as free-form text from CLIP embeddings.

Among the studies that augment aligned multimodal encoders with a medium-sized fusion network, two primary design paradigms for modality fusion emerge: (1) approaches such as CLIPCap and CLIPort that employ linear modulation of visual features via textual cues (*e.g.*, using FiLM), and (2) strategies like CLIPSeg and NOVIC that leverage a single-stream transformer-based fusion network to generate a unified joint output from pretrained text and image features. Due to the fundamental differences between FiLM-based modulation and transformer-based fusion, their performance in CL settings can differ significantly. Consequently, the decision on which modality interaction mechanism to prioritize during CL may depend on the specific performance characteristics and trade-offs of each approach. To examine these differences, we will conduct a comprehensive evaluation and analysis of both fusion network architectures in the CL experiments detailed in Chapter 6.

2.2.3 Taxonomy of Vision-Language Models

Having examined a variety of VLM architectures in earlier sections, we now establish a systematic framework that categorizes these models according to their architectural designs and fusion strategies, as illustrated in Figure 2.6. Kim et al. (2021) introduce a taxonomy of VLMs based on two principal dimensions: (1) the distribution of expressiveness between the modalities, determined by the allocation of dedicated parameters and computational resources to each modality; and (2) the depth of the modality interaction network. Their taxonomy delineates four distinct VLM types, as depicted in Figures 2.6a to 2.6d. In this thesis, we extend the taxonomy by introducing a fifth category, which is visualized in Figure 2.6e. This novel category encapsulates recent advancements in dual encoder models, particularly the developments that have emerged following CLIP (*cf.* Section 2.2.2). Specifically, models in this category train a mid-sized fusion network atop a fixed dual encoder architecture to effectively address complex downstream tasks that demand enhanced multimodal reasoning.

Models in the first category, shown in Figure 2.6a, represent the majority of fusion encoders discussed in Section 2.2.1, such as VilBERT, LXMERT, *etc.* Fusion encoder VLMs train a deep modality interaction network, which is typically a



(a) IE > MI > TE **(b)** MI > IE = TE **(c)** IE > TE > MI

(d) IE = TE ≫ MI **(e)** IE = TE > MI

Figure 2.6: Comparison of fusion strategies in VLMs. Categories **(a)** and **(b)** refer to *fusion encoders*, which use deep fusion networks to capture complex multimodal interactions. VLMs from categories **(c)** and **(d)** use *dual encoders* to align representations of each modality via shallow interactions. Models in category **(e)** extend dual encoders by incorporating a deep fusion network to enhance modality interaction. The higher the box, the deeper the network it represents. **IE** = Image Encoder, **TE** = Text Encoder, **MI** = Modality Interaction. Adapted from Kim et al. (2021).

transformer encoder, to integrate visual and linguistic information. Visual features are generally extracted at the pixel level via ResNet, as region-of-interest features via Faster R-CNN, or as sequences of patch embeddings via ViT.

In contrast to first-category models that rely on deep, computationally intensive visual embedding layers, models in the second category, as illustrated in Figure 2.6b, employ shallow embedding layers for both raw pixel inputs and text tokens. These models prioritize early-stage modality fusion and allocate the majority of computational resources to modeling crossmodal interactions. A notable example of this approach is ViLT (Kim et al., 2021).

The third VLM category, illustrated in Figure 2.6c, comprises *visual semantic embedding models*, including VSE++ (Lee et al., 2018) and SCAN (Faghri et al., 2018). These models represent some of the earliest dual encoder architectures designed to map text and images into a shared embedding space. They utilize separate embedding networks for textual and visual inputs, where the visual encoder is typically a computationally intensive CNN, and the text encoder is generally a more lightweight RNN. Multimodal interaction in these models is facilitated through similarity matching between text and image embeddings, achieved by parameter-free (*e.g.*, dot product) or lightweight parameterized methods (*e.g.*, shallow attention).

The fourth category, depicted in Figure 2.6d, comprises contemporary dual encoder models that employ separate, yet equally sized, transformer encoders for each modality. Notable examples in this category include CLIP and ALIGN (Jia et al., 2021). Much like the visual semantic embedding models in the third category, these models implement shallow modality interaction—typically via a dot product—to align multimodal representations. Although they demonstrate strong performance in zero-shot image classification and retrieval through prompting, their ability to tackle more complex VL reasoning tasks remains limited (*cf*. Section 2.2.2).

Finally, the fifth and last category, as depicted in Figure 2.6e, includes a diverse array of recent VLMs that extend the capabilities of dual encoders to VL tasks beyond classification and retrieval. These models typically consist of two distinct, approximately equal-sized transformer encoders—one for producing textual embeddings and the other for visual embeddings—along with an additional multi-layer modality interaction network. Approaches that build upon the most well-known dual encoder, CLIP, such as CLIPCap and CLIPSeg, fall within this category.

Given the five different types of VLMs, selecting the appropriate VLM for our CL experiments in Part III is essential. Since unimodal representations from transformer-based PTMs are inherently generic and powerful, we will keep the modality-specific encoders (*i.e.*, IE and TE in Figure 2.6) frozen. Our primary focus will therefore be on learning the modality fusion parameters to flexibly adapt to downstream tasks.

In the context of CL, the selected VLM must produce joint representations that are both sufficiently expressive to accommodate new tasks and robust enough to retain knowledge gained from previous ones. Fusion encoder architectures from the first two categories typically employ shallow text encoders that are fine-tuned alongside the modality interaction network. This constraint renders them unsuitable for our experiments, as they tend to lack robustness and conflict with our objective of maintaining fixed, pretrained visual and textual encoders to fully leverage their potent feature extraction capabilities. Similarly, under our constraint of frozen modality-specific encoders, the shallow modality interaction mechanisms in dual encoder architectures from the third and fourth categories cannot effectively learn task-specific representations, limiting their adaptability to new tasks.

VLMs in the fifth category integrate state-of-the-art, pretrained text and image encoders with a moderately deep modality interaction network. This design is the most effective for the multimodal CL scenarios we will investigate in this thesis, given the aforementioned assumptions and desired VLM properties. However, models in this category are typically resource intensive—in terms of both runtime and memory usage—since all three components (the image encoder, text encoder, and modality interaction network) are deep networks. As we intend to keep the parameters of the pretrained image and text encoders fixed, it is crucial to design a modality interaction network that efficiently leverages these high-quality, modality-specific representations while remaining as shallow or lightweight as possible. This design constraint is essential for achieving an optimal balance between the advantageous properties for CL and the associated computational costs.

2.3 Chapter Summary

In this chapter, we examined the evolution of representation learning within the realm of foundation models, beginning with an in-depth exploration of the transformer encoder–decoder architecture. This marked a paradigm shift from training models from scratch to leveraging self-supervised pretraining on large-scale datasets, subsequently refined through supervised fine-tuning on task-specific data. Transformer-based foundation models have streamlined architectural design by obviating the need for computationally intensive mechanisms like recurrence and convolution, while also demonstrating remarkable generalization across diverse modalities, tasks, and domains. We underscored the transformative role of the transformer in advancing deep learning—most notably in natural language processing and computer vision—with landmark models such as BERT and ViT epitomizing this evolution.

Moreover, we reviewed various parameter-efficient fine-tuning strategies that adapt foundation models to downstream tasks while substantially reducing computational overhead. These techniques are particularly valuable in low-data or resource-constrained settings, making them indispensable in efforts to deploy foundation models in complex and dynamic learning environments.

Motivated by the multisensory nature of human perception in real-world settings, we also highlighted significant advancements in multimodal learning driven by foundation models. We discussed key design principles for the integration and fusion of heterogeneous modalities. Similar to their successful application in unimodal applications, large-scale foundation models can capture rich, comprehensive representations across multiple modalities. This capability enables effective adaptation to a broad spectrum of downstream multimodal tasks, often requiring minimal fine-tuning data. Building on these insights, we proposed a taxonomy for vision–language models and provided a rationale for emphasizing deep modality interaction networks constructed atop pretrained representation alignment models such as CLIP.

Despite the broad applicability enabled by pretraining on vast, web-derived datasets followed by fine-tuning on smaller, annotated datasets, foundation models inherently assume that input samples are independently drawn from a fixed, stationary distribution. Although self-supervised upstream pretraining confers enhanced robustness and generalizability, this thesis will demonstrate that such models still face significant limitations in dynamic, real-world environments where data distributions evolve over time. Thus, maintaining high performance under distributional shifts remains an open challenge, yet important prerequisite for pretrained models for successful real-world deployment. In the subsequent chapter, we will develop the theoretical framework for continual learning, discuss its importance, and review different methodologies for addressing this challenge.

Chapter 3

Continual Learning

Continual learning addresses the challenge of learning from a continuous, potentially infinite stream of information, which may involve multiple tasks and be subject to distributional shifts (Chen and Liu, 2018). The primary objective of CL is to leverage knowledge acquired over time to improve problem-solving capabilities and facilitate future learning, while mitigating *catastrophic forgetting* to retain previously learned knowledge. In the literature, CL is often discussed using terms like *lifelong learning*, *sequential learning*, and *incremental learning*, each with nuanced differences in focus. While *lifelong learning* emphasizes open-ended knowledge accumulation and refinement, *sequential learning* addresses the challenge of ordered task learning under shifting distributions, and *incremental learning* focuses on updating models without retraining from scratch. Despite these nuances, the terms describe largely overlapping concepts and will be treated synonymously throughout this thesis.

This chapter highlights the importance of CL for both biological and artificial systems, while identifying the challenges and limitations inherent in strategies designed to address this problem. We begin by discussing the key biological mechanisms underpinning CL in Section 3.1. Following this, we provide a concise review of the most relevant desiderata, evaluation metrics, learning scenarios, and paradigms for CL in artificial neural networks in Section 3.2. To bridge the gap with the previous chapter and the core focus of this thesis, we explore CL in the context of foundation models in Section 3.3. Finally, we summarize and conclude this chapter in Section 3.4, where we also highlight the remaining challenges and unresolved open questions.

3.1 Biological Aspects of Continual Learning

Learning is defined as the process by which experiences lead to behavioral adaptations (Kandel and Hawkins, 1992). Throughout their lifespans, humans and other animals demonstrate an exceptional ability to acquire, integrate, refine, and transfer knowledge in response to novel stimuli. This section presents a succinct examination of the biological and neurocognitive mechanisms in the mammalian brain that facilitate continuous learning and adaptation, which have served as inspiration for various computational methods and algorithms for CL, including several discussed in this thesis.

3.1.1 Stability-Plasticity Dilemma

The *stability-plasticity dilemma* (Grossberg, 1982) represents a fundamental challenge in neuroscience, particularly in understanding how the brain balances retaining learned knowledge (*stability*) with integrating novel information from new experiences or learning events (*plasticity*). Stability, in biological terms, refers to the preservation and maintenance of synaptic strengths and neural circuits that underlie the encoding and storage of long-term memories, while plasticity involves dynamic remodeling of synaptic connections and neural pathways, encompassing both structural and functional changes in response to external stimuli. A regulated balance between these mechanisms is essential for long-term memory retention in neural circuits (Abraham and Robins, 2005; Takeuchi et al., 2014).

This trade-off between stability and plasticity is deeply rooted in the neural mechanisms that underpin learning and memory. For example, *Long-Term Potentiation* (LTP) and *Long-Term Depression* (LTD) are essential mechanisms of synaptic plasticity that contribute to experience-dependent changes in brain function by modifying synaptic strength (Malenka and Bear, 2004). LTP, first demonstrated in rabbits by Bliss and Lømo (1973) and later observed across various brain regions and species, is defined as a long-lasting increase in synaptic strength following repeated high-frequency stimulation. A key mechanism of LTP involves increasing the sensitivity of the postsynaptic neuron to neurotransmitters, primarily through the insertion of additional AMPA receptors into the postsynaptic membrane (Abraham, 2003; Shepherd and Huganir, 2007). Conversely, LTD is characterized by a long-term reduction in synaptic efficacy, typically after low-frequency stimulation. This reduction is crucial for processes such as synaptic pruning, in which unnecessary or redundant synaptic connections are weakened or eliminated to refine neural circuits (Bear and Abraham, 1996). Furthermore, both LTP and LTD are intricately linked to the concept of meta-plasticity, a higher-order regulatory mechanism in which prior synaptic activity not only adjusts synaptic strength but also modulates the threshold for future potentiation or depression (Abraham and Bear, 1996; Abraham, 2008).

Striking a balance between stability and plasticity is essential for both immediate synaptic changes and the broader processes involved in *memory consolidation* and *reconsolidation*. Memory consolidation refers to the process by which newly acquired information becomes stable and integrated into long-term memory over time. This process involves not only strengthening specific synapses, *i.e.*, *synaptic consolidation*, but also reorganization of neural networks to support long-term storage of memories, *i.e.*, *systems consolidation* (McGaugh, 2000). However, the acquisition of new information can potentially disrupt existing memories, a phenomenon known as *retroactive interference* (Wixted, 2004). Reconsolidation theory posits that when a consolidated memory is retrieved, it temporarily becomes labile and subject to modification or updating before being re-stabilized (Dudai, 2004).

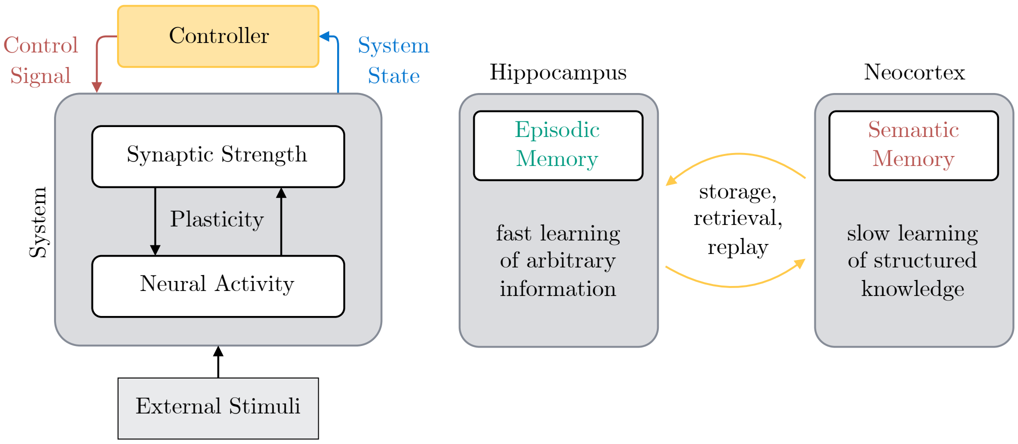
As the brain ages, the dynamics of stability and plasticity shift significantly, with critical implications for learning and memory throughout the lifespan (Peters, 2006). During early development, the brain exhibits increased plasticity, particularly during critical periods, when certain experiences can profoundly shape neural circuits and behavioral outcomes (Knudsen, 2004). This plasticity is crucial for acquiring complex skills such as language and sensory processing, as highlighted by studies on early developmental stages (Hensch, 2005; Werker and Hensch, 2015). However, as individuals age, the brain progressively prioritizes stability, focusing on the retention of existing knowledge over the incorporation of new information. This transition is associated with a reduction in synaptic plasticity and an increase in mechanisms that promote stability, including enhanced inhibitory processes and the strengthening of established synaptic connections (Burke and Barnes, 2006; Morrison and Baxter, 2012).

Hedden and Gabrieli (2004) highlight that these neural changes result in contrasting patterns of cognitive function, where abilities such as encoding new memories, working memory, and processing speed decline, while other cognitive functions, including autobiographical memory and semantic knowledge, remain relatively stable. Although increasing stability supports the preservation of long-term memories and acquired skills, it comes at the cost of reduced behavioral flexibility and adaptability (Lövdén et al., 2010). This shift is further associated with a reduction in cognitive reserve, diminishing the brain's ability to buffer against age-related cognitive decline (Stern, 2009).

3.1.2 Hebbian and Homeostatic Plasticity

One of the first and most influential theories on stimulus-driven neural responses was introduced by Hebb (1949). This theory posits that when a presynaptic neuron, *i.e.*, the signal-sending neuron, consistently triggers the activation of a postsynaptic neuron, *i.e.*, the signal-receiving neuron, the synaptic connection between them is strengthened. In essence, frequent activation of the postsynaptic neuron by the presynaptic neuron reinforces their connection, thus enhancing the efficiency of future communication. This principle, often encapsulated by the phrase "neurons that fire together wire together," laid the foundation for understanding how repeated experiences can induce lasting changes in neural circuitry. A simplified mathematical formulation of *Hebbian plasticity*, developed after Hebb's original work, updates the synaptic weight w_{ij} between neuron i and neuron j according to the rule:

$$\Delta w_{ij} = \eta \cdot x_i \cdot y_j, \tag{3.1}$$



(a) Hebbian and homeostatic plasticity

(b) CLS theory

Figure 3.1: Models of learning and memory in the brain. (a) Hebbian plasticity strengthens neural connections in response to external stimuli, while homeostatic plasticity maintains neural stability amidst these changes by computing and returning a feedback control signal based on the observed system state. **(b)** The CLS theory suggests that the hippocampus rapidly encodes new memories for short-term storage, while the neocortex gradually consolidates these memories over time for long-term storage and integration with existing knowledge. Adapted from Parisi et al. (2019).

where x_i and y_j represent the activities of the presynaptic and postsynaptic neurons, respectively, and η is a small positive constant representing the learning rate.

Hebbian plasticity is fundamental for learning and memory formation as it enables the encoding of synaptic associations and the reinforcement of frequently co-activated neural circuits (Bi and Poo, 1998; Magee and Johnston, 1997). However, unregulated Hebbian plasticity can lead to *runaway synaptic dynamics*, where the most active synapses disproportionately strengthen, potentially causing network overexcitation and disrupting the balance of neural function (Bienenstock et al., 1982; Miller, 1996). To prevent such instability, Hebbian plasticity is modulated by *homeostatic plasticity*, a set of regulatory mechanisms that stabilize neural activity by globally or locally adjusting synaptic strengths to maintain them within an optimal range (Turrigiano, 1999). This regulation prevents excessive synaptic potentiation and counteracts the harmful effects of excessive synaptic inhibition, which can contribute to memory dysfunction (Vogels et al., 2011). Thus, homeostatic processes maintain the functional integrity of neural circuits, ensuring that Hebbian modifications allow the brain to remain flexible for new learning while safeguarding the stability of existing memories (Turrigiano and Nelson, 2004; Pozo and Goda, 2010). A schematic illustration of Hebbian and homeostatic plasticity is provided in the left part of Figure 3.1.

One prominent type of homeostatic plasticity is *synaptic scaling*, where the strength of all synapses in a neuron is adjusted uniformly through a multiplicative scaling factor m. This scaling factor acts as a modulatory signal to stabilize the neuron's overall firing rate while preserving the relative differences between synaptic strengths, ensuring that the proportional synaptic weight distribution remains intact (Turrigiano, 2008; Davis, 2006). The change in synaptic weight w_{ij} is given by the following equation:

$$\Delta w_{ij} = m_j \cdot \eta \cdot x_i \cdot y_j, \tag{3.2}$$

where m_j is unique to each postsynaptic neuron j. The incorporation of modulatory feedback loops in Hebbian neural networks has recently garnered significant attention, leading to the development of various approaches that utilize biologically plausible learning mechanisms (Grant et al., 2017; Payeur et al., 2021; Tang et al., 2023). In particular, the interaction between Hebbian and homeostatic plasticity stabilizes synaptic changes within individual neurons, but additionally plays a crucial role in the broader organization of neural circuits into functional structures (Turrigiano, 2012). The foundational work of Mountcastle (1957), and later of Hubel and Wiesel (1962), demonstrates that neurons in the cat brain are organized into cortical columns, or *brain maps*, which are selectively responsive to specific sensory stimuli. These neurons are spatially arranged to systematically reflect properties of sensory input, such as the orientation of edges in the visual field.

Self-organizing maps. The concept of brain maps has been applied effectively in artificial neural networks, particularly in Kohonen's Self-Organizing Map (SOM; Kohonen, 1990). The unsupervised SOM algorithm utilizes principles of topological mapping and Hebbian plasticity to develop spatially organized representations of input data through competitive Hebbian learning (Martinetz, 1993). SOMs map high-dimensional input data onto a lower-dimensional (often two-dimensional) grid of neurons, where each neuron corresponds to a region in the input space. During training, input vectors are compared with the neurons' weight vectors, and the neuron with the closest match—based on a predefined distance metric, typically Euclidean distance—is designated as the Best Matching Unit (BMU). Given a set of SOM nodes $\mathcal{N}$ and some input vector $\boldsymbol{x}$, the BMU $\boldsymbol{b}$ is given by

$$\boldsymbol{b} = \underset{\boldsymbol{n} \in \mathcal{N}}{\arg\min}\, ||\boldsymbol{x} - \boldsymbol{n}||, \tag{3.3}$$

where $||\cdot||$ denotes the L^2 norm. The BMU and its neighboring neurons adjust their weight vectors to better approximate the input vector, with the degree of adaptation diminishing as a function of distance from the BMU. Through repeated iterations, neurons self-organize, resulting in similar input patterns being mapped to neighboring neurons, thereby preserving the topological relationships of the input space.[1]

[1] For an in-depth examination of the theoretical foundations and applications of SOMs, refer to Kohonen (2013).

Self-organizing incremental neural networks. The Self-Organizing Incremental Neural Network (SOINN; Shen and Hasegawa, 2010) improves the SOM algorithm in several key aspects. First, unlike SOM, which requires the entire dataset to be available before training, SOINN supports online continual learning, allowing the network to dynamically update its structure with each new input vector. Second, SOINN incorporates a noise-resistant pruning mechanism that removes redundant nodes to produce a more compact and storage-efficient representation of the input space. Third, SOINN dynamically adjusts its network structure by modulating the number of nodes and connections based on the statistical properties and distribution of the input data. Finally, SOINN accommodates both unsupervised and semi-supervised learning through an automatic node-labeling process, making it especially useful for classification tasks. SOINN+ (Wiwatcharakoses and Berrar, 2020) extends the original SOINN algorithm towards a novel node deletion mechanism based on idle time, trustworthiness, and inactivity of network units. This advancement enables SOINN+ to demonstrate increased resilience to noisy data and to learn high-quality topologies from the input domain, while maintaining a smaller number of nodes compared with the original SOINN algorithm.

In this thesis, we refrain from an exhaustive examination of the mechanisms for node creation, adjustment, and deletion inherent to the dynamic self-organization process, as these are comprehensively detailed in the original SOM, SOINN, and SOINN+ publications. Instead, our focus is on leveraging the self-supervised node labeling strategy of the SOINN+ algorithm to augment input features extracted by foundation models during CL, which will be further detailed in Chapter 4.

3.1.3 Complementary Learning Systems

The *Complementary Learning Systems* (CLS) theory, initially proposed by McClelland et al. (1995), provides a comprehensive framework for understanding how the brain resolves the stability-plasticity dilemma by delineating the roles of two interacting memory systems: the hippocampus and the neocortex. According to this theory, these systems have complementary functions in learning and memory, with the hippocampus specializing in the rapid acquisition of new episodic or factual information with a high learning rate, and the neocortex gradually integrating this information into generalized long-term memory with a slower learning rate. Hebbian learning mechanisms provide the foundation for the consolidation mechanisms in both the hippocampus and the neocortex to strengthen representations activated within each system and to facilitate coordination between fast and slow learning processes (O'Reilly and Rudy, 2000). A brief overview of the CLS theory is shown on the right side of Figure 3.1.

The hippocampus plays a critical role in the rapid acquisition of new experiences, particularly in the temporary storage of episodic and contextual information. It underlies synaptic consolidation and rapid plasticity mechanisms such as LTP to support rapid memory encoding (Bliss and Lømo, 1973; Bliss and Collingridge, 1993). This plasticity is essential for forming detailed and contextually rich representations of recent experiences, significantly contributing to episodic memory and early stages of learning (Eichenbaum, 2004). However, this same plasticity renders the hippocampus a transient storage system, susceptible to interference from subsequent learning, and thus unsuitable for stable, long-term retention of knowledge (Frankland and Bontempi, 2005).

Over time, the hippocampus undergoes reactivation to replay its encoded memories, promoting their gradual transfer to the neocortex, particularly during periods of rest and sleep (Wilson and McNaughton, 1994; Rasch and Born, 2013). The neocortex plays a pivotal role in systems consolidation, a process in which these memories are gradually reorganized and integrated into existing cortical networks to support the abstraction of knowledge across multiple experiences (Takashima et al., 2009). Due to its lower synaptic plasticity and higher structural stability compared with other brain regions, such as the hippocampus, the neocortex is particularly well suited for preserving generalized knowledge over extended time periods (Frankland and Bontempi, 2005; Kandel et al., 2013).

CLS theory offers a framework for tackling CL challenges in artificial neural networks, particularly in mitigating catastrophic forgetting. This theory has inspired dual-memory models that mirror the brain's division of labor between the hippocampus and the neocortex. Specifically, these models feature a fast-learning module—akin to the hippocampus—that rapidly encodes new experiences, alongside a slow-learning module—analogous to the neocortex—that gradually integrates and consolidates these experiences into a stable, long-term knowledge base (Kumaran et al., 2016). In Chapter 4, we introduce a novel CL methodology that leverages this dual-memory architecture derived from CLS theory.

3.2 Continual Learning with Artificial Neural Networks

Building on our exploration of the biological foundations of CL, we now shift focus to continual machine learning, where insights from biological systems can inform or enhance the design of AI systems. The mechanisms by which the mammalian brain acquires and integrates knowledge across the lifespan serve as a guiding framework for developing machine learning models capable of learning and evolving over time without forgetting past knowledge. In this section, we formally define the problem of continual machine learning and examine its key objectives, methodologies, recent advancements, and persistent challenges that shape this dynamic and rapidly evolving field.

3.2.1 Problem Formulation

In a traditional supervised machine learning setting, a model observes independent and identically distributed (i.i.d.) data from a training set $\mathcal{D}$ of size $N = |\mathcal{D}|$, where each sample is drawn from a fixed but unknown joint probability distribution

$p(\mathcal{X}, \mathcal{Y})$ across the sample space $\mathcal{X}$ and label space $\mathcal{Y}$. The number of distinct labels, corresponding to the number of classes in a classification problem, is denoted by C. The learning task involves finding a function $f_\theta : \mathcal{X} \to \mathcal{Y}$, parameterized by θ, that best approximates the underlying relationship between inputs and outputs. This is achieved by minimizing an empirical risk function, which serves as a proxy for the expected loss over the data distribution. The empirical risk is defined as

$$\sum_{n=1}^{N} \mathcal{L}\left(f_\theta(\boldsymbol{x}_n), y_n\right), \tag{3.4}$$

where $\mathcal{L}(\cdot,\cdot)$ is a predefined loss function that quantifies the error between the predicted labels $f_\theta(\boldsymbol{x}_n)$ of the n^{th} input sample $\boldsymbol{x}_n$ and the associated ground-truth label y_n. The goal is to determine the optimal parameters θ^* that minimize the empirical risk:

$$\theta^* = \arg\min_\theta \mathcal{L}(\theta). \tag{3.5}$$

This approach, known as empirical risk minimization, critically relies on the *i.i.d. assumption* and the stationarity of the data distribution. Under these conditions, minimizing the empirical risk is expected to result in a model that generalizes well to new, unseen data drawn from the same distribution. In practice, the i.i.d. assumption requires that the entire training set $\mathcal{D}$ is accessible at once, which is often unrealistic when new data become progressively available during training, *e.g.*, in the form of real-time sensory data, or when collected under varying or evolving environmental conditions. A pragmatic solution to preserve the i.i.d. assumption in this setting would be to repeatedly retrain a model on the entire historical dataset each time new data become available or following each distributional shift. However, frequent retraining is prohibitively expensive due to the growing size of models and the superlinear increase in computational resources required to train them (Kaplan et al., 2020). Moreover, this approach may be infeasible when historical data are inaccessible due to privacy, storage, or regulatory constraints.

Continual learning is a subfield of machine learning that aims to address the challenge of learning from non-stationary data distributions. Approaches to CL seek to enhance the robustness[2] and applicability of models in dynamic, real-world environments. Practically, instead of being trained on a single set of i.i.d. data $\mathcal{D}$, a model in a CL scenario is trained on an ordered sequence of tasks with indices $t \in \{1, \ldots, T\}$, where a *task* is defined by its set of N_t annotated training samples $\mathcal{D}_t = \{(\boldsymbol{x}_{t,n}, y_{t,n})\}_{n=1}^{N_t}$ following the distribution $p(\mathcal{X}_t, \mathcal{Y}_t)$. Depending on the CL setting, these training samples can arrive one sample at a time, in batches, or all at once. The objective function in CL is similar to that of multi-task learning (*cf.*

[2] We broadly define *robustness* as a model's ability to consistently achieve high performance and generalization across different tasks, irrespective of variations in how training data are presented to the model.

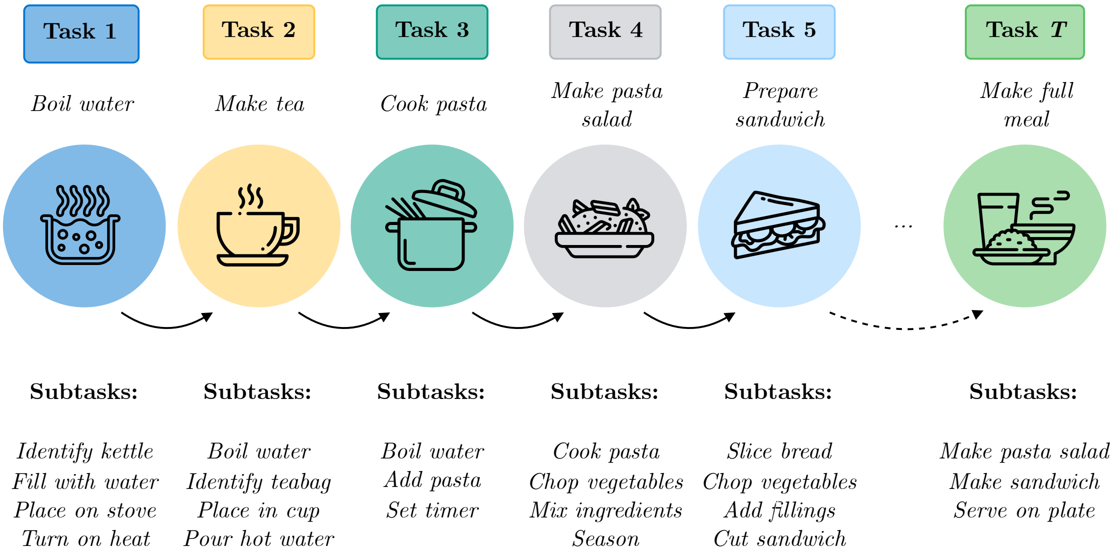


Figure 3.2: Overview of a continual learning scenario. In a CL setting, a model is trained on a series of tasks, each potentially comprising explicit or implicit subtasks. Every task is represented by a training dataset or a series of training episodes and is governed by its own probability distribution. Because tasks and subtasks may (re-)appear at any time, previously acquired knowledge can be leveraged to enhance transfer across the learning process.

Chen and Liu, 2018), given by

$$\sum_{t=1}^{T}\sum_{n=1}^{N_t}\mathcal{L}\left(f_\theta(\boldsymbol{x}_{t,n}), y_{t,n}\right), \tag{3.6}$$

with the key distinction that CL focuses on developing models capable of effectively learning from a *sequence* of tasks $(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T)$ rather than being trained on all tasks simultaneously. A visual representation of an exemplary CL setting is shown in Figure 3.2.

The primary challenge in CL is mitigating catastrophic forgetting, a phenomenon in which a neural network's performance on previously learned tasks declines as new tasks are introduced. This degradation arises from shared parameters between tasks, which lead to interference between old and new knowledge. CL also aims to facilitate *forward transfer*, where knowledge acquired from previous tasks enhances performance on subsequent tasks, and *backward transfer*, in which learning new tasks improves performance on previously learned tasks. For instance, in the kitchen scenario depicted in Figure 3.2, once the system masters boiling water for tea, it can reuse core principles (*e.g.*, managing water temperature) when cooking pasta, illustrating forward transfer. In contrast, improvements in temperature control or stovetop usage gained while preparing pasta can refine the original boiling process, exemplifying backward transfer.

3.2.2 Key Features and Desiderata

The criteria for evaluating the extent to which a model is capable of CL must be defined both in terms of the model's capabilities and the scenario in which it operates. A sufficiently challenging CL scenario is crucial for demonstrating that a model performing well within it truly exhibits CL capabilities. Drawing from the insights of Kudithipudi et al. (2022), Hadsell et al. (2020), and Farquhar and Gal (2019), we define the following key features and desiderata that emerge in this context:

1. **Learning efficiency**: The learning efficiency of a model is largely determined by its capacity to transfer and adapt knowledge across tasks. When data availability is inconsistent, such as in scenarios where tasks exhibit significant data imbalance, models must often rapidly adapt with minimal information, using advanced techniques like few-shot learning. Furthermore, models may operate in single-epoch or online learning settings, allowing only a single pass over the training data. An effective strategy for improving learning efficiency involves identifying and reusing shared knowledge across tasks, which aligns closely with the objectives of systematic or compositional generalization approaches.

2. **Overcoming forgetting**: Models must retain previously acquired knowledge during incremental learning, particularly when access to historical data is restricted. When a model, already trained on certain tasks, is exposed to a new task, it adjusts (or overwrites) its parameters based on the probability distribution of the new task. Without an explicit prevention mechanism, this adjustment compromises the model's performance on previously learned tasks, as prior knowledge is inadequately preserved.

3. **Task-agnostic learning**: A model should operate without explicit definitions or boundaries between tasks, particularly during testing. The ideal model should autonomously identify and categorize the task it needs to address. This requires the model to possess advanced inference capabilities, enabling it to interpret and respond to ambiguous or incomplete task signals.

4. **Open-ended learning**: Real-world applications often involve learning a potentially unlimited number of diverse tasks. Consequently, a model must be able to acquire a new task at any time, regardless of how many tasks it has previously learned.

5. **Resource efficiency**: A model that requires storing all historical data for repetitive retraining during incremental learning imposes increasingly unsustainable memory and computational demands, rendering it impractical for real-world applications. Similarly, introducing a new model for each additional task is both cost-prohibitive and detrimental to cross-task knowledge transfer. For effective deployment in realistic, resource-limited environments,

the model must operate within strict resource constraints—specifically regarding memory, time, and computation—without sacrificing overall performance.

6. **Exemplar-free retention**: A model must be capable of knowledge retention and transfer without revisiting previously seen raw data, or *exemplars*. This requirement is critical in scenarios governed by privacy constraints or legal regulations that prohibit the storage or reuse of historical data, as well as in cases of real-time processing with high data throughput, where storing past data is either impractical or impossible due to time or storage constraints. Potential strategies for exemplar-free retention include (i) storing compact latent representations for replay, (ii) employing generative models, (iii) using knowledge distillation, or (iv) prototyping methods. Beyond feature and generative replay approaches using strategies (i) or (ii), which are commonly referred to as *exemplar-free*, knowledge distillation and prototype-based methods using strategies (iii) or (iv) are additionally considered *rehearsal-free*.

While these characteristics guide the evaluation of CL methods, they often involve trade-offs. For instance, achieving high learning efficiency may require storing large amounts of historical data, which can conflict with goals like resource efficiency and data-free retention. Therefore, a practical strategy is to prioritize the characteristics most relevant to the specific application. In time-sensitive domains, such as autonomous driving systems, computational and runtime efficiency are paramount, often taking precedence over memory constraints. Conversely, in fields like healthcare, efficient data usage becomes critical due to limited access to historical records and incoming data. Nevertheless, this thesis will present CL methods that achieve competitive performance in their respective applications without significant compromises across these criteria.

3.2.3 Evaluation Metrics

In contrast to traditional machine learning research, which primarily reports a success metric evaluated on the entire dataset $\mathcal{D}$ after training, CL research places additional emphasis on model performance over time. Consequently, CL evaluation metrics are typically measured not only at the end of training but also at frequent intervals during the sequential learning phase, *e.g.*, after training on each task $\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_T$.

Metrics for evaluating CL methods have been proposed in various works with minor variations (Lopez-Paz and Ranzato, 2017; Chaudhry et al., 2018; Kemker et al., 2018; Wang et al., 2024). Common to all of them is the measurement of CL performance along three major dimensions: the ability of a model to retain knowledge (*memory stability*), its capacity to learn new tasks (*learning plasticity*), and its capability to balance these two objectives (*overall performance*). In this thesis, we primarily adopt the CL evaluation metrics introduced by Wang et al. (2024). Specifically, let $a_{t,i} \in [0, 1]$ denote some success metric, *e.g.*, classification accuracy, measured on the test set of the i^{th} task after incremental learning of the t^{th} task, where $i \leq t$.

Learning plasticity. To measure the capability of a model to leverage knowledge from prior tasks to better learn the current task, we formally define the *Forward Transfer* (FWT) as

$$\text{FWT}_t = \frac{1}{t-1} \sum_{i=2}^{t} a_{i-1,i} - \tilde{a}_i, \tag{3.7}$$

where $\tilde{a}_i$ denotes the performance on the i^{th} task at (random) initialization of the reference model.

Memory stability. The ability of a model to retain knowledge from previous tasks in an incremental learning setting can be quantified using two different metrics: *Forgetting Measure* (FM) and *Backward Transfer* (BWT). Both metrics are inversely related: positive forgetting reflects a *decline* in performance on a task i after learning a subsequent task $t > i$, while positive backward transfer denotes an *improvement* in performance on task i following the acquisition of the t^{th} task. Formally, the forgetting measure is expressed as

$$\text{FM}_t = \frac{1}{t-1} \sum_{i=1}^{t-1} \max_{j \in \{1,\ldots,t-1\}} a_{j,i} - a_{t,i}, \tag{3.8}$$

while backward transfer is computed as

$$\text{BWT}_t = \frac{1}{t-1} \sum_{i=1}^{t-1} a_{t,i} - a_{i,i}. \tag{3.9}$$

Although few theoretical studies examine backward transfer (*e.g.*, Lin et al., 2022a; Benavides-Prado and Riddle, 2022), they typically restrict their analysis to a single target task and assume a high degree of similarity between tasks. However, in practice, such conditions are rarely met. Given the more complex scenarios explored in this thesis that extend beyond the limited scope of the aforementioned studies, we will show that forgetting still remains a significant challenge. Thus, in the remainder of this thesis, we will focus on FM as a representative measure of memory stability.

Overall performance. The most commonly used metric for evaluating and comparing CL methods is *Average Accuracy* (AA), formally defined as

$$\text{AA}_t = \frac{1}{t} \sum_{i=1}^{t} a_{t,i} \tag{3.10}$$

AA_t implicitly captures both learning plasticity (the ability to acquire new knowledge) and memory stability (the ability to retain past knowledge), which makes it

Scenario	Training	Evaluation
TIL	$\{\{\mathcal{D}_t, t\}_{e=1}^{E}\}_{t=1}^{T}$; $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset \; \forall \; i \neq j$	$\{p(\mathcal{X}_t), t\}_{t=1}^{T}$
DIL	$\{\{\mathcal{D}_t, t\}_{e=1}^{E}\}_{t=1}^{T}$; $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i = \mathcal{Y}_j \; \forall \; i \neq j$	$\{p(\mathcal{X}_t)\}_{t=1}^{T}$
CIL	$\{\{\mathcal{D}_t, t\}_{e=1}^{E}\}_{t=1}^{T}$; $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset \; \forall \; i \neq j$	$\{p(\mathcal{X}_t)\}_{t=1}^{T}$
OCL	$\{\{\{\mathcal{D}_{t,b}\}_{b \in \mathcal{B}_t}\}_{e=1}^{E}\}_{t=1}^{T}$, $\lvert b \rvert = 1$, $E = 1$; $p(\mathcal{X}_i) \neq p(\mathcal{X}_j) \; \forall \; i \neq j$	$\{p(\mathcal{X}_t)\}_{t=1}^{T}$

Table 3.1: Formal comparison of continual learning scenarios. $\mathcal{D}_t$: training set for task t. $\mathcal{D}_{t,b}$: training set for task t and batch b. $\lvert b \rvert$: batch size of batch b. $\mathcal{B}_t$: space of incremental batches for task t. E: number of training epochs for task t. $\mathcal{X}_t$: input data space for task t. $p(\mathcal{X}_t)$: probability distribution over $\mathcal{X}_t$. $\mathcal{Y}_t$: output label space for task t. Note that $E = 1$ corresponds to single-epoch training in any CL scenario. Adapted from Wang et al. (2024).

an effective measure of a model's ability to balance the stability-plasticity trade-off, as discussed in Section 3.1.1. Given its comprehensive nature, AA will be used as the primary evaluation metric for baseline comparisons throughout this thesis. However, we recognize that the importance of retaining old knowledge versus acquiring new knowledge may vary depending on the specific application scenario.

3.2.4 Learning Scenarios

CL encompasses a range of scenarios in which models must adapt to new information while maintaining knowledge from previous experiences. These scenarios differ based on various factors, such as the structure and distinctiveness of tasks, as well as the availability of task-specific information during training and evaluation. The scenarios most commonly studied in CL are Task-Incremental Learning (TIL), Domain-Incremental Learning (DIL), Class-Incremental Learning (CIL), and Online Continual Learning (OCL). Each of these approaches presents distinct characteristics and challenges, as outlined in Table 3.1. In line with previous works (Van De Ven et al., 2022; Delange et al., 2021), we define each CL scenario in the context of classification problems. Nevertheless, these definitions can be extended to other types of learning tasks by substituting $\mathcal{X}$ and $\mathcal{Y}$ in Table 3.1 with the corresponding types of inputs and outputs.

Task-incremental learning. TIL involves training a model to learn a series of distinct tasks sequentially, where both the input data distribution and the output label sets may vary between tasks. During evaluation, the model is provided with a task identifier that indicates the relevant set of output classes for the current task. This means the model only needs to discriminate among classes within the specified task, rather than across all tasks. Consequently, TIL is considered less challenging compared with other CL scenarios, as it reduces complexity by limiting class differentiation to a single task at a time.

Domain-incremental learning. DIL requires solving the same classification problem across multiple domains or contexts that are introduced sequentially. Unlike TIL, the input data distribution changes between domains, while the output label set remains fixed. The primary challenge in DIL is to enable the model to generalize across new domains by adapting to shifts in input distributions without sacrificing performance on the shared output classes. Crucially, the model is not provided with an explicit domain identifier during evaluation, forcing it to autonomously detect and adapt to domain shifts. This lack of direct guidance makes DIL more difficult than TIL, as the model must maintain consistent performance despite variations in input conditions.

Class-incremental learning. CIL requires a model to learn new classes sequentially, with both new input data and new sets of output labels introduced with each incoming task. Unlike TIL, the model is not provided with a task identifier during evaluation, which requires it to discriminate between all classes learned so far, rather than just those within a specific task. CIL is considered more challenging than TIL due to the model's need to integrate and retain knowledge across an ever-growing set of classes without explicit task boundaries or task-specific information.

Online continual learning. OCL presents unique challenges that extend beyond those in traditional CL settings. It requires models to learn from a single-pass data stream, where the underlying data distribution gradually shifts or evolves over time. Unlike in TIL, where task boundaries are clearly defined, or in DIL and CIL, where data are introduced in discrete phases, OCL necessitates continuous updates as each data point is received. Furthermore, unlike all other scenarios, OCL is *task-free*, meaning task identities are entirely unknown or undefined at any point in time. This real-time learning process demands a model to rapidly acquire new knowledge from limited examples while simultaneously retaining and consolidating previously acquired information, all without explicit task demarcation during training or inference. As a result, OCL is often regarded as the most challenging CL scenario.

3.2.5 Strategies and Approaches

This section explores three primary strategies to mitigate catastrophic forgetting in CL: regularization, replay, and dynamic architectures. Each strategy is inspired by the biological mechanisms of learning and memory discussed in Section 3.1.

Regularization-based methods apply constraints to either model parameters or behavior. Parameter regularization mimics synaptic consolidation in the brain, stabilizing memories by selectively reinforcing connections critical to previously learned information. Behavioral (or functional) regularization extends this idea by ensuring that the model's responses to prior tasks remain stable, analogous to the brain's preservation of neural activity patterns that are crucial to maintaining functional integrity across time and tasks. *Replay-based* methods periodically reintroduce past experiences to the model, similar to hippocampal replay in the brain, where neurons reactivating neural patterns associated with prior experiences during rest and sleep to facilitate memory consolidation and retention. Lastly, *architecture-based* methods divide the model into task-specific subnetworks, mirroring the modular organization of the brain, where distinct regions are specialized for different types of information processing. This approach effectively reduces interference between new and old memories by compartmentalizing knowledge.

Several hybrid CL approaches integrate elements from the core strategies of regularization, replay, and dynamic architectures to take advantage of their complementary strengths while mitigating individual limitations. For example, certain methods combine replay and regularization by selectively reintroducing past experiences and simultaneously constraining parameter updates to preserve previously acquired knowledge. Although this section primarily outlines methods that apply the core strategies of CL separately, the methods proposed and evaluated in this thesis also incorporate such hybrid approaches. Moreover, the advent of large-scale foundation models has led to the development of novel CL strategies that extend beyond traditional techniques by capitalizing on the enhanced robustness against catastrophic forgetting achieved through upstream self-supervised pretraining. This advancement opens up new opportunities for continual fine-tuning and effective knowledge transfer across downstream tasks, as will be discussed in detail in Section 3.3.

Regularization-based Approach

Regularization-based approaches have emerged as a natural extension of traditional machine learning techniques to address the stability-plasticity dilemma in CL. These methods are grounded in optimization principles and integrate smoothly with standard training pipelines, making them applicable to a wide range of models and CL scenarios. In CL, regularization methods primarily ensure that models retain the ability to solve previously learned tasks by either constraining updates to parameters considered critical for those tasks or by promoting output consistency across tasks, despite changes in model parameters. An illustrated overview of regularization-based approaches is provided in Figure 3.3.

Parameter regularization. Parameter regularization, also commonly referred to as *prior-focused* regularization, aims to limit the plasticity of neural models by selectively restricting parameter updates. Elastic Weight Consolidation (EWC; Kirkpatrick et al., 2017) implements this approach by incorporating a penalty term into the loss function that discourages substantial alterations to parameters deemed essential for previous tasks. The loss function during training of task $t > 1$

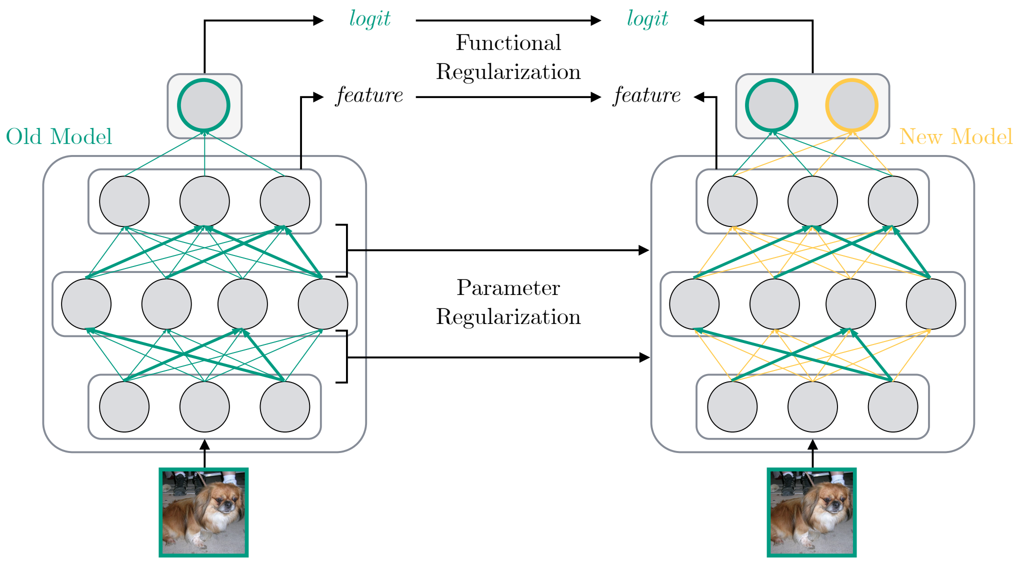


Figure 3.3: Regularization-based methods for continual learning. Parameter regularization penalizes changes to key parameters identified through heuristic importance measures. Functional regularization ensures that the outputs (logits or features) of the new model remain consistent with those of the old model for data from previous tasks. Adapted from Wang et al. (2024).

is expressed as

$$\mathcal{L}_{\text{EWC}}(\theta) = \mathcal{L}_t(\theta) + \lambda \sum_{i=1}^{t-1} \sum_j F_j^{(i)} \left(\theta_j - \hat{\theta}_j^{(i)}\right)^2, \tag{3.11}$$

where $\mathcal{L}_t$ is the loss for the current task t based on a predefined loss function, λ is a regularization coefficient controlling the trade-off between preserving knowledge from previous tasks and learning the new one, $F_j^{(i)}$ is the Fisher information matrix, which reflects the importance of parameter j for task i, and $\hat{\theta}_j^{(i)}$ represents the value of parameter j after training on the i^{th} task. The Fisher information matrix quantifies the sensitivity of the log-likelihood function (or loss function in practical applications) with respect to changes in each parameter and is typically computed using the gradients of the loss function with respect to each parameter, as shown below:

$$F_j^{(i)} = \frac{1}{N_i} \sum_{n=1}^{N_i} \left(\frac{\partial \mathcal{L}(\boldsymbol{x}_{i,n}; \theta)}{\partial \theta_j} \right)^2 \tag{3.12}$$

Here, N_i is the number of samples for the i^{th} task, and $\mathcal{L}(\boldsymbol{x}_{i,n}; \theta)$ represents the loss function for sample $\boldsymbol{x}_{i,n}$ given the model parameters θ. The significance of a parameter in a model's predictions correlates with its Fisher information, with higher Fisher information indicating a greater importance of the parameter. EWC aims to preserve these critical parameters during subsequent learning phases by adjusting their importance at the end of each task. However, computing a separate

Fisher information matrix for each task leads to poor scalability as the number of tasks increases. To address this limitation, two follow-up approaches have been proposed: Online Elastic Weight Consolidation (O-EWC; Schwarz et al., 2018) and Synaptic Intelligence (SI; Zenke et al., 2017).

In O-EWC, the Fisher information is no longer accumulated over all past tasks but is instead updated in an online manner. This is achieved by maintaining a running estimate of the Fisher information matrix, denoted as $\bar{F}$, which is updated after each task. This yields the following modified loss function during the training of task t:

$$\mathcal{L}_{\text{O-EWC}}(\theta) = \mathcal{L}_t(\theta) + \lambda \sum_j \bar{F}_j^{(t-1)} \left(\theta_j - \hat{\theta}_j^{(t-1)}\right)^2, \tag{3.13}$$

where $\bar{F}_j^{(t-1)}$ represents the accumulated Fisher information estimate for parameter j with respect to all prior tasks, and $\hat{\theta}_j^{(t-1)}$ denotes the value of parameter j after training on the previous task $t-1$. $\bar{F}_j^{(t-1)}$ is formally expressed as

$$\bar{F}_j^{(t-1)} = \gamma \bar{F}_j^{(t-2)} + F_j^{(t-1)}, \tag{3.14}$$

where γ is a decay factor that controls how much the Fisher information from prior tasks influences the overall estimate $\bar{F}_j$.

Similarly to O-EWC, SI accumulates information on parameter importance in an online fashion by capturing the sensitivity of the loss function to each parameter after every training step and penalizing changes accordingly. The modified loss function during the training of task t in SI is defined as

$$\mathcal{L}_{\text{SI}}(\theta) = \mathcal{L}_t(\theta) + \lambda \sum_j \Omega_j^{(t-1)} \left(\theta_j - \hat{\theta}_j^{(t-1)}\right)^2, \tag{3.15}$$

where the importance score $\Omega_j^{(t-1)}$ of parameter θ_j is calculated as

$$\Omega_j^{(t-1)} = \sum_{i=1}^{t-1} \frac{\omega_j^{(i)}}{\left(\Delta\theta_j^{(i)}\right)^2 + \xi}, \tag{3.16}$$

with $\Delta\theta_j^{(i)}$ representing the total change in parameter θ_j after training on the i^{th} task, and ξ being a small positive constant. The term $\omega_j^{(i)}$ denotes the path integral of the gradient during training, and it is computed by accumulating the product of the gradient of the loss with respect to the j^{th} parameter and the change in this parameter over each training step s:

$$\omega_j^{(i)} = \sum_s \frac{\partial \mathcal{L}_s(\theta)}{\partial \theta_j^{(i)}} \Delta\theta_j^{(s)}, \tag{3.17}$$

where $\Delta\theta_j^{(s)}$ is the change in the j^{th} parameter after performing the s^{th} update step and $\mathcal{L}_s$ is the total loss measured at the s^{th} update step.

Unlike SI, which tracks the importance of all parameters across the entire sequence of task trajectories, O-EWC is more memory-efficient by maintaining and updating only a diagonal Fisher information matrix. The use of the Fisher information matrix in O-EWC is theoretically grounded in Bayesian principles, where the diagonal Fisher matrix approximates the posterior distribution as a Gaussian, with the mean corresponding to the optimal parameters after learning a previous task. This approach potentially yields more accurate estimates of parameter importance compared with SI, which derives its measure empirically from the trajectory of weight changes during training. As a result, O-EWC will be utilized as a representative baseline for prior-focused regularization in this thesis.

Functional regularization. Functional regularization, also referred to as *data-focused* regularization, aims to preserve the model's output behavior on previously learned tasks while acquiring new knowledge. Unlike parameter regularization methods, which directly restrict updates to model parameters, functional regularization focuses on maintaining consistent predictions across tasks. This approach is particularly effective when prior tasks' outputs are meaningful and not overly noisy. A prominent example is Learning without Forgetting (LwF; Li and Hoiem, 2018), which incorporates a regularization term in the loss function to reduce the divergence between the model's current predictions on new tasks and its stored predictions (*soft targets*) from earlier tasks. In LwF, the loss function for training task $t > 1$ is defined as

$$\mathcal{L}_{\text{LwF}}(\theta) = \mathcal{L}_t(\theta) - \lambda \sum_{i=1}^{t-1} \sum_{y \in \mathcal{Y}_i} p_{\hat{\theta}^{(t-1)}}(y|\boldsymbol{x}) \log(p_\theta(y|\boldsymbol{x})), \tag{3.18}$$

where $\mathcal{Y}_i$ is the set of output classes from the i^{th} task and $\hat{\theta}^{(t-1)}$ are the parameters at the end of training on task $t-1$. $p_\theta(y|\boldsymbol{x})$ is the output of the temperature-controlled softmax function (*i.e.*, the probability that sample $\boldsymbol{x}$ belongs to class y), as predicted by the model parameterized with θ.

Replay-based Approach

Rooted in CLS theory (*cf.* Section 3.1.3), replay-based, or *rehearsal*, techniques address the issue of catastrophic forgetting by approximating the data distributions of previously learned tasks and periodically retraining the model on these approximations. Typically, these methods either store a subset of past training examples in an episodic memory buffer for periodic replay during new task training, leverage generative models to synthesize data that mimic the statistical properties of earlier tasks, or reconstruct the distribution of latent features—rather than raw data—using, *e.g.*, prototypical methods or latent-space generative models. Figure 3.4 provides an overview of these replay-based CL strategies.

Experience replay. Experience Replay (ER) is the most straightforward and widely used replay strategy in CL, as shown in works such as Chaudhry et al.

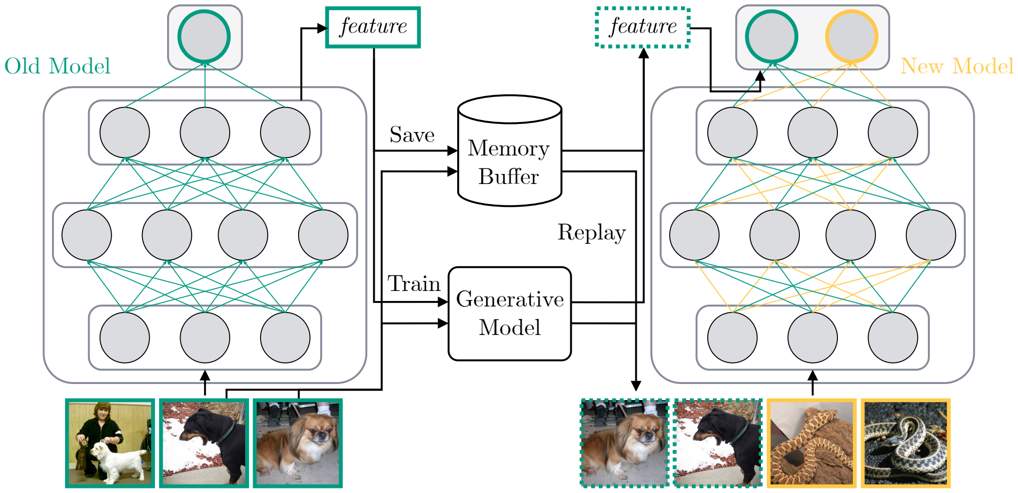


Figure 3.4: Replay-based methods for continual learning. Experience replay periodically retrains a model using historical data stored in and sampled from a memory buffer. In contrast, generative replay retrains the model using synthetic data instead of real data. Latent (or feature) replay, on the other hand, reconstructs the distribution of features rather than the raw data. Adapted from Wang et al. (2024).

(2019), Hayes et al. (2019), and Riemer et al. (2019). ER operates by maintaining a small, fixed-size memory buffer, $\mathcal{M}$, which stores a subset of examples from previous tasks. In classification problems, this buffer can be organized into label-wise replay buffers, denoted as $\mathcal{M} = \bigcup_{y \in \mathcal{Y}} \mathcal{M}_y$, where $\mathcal{Y}$ represents the set of possible labels. A more problem-agnostic approach, which requires the availability of task identifiers during training, is to organize $\mathcal{M}$ into task-specific replay buffers, resulting in $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_t\}$ upon completion of task t. This task-specific organization enables balanced task sampling based on predefined criteria, such as ensuring equal representation of each prior task during replay. During the training of each new task, this memory buffer helps mitigate catastrophic forgetting by mixing current task data with memory samples. Assuming per-task replay buffers, the learning process for each task $t > 1$ is guided by a modified loss function:

$$\mathcal{L}_{\text{ER}}(\theta) = \mathcal{L}_t(\theta) + \lambda \sum_{i=1}^{t-1} \mathcal{L}(\theta; \mathcal{M}_i), \tag{3.19}$$

where λ is a hyperparameter that controls the contribution of replayed samples to the overall loss, analogous to its role in regularization terms. In practice, replay samples are often processed as a (mini-)batch that is interleaved with the current task's (batched) data, typically with $\lambda = 1$. The objective of minimizing the loss function by updating θ can be achieved either through joint training on the current task data and memory samples or by alternating between them.

The memory buffer $\mathcal{M}$ is frequently updated with new examples from the current task. Various strategies have been proposed to effectively manage and populate this buffer. One of the most widely used strategies is *reservoir sampling* (Vitter, 1985), which was later adopted by Riemer et al. (2019). Reservoir sampling maintains a random subset of exemplar samples in the buffer, with the probability that a sample is selected decreasing as more data points are observed. Another approach is the *ring buffer*, used by Lopez-Paz and Ranzato (2017), where a fixed buffer size is allocated to each class, and recent examples are stored in a first-in-first-out manner. A different method for buffer management is the *k-means* approach, where the memory is populated with examples closest to the centroids in the feature space, calculated using an online k-means clustering algorithm (MacQueen, 1967). Similarly, the *mean of features* strategy, as demonstrated by Rebuffi et al. (2017b), retains examples closest to the running average feature vector for each class, ensuring that the most representative examples are stored. Lastly, Aljundi et al. (2019) propose a *gradient-based sampling* method, in which samples are selected based on maximizing gradient diversity, ensuring that the buffer effectively captures the diversity of previous tasks.

Generative replay. Generative replay and pseudo-rehearsal techniques address the limitations associated with storing large amounts of data, especially as the number of tasks grows large. These methods utilize generative models to synthesize samples that mimic the underlying distribution of previous tasks. By generating representative data from prior tasks, these techniques substantially reduce memory requirements while preserving task-specific knowledge to a significant extent. In particular, methods such as Deep Generative Replay (DGR; Shin et al., 2017) and Reinforcement-Pseudo-Rehearsal (RePR; Atkinson et al., 2021) leverage Generative Adversarial Networks (GANs; Goodfellow et al., 2014) to produce high-fidelity synthetic data. These pseudo-realistic samples are then replayed alongside data from new tasks to preserve knowledge. Alternatively, Lifelong Generative Model (LGM; Ramapuram et al., 2020) generates synthetic samples in a latent space, which significantly reduces computational complexity compared to generating data in the original input space. This approach employs a student-teacher Variational Autoencoder (VAE; Kingma and Welling, 2013) architecture, which effectively compresses task data into a lower-dimensional latent space for replay while balancing compression with knowledge retention.

Although generative models can alleviate memory constraints by synthesizing data on demand, they frequently produce samples with reduced diversity or fidelity, a limitation arising from phenomena such as mode collapse and discrepancies between the generated and true data distributions. Consequently, these models often require a substantially larger number of synthetic samples and require more frequent sampling to achieve performance levels comparable to the standard ER method (Van De Ven et al., 2022; Goodfellow et al., 2014).

Latent replay. In contrast to experience and generative replay, latent replay (or feature replay) stores hidden features rather than raw data. This approach is particularly advantageous when dealing with high-dimensional input data, such as images or audio, where storing raw data is resource-intensive and poses privacy concerns due to the sensitive nature of the information. Thus, latent replay presents a compelling strategy to enhance memory efficiency and preserve privacy. However, a key challenge in latent replay is representational drift: as the model's internal representations evolve over time, the stored latent features may become misaligned with the updated model, and thus result in performance degradation on previous tasks. This problem is particularly evident when training models from scratch, where a high degree of model plasticity is required to accommodate new information.

To address this issue, Generative Feature Replay (GFR; Liu et al., 2020a) and Feature Adaptation (FA; Iscen et al., 2020) perform feature distillation using stored features, whereas Prototype Reminiscence and Asymmetric Knowledge Aggregation (PRAKA; Shi and Ye, 2023) leverages class prototypes for distillation. Incremental Learning with Dual Memory (IL2M; Belouadah and Popescu, 2019) employs a dual-memory strategy that combines experience replay with latent replay based on recovered class statistics. Revived Evanescent Representations (RER; Toldo and Ozay, 2022) estimates the representational drift to update stored features. Replay using Memory Indexing (REMIND; Hayes et al., 2020) and Feature Translation for Incremental Learning (FeTrIL; Petit et al., 2023) freeze specific parts of the feature extractor: REMIND freezes early layers and replays compressed intermediate representations to adjust later layers, while FeTrIL freezes the entire feature extractor after the first incremental learning stage and subsequently uses latent replay to refine the prediction layer.

Traditional latent replay methods for CL typically rely on feature extractors or encoders that are either trained from scratch or require additional fine-tuning due to their limited capacity (*e.g.*, ResNet-18). In contrast, foundation models function as generic feature extractors that deliver stable yet expressive compressed representations of inputs. By keeping the encoder parameters of a foundation model fixed during incremental learning, we naturally mitigate representational drift. This approach not only overcomes the inherent limitations of latent replay but also preserves its efficiency and privacy advantages. This principle forms the core of the CL methodology presented in Chapter 7.

Architecture-based Approach

One significant limitation of regularization-based and replay-based strategies for CL is their reliance on a fixed set of model parameters that are fully shared across all tasks. This shared parameter space makes them vulnerable to interference between sequentially learned tasks, as optimizing parameters for one task may negatively impact the performance of others. To address this challenge, architecture-based methods for CL allocate separate resources for different tasks to prevent competition for structural resources within the model. As illustrated in Figure 3.5, these approaches aim to minimize interference by either expanding the model, se-

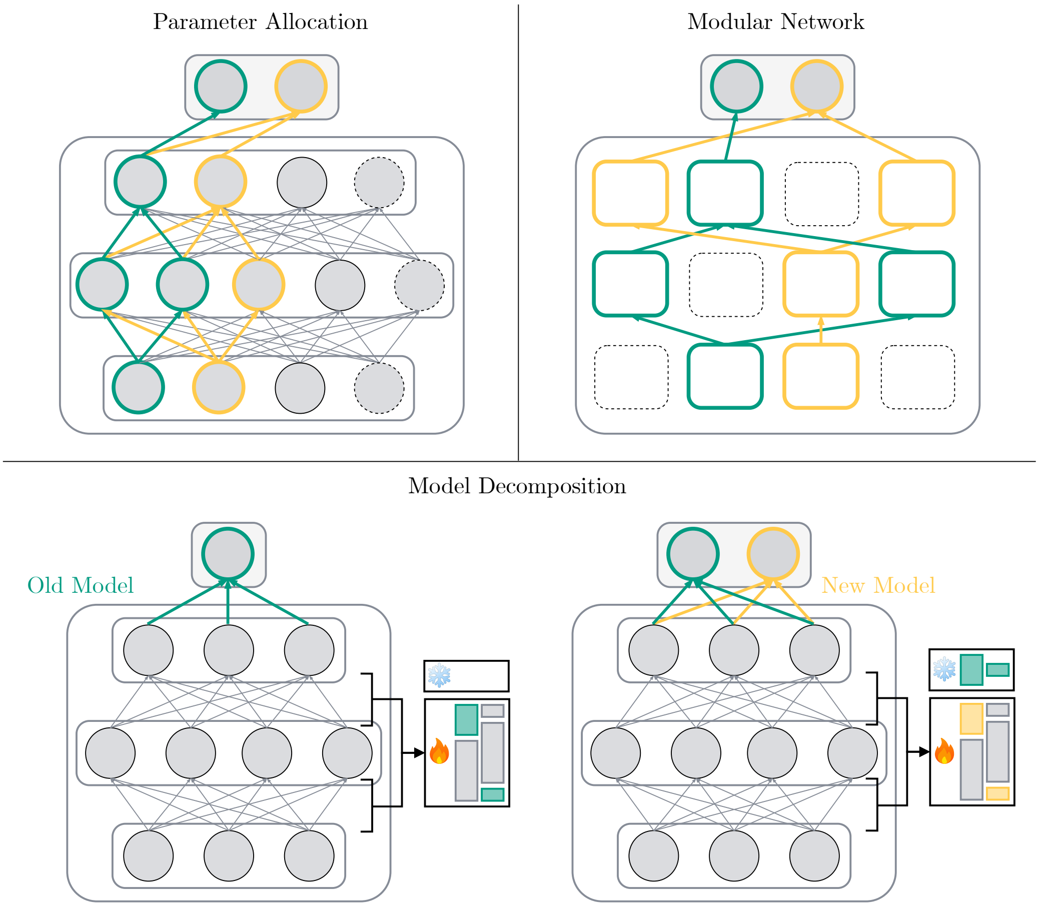


Figure 3.5: Architecture-based methods for continual learning. The primary goal of corresponding approaches is to construct task-specific parameters that minimize interference across tasks. Parameter allocation approaches learn task-specific parameters within either a fixed or dynamically expanding model architecture. Modular network approaches take a more coarse-grained perspective by creating task-specific parallel subnetworks or modules. Finally, model decomposition techniques explicitly partition the network into shared and task-specific components, typically allowing the latter to expand. In the illustrated example, a distinct subset of model parameters is buffered for each task. Adapted from Wang et al. (2024).

lectively activating distinct subsets of parameters, or utilizing a modular network structure that assigns dedicated components to individual tasks.

Parameter allocation. In parameter allocation, each task is assigned a dedicated subspace of parameters within a dynamically expanded or fixed-size network. Approaches that *dynamically expand* the network offer flexibility by adjusting the model structure as new tasks emerge. For instance, Dynamically Expandable Network (DEN; Yoon et al., 2018) adds or duplicates neurons when the current network configuration cannot adequately learn a new task. Expanding on this idea, Compacting, Picking, and Growing (CPG; Hung et al., 2019) enhances scalability by enabling both network expansion and compression through operations such as compacting, splitting, and merging. While these structural adaptations help manage the increasing complexity of tasks, they require careful oversight to prevent excessive resource consumption. Consequently, achieving sublinear model growth with respect to the number of tasks is a desirable goal for CL methods involving network expansion.

Methods that maintain a *fixed* network structure provide a direct solution to the problem of excessive consumption of computational and memory resources, which arises from the unbounded growth of model parameters. A widely adopted category within this framework is masking methods, exemplified by Piggyback (Mallya et al., 2018), Hard Attention to the Task (HAT; Serra et al., 2018), and Winning SubNetworks (WSN; Kang et al., 2022). These methods allocate task-specific resources by applying binary or soft masks over the trainable parameters. After training each task, they freeze critical (*masked*) parameters to prevent interference with subsequent tasks.

Other methods within this framework focus on explicitly identifying and reallocating parameters to enable more dynamic and flexible parameter management. To achieve this, they employ strategies such as iterative pruning (Mallya and Lazebnik, 2018), node-based uncertainty estimation (Ahn et al., 2019), and activation-based parameter selection (Jung et al., 2020). An inherent limitation of fixed-capacity models is *parameter saturation*, where all parameters are fully allocated to learned tasks, leaving no capacity for the model to effectively learn new tasks. To address this issue, strategies for optimizing parameter reuse, such as sharing parameters across tasks to minimize interference, or implementing structural sparsification, like pruning less important parameters to free up capacity, are necessary to maintain model plasticity.

Model decomposition. Model decomposition methods for CL involve the *explicit* partitioning of a model into task-sharing and task-specific components. This strategy offers high interpretability because explicit partitioning into task-sharing and task-specific components allows practitioners to directly observe and analyze how the model allocates and utilizes knowledge for different tasks. The task-specific components may include elements such as network layers (Loo et al., 2021), parallel networks (Wu et al., 2021c), or intermediate features (Hurtado et al., 2021; Abati et al., 2020). Additive Parameter Decomposition (APD; Yoon et al., 2020) partitions a trainable network into task-shared and sparse task-specific parameters by using small mask vectors to delineate these partitions. This approach ensures that task-specific parameters from previous tasks remain largely unchanged, with updates concentrated primarily on shared parameters to accommodate new tasks. In contrast, Reparameterized Convolutions for Multi-task learning (RCM; Kanakis et al., 2020) decompose convolution parameters into a shared, non-trainable filter bank and task-specific modulators. Unlike APD, which allows for repeated updates to shared parameters, RCM eliminates task interference by design, allowing each task to optimize only its modulators while preventing any updates to shared filters.

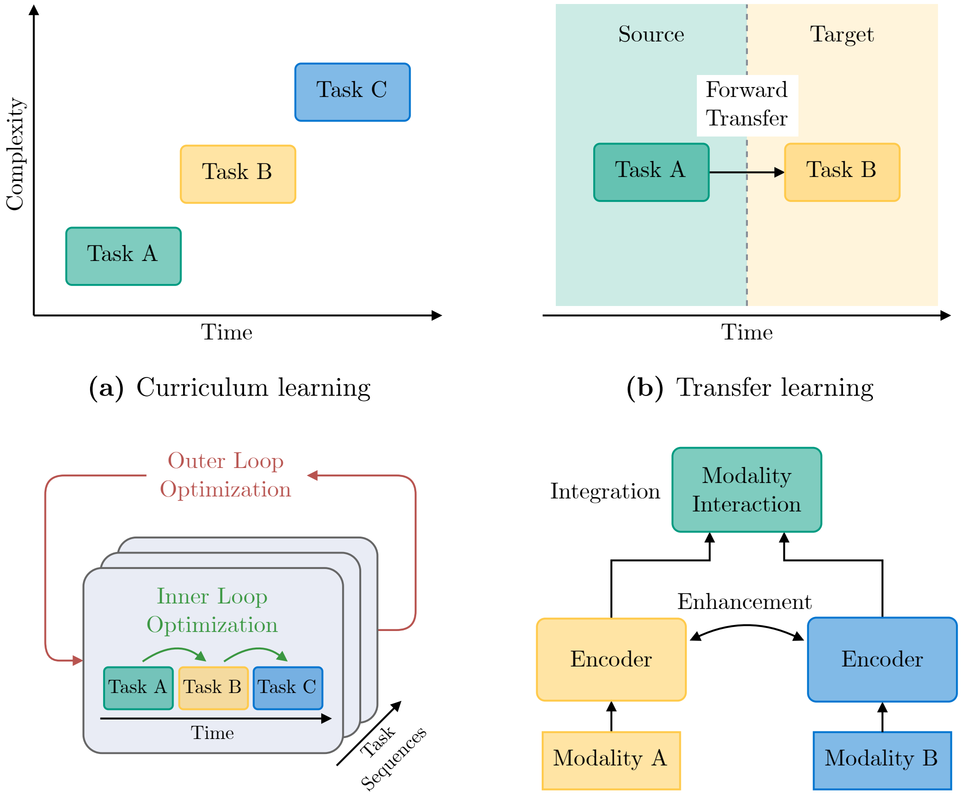
A common challenge with model decomposition approaches in CL is the linear increase in the number of task-specific components with the number of tasks. Consequently, as with parameter allocation methods involving network expansion, optimizing the memory and computational demands of these components is essential for maintaining the scalability of such approaches.

Modular networks. Modular network approaches selectively activate or combine parallel modules or subnetworks to handle a sequence of tasks. Unlike parameter allocation methods, which operate at the parameter level, these approaches function at a higher, modular level. Additionally, in contrast to model decomposition techniques, which impose predefined task-specific and task-sharing parameters, modular networks dynamically select parameters during training, allowing for more flexible adaptation to different tasks. However, this flexibility may come at the cost of interpretability, as it becomes difficult to trace how specific parameter selections impact the model's learning dynamics or correspond to the inherent characteristics of the tasks.

One of the pioneering methods in this domain is Progressive Neural Network (PNN; Rusu et al., 2016), which introduces a framework where a new subnetwork is added for each task, allowing previously learned representations to be reused through lateral connections to promote knowledge transfer. Building on this concept, Expert Gate (Aljundi et al., 2017) similarly adds a new expert subnetwork for each task during training, but incorporates a dynamic gating mechanism to select the most relevant expert module during inference. PathNet (Fernando et al., 2017) takes a different approach, employing evolutionary algorithms to select the optimal pathway through a fixed-size network of parallel modules for each task, with the pathways evolving over multiple generations to improve task performance. In contrast, Local Module Composition (LMC; Ostapenko et al., 2021) dynamically creates, composes, and reuses a set of local modules for each new task without freezing specific pathways to maintain the model's structural flexibility.

3.2.6 Related Machine Learning Paradigms

CL shares commonalities with several machine learning paradigms, each offering distinct approaches to managing dynamic, evolving, or multimodal data streams. These paradigms provide valuable information on strategies to improve model performance and address challenges such as knowledge transfer and adaptation. In this section, we examine four major paradigms—curriculum learning, transfer learning, meta-learning, and multimodal learning—detailing their contributions, distinctions, and relevance to CL. By analyzing how these paradigms influence and enhance CL methodologies, we aim to highlight their impact on advancing the field.



(a) Curriculum learning

(b) Transfer learning

(c) Meta-learning

(d) Multimodal learning

Figure 3.6: Machine learning paradigms related to continual learning. **(a)** Curriculum learning structures tasks in an increasingly complex sequence to achieve competence in the final, most challenging task. **(b)** Transfer learning leverages knowledge from a source task to enhance performance on a target task. **(c)** Meta-learning utilizes nested optimization loops to learn implicit inductive biases across tasks for improved future learning. **(d)** Multimodal learning integrates multiple modalities to improve a model's ability to learn from diverse input types.

Curriculum learning. In Section 3.1.1, we discussed *developmental learning*, the ability of humans to rapidly acquire foundational skills during critical periods of increased plasticity and stabilize the circuits needed for future learning during synaptic stabilization (Hensch, 2005; Knudsen, 2004). Once stabilized, these circuits offer a scaffold for cumulative skill acquisition and efficient integration of new knowledge with existing cognitive structures (Huttenlocher, 2009).

Curriculum learning (Bengio et al., 2009; Wang et al., 2022a) mimics human developmental processes in machine learning by organizing training tasks from simpler to more complex, thus simulating the skill acquisition process in humans. The most challenging task is typically the last, and solving it requires synthesizing knowledge from earlier, simpler tasks. This approach is especially effective under constraints such as limited training time or noisy data (Wu et al., 2021b), both of which are common in real-world curriculum learning scenarios. In addition, curriculum learning can enable the mastery of challenging tasks that might be unsolvable with traditional methods (Lee et al., 2022; Florensa et al., 2017) and is beneficial in learning scenarios involving significant shifts in task or data distributions (Wang et al., 2019; Shu et al., 2019).

Despite these similarities, two key differences set curriculum learning apart from CL. First, in CL, the sequence of input data is typically unpredictable, whereas in curriculum learning, the sequence is intentionally structured. Second, curriculum learning usually focuses on optimizing performance on the *final*, most complex task rather than achieving balanced performance across *all* tasks. Nonetheless, incorporating an automated curriculum—in which the model self-selects the task at each learning stage—may improve knowledge transfer and data efficiency in a CL setting (Say and Oztop, 2023; Mendez-Mendez et al., 2023).

Transfer learning. *Transfer learning* (Zhuang et al., 2021) involves leveraging knowledge acquired from one task (the *source*) and applying it to another, related task (the *target*). The main objective is to exploit similarities between the source and target tasks to enhance performance on the target task, especially when the training data for the target is limited. An important subfield of transfer learning is *domain adaptation*, where the source and target tasks are similar but originate from different domains (Farahani et al., 2021). Similarly to transfer learning, domain adaptation is unidirectional, generally involving a one-way transition from source to target.

Although transfer learning and CL both aim to leverage prior knowledge to improve performance on new tasks, they differ fundamentally in their approaches. Transfer learning typically addresses a single distributional shift and often does not involve further model adaptation beyond the initial transfer. It is primarily focused on optimizing performance on the target task, with less concern for the source task's performance. In contrast, CL is designed for sequential learning of multiple tasks in an open-ended manner while balancing knowledge transfer and retention. Farquhar and Gal (2019) elaborate in detail why excelling in a two-task transfer does not guarantee that a model can effectively learn a long sequence of tasks without forgetting.

Meta-learning. The CL strategies discussed in Section Section 3.2.5 generally rely on manually designed mechanisms or model architectures to introduce inductive biases that facilitate learning. In contrast, *meta-learning* (Thrun and Pratt, 1998; Vilalta and Drissi, 2002) aims to automatically learn these inductive biases, with the primary goal of enhancing forward transfer and adaptation across tasks. Often referred to as "learning to learn," this approach seeks to enable models to quickly acquire new tasks with minimal examples by exploiting structural similarities with previously learned tasks, akin to transfer learning. Intuitively, meta-learning operates on two distinct time scales within a two-loop optimization process: the *inner loop* and the *outer loop*. The inner loop focuses on adapting the model to one or more specific tasks using a limited number of data points, effectively simulating the initial learning experience. In contrast, the outer loop optimizes the model's parameter initialization and learning strategy by evaluating performance across multiple inner-loop adaptations. This dual-loop optimization framework fine-tunes the model's learning process, ensuring that it can generalize more effectively and adapt more quickly to new tasks by leveraging previous experience.

In the context of CL, outer loops can be configured to optimize performance in non-stationary learning environments (Banayeeanzade et al., 2021). However, meta-learning alone has limited ability to fully mitigate catastrophic forgetting, as it is not inherently designed to maintain high performance on previously learned tasks. Therefore, it is typically combined with additional CL strategies, such as memory replay (Cetin et al., 2023; Ho et al., 2024) or parameter regularization (Wu et al., 2024b).

Multimodal learning. *Multimodal learning* refers to the process by which a model integrates and leverages information from multiple sensory modalities—such as vision, sound, and touch—to enhance learning and decision-making (Baltrušaitis et al., 2019). This paradigm is inspired by the human ability to naturally combine information from different sensory stimuli to form a coherent understanding of the world (Stein and Stanford, 2008; Calvert et al., 2004). By compensating for the limitations of one modality with the strengths of another, multimodal learning enables models to discover latent correlations among various sensory inputs. Furthermore, it broadens the applicability and generalizability of models across various tasks and environmental conditions (Ngiam et al., 2011).

Unlike CL, which is inherently sequential, multimodal learning emphasizes the integration of multiple modalities over managing the temporal sequence of task learning. Nevertheless, in the context of CL, incorporating multimodal learning can harness the diverse and complementary nature of multimodal data, opening avenues for deploying CL methods in real-world environments. By simultaneously training on and integrating information from multiple sensory streams, models can develop more robust and reusable representations that are less likely to degrade when encountering distributional shifts or novel tasks (Baltrušaitis et al., 2019). For instance, when visual information is unavailable, changes over time, or is ambiguous, a model trained with multimodal learning might still perform well by relying on other modalities such as textual input to retain task-relevant knowledge. This flexibility not only improves the performance of the model on new tasks, but also helps to retain previously learned tasks by reinforcing representations through multiple channels (Srivastava and Salakhutdinov, 2012; Parisi et al., 2019).

Among the various applications of multimodal learning, VL grounding is particularly prominent (*cf*. Section 2.2). Extensive research has explored the intersection of CL and visually grounded language learning, utilizing both diagnostic (Skantze and Willemsen, 2022; Greco et al., 2019; Liu et al., 2023) and real-world datasets (Srinivasan et al., 2022; Jin et al., 2020, 2024). These studies consistently demonstrate that traditional CL methods, originally designed for unimodal learning, often struggle to balance the trade-off between mitigating forgetting and enabling effective cross-task knowledge transfer in multimodal contexts.

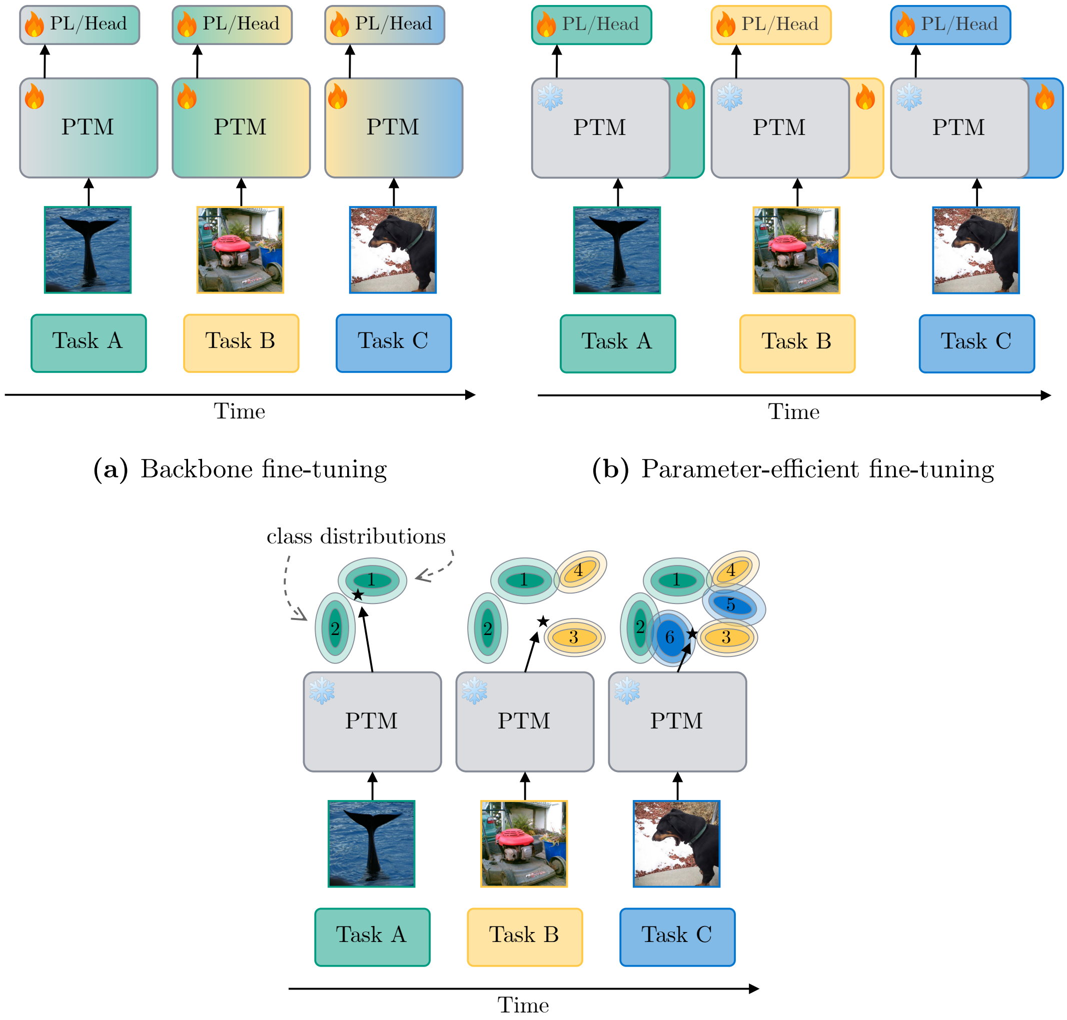
While some of these studies propose novel multimodal CL methods that enhance performance compared with traditional CL baselines, they generally fail to provide a comprehensive understanding of how insights into learning dynamics or the architectural design of modality integration networks can be leveraged to maximize the reusability and robustness of learned representations. As we will demonstrate in Chapter 6, identifying and safeguarding components of a modality interaction network—whether at the level of individual parameters, neurons, or layers—that are specialized in learning visuo-linguistic concepts frequently recurring in subsequent tasks, such as the colors and shapes of objects, could significantly contribute to producing more reusable knowledge in a multimodal CL setting.

3.3 From Upstream Pretraining to Downstream Continual Learning

The advent of the foundation model era has significantly transformed the landscape of CL, leading to the development of numerous novel approaches specifically designed for PTMs. Unlike models trained from scratch, PTMs possess extensive foundational knowledge acquired through large-scale pretraining on diverse datasets. This broad understanding enables them to adapt more efficiently to downstream tasks, often requiring significantly fewer labeled examples for fine-tuning. Consequently, CL in the context of PTMs presents unique challenges that differ from traditional CL settings. The primary focus shifts from building new capabilities from the ground up to preserving and efficiently repurposing existing knowledge, ensuring that the models retain their broad understanding while incrementally specializing in downstream tasks. Approaches to CL in PTMs (*cf.* Figure 3.7) vary based on which parameters are updated during training and whether pretrained representations are kept fixed or adapted as new tasks are learned.

3.3.1 Backbone Adaptation Methods

Most CL methods introduced during the early stages of the foundation model era involved updating all the parameters of a pretrained backbone, a practice known as *full-body adaptation*. This trend can be attributed to the relatively modest size and capacity of pioneering transformer models like BERT and GPT-2, released



(a) Backbone fine-tuning

(b) Parameter-efficient fine-tuning

(c) Prototype construction

Figure 3.7: Strategies for downstream continual learning with foundation models. (a) Backbone fine-tuning adapts to each new task by updating all PTM parameters. **(b)** Parameter-efficient fine-tuning selectively freezes the PTM parameters and updates only task-specific PEFT parameters (*e.g.*, prompts or intra-layer adapters). Task-specific parameters from previous tasks are stored in memory and inserted into the model as needed during inference. **(c)** Prototype-based methods maintain and update class prototypes and statistics to represent features extracted from a fixed PTM. During inference, class assignment is achieved by comparing new input features with all class prototypes via similarity matching.

in 2018 and 2019. These models were significantly smaller than the much larger foundation models that were introduced just a few years later.[3]

[3] For example, the medium-sized version of LLaMA 3 (Dubey et al., 2024) has 70 billion trainable parameters, approximately 636 times more than the 110 million parameters in the BERT$_{\text{BASE}}$ model.

Early CL methods in NLP that modify the parameters of pretrained backbone models typically incorporate replay strategies. The CL version of Memory-based Parameter Adaptation (MbPA++; de Masson d' Autume et al., 2019) employs experience replay and test-time adaptation to maintain BERT's high performance across a sequence of downstream NLP tasks. To mitigate the memory overhead of handling an ever-growing set of tasks, Language Modeling for Lifelong Language Learning (LAMOL; Sun et al., 2020a) introduces pseudo-sample generation for rehearsal by providing GPT-2 with special input prompts containing task identifiers and task-specific instructions, guiding it to generate appropriate data for each task. Distill and Replay (DnR; Sun et al., 2020b) builds on LAMOL by further reducing model complexity through knowledge distillation in a teacher-student framework.

Meta-learning, which aims at sample-efficient adaptation to new tasks within an inner optimization loop (*cf*. Section 3.2.6), is particularly well suited for integration with PTM adaptation in the context of CL, where rapid adaptation is crucial. Holla et al. (2021) combine the experience replay strategy of MbPA++ with two meta-learning algorithms, *i.e.*, Online aware Meta-learning (OML; Javed and White, 2019) and A Neuromodulated Meta-Learning Algorithm (ANML; Beaulieu et al., 2020), to introduce the two methods OML-ER and ANML-ER. Unlike LAMOL and DnR, their approach supports single-epoch training and is task-agnostic, although it requires storing historical data in a memory buffer.

The emergence of transformer-based architectures in CV has led to the development of several full-body adaptation CL methods tailored for pretrained image recognition models. Slow Learner with Classifier Alignment (SLCA; Zhang et al., 2023) proposes a dual learning rate strategy, using a smaller learning rate to update the pretrained ViT backbone while applying a larger learning rate to update the linear probing layer. Additionally, SLCA accumulates class distributions and performs pseudo-rehearsal on synthesized class-wise features to rectify the linear probe. In contrast, L2 (Smith et al., 2023b) introduces regularization of self-attention parameters in a ViT during downstream continual fine-tuning.

3.3.2 Parameter-efficient Adaptation Methods

In Section 2.1.4, we discussed various PEFT strategies, emphasizing their computational and data efficiency compared with full-body adaptation. These methods have garnered increasing attention for their application in downstream CL with PTMs. The key motivation behind using PEFT lies in its ability to preserve the knowledge encoded within the pretrained backbone by limiting the training process to a small subset of parameters. This selective adaptation significantly reduces the computational overhead associated with full-model fine-tuning while retaining the core knowledge of the PTM. Additionally, since PEFT parameters operate as modular plug-ins that can be easily removed or exchanged, this modularity provides inherent resilience to catastrophic forgetting.

One line of research involves utilizing *prompt* or *prefix tuning* for CL with PTMs. Learning to Prompt (L2P; Wang et al., 2022f) maintains a pool of key-value prompts, where task-relevant prompts are dynamically selected by querying based on input instance features. DualPrompt (Wang et al., 2022e) extends L2P by introducing two types of prompts that encode task-specific and task-invariant knowledge in the adaptation process. S-Prompt (Wang et al., 2022b) learns task-specific prompts in a separate space for each task and explicitly infers task identity via k-NN during inference. Domain-Adaptive Prompt (DAP; Jung et al., 2023) proposes an adaptive prompt generator that provides instance-specific prompts as an alternative to the prompt pool used in previous methods. Progressive Prompt (Razdaibiedina et al., 2023) gradually expands prompts with each new task to be learned. Finally, Continual Decomposed Attention-Based Prompting (CODA-Prompt; Smith et al., 2023a) learns a combination of prompt components through input-conditioned attention weights.

As a parallel line of research to prompt learning for CL, which focuses on task-specific parameters at the input level, several methods have been proposed to enhance CL capabilities by integrating task-specific modules into PTMs at a structural level. Ke et al. (2021a,b) integrate a *capsule network* (Hinton et al., 2011) into a PTM, comprising a knowledge-sharing module and a task-specific module. Zhang et al. (2020) train a task-specific *side-network* whose outputs are fused with the PTM's outputs via summation. Zhao et al. (2022) employ *intra-layer adapters* and teacher-student training to perform CL in semi-supervised learning settings. Ermis et al. (2022) utilize a distillation mechanism to merge adapter parameters while controlling memory growth as the number of tasks increases. More recently, Yu et al. (2024) employ adapters within a mixture-of-experts framework to collaboratively address tasks in an incremental setting.

Overall, PEFT methods offer promising rehearsal-free alternatives to the full-body adaptation approaches to CL discussed in Section 3.3.1, which typically rely on replay mechanisms to mitigate forgetting caused by substantial parameter updates. However, integrating task-specific parameters into a PTM for each new task requires knowledge of the task identity during inference to select the appropriate parameter set. Consequently, many of these methods are restricted to TIL settings, where task information is available at test time. Leveraging PEFT methods in more challenging CL scenarios without task information during inference remains an active area of research, which will be addressed later in Chapter 5 of this thesis.

3.3.3 Prototype-based Methods

Both full-body adaptation and PEFT methods adjust the feature space by updating fully connected linear probes, which are optimized via softmax functions and iterative gradient steps. In scenarios with imbalanced or limited data, these updates disproportionately emphasize the most recent tasks, leading to *task-recency bias*. Specifically, the model tends to prioritize newly introduced classes or data at the expense of previously learned tasks (Mai et al., 2021; Rymarczyk et al., 2023; Wang et al., 2023). Moreover, because a linear probe can be viewed as a shallow prediction network—essentially a single fully connected layer—it is inherently more brittle and prone to catastrophic forgetting compared to deeper trainable networks, which requires additional mechanisms to mitigate this vulnerability (Ramasesh et al., 2021; Zhang et al., 2023).

As an alternative to the aforementioned methods, constructing *prototypes* directly from pretrained features has emerged as a cost-effective approach to mitigate forgetting in CL while leveraging the powerful representations from large-scale pretraining. Prototypes are representative vectors that capture the key characteristics of the instances they summarize. They are commonly used in classification tasks, where they are referred to as *class prototypes*, and the corresponding approaches that use class prototypes are referred to as *class-prototype methods*.

Class-prototype methods extract features from the last layer of a PTM encoder h and aggregate them to construct a representative prototype for each class. The most straightforward of these methods is the Nearest Mean Classifier (NMC), which computes the class prototype $\overline{\boldsymbol{c}}_y$ for each class $y \in \mathcal{Y}$ by averaging the features of its training samples extracted at the last (L^{th}) encoder layer, defined as

$$\overline{\boldsymbol{c}}_y = \frac{1}{K} \sum_{t=1}^{T} \sum_{n=1}^{N_t} [\![y = y_{t,n}]\!] \; h_L(\boldsymbol{x}_{t,n}), \tag{3.20}$$

with $[\![\cdot]\!]$ denoting the Iverson bracket (Iverson, 1962), $(\boldsymbol{x}_{t,n}, y_{t,n})$ representing the n^{th} sample-label pair of the t^{th} task, and $K = \sum_{t=1}^{T} \sum_{n=1}^{N_t} [\![y = y_{t,n}]\!]$.

In the inference phase, the NMC assigns each test sample to the class for which the prototype most closely aligns with its feature vector. This is achieved by either minimizing the Euclidean distance (Janson et al., 2022) or maximizing the cosine similarity (Zhou et al., 2024). When employing cosine similarity and given a test sample $\boldsymbol{x} \in \mathcal{D}_{\text{test}}$, the class label prediction is determined by

$$\hat{y} = \underset{y \in \{1,\ldots,C\}}{\arg\max}\, s_y, \qquad s_y := \frac{h_L(\boldsymbol{x})^T \, \overline{\boldsymbol{c}}_y}{\|h_L(\boldsymbol{x})\| \cdot \|\overline{\boldsymbol{c}}_y\|}, \tag{3.21}$$

where $\|\cdot\|$ denotes the L^2 norm. Both Janson et al. (2022) and Zhou et al. (2024) find that the conceptually simple NMC outperforms prompt learning methods like L2P and DualPrompt, while being entirely training-free. However, Panos et al. (2023) observe that the assumption of isotropic feature covariance, *i.e.*, features being mutually uncorrelated, made under Equation (3.21) does not hold for PTMs. To account for correlations between features and to better "calibrate" similarity measures, they propose using class-prototype methods that leverage second-order statistics to capture the covariance information of instance features.

McDonnell et al. (2023) propose a method based on the closed-form ordinary least squares solution to ridge regression (Murphy, 2012) to decorrelate class prototypes during CL. This method computes the Gram matrix $\boldsymbol{G}$ as summation over outer products as

$$\boldsymbol{G} = \sum_{t=1}^{T} \sum_{n=1}^{N_t} h_L(\boldsymbol{x}_{t,n}) \otimes h_L(\boldsymbol{x}_{t,n}) \tag{3.22}$$

and aggregates the class prototypes (or regressands) $\boldsymbol{c}_y$ via summation—rather than averaging to obtain $\overline{\boldsymbol{c}}_y$ as in Equation (3.20) (with the $\frac{1}{K}$ term omitted)—to yield:

$$\hat{y} = \underset{y \in \{1,\ldots,C\}}{\arg\max}\, s_y, \quad s_y := h_L(\boldsymbol{x})^T \left(\boldsymbol{G} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{c}_y \tag{3.23}$$

for a regression parameter $\lambda \geq 0$ and d_L-dimensional identity matrix $\boldsymbol{I}$. Although Equation (3.21) and Equation (3.23) are defined with respect to the maximum number of classes C after observing all T tasks, they can be applied after seeing any training sample $\boldsymbol{x}_{t,n}$ with any $t \leq T$ and any $n \leq N_t$, as class prototypes and Gram matrices can be updated online. When denoting the extracted features of some input datum $\boldsymbol{x}_{t,n} \in \mathcal{D}_t$ as $\boldsymbol{h}_{t,n} = h_L(\boldsymbol{x}_{t,n})$ and considering $\boldsymbol{H} \in \mathbb{R}^{N \times d_L}$ as concatenated row-vector features $\boldsymbol{h}_{t,n}$ of all N training samples, Equation (3.22) is reduced to $\boldsymbol{G} = \boldsymbol{H}^T\boldsymbol{H}$, which corresponds to the original definition of a Gram matrix used in the closed-form ridge estimator (Hoerl and Kennard, 1970).

The use of first- or second-order feature statistics has recently inspired several class-prototype methods for CL with PTMs. For example, Janson et al. (2022) employ the straightforward NMC without any updates to the PTM parameters. In contrast, Panos et al. (2023) propose First Session Adaptation (FSA) combined with an incremental variant of linear discriminant analysis. FSA addresses the limitations of prototype-based methods that rely on static representations for prototype extraction by (i) fine-tuning the model parameters during training on the first task, $\mathcal{D}_1$, to bridge the domain gap between pretraining and fine-tuning data, and (ii) extracting prototypes only afterward, when no further optimization steps are performed.

APER (Zhou et al., 2024) combines FSA with NMC as the class-prototype method. RanPAC (McDonnell et al., 2023) extends this approach by incorporating high-dimensional random feature projections, aiming to decorrelate the class prototypes of a PTM, thereby enhancing their separability. Notably, while Panos et al. (2023) modify all backbone parameters during FSA, both APER and RanPAC keep the backbone frozen, instead updating PEFT parameters using different strategies (*cf*. Section 2.1.4).

Prototype-based methods have shown promise in CL with PTMs, especially in rehearsal-free settings. However, they face two major drawbacks. First, they universally assume that class-specific information is fully captured in the foundation model's final-layer representations (obtained by applying $h_L(\cdot)$, *e.g.*, in Equation (3.20), where L denotes the last layer of encoder h). Whether these final-layer features provide sufficient information for a prototype-based classifier to optimally distinguish between classes remains unclear—a question we address in detail in

Chapter 5. Second, prototype-based methods inherently rely on measuring similarity between data points and prototypes within a feature space, making them naturally suited for classification tasks where the objective is to assign discrete labels based on proximity. However, they lack mechanisms to model complex relationships, contextual nuances, and structural dependencies within the data. As a result, they are not well equipped to handle tasks that require a deeper understanding, such as contextual interpretation or reasoning about the relationships between different elements in the input.

To address these shortcomings, we outline in Chapter 7 a novel CL method that synergistically combines replay-based and prototype-based paradigms. Our approach maintains a compact set of latent representations for each class—effectively serving as prototypes—and incorporates targeted noise into these stored latents. This strategy preserves essential ground-truth information, which is challenging to reproduce synthetically in non-classification contexts, while also leveraging the statistical properties of the prototypes to generate pseudo-latent representations. The resulting method enhances generalization in CL tasks that involve complex multimodal reasoning.

3.4 Chapter Summary

In this chapter, we formally introduced the CL problem and explored key methods and paradigms that enable models to learn from a continuous, non-stationary stream of information. Inspired by biological principles of the mammalian brain, CL has guided the development of several computational approaches. Traditional strategies such as regularization, replay, and dynamic architectures have proven effective in small neural networks trained from scratch. However, applying these strategies to large, parameter-heavy foundation models demands more thoughtful consideration, given the increased resource constraints these models impose.

While there has been progress in adapting CL to PTMs, many challenges remain. A central issue is deciding when and how extensively backbone parameters should be updated during sequential fine-tuning. Over-updating the backbone risks catastrophic forgetting, while insufficient updates can cause poor adaptation to new tasks. PEFT methods offer a promising solution that circumvents this trade-off by allowing for efficient task-specific adaptation without altering the backbone. However, further research is needed to evaluate their effectiveness, especially in task-agnostic CL scenarios, where task boundaries are unknown or undefined, especially during inference.

CL also faces inherent challenges due to often conflicting goals. The primary objective of CL is to balance the retention of prior knowledge with the integration of new information. However, the performance of various CL methods can vary significantly across different application areas. For instance, techniques optimized for image recognition may struggle in complex, multimodal reasoning scenarios.

Moreover, certain application domains may impose constraints on the storage of historical data due to privacy or regulatory concerns, or limit the resource consumption of CL methods. These challenges persist regardless of whether models are pretrained or randomly initialized. Thus, there is a need for carefully designed CL approaches that are versatile and scalable to apply across a broad range of application areas, while acknowledging that there is no one-size-fits-all solution to CL, a point that we will consistently demonstrate throughout this thesis.

We identify two promising yet underexplored research avenues that could greatly enhance the downstream continual adaptation of PTMs. First, in Chapters 4 and 5, we will explore whether PTMs inherently generate task-agnostic knowledge that can be systematically utilized to enhance CL. By identifying and leveraging this knowledge to augment pretrained representations, we introduce a form of "inherent resilience" into the PTM that relies less on task-specific modifications and more on generalized, reusable knowledge. Second, in Chapters 6 and 7, we investigate the underutilized potential of multimodal learning within CL. Our efforts focus on identifying network structures capable of retaining transferable knowledge when integrating features from multimodal foundation models, as well as on maximizing and refining knowledge gained from a limited set of extracted multimodal features to recover old data distributions. By advancing these research directions, we seek to develop more robust CL strategies that increase the practical impact of PTMs across diverse, real-world environments.

Part II

Unimodal Representation Augmentation

Chapter 4

Unsupervised Representation Modeling via Self-organization

Pretrained foundation models are susceptible to catastrophic forgetting when fine-tuned sequentially on multiple downstream tasks, as optimizing for new tasks can overwrite previously acquired knowledge. To address this limitation, we propose a dual-memory framework for CL that minimizes disruptive shifts in the latent representations of PTMs. Our approach is designed to maintain robust classification performance regardless of (i) the sequence in which tasks are introduced and (ii) the variability in dataset sizes across tasks. A central component of our method is the unsupervised modeling of latent representations via topological mapping. We hypothesize that this technique intrinsically stabilizes the learning dynamics by preserving the underlying manifold structure of pretrained representations during continual fine-tuning, thereby enhancing robustness against forgetting. We validate our method by comparing it with state-of-the-art replay-based CL baselines and demonstrate its efficacy in maintaining high performance, particularly in scenarios where data from earlier training phases become underrepresented.[1]

4.1 Motivation

In Section 2.1.2, we described how pioneering models like BERT have become fundamental in advancing NLP following the advent of the transformer architecture. These transformer-based models, pretrained on massive text corpora, capture complex linguistic structures and yield rich, transferable representations for a wide array of downstream tasks. Conventional fine-tuning procedures for these models typically assume that the training data are drawn from a stationary distribution and are independently and identically distributed, mirroring the assumptions made in traditional supervised learning for models trained from scratch. However, as detailed in Chapter 3, these assumptions often do not hold in real-world scenarios,

[1] The source code is made available at https://github.com/knowledgetechnologyuhh/drill.

where models are exposed to continuously evolving data streams reflecting changes in context, user preferences, or domains. Consequently, when a PTM is sequentially fine-tuned on new tasks or datasets, the optimization process can overwrite previously acquired knowledge, leading to catastrophic forgetting.

One avenue to address these challenges and bridge the gap between the strengths of PTMs and the demands posed by CL is exploring innovative approaches to leverage the innate knowledge representations within PTMs to enhance robustness against forgetting in CL settings. In this chapter, we aim at answering the questions (i) whether integrating self-organizing network architectures with PTMs to maintain topological relationships in the feature space can preserve learned knowledge and (ii) how such architectures impact the learning of underrepresented or minority classes in imbalanced datasets.

As for the first research question, our central hypothesis is that integrating self-organizing network mechanisms with PTMs can significantly mitigate catastrophic forgetting by maintaining the structural integrity of learned representations. Self-organizing networks, such as SOINN (*cf*. Section 3.1.2), construct prototypes as weight vectors representing clusters or patterns in the input data. These prototypes effectively summarize the data characteristics and are updated during training to align with the input space while maintaining topological relationships. In NLP, texts belonging to the same category or class often cluster around prototypical examples due to shared semantic or syntactic features. As for the second research question, we additionally hypothesize that self-organizing network architectures inherently favor smaller or underrepresented classes, as these may occupy distinct regions in the feature space, which prompts the network to allocate dedicated nodes or prototypes to them and encourages the long-term preservation of these nodes.

Building on this, we present **D**ynamic **R**epresentations for **I**mbalanced **L**ifelong **L**earning (**DRILL**), a novel CL method that integrates self-organizing network principles with PTMs. Drawing on the success of replay strategies in PTM-based CL and grounded in CLS theory (*cf*. Section 3.1.3), DRILL employs a dual-memory design: an *episodic memory* buffer periodically replays prior examples, while a self-organizing network acts as *semantic memory*, which enriches PTM-derived features with prototypical representations. This network incrementally adapts to new inputs while preserving the latent topology (*i.e.*, the structural arrangement and relationships among features). This dual-memory framework provides three key advantages: (i) it controls representational drift by maintaining stable feature mappings and thus reducing abrupt changes in representations that typically result in forgetting, (ii) it mitigates task-recency bias by recalibrating the PL to prevent it from overly predicting classes from the most recent task, and (iii) it favors underrepresented classes by allocating resources to sparsely populated regions of the feature space.

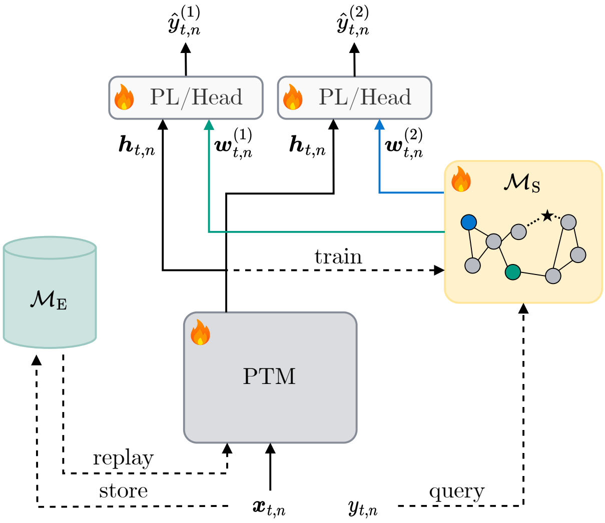


Figure 4.1: Overview of DRILL. For each input sample $(\boldsymbol{x}_{t,n}, y_{t,n})$, drawn either from the continual data stream or the episodic memory $\mathcal{M}_{\mathrm{E}}$, the PTM extracts features $\boldsymbol{h}_{t,n}$ from $\boldsymbol{x}_{t,n}$. These features are used to train the semantic memory $\mathcal{M}_{\mathrm{S}}$. Simultaneously, the ground truth label $y_{t,n}$ is used to retrieve two class-prototype vectors $\boldsymbol{w}_{t,n}^{(1)}$ and $\boldsymbol{w}_{t,n}^{(2)}$ from $\mathcal{M}_{\mathrm{S}}$. Each prototype vector is concatenated with the extracted features $\boldsymbol{h}_{t,n}$ to yield two separate input vectors to the prediction layer and finally two label predictions which are each compared with $y_{t,n}$ for loss calculation.

Our contributions made in this chapter are twofold:

(1) We introduce two imbalanced sampling strategies for CIL where underrepresented classes appear early or late in the training data stream, respectively, and show that existing baseline CL methods show high task order sensitivity and low overall performance in either scenario.

(2) We additionally propose DRILL as a novel dual-memory CL method and empirically demonstrate that DRILL achieves strong performance against baseline methods in CL settings with class imbalances. We find that DRILL depicts lower sensitivity to task ordering and random initialization and is particularly effective in settings where underrepresented classes appear early during training.

This chapter is organized as follows: we lay the theoretical foundation for our DRILL method in Section 4.2. Subsequently, we describe our training details and conduct an experimental evaluation in Section 4.3. We discuss our findings in Section 4.4 and summarize this chapter in Section 4.5.

4.2 Proposed Method: DRILL

An overview of our DRILL training pipeline is provided in Figure 4.1. The model architecture comprises four main elements: (1) an episodic memory module $\mathcal{M}_\mathrm{E}$, (2) a semantic memory module $\mathcal{M}_\mathrm{S}$, (3) a PTM h_ϕ, and (4) a PL g_ψ. Components (1) and (2) together form the dual-memory system, while components (3) and (4) constitute the base model $f = g \circ h$. We adopt the CL problem formulation introduced in Section 3.2.1.

Upon receiving the n^{th} input sample $\boldsymbol{x}_{t,n}$ from the t^{th} task in the data stream, the PTM processes the input through its encoder to extract a d-dimensional feature representation, denoted as $\boldsymbol{h}_{t,n} = h_\phi(\boldsymbol{x}_{t,n})$. In transformer-based architectures, this representation is typically derived from the [CLS] token embedding of the final encoder layer. However, depending on the specific encoder architecture, the extracted feature can be any intermediate hidden representation, such as the flattened feature maps from a CNN encoder.

Using the ground-truth label $y_{t,n}$, the semantic memory $\mathcal{M}_\mathrm{S}$ is queried to retrieve two d-dimensional prototypical weight vectors, $\boldsymbol{w}^{(1)}_{t,n}$ and $\boldsymbol{w}^{(2)}_{t,n}$ (see Section 4.2.2 for additional details). The extracted sample features $\boldsymbol{h}_{t,n}$ are then concatenated with each of the prototype vectors to produce two distinct input vectors, $[\boldsymbol{w}^{(1)}_{t,n}; \boldsymbol{h}_{t,n}]$ and $[\boldsymbol{w}^{(2)}_{t,n}; \boldsymbol{h}_{t,n}]$. Each concatenated vector is then passed through the PL for loss computation and gradient calculation.

Therefore, during training, we optimize the combined set of all trainable parameters $\theta = \phi \cup \psi$ of the base model f_θ, which is defined by:

$$f_\theta(\boldsymbol{x}_{t,n}) = \hat{y}^{(i)}_{t,n} = g_\psi\left(\left[\boldsymbol{w}^{(i)}_{t,n}\ ;\ h_\phi(\boldsymbol{x}_{t,n})\right]\right) \tag{4.1}$$

where $[\cdot;\cdot]$ denotes the concatenation operator, and $i \in \{1, 2\}$.

4.2.1 Meta-learning with Experience Replay

Building upon the work of Holla et al. (2021), who demonstrated the efficacy of meta-learning in conjunction with experience replay, we integrate a meta-learning strategy within the DRILL framework and adopt their experimental protocol for constructing training episodes and managing experience replay. The pseudocode outlining the meta-training and meta-testing procedures for DRILL is provided in Algorithms 1 and 2, respectively. Let $\mathcal{B}_t$ denote the set of incremental batches associated with the t^{th} task, where each batch $b \in \mathcal{B}_t$ has a uniform size $|b|$. Each batch is stored in the episodic memory $\mathcal{M}_\mathrm{E}$, which has a maximum capacity B, using reservoir sampling as described in Section 3.2.5. In contrast to conventional sequential task training, our approach processes data in a streaming fashion by organizing it into episodes. Specifically, each episode i comprises $s + 1$ batches, with the first s batches forming the support set $\mathcal{S}_i$ and the final batch serving as the query set $\mathcal{Q}_i$.

Algorithm 1: DRILL (Meta-)Training

```
Input: model parameters $\theta = \phi \cup \psi$, replay interval $R_I$, replay frequency
       $R_F$, replay ratio $r$, episodic memory capacity $B$, support set buffer
       size $s$, inner-loop learning rate $\alpha$, outer-loop learning rate $\beta$, pull
       factor $\eta$
# Initialization
$\mathcal{M}_\text{E} \leftarrow \emptyset$
$\mathcal{M}_\text{S} \leftarrow$ three $d$-dimensional random vectors
# Continual Learning with DRILL
for episode $i = 1, 2, ...$ do
    $\mathcal{S}_i \leftarrow s$ batches from the stream
    if $i$ mod $R_F = 0$ then
        $\mathcal{Q}_i \leftarrow$ sample($\mathcal{M}_\text{E}$, $\lfloor r \cdot R_I \rfloor$)
    end
    else
        $\mathcal{Q}_i \leftarrow$ next batch from the stream
        write($\mathcal{M}_\text{E}, \mathcal{Q}_i, B$)
    end
    write($\mathcal{M}_\text{E}, \mathcal{S}_i, B$)
    for every sample $(\boldsymbol{x}_j, y_j) \in \mathcal{S}_i \cup \mathcal{Q}_i$ do
        $\boldsymbol{h}_j \leftarrow h_\phi(\boldsymbol{x}_j)$
        train($\mathcal{M}_\text{S}, \boldsymbol{h}_j, \eta$)
        $\boldsymbol{w}_j^{(1)}, \boldsymbol{w}_j^{(2)} \leftarrow$ sample($\mathcal{M}_\text{S}, y_j$)
        predict $\hat{y}_j^{(1)}, \hat{y}_j^{(2)}$ using Equation (4.1)
    end
    update $\psi'$ using SGD (cf. Equation (4.3))
    update $\theta$ using Adam (cf. Equation (4.4))
end
```

Let R_I denote the replay interval. After observing R_I samples from the data stream, we randomly draw $\lfloor r \cdot R_I \rfloor$ samples from $\mathcal{M}_\text{E}$ for rehearsal, where $r \in [0, 1]$ is the replay ratio. Using the replay interval and ratio, we calculate the replay frequency as

$$R_F = \left\lceil \frac{R_I/|b| + 1}{s + 1} \right\rceil \tag{4.2}$$

Therefore, every $R_F{}^\text{th}$ episode can be considered a replay episode, in which the query set is not composed of data from the stream, but from the episodic memory module $\mathcal{M}_\text{E}$.

During inner-loop meta-optimization of the i^th episode, the PTM parameters ϕ are fixed while the PL parameters ψ are updated using Stochastic Gradient Descent (SGD) with an inner-loop learning rate α. This update is performed as

$$\psi' \leftarrow \text{SGD}(\mathcal{L}_i(\phi, \psi), \mathcal{S}_i, \alpha) \tag{4.3}$$

Algorithm 2: DRILL (Meta-)Testing

```
Input: trained model parameters $\theta = \phi \cup \psi$, support set buffer size $s$,
       inner-loop learning rate $\alpha$
$\mathcal{S} \leftarrow$ sample($\mathcal{M}_\mathrm{E}, s \cdot |b|$)
$\mathcal{Q} \leftarrow \mathcal{D}_\text{test}$
update $\psi'$ using Equation (4.3)
for $\boldsymbol{x}_j \in \mathcal{Q}$ do
    $\boldsymbol{h}_j \leftarrow h_\phi(\boldsymbol{x}_j)$
    $\boldsymbol{w}_j \leftarrow \mathbf{0}$
    predict $\hat{y}_j$ using Equation (4.1)
end
```

Subsequently, during the outer-loop optimization for the i^th episode, both the PTM parameters ϕ and the meta-updated PL parameters ψ' are fine-tuned on the query set $\mathcal{Q}_i$. All model parameters θ are updated using the Adam optimizer (Kingma and Ba, 2015) with an outer-loop learning rate β, resulting in:

$$\theta' \leftarrow \mathrm{Adam}(\mathcal{L}_i(\phi, \psi'), \mathcal{Q}_i, \beta) \tag{4.4}$$

$\mathcal{L}_i$ denotes the cross-entropy loss on all batches of $\mathcal{S}_i$ in Equation (4.3) and $\mathcal{Q}_i$ in Equation (4.4), respectively.

4.2.2 Representation Sampling from Semantic Memory

Unlike the episodic memory $\mathcal{M}_\mathrm{E}$, which is used exclusively for experience replay, we utilize the semantic memory $\mathcal{M}_\mathrm{S}$ to retrieve class-prototype features that augment the features of the current input sample. We hypothesize that these class prototypes serve as regularizers for PL fine-tuning by mitigating its bias towards the most recent data.

In the semantic memory $\mathcal{M}_\mathrm{S}$, we store the set of all nodes of the underlying self-organizing network, denoted as $\mathcal{N}$, along with a lookup table that tracks the frequency with which each node has served as BMU for each label $y \in \mathcal{Y}$. We assume that the more often a node has been a BMU for input samples belonging to some class y, the better it represents samples of this class. The table has dimensions $|\mathcal{N}| \times |\bigcup_{i=1}^{t} \mathcal{Y}_i|$, where $|\bigcup_{i=1}^{t} \mathcal{Y}_i|$ denotes the total number of classes encountered up to the training of the t^th task. As new classes are introduced, the table can be dynamically expanded by adding new columns initialized to zero.

During training, for each input feature vector $\boldsymbol{h}_{t,n}$, we retrieve its BMU using Equation (3.3) and nodes in $\mathcal{M}_\mathrm{S}$ are updated according to the algorithm of the underlying topology-mapping network. Additionally, utilizing the ground-truth label $y_{t,n}$ and the lookup table, we identify the two neurons with neural weights $\boldsymbol{w}_{t,n}^{(1)} \in \mathbb{R}^d$ and $\boldsymbol{w}_{t,n}^{(2)} \in \mathbb{R}^d$ that have most frequently served as BMUs for class $y_{t,n}$. We choose to retrieve two neurons from $\mathcal{M}_\mathrm{S}$ rather than just one to enhance the diversity of prototypical augmentation features.

These two *winning nodes* are then concatenated with the latent features $\boldsymbol{h}_{t,n}$ extracted by the PTM, resulting in two inputs to the PL derived from a single PTM output. During inference, since the ground-truth label $y_{t,n}$ is unknown, we cannot query $\mathcal{M}_\mathrm{S}$ for class-specific prototypical features. Therefore, $\boldsymbol{h}_{t,n}$ is concatenated with a zero vector $\mathbf{0} \in \mathbb{R}^d$ before being fed into the PL.

For the experimental evaluation in Section 4.3, the semantic memory $\mathcal{M}_\mathrm{S}$ will be updated using the SOINN+ algorithm (Wiwatcharakoses and Berrar, 2020). Unlike the original SOINN (*cf*. Section 3.1.2), which deletes nodes based solely on elapsed time, SOINN+ additionally evaluates nodes by their trustworthiness and utility within the topology mapping network. Nodes in sparsely populated regions often represent edge cases, minor clusters, or transient concepts that might reoccur. Prematurely deleting these nodes risks losing the ability to recognize recurring patterns. SOINN+ mitigates this risk by conservatively retaining low-density nodes until clear evidence of irrelevance or noise emerges through failed trustworthiness and utility evaluations. This ensures the model robustly preserves relevant structures, even under sudden or recurring concept drifts.

4.3 Experiments

In the following, we present experiments comparing DRILL with baseline CL methods. Our analysis includes detailed descriptions of the benchmarks employed, the training protocols implemented, and an extended experimental investigation into the variability and the effects of feature integration strategies on overall CL performance.

4.3.1 Benchmarks

To comprehensively assess the CL capabilities of our method, we sequentially train it on five text classification datasets introduced by Zhang et al. (2015). These datasets encompass four distinct tasks: sentiment analysis, news topic classification, question-answer classification, and ontology categorization. A summary of all datasets is provided in Table 4.1 and examples can be found in Table 4.2. Recognizing the significant influence of task order on CL performance, as highlighted by de Masson d' Autume et al. (2019), we arrange the datasets into four randomized permutations. The incremental learning process across these datasets is treated as CIL, where new classes are introduced with each task, with the only exception of the sentiment analysis datasets, where label spaces are unified due to overlapping classes.

Previous studies on CL for text classification with PTMs (Holla et al., 2021; de Masson d' Autume et al., 2019; Sun et al., 2020a,b) have typically employed balanced sampling, where an equal amount of training data is sampled from each of the five original datasets. However, equally balancing data across datasets may not accurately reflect the efficacy of a CL method in more realistic scenarios, where

Classification Domain	**Dataset**	**Classes** (C)	**Order Position**			
			I	**II**	**III**	**IV**
Sentiment	**Amazon Reviews**	5	4	4	3	3
	Yelp Reviews	(merged)	1	5	1	2
News Topic	**AGNews**	4	2	3	5	1
Question Topic	**Yahoo! Answers**	10	5	2	2	4
Ontology	**DBPedia**	14	3	1	4	5
	Total:	**33**				

Table 4.1: Summary of the text classification datasets. All five text classification datasets were originally published by Zhang et al. (2015). Each dataset contains 7 600 test samples, with the number of training samples determined by the respective order and sampling strategy. Classes in the Amazon and Yelp Reviews datasets are merged, given their shared five-star rating scale. Examples for each dataset are provided in Table 4.2. In the CIL setting, a model is trained sequentially on all five datasets (*i.e.*, tasks) for each of the four task orders (I–IV).

tasks often vary in training duration and the number of steps. To better simulate practical conditions, we introduce two novel sampling techniques: *Progressive Oversampling* (PO) and *Progressive Undersampling* (PU). These methods exponentially increase or decrease the number of samples for each successive task, respectively, defined as follows.

For the PO setting

$$N_{t+1}^{(\mathrm{PO})} \leftarrow 2 \cdot N_t^{(\mathrm{PO})}, \tag{4.5}$$

and for the PU setting:

$$N_{t+1}^{(\mathrm{PU})} \leftarrow \left\lfloor \frac{N_t^{(\mathrm{PU})}}{2} \right\rfloor, \tag{4.6}$$

where N_t denotes the number of samples for task t. Both sampling techniques enable the simulation of two contrasting CL scenarios, where data from either earlier or later stages are markedly underrepresented.

For evaluation, we adhere to the methodology of previous studies by randomly selecting 7 600 samples from each of the five datasets, resulting in a total of 38 000 instances in the test dataset. However, differing from prior works—which utilize 115 000 training samples per dataset, totaling 575 000 training samples—we adopt the aforementioned PO and PU sampling strategies for training dataset construction. We set 115 000 as the maximum number of training samples for the most overrepresented dataset and apply the progressive imbalancing techniques, initializing with $N_0^{(\mathrm{PU})} = 115\,000$ for PU and $N_0^{(\mathrm{PO})} = 7\,187$ for PO. This strategy yields a total training set size of 222 812 samples for each sampling method.

Dataset	Input Example	Label
Amazon Reviews	These are great blocks but they are definitely not worth the listed \$79.99. They can be purchased for about \$20 many other places.	★★★☆☆
Yelp Reviews	The only thing worse than the food is the service.	★☆☆☆☆
AGNews	South Africa won the gold medal Sunday in the men's 400-meter freestyle relay with a world-record time of 3 minutes 13.17 seconds.	Sports
Yahoo! Answers	When is Thanksgiving celebrated in Canada?	Society & Culture
DBPedia	Witches Brew is an underground European record label which mainly sells music via online distribution.	Company

Table 4.2: Examples of the five text classification datasets. Amazon and Yelp reviews require assigning each text input to one of five star ratings. AGNews classifies texts into different news topics. Yahoo! Answers categorizes texts by question domains. DBPedia assigns texts to specific encyclopedia article topics.

4.3.2 Training Details

We employ $\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019) as the PTM $h_\phi(\cdot)$, which consists of 12 transformer encoder layers with a token embedding dimensionality of $d = 768$. The PL $g_\psi(\cdot)$ is implemented as a fully-connected linear layer followed by a softmax activation function to generate class probabilities. SOINN+ (*i.e.*, the semantic memory $\mathcal{M}_{\text{S}}$) is governed by a hyperparameter called the pull factor η, which determines the influence of new observations on neighboring nodes within the network. Given that Wiwatcharakoses and Berrar (2020) demonstrate the relative insensitivity of the network to the specific value of η, we fix it at a constant value of $\eta = 50$, as used in the original SOINN+ paper, throughout our experiments.

To evaluate the effectiveness of DRILL, we compare it with several baseline methods: Sequential Fine-Tuning (**SFT**) updates both the PTM and the PL without incorporating any CL mechanisms. It is typically regarded as the lower bound for CL performance as a result of its susceptibility to catastrophic forgetting. **ER** (*cf.* Equation (3.19)) extends SFT by introducing experience rehearsal, where historical exemplars are stored in the episodic memory $\mathcal{M}_{\text{E}}$. **ANML-ER** and **OML-ER** (Holla et al., 2021) integrate meta-learning algorithms with experience replay. For a fair comparison, we standardize the memory writing and rehearsal policies across ER, ANML-ER, OML-ER, and our proposed DRILL method. Additionally, to assess the upper performance limit achievable through full-body adaptation in a traditional batch learning setting, we include Joint Fine-Tuning (**JFT**) in our evaluation. In JFT, the PTM and PL parameters are jointly optimized over all tasks simultaneously.

All baseline methods are trained with a batch size of $|b| = 8$. We apply batch normalization (Ioffe and Szegedy, 2015) and truncate the input sequences of the PTM BERT$_{\text{BASE}}$ to a maximum length of 448 tokens. The reported performance metric is average accuracy after training on the last task (AA$_T$, *cf*. Equation (3.10)), with the success metric $a_{T,t}$ denoting the macro F_1 score on task t after incremental training on the T^{th} (*i.e.*, last) task. Each model configuration is trained using three random seeds. Learning rate settings are obtained through hyperparameter tuning, performed on OML-ER as the representative of the meta-learning-based approaches and on SFT for the other methods. The optimal hyperparameters, found based on training on the full dataset without imbalanced sampling, using task order I (*cf*. Table 4.1) and a fixed random seed of 42, were subsequently applied to all models within each respective group. Thus, for the three meta-learning-based CL methods—DRILL, OML-ER, and ANML-ER—we set the inner-loop learning rate to $\alpha = 8 \times 10^{-3}$ and the outer-loop learning rate to $\beta = 1.5 \times 10^{-5}$. The remaining baselines—SFT, ER, and JFT—are trained with a learning rate of 1×10^{-5}.

Each model is trained for a single epoch ($E = 1$), except for JFT, which is trained for two epochs ($E = 2$). We set the support set size per episode uniformly to $s = 5$. Following the rehearsal and evaluation protocol of Holla et al. (2021), we define the rehearsal interval as $R_I = 9\,600$ and the replay ratio as $r = 1\%$. Accordingly, after processing every 9 600 samples from the data stream, we sample 96 instances (*i.e.*, 1% of R_I) from the episodic memory $\mathcal{M}_{\text{E}}$ for rehearsal. The maximum capacity of the episodic memory buffer is fixed at $B = 2\,000$ for all baselines that use experience replay. In accordance with standard meta-learning evaluation practices, we construct five evaluation episodes, each employing the test datasets as query sets.

4.3.3 Baseline Comparison

The main results for the single-epoch CIL setting are summarized in Table 4.3. The persistently poor performance of SFT illustrates the susceptibility of full-body adaptation strategies to catastrophic forgetting when no additional CL mechanisms are used. This finding emphasizes the need to integrate such mechanisms to preserve high performance across tasks.

In the PO setting, which is characterized by a significant underrepresentation of early tasks, DRILL outperforms all baseline methods by an absolute margin of at least 1.5%, with ANML-ER emerging as the second-best performer. Similarly, in the PU setting, where the number of samples per task is reduced by half at each training stage, DRILL achieves an improvement in average model performance of at least 0.7% compared to OML-ER, the next best baseline.

Although the relatively high standard errors warrant cautious interpretation, these performance differences underscore the effectiveness of DRILL in mitigating catas-

Method	PO	PU
JFT	77.4 ± 0.3	77.7 ± 0.2
SFT	23.2 ± 0.7	27.6 ± 1.4
ER	61.9 ± 1.9	48.1 ± 1.8
ANML-ER	63.8 ± 1.9	55.9 ± 3.3
OML-ER	62.9 ± 1.5	58.3 ± 2.8
DRILL	**65.3** ± 1.1	**59.0** ± 1.8

Table 4.3: Performance comparison of experience replay methods in the single-epoch CIL setting. Reported scores are averaged macro F_1 scores after incremental training across four permutations of task orders and three random seeds. Standard errors after ±. **PO** = Progressive Oversampling, **PU** = Progressive Undersampling.

trophic forgetting and suggest its potential to narrow the gap toward the upper performance bound, as demonstrated by the JFT baseline. Notably, DRILL is particularly beneficial in the PO setting, where early tasks are learned from only a few examples. This scenario exacerbates the inherent recency bias in the PL to disproportionately favor recently learned classes. The observation of an absolute 4.4% performance drop for SFT in the PO setting relative to the PU setting further highlights the importance of CL mechanisms that counteract the forgetting of early, underrepresented data in the face of substantial subsequent representational drifts.

4.3.4 Variability Analysis

We investigate the sensitivity of the baseline methods to random parameter initialization and task ordering within the context of CL. To analyze the effect of random initialization, we present in Figure 4.2 the distribution and skewness of performance scores under consistent conditions across all baselines. The results indicate that DRILL achieves a higher median performance and exhibits lower variability compared with all other replay-based baselines. Evidently, the ER method shows particularly higher variability in the PU setting than meta-learning methods. This observation suggests that meta-learning, which is explicitly designed for rapid task-specific adaptation, enables models to robustly learn new tasks with limited samples regardless of specific experimental conditions.

To gain deeper insights into the impact of task ordering on model performance in the single-epoch CIL setting, we present the performance metrics of all baselines for each imbalanced sampling setting and task order configuration (averaged across three random seeds) in Table 4.4. While DRILL outperforms all replay-based CL methods in only three of the eight configurations, it significantly reduces variability—as indicated by the standard error across all experimental setups—by between 24% and 41% compared with ER, ANML-ER, and OML-ER, respectively.

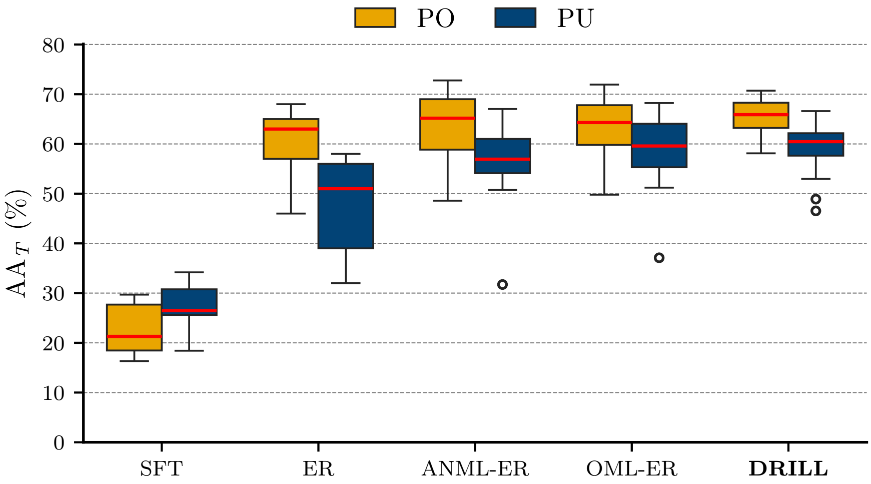


Figure 4.2: Variability comparison of DRILL with other baselines in the single-epoch CIL setting. Performance scores of all comparison models are averaged across four task orderings and three random seeds. Median performance is highlighted in red.

	Order (PO)				Order (PU)				
Method	**I**	**II**	**III**	**IV**	**I**	**II**	**III**	**IV**	∅
JFT	77.9	78.7	76.2	76.7	77.7	76.4	78.3	78.2	77.5 ± 0.4
SFT	17.4	27.6	26.6	21.0	23.7	32.7	28.8	25.0	25.4 ± 1.7
ER	55.3	67.9	58.6	**65.7**	44.2	57.7	53.5	37.0	55.0 ± 3.7
ANML-ER	66.7	**70.5**	55.0	62.9	57.0	58.6	62.7	45.2	59.8 ± 3.1
OML-ER	**70.2**	64.9	52.2	64.4	56.0	**62.0**	**66.5**	48.7	60.6 ± 2.9
DRILL	68.4	68.1	**59.1**	65.5	**61.8**	61.6	62.9	**49.5**	**62.1 ± 2.2**

Table 4.4: Variability of DRILL and other CL baselines across task orders. Each performance score AA$_T$ (%) is reported separately for each of the four task orders and two sampling strategies. ∅ represents the mean performance and standard error across all task orderings, random seeds, and sampling strategies. Standard errors for individual scores are omitted for clarity.

This reduction is particularly critical, as high variability can undermine the reliability of model performance in practical applications. In real-world scenarios, where models often have only a single opportunity to learn from streaming data without the possibility of repeated retraining, consistent performance is essential. Consequently, lower variability provides a more reliable estimate of model performance in an open CL setting.

4.3.5 Impact of Feature Integration Strategies

Given that the original ANML algorithm (Beaulieu et al., 2020), which underpins the ANML-ER method, fuses latent features from two PTMs via multiplication—

	Order (PO)				Order (PU)				
Fusion Strategy	**I**	**II**	**III**	**IV**	**I**	**II**	**III**	**IV**	∅
Multiplication	23.2	36.5	37.1	37.6	58.0	51.7	41.0	**50.4**	41.9 ± 6.1
Concatenation	**68.4**	**68.1**	**59.1**	**65.5**	**61.8**	**61.6**	**62.9**	49.5	**62.1** ± **2.2**

Table 4.5: Comparison of representation fusion strategies. Average performance of DRILL after training is reported for every task order and sampling strategy. ∅ represents the mean performance and standard error across all task orderings, random seeds, and sampling strategies. Standard errors for individual scores are omitted for clarity.

in contrast to the concatenation used with DRILL—we aim to investigate how the integration strategy of latent features affects the performance of our DRILL method. To this end, we compare the average performance after training across various imbalanced sampling strategies and task orderings for integrating features via multiplication and concatenation, as presented in Table 4.5. Specifically, *Multiplication* involves replacing the concatenation $[\boldsymbol{w}_{t,n}^{(i)}\ ;\ h_\phi(\boldsymbol{x}_{t,n})]$ in Equation (4.1) with an element-wise multiplication $\boldsymbol{w}_{t,n}^{(i)} \otimes h_\phi(\boldsymbol{x}_{t,n})$, while *Concatenation* corresponds to the original DRILL method described in Section 4.2.

The results show that the use of multiplicative fusion between the prototypical features retrieved from the semantic memory $\mathcal{M}_\mathrm{S}$ and the features extracted by the PTM results in a significant absolute performance reduction of 20.2%, as well as a doubling of performance variability, compared with the original DRILL method. Across all task orders and sampling strategies, multiplicative integration proves inferior to feature concatenation in all configurations except one.

One potential explanation for this performance degradation is rooted in the design of ANML, where *gating layers* preceding the fusion operation are learned in a supervised manner. This supervised learning of gating parameters helps stabilize the multiplicative interactions between features. In contrast, our method utilizes SOINN+, an unsupervised algorithm for semantic memory, which does not allow for the optimization of parameters to stabilize the multiplicative gating and thus the overall learning trajectory. As a result, the multiplicative fusion mechanism adversely affects CL performance by introducing instability and divergence to the training process. Therefore, concatenation proves to be a more suitable strategy for feature integration when utilizing unsupervised self-organizing networks for prototype sampling.

4.4 Discussion

The results in the single-epoch CIL setting demonstrate that DRILL, as a dual-memory strategy employing a self-organizing network architecture for prototype augmentation, has the potential to enhance performance and reduce sensitivity to model parameter initialization and task ordering. DRILL exhibits particular effectiveness in preserving task-specific knowledge when early classes (or tasks) are significantly underrepresented. This underscores its strength in long-term retention under low-data conditions. We attribute this effectiveness to the conservative approach of the SOINN+ algorithm to deleting nodes that, although significantly different from others, possess high *trustworthiness* in accurately representing specific data subsets. Consequently, a small set of nodes representing heavily underrepresented classes are more likely to be maintained and frequently retrieved from the SOINN+ network, thereby being used to guiding the PL towards confidently predicting these classes.

A well-recognized limitation in combining self-organizing neural networks with feed-forward networks for supervised learning is their tendency to capture the entire evolution of hidden representations in the feature space, including obsolete knowledge. This can lead to a deterioration in the quality of nodes representing subsets of the training data when updates are made to the PTM. The DRILL architecture largely mitigates this issue by freezing the PTM parameters during inner-loop optimization and by retrieving the most frequently activated SOINN+ nodes instead of the BMUs directly, a strategy that is less sensitive to representational drift. Freezing the PTM parameters ensures a more stable latent data distribution over time and preserves the integrity of the learned features. Additionally, selecting the most frequently activated nodes increases the likelihood that neural units aligned with the current input distribution are considered high-quality class representatives, thereby improving the model's predictive accuracy.

An important consideration is the computational overhead introduced by integrating self-organizing networks into the DRILL framework. While SOINN+ effectively preserves the topological structure of the feature space and aids in mitigating catastrophic forgetting, it adds complexity in terms of memory usage and processing time. This issue becomes particularly evident when dealing with high-dimensional data or a great diversity of input feature vectors, as the topology mapping network may grow substantially to accommodate new patterns. Moreover, unsupervised self-organizing networks do not support batched learning and must compute pairwise distances between each newly observed input feature vector and all existing network nodes. In our experiments, this resulted in the SOINN+ comprising approximately between 10% and 15% of the number of input samples, which amounts to several thousand nodes. Addressing this challenge is therefore critical for the practical deployment of DRILL in real-world scenarios where computational resources are often constrained.

To mitigate these issues, future work could explore strategies to optimize the efficiency of the semantic memory module. Potential solutions include implementing more aggressive node pruning techniques to remove redundant or less informative nodes, employing approximate nearest neighbor search algorithms to expedite BMU retrieval and neighborhood adjustment, or incorporating parallel processing to handle the increased computational load. Balancing the benefits of topological mapping for CL performance with the practical constraints of computational resources is essential not only for scaling DRILL to more extensive and diverse CL scenarios, but also for its practical applicability in resource-constrained environments.

4.5 Chapter Summary

In this chapter, we introduced DRILL, a novel dual-memory strategy that integrates unsupervised self-organizing networks with PTMs to mitigate catastrophic forgetting in CL settings. DRILL preserves topological relationships within the feature space and augments latent features with representative class prototypes to stabilize the training of the prediction layer and increase the model's robustness against forgetting old classes. We conducted experiments in imbalanced data settings, a common challenge in real-world applications, and demonstrated that DRILL is particularly effective when underrepresented classes appear early in the training stream. DRILL considerably reduces the sensitivity to task order and random initialization, thereby providing more reliable performance estimates in real-world CL scenarios compared with prior CL methods.

Recognizing the computational overhead associated with extracting prototypes from a topological mapping network for feature augmentation, the subsequent chapter investigates leveraging the rich intra-layer features of PTMs for prototype construction. This approach aims to achieve similar benefits in mitigating forgetting while enhancing scalability and efficiency.

Chapter 5

Prototyping with Intermediate Features

CL with foundation models necessitates robust methodologies that capitalize on the rich and versatile representations derived from large-scale pretraining. These methods must also exhibit the flexibility to sequentially learn novel tasks that extend beyond the original pretraining distribution. Existing approaches, including our DRILL method (*cf*. Chapter 4), predominantly focus on disentangling high-level instance- or class-specific features at the final representation layer, thereby overlooking the potential of intermediate representations to capture low- and mid-level features. Such features are more invariant to domain shifts and can enhance generalization across tasks.

In this chapter, we introduce LayUP, a novel prototype-based approach to CL that leverages second-order feature statistics from multiple intermediate layers of a PTM. By harnessing the hierarchical representations within PTMs, LayUP captures richer and more discriminative features that improve learning efficiency and robustness. Our method is conceptually simple, does not require access to prior data, and operates seamlessly with any foundation model, making it both adaptable and practical for a wide range of applications. We conduct extensive empirical evaluations, in which we demonstrate that LayUP exceeds state-of-the-art performance on four out of seven CIL benchmarks, all three DIL benchmarks, and six out of seven OCL benchmarks. Notably, it achieves these results while significantly reducing memory and computational requirements compared with existing methods. These findings underscore that fully exploiting the representational capacities of PTMs in CL settings extends well beyond their final embeddings.[1]

5.1 Motivation

The advent of foundation models has sparked significant interest in developing CL methods that leverage the powerful representations obtained through large-

[1] The source code is made available at `https://github.com/ky-ah/LayUP`.

scale self-supervised pretraining. CL with PTMs operates under the premise that a comprehensive all-purpose feature extractor not only facilitates robust knowledge transfer but also offers enhanced resilience against catastrophic forgetting during incremental adaptation to downstream tasks. This shift aligns with the broader trend in machine learning toward utilizing PTMs to improve performance and efficiency across a variety of tasks (*cf*. Section 2.1).

In Section 3.3, we outlined three principal strategies adopted by recent CL approaches with PTMs: (i) carefully fine-tuning the parameters of the PTM (full-body adaptation methods), (ii) keeping the PTM parameters fixed while learning a small set of additional parameters (PEFT methods), or (iii) extracting prototypical vectors from PTMs without further fine-tuning (class-prototype methods). Notably, all three strategies—except for early CL methods with PTMs employing full-body adaptation (*cf*. Section 3.3.1)—share the beneficial property of not relying on rehearsal from historical data to perform well, unlike many of the top-performing CL strategies used for models trained from scratch. As described in Section 3.2.2, exemplar-free retention is a desirable characteristic of CL methods, as it circumvents issues related to data privacy, regulatory compliance, and memory overhead.

Despite these advancements, determining the optimal strategy to balance the trade-offs among effectively utilizing pretrained features, ensuring resource efficiency, and maintaining robustness to domain shifts in CL settings remains an open question. As discussed in Section 3.3.3, both full-body adaptation methods and PEFT methods update a linear probe through iterative gradient descent during training, making them susceptible to catastrophic forgetting and task-recency bias. DRILL (*cf*. Chapter 4), although prototype-based, preserves this gradient-driven probing strategy and therefore inherits the same weaknesses. Conversely, class-prototype methods that classify by directly comparing extracted features with stored prototypes entirely bypass extra fine-tuning. This property makes them intrinsically more robust to forgetting and markedly more resource-efficient than the previous strategies.

However, existing class-prototype methods for CL typically rely solely on features extracted from the last layer of the PTM for prototype construction. This approach may be insufficient, especially as the divergence between the pretraining and fine-tuning domains increases. High-level features from the final layer are often tailored to the pretraining tasks and may not generalize well to new domains, leading to reduced class separability in the target domain (Yosinski et al., 2014). Consequently, a key challenge is to develop a prototype-based classifier that holistically leverages representations from multiple layers of the pretrained feature extractor, thereby capturing a richer set of features while maintaining low memory and computational overhead. In this chapter, we seek to answer the question of whether augmenting last-layer features with intermediate features extracted from a PTM can enhance class separability and improve performance in prototype-based CL.

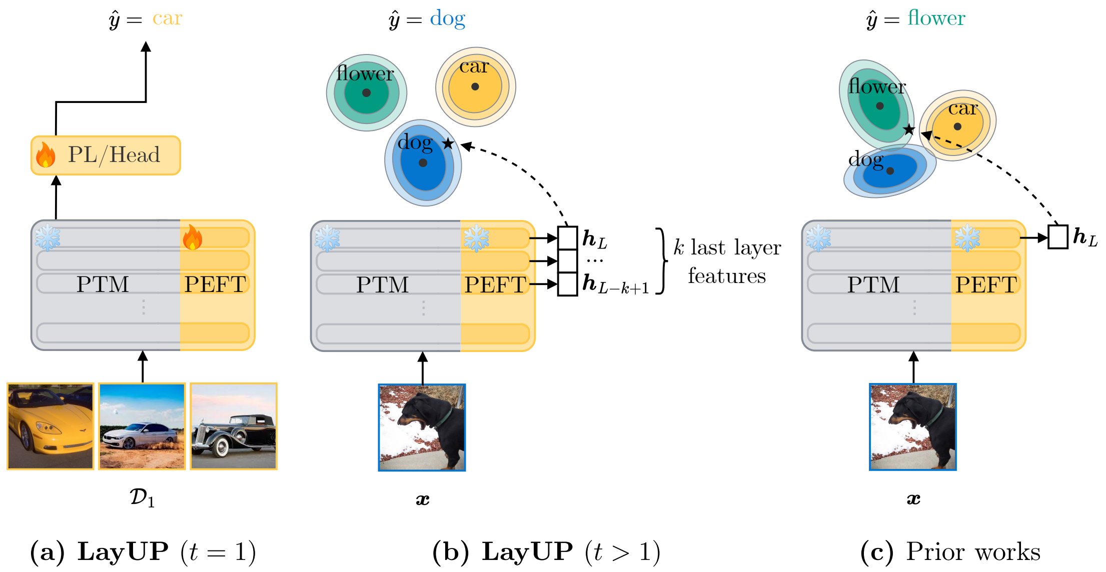
In this context, we hypothesize that aggregating features from multiple layers to construct a classifier based on first-order (class prototypes) and second-order (Gram matrix) feature statistics allows for a more effective utilization of representations by leveraging information from multiple levels of abstraction. More specifically, this multi-layer approach captures both high-level semantic information and lower-level features that are crucial for generalization across domains. Our strategy is inspired by prior research in Neural Style Transfer (Gatys et al., 2016; Jing et al., 2020), which involves applying the artistic style of one image to the content of another—a problem that can be interpreted through the lens of domain adaptation (Li et al., 2017). Neural Style Transfer requires the disentanglement of image information into content and style components (*e.g.*, textures, patterns, colors). In this context, content information is captured through feature activations at different layers (analogous to computing multi-layer class prototypes), while style information is encapsulated by the correlations between features across layers (equivalent to computing multi-layer Gram matrices).

To this end, we propose Multi-**Lay**er **U**niversal **P**rototypes (**LayUP**), a novel class-prototype method for CL grounded in the strategy of disentangling style and content information across multiple encoder layers of a PTM. By leveraging both first-order and second-order statistics from multiple layers, LayUP captures rich and fine-grained information about images to perform a more informed classification that is less sensitive to domain shifts. We also explore various PEFT strategies to further refine the intermediate representations obtained by LayUP and allow it to bridge the gap between source and target domains. Across multiple CL benchmarks and learning settings, our method improves performance and narrows the gap to the upper bound by as much as 80% (Stanford Cars-196 in the CIL setting; *cf.* Section 5.3.3) compared with the next best baseline, while also reducing its memory and compute requirements by up to 81% and 90%, respectively (*cf.* Section 5.2.3). Beyond these improvements, LayUP serves as a versatile plug-in to enhance existing class-prototype methods, consistently delivering absolute performance gains of up to 31.1% (*cf.* Section 5.3.8).

Our contributions are threefold:

(1) We examine in detail the CL strategy using PTMs and demonstrate why it benefits from directly extracting intermediate representations for classification. We further demonstrate the advantages of leveraging cross-correlations of features within and between multiple layers to decorrelate class prototypes.

(2) Building upon our insights, we introduce LayUP, a novel class-prototype approach to CL that capitalizes on second-order statistics of features from multiple layers of a PTM. Drawing inspiration from prior works, we experiment with various PEFT strategies and extend our approach towards FSA. LayUP is conceptually straightforward, resource-efficient in terms of memory and computation, and integrates seamlessly with any PTM.



(a) **LayUP** $(t = 1)$ (b) **LayUP** $(t > 1)$ (c) Prior works

Figure 5.1: Overview of LayUP. (a) LayUP incorporates adaptation of PEFT parameters during training on the first task to bridge the domain gap from pretraining to fine-tuning domains. **(b)** It enhances the last-layer representations of PTMs by incorporating features from intermediate layers, resulting in more accurately calibrated similarity measures for prototype-based classification and greater robustness to domain gaps in the CL setting. **(c)** Prior works on CL with PTMs, on the other hand, use only final representation layer features for classification. Stars ($\star$) denote extracted features and circles ($\bullet$) denote class prototypes.

(3) We report performance improvements with two pretrained ViT-B/16 models on the majority of benchmarks for the CIL and DIL settings, as well as in the challenging OCL setting. We demonstrate that LayUP is particularly effective under significant distributional shifts and in low-data regimes.

This chapter is organized as follows: we establish the theoretical foundation for our LayUP method in Section 5.2. Subsequently, we detail our training procedures and present an extensive experimental evaluation in Section 5.3. We discuss our findings and their implications in Section 5.4, and conclude the chapter in Section 5.5.

5.2 Proposed Method: LayUP

An overview of our proposed LayUP method is depicted in Figure 5.1. The model architecture integrates a PTM h, PEFT parameters φ, and a PL g_ψ parameterized by ψ. We adopt the formulation of the CL problem introduced in Section 3.2.1 and employ the definitions of prototype-based classifiers as described in Section 3.3.3.

We argue that combining (i) the augmentation of last-layer representations with hierarchical multi-layer features and (ii) the decorrelation of these multi-layer features—which capture diverse image properties such as content and style—via a Gram matrix transformation enhances robustness to domain shifts and thereby improves generalizability to downstream continual tasks. To operationalize this, we integrate embeddings from multiple layers through concatenation into a ridge regression estimator, as defined in Equation (3.23). Let

$$h_{-k:}(\boldsymbol{x}) = [h_{L-k+1}(\boldsymbol{x}); \ldots ; h_{L-1}(\boldsymbol{x}); h_L(\boldsymbol{x})] \tag{5.1}$$

denote the concatenated input features of some input sample $\boldsymbol{x} \in \mathcal{D}_{\text{test}}$, extracted from the last k layers of the PTM $h(\cdot)$. Such features can be, *e.g.*, [CLS] token embeddings of a transformer encoder or flattened feature maps of a CNN or ResNet encoder. This leads to a modified estimator for the predicted label:

$$\hat{y} = \underset{y \in \{1,\ldots,C\}}{\arg\max}\, s_y, \quad s_y := h_{-k:}(\boldsymbol{x})^T \ \left(\boldsymbol{G} + \lambda \boldsymbol{I}\right)^{-1} \ \boldsymbol{c}_y, \tag{5.2}$$

where $\boldsymbol{I}$ is the $d_{(k)} \times d_{(k)}$-dimensional identity matrix with $d_{(k)} = (d_{L-k+1} + \cdots + d_{L-1} + d_L)$, $\lambda \geq 0$ is the regression parameter, and $\boldsymbol{G}$ is the multi-layer Gram matrix, defined as

$$\boldsymbol{G} = \sum_{t=1}^{T} \sum_{n=1}^{N_t} h_{-k:}(\boldsymbol{x}_{t,n}) \otimes h_{-k:}(\boldsymbol{x}_{t,n}) \tag{5.3}$$

Similarly to the ridge classifier defined in Equation (3.23), the LayUP estimator in Equation (5.2) corresponds to the closed-form solution to the regularized ridge regression problem when considering $\boldsymbol{H} \in \mathbb{R}^{N \times d_{(k)}}$ as the matrix of concatenated row vector features and expressing $\boldsymbol{G}$ as $\boldsymbol{H}^\top \boldsymbol{H}$. In other words, when not employing CL, optimizing a linear probe with weight matrix $\boldsymbol{W}$ using SGD with mean squared error loss and weight decay equal to λ yields a loss lower bounded by that achieved by directly computing the closed-form solution

$$\boldsymbol{W} = (\boldsymbol{H}^\top \boldsymbol{H} + \lambda \boldsymbol{I})^{-1} \boldsymbol{H}^\top \boldsymbol{Y}, \tag{5.4}$$

where $\boldsymbol{Y}$ is the target label matrix of the training dataset $\mathcal{D}$.

5.2.1 Prototyping from Second-order Intermediate Features

To motivate our approach, we show that intermediate representations encode rich class-specific statistics that can be exploited for prototype-based classification. Using the split CIFAR-100 dataset (Krizhevsky, 2009) and ImageNet-R (Hendrycks et al., 2021a), we construct prototype-based classifiers at each intermediate layer $l \in \{1, \ldots, L-1\}$ of two ViT models by applying Equation (3.23). We then quantify the percentage of classes for which the classifier at layer l outperforms the classifier at the final layer L (with $L = 12$ for ViT-B/16).

As shown in Figure 5.2, prototype-based classifiers derived from the last five intermediate layers outperform the final-layer classifier for up to 32% of the classes

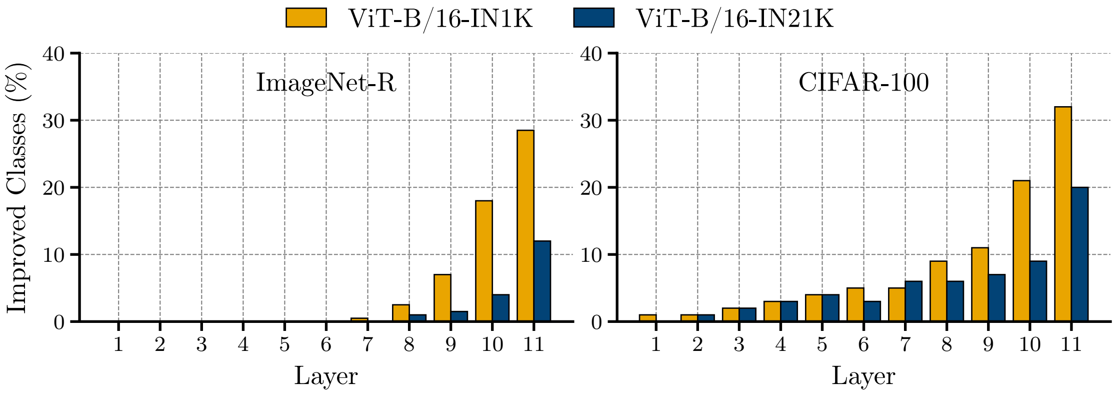


Figure 5.2: Classification performance of intermediate layers. Comparison for two different ViT-B/16 models and split ImageNet-R (*left*) and CIFAR-100 (*right*) datasets. For each intermediate layer $l \in \{1, \ldots, L-1\}$ (where $L = 12$ denotes the final representation layer), the bars represent the percentage of classes for which a classifier, utilizing Equation (3.23) at layer l, surpasses the accuracy of the classifier at the L^{th} layer.

across both datasets. Notably, this advantage is more pronounced for the ViT-B/16-IN1K model—which is initially pretrained on ImageNet-21K (21 000 classes) and subsequently fine-tuned on ImageNet-1K (1 000 classes)—than for the ViT-B/16-IN21K model, which is solely trained on ImageNet-21K. This is likely because fine-tuning on ImageNet-1K further sharpens and aligns intermediate features around the 1 000 natural object categories that also underlie ImageNet-R and many CIFAR-100 classes, yielding more compact, class-discriminative clusters that a prototype-based rule can exploit more effectively than the broader, less specialized features of the model trained only on ImageNet-21K. The results given suggest that intermediate representations capture hierarchical and complementary features that allow for effective class discrimination across varying levels of abstraction.

We identify two intuitive methods to integrate intermediate representations into the class-prototype classifier: (1) averaging over k separate classifiers according to Equation (3.23), and (2) concatenating representations from the last k layers to obtain shared class prototypes according to Equation (5.2). Figure 5.3 presents average accuracies for two split datasets across different values of k. Notably, for each k, transforming concatenated features via Gram matrix inversion, as defined by strategy (2), consistently outperforms the averaging of separate classifiers, as per strategy (1). This finding suggests that the correlations across layers, encapsulated in the shared Gram matrix (*cf*. Equation (5.3)), provide significant additional information that enhances class separability. These results highlight the importance of capturing inter-layer dependencies to improve classification performance, indicating that jointly integrating representations is more effective than treating each layer independently.

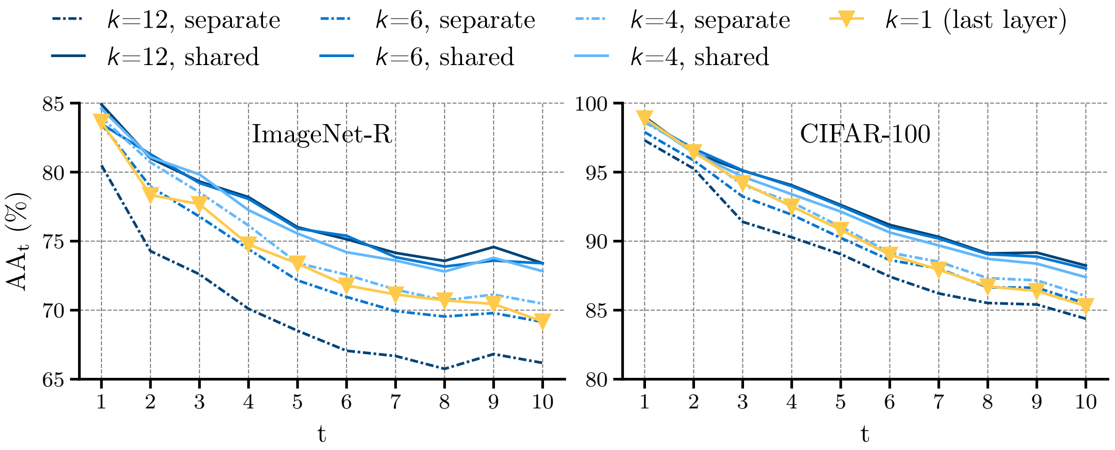


Figure 5.3: Comparison of strategies to integrate intermediate representations. LayUP implementations for different values of k using a *shared* representation and Gram matrix as in Equation (5.2) versus averaging over *separate* ridge (or Gram) classifiers for each layer using Equation (3.23). Results are reported as average accuracy scores over CL training on split ImageNet-R (*left*) and CIFAR-100 (*right*) datasets following phase B of Algorithm 3.

Figure 5.3 demonstrates that increasing k, *i.e.*, the number of layers from which representations are concatenated, generally enhances the performance of a prototype-based classifier. In particular, the performance improvement when increasing k from $k = 4$ to $k = 6$ is considerably more pronounced than the improvement from $k = 6$ to $k = 12$, suggesting diminishing returns beyond a certain depth. Given that larger k values incur higher computational and memory costs—owing to the expanded dimensions of the Gram matrix $\boldsymbol{G}$ and class prototypes $\boldsymbol{c}_y$—choosing $k =$ 6 strikes a practical balance between performance and resource efficiency. Although concatenating features from multiple consecutive layers inherently consumes more memory than using a final-layer classifier or averaging over layer-wise classifiers, we will demonstrate in Section 5.2.3 that our method nevertheless remains significantly more memory- and computation-efficient than other competitive class-prototype methods for CL.

Interestingly, the performance of the final-layer classifier ($k = 1$) is approximately equivalent to that achieved by averaging over separate classifiers for $k = 4$ and $k = 6$. This finding challenges the notion that combining class embeddings from multiple consecutive layers—each individually proficient at predicting specific classes (*cf*. Figure 5.2)—introduces noise that degrades performance when averaged. Instead, it suggests that the final layer encapsulates the most discriminative features, while averaging over multiple layers may not yield additional performance gains, potentially due to redundancy or the incorporation of less discriminative features from earlier layers.

5.2.2 Combination with Parameter-efficient Model Adaptation

A notable advantage of class-prototype methods for CL is their inherent compatibility with PEFT and full-body adaptation techniques. However, updating the feature extractor parameters typically induces shifts in the latent representations, which can misalign the stored class prototypes. Therefore, any strategy aimed at adapting the latent representations must be meticulously designed to suit the specific class-prototype method in use. Recent studies (Panos et al., 2023; Zhou et al., 2024) have demonstrated that applying FSA on the initial dataset $\mathcal{D}_1$ effectively bridges the domain gap between PTM pretraining and downstream fine-tuning data. They further recommend accumulating the class prototypes after the FSA phase, thereby preserving the integrity of the prototypes while enabling the model to assimilate new information and maintaining full compatibility with established CL paradigms.

To preserve the robust general-purpose features of the PTM and reduce computational complexity, we apply FSA to an additional set of learnable PEFT parameters while keeping the PTM parameters fixed throughout the training process. To update these PEFT parameters during FSA, we train a linear PL with the number of output neurons equal to the number of unique labels in $\mathcal{D}_1$ using the Adam optimizer (Kingma and Ba, 2015) and cross-entropy loss. After completing this initial training phase on $\mathcal{D}_1$, the PL is discarded. In the subsequent CL phase, both the PTM and PEFT parameters remain frozen; only $\boldsymbol{G}$ and $\boldsymbol{c}_y$ are updated. Consistent with previous studies (Zhou et al., 2024; McDonnell et al., 2023), we experiment with VPT (Jia et al., 2022), SSF (Lian et al., 2022), and AdaptFormer (Chen et al., 2022) as PEFT methods.[2]

Algorithm 3 details the pseudocode for LayUP training within the CIL setting, and Algorithm 4 outlines the corresponding testing procedure. A primary advantage of our method is that both the Gram matrix $\boldsymbol{G}$ and the class prototypes $\boldsymbol{c}_y$ are updated incrementally on a per-sample basis. Consequently, Equation (5.2) can be recomputed after processing each new data point, thereby ensuring compatibility with online or streaming learning scenarios.

To determine the optimal value for the regression parameter λ, we employ a cross-validation approach as follows: for each task t, we stratify the training data by their ground truth labels and conduct a four-fold cross-validation. In each fold, we temporarily update $\boldsymbol{G}$ and $\boldsymbol{c}_y$ for all labels $y \in \mathcal{Y}_t$ and evaluate the classifier's performance on the validation fold across a range of λ values.[3] After aggregating the accuracies across all folds for each λ, we select the parameter that yields the highest mean accuracy. This selected λ is then used to update $\boldsymbol{G}$ and $\boldsymbol{c}_y$ for all $y \in \mathcal{Y}_t$ using the complete training dataset for task t. We then proceed to task

[2] More details of the PEFT methods used in this chapter can be found in Section 2.1.4.

[3] The search space for the regression parameter λ is defined as follows: $\lambda = 10^x$, $x \in \{-8, -4, -2, -1, -0.5, 0, 0.5, 1, 2, 4, 8\}$.

Algorithm 3: LayUP Training

```
Input: PEFT parameters $\varphi$, PL parameters $\psi$, maximum layer depth $k$
# Initialization
$\boldsymbol{G} \leftarrow \mathbf{0} \in \mathbb{R}^{d_{(k)} \times d_{(k)}}$
$\boldsymbol{c}_y \leftarrow \mathbf{0} \in \mathbb{R}^{d_{(k)}} \ \forall \ y \in \mathcal{Y}$
# Phase A: First Session Adaptation
for every sample $(\boldsymbol{x}, y) \in \mathcal{D}_1$ do
    collect $h_L(\boldsymbol{x})$
    update $\varphi$ and $\psi$ using Adam
end
# Phase B: Continual Learning with LayUP
for task $t = 1, \dots, T$ do
    for every sample $(\boldsymbol{x}, y) \in \mathcal{D}_t$ do
        collect $h_{-k:}(\boldsymbol{x})$ using Equation (5.1)
        $\boldsymbol{G} \leftarrow \boldsymbol{G} + h_{-k:}(\boldsymbol{x}) \otimes h_{-k:}(\boldsymbol{x})$
        $\boldsymbol{c}_y \leftarrow \boldsymbol{c}_y + h_{-k:}(\boldsymbol{x})$
    end
    optimize $\lambda$ and compute $(\boldsymbol{G} + \lambda \boldsymbol{I})^{-1}$
end
```

Algorithm 4: LayUP Testing

```
Input: trained PEFT parameters $\varphi$ (if FSA applied), maximum layer
       depth $k$, Gram matrix $\boldsymbol{G}$, class prototypes $\boldsymbol{c}_y \ \forall \ y \in \mathcal{Y}$
for every sample $\boldsymbol{x} \in \mathcal{D}_{\text{test}}$ do
    collect $h_{-k:}(\boldsymbol{x})$ using Equation (5.1)
    predict $\hat{y}$ using Equation (5.2)
end
```

$t + 1$ to repeat the optimization process. Importantly, this procedure adheres to the constraints of rehearsal-free CL, as it does not require access to data from previous tasks at time t.

As the cross-validation procedure for selecting λ involves repeated passes over the data of the current task t, it is not compatible with strict OCL scenarios, where only a single pass over the data is permissible. Thus, for the OCL comparison, we omit the FSA stage (phase A in Algorithm 3) and utilize a fixed regularization parameter λ. This way, we maintain adherence to the online learning paradigm while enabling a fair and consistent comparison across all baseline methods.

5.2.3 Memory and Runtime Complexity Comparisons

We assess the memory and runtime overhead of our LayUP method relative to three competitive CL approaches for PTMs: APER (Zhou et al., 2024), SLCA (Zhang et al., 2023), and RanPAC (McDonnell et al., 2023). We assume a typical number of classes, $C = 200$, and AdaptFormer as the representative PEFT method. We exclude the memory footprint of the ViT backbone and the PL parameters from our comparison, as these components are common across all methods. Additionally, we do not consider the memory requirements of the class prototypes, since they are utilized by every approach and contribute negligibly to the overall storage demands.

LayUP stores approximately one million PEFT parameters and an additional multi-layer Gram matrix in memory. The size of this matrix depends on the choice of k and is calculated as $d^2_{(k)} = (d_{L-k+1} + \cdots + d_{L-1} + d_L)^2$. In typical transformer encoders, including ViT-B/16, each layer outputs tokens of the same dimensionality; specifically, $d_l = 768$ for every layer l. With $k = 6$, as used in the main experiments in this chapter, the Gram matrix stores approximately 21 million values. In comparison, RanPAC, using a random projection dimension of $M = 10\,000$ as in the original paper, updates a Gram matrix of size $M^2 = 100$ million entries and requires approximately 11 million additional parameters for the random projection layer and PEFT parameters. SLCA stores C additional covariance matrices of size d_L^2, resulting in a total of approximately 118 million entries. Finally, APER stores a second copy of the feature extractor (*i.e.*, the ViT-B/16 model), necessitating 84 million additional parameters. Assuming comparable memory costs for model parameters and matrix entries, our method reduces the additional *memory requirements* by approximately 81%, 82%, and 75% compared with RanPAC, SLCA, and APER, respectively.

Considering *runtime complexity during training*, APER updates all parameters of a ViT-B/16 model during the first task adaptation phase, resulting in significant computational overhead due to full fine-tuning of approximately 84 million parameters. SLCA conducts full-body adaptation of the ViT-B/16 backbone during slow learning and performs Cholesky decomposition (Cholesky, 1924) on class-wise covariance matrices for pseudo-rehearsal, which requires $C \cdot d_L^3 \approx 10^{11}$ operations. In contrast, RanPAC and LayUP update only PEFT parameters during training and perform matrix inversion during inference, albeit with differently constructed and sized Gram matrices. For $k = 6$, the matrix inversion requires $(d_{L-k+1} + \cdots + d_{L-1} + d_L)^3 \approx 10^{11}$ operations for LayUP and $10\,000^3 = 10^{12}$ for RanPAC. Thus, LayUP reduces RanPAC's *runtime complexity during inference* by up to 90%.

5.3 Experiments

In the following, we conduct a series of experiments to evaluate the performance of our LayUP method across various CL settings. We start with an overview of the benchmarks and training details, followed by empirical comparisons of our approach against recent strong baselines in CIL, DIL, and OCL settings. Subsequently, we perform ablation studies to investigate the key components of our method and analyze how different configurations of the last k layers affect classifier

Setting	Dataset	T	N_{train}	N_{test}	C
CIL/OCL	**CIFAR**	10	50 000	10 000	100
	IN-R	10	24 000	6 000	200
	IN-A	10	6 056	1 419	200
	CUB	10	9 465	2 323	200
	OB	10	89 668	5 983	300
	VTAB	5	1 796	8 619	50
	Cars	10	8 144	8 041	196
DIL	**CDDB-H**	5	16 068	10 706	2
	S-DomainNet	6	20 624	176 743	345
	IN-R (D)	15	24 000	6 000	200

Table 5.1: Summary of the image classification datasets. For the baseline comparisons in Section 5.3.3, we report the CL settings and the number of tasks (T), along with the number of training samples (N_{train}), validation/test samples (N_{test}), and classes (C). In the CIL and OCL settings, the datasets are partitioned by classes. Although IN-R was originally proposed for domain generalization tasks, to the best of our knowledge, it has not been previously utilized in the DIL setting.

performance. Furthermore, we examine the range of classes that benefit from hidden features and conclude our empirical analysis by assessing LayUP as a plug-in for other class-prototype CL methods.

5.3.1 Benchmarks

For the CIL and OCL settings, we employ seven representative split datasets: CIFAR-100 (CIFAR; Krizhevsky, 2009), ImageNet-R (IN-R; Hendrycks et al., 2021a), ImageNet-A (IN-A; Hendrycks et al., 2021b), CUB-200 (CUB; Wah et al., 2011), OmniBenchmark (OB; Zhang et al., 2022), Visual Task Adaptation Benchmark (VTAB; Zhai et al., 2020), and Stanford Cars-196 (Cars; Krause et al., 2013). Consistent with established practices, we partition all datasets into $T = 10$ tasks for baseline comparisons in the CIL and OCL settings, except for VTAB, which is conventionally divided into $T = 5$ tasks. Additionally, we report results for task counts of $T = 5$ and $T = 20$ in the variability analysis in Section 5.3.5. For the DIL comparisons, baselines are trained on the Continual Deepfake Detection Benchmark Hard (CDDB-H; Li et al., 2023a) with $T = 5$, a subsampled version of DomainNet (S-DomainNet; Peng et al., 2019) with $T = 6$, and a domain-incremental formulation of ImageNet-R (IN-R (D)) with $T = 15$. A summary of the datasets used in the baseline comparisons is shown in Table 5.1.

Datasets for the CIL and OCL settings. CIFAR comprises 100 classes of natural images across various domains and topics, closely aligning with the distribution of pretraining datasets like ImageNet-1K and ImageNet-21K. IN-R includes image categories overlapping with ImageNet-1K but features out-of-distribution samples, such as challenging examples or newly collected data in various styles. Similarly, IN-A shares categories with ImageNet-1K, but contains real-world adversarially filtered images designed to mislead classifiers pretrained on ImageNet. The CUB dataset is a specialized collection of labeled images of 200 bird species, capturing a wide range of poses and backgrounds. OB serves as a compact benchmark intended to assess the generalization capabilities of PTMs across semantic super-concepts or realms, including images from 300 categories representing distinct concepts. The VTAB benchmark, as utilized in our work, consists of five datasets—Resisc45 (Cheng et al., 2017), DTD (Cimpoi et al., 2014), Pets (Parkhi et al., 2012), EuroSAT (Helber et al., 2019), and Flowers (Nilsback and Zisserman, 2006)—and is commonly split into $T = 5$ tasks in the CL setting. The Cars dataset comprises images of cars belonging to 196 distinct combinations of make and model.

Datasets for the DIL setting. The CDDB-H benchmark is designed to evaluate deepfake detection models under continuously evolving attack scenarios, including methods such as *GauGAN*, *BigGAN*, *Wild*, *WhichFaceIsReal*, and *SAN*. DomainNet is a multi-source domain adaptation benchmark encompassing six image style domains: *real*, *quickdraw*, *painting*, *sketch*, *infograph*, and *clipart*. Due to the extensive size of the original DomainNet training set (exceeding 400 000 samples), we employ S-DomainNet, a subsampled version that restricts the number of training samples to ten per each of the 345 classes across the six domains while preserving the original validation set. Finally, IN-R (D) encompasses a sequence of fifteen image style domains: *sketch*, *art*, *cartoon*, *deviantart*, *embroidery*, *graffiti*, *graphic*, *misc*, *origami*, *painting*, *sculpture*, *sticker*, *tattoo*, *toy*, and *videogame*.

5.3.2 Training Details

Consistent with prior studies (Wang et al., 2022e,f; Zhou et al., 2024; Zhang et al., 2023; McDonnell et al., 2023), we conduct experiments using two ViT-B/16 models as PTM h, both consisting of 12 transformer encoder layers with a token embedding dimensionality of $d_1 = d_2 = \cdots = d_L = 768$. The first ViT model (**ViT-B/16-IN21K**) is pretrained in a supervised manner on the full ImageNet-21K dataset (Deng et al., 2009). The second ViT model (**ViT-B/16-IN1K**) is additionally fine-tuned on the ImageNet-1K dataset (Krizhevsky et al., 2012). During FSA in both CIL and DIL settings, the PEFT parameters φ are trained for $E = 20$ epochs using a batch size of $|b| = 48$. The PL g_ψ, which is used only during the FSA stage, is implemented as a fully-connected linear layer followed by a softmax activation function to generate class probabilities. We utilize the Adam optimizer (Kingma and Ba, 2015) with default parameters and employ a cosine annealing learning rate schedule starting from an initial learning rate of 3×10^{-2}. For VPT, we train five prompt tokens, and for AdaptFormer, we use a bottleneck dimension of 16. All baseline methods are rehearsal-free and trained on the same ViT backbones as described above.

Data augmentation is uniformly applied to all training datasets. Specifically, during training, images are randomly cropped and resized to 224×224 pixels. The cropping scale is randomly selected between 70% and 100% of the original dimensions, with the aspect ratio constrained between 3:4 and 4:3. Additionally, horizontal flipping is performed with a probability of 0.5, and brightness, contrast, saturation, and hue are each randomly adjusted within a 10% range. For the CIFAR-100 dataset, due to its original resolution of 32×32 pixels, images are directly resized to 224×224 pixels without cropping. During inference, images across all datasets are simply resized to 224×224 pixels without any further augmentation or preprocessing.

Based on the observations detailed in Section 5.2.1, we found that utilizing features from the latter half of the network layers for prototype construction offers a balance between performance and computational efficiency. Therefore, all experiments are conducted with $k = 6$, unless specified otherwise. For a comprehensive comparison of performance across different values of k for all datasets, refer to Section 5.3.6.

Throughout this chapter and unless otherwise specified, all tabulated results report the average accuracy after learning the final task (AA_T; *cf*. Equation (3.10)) using a fixed random seed of 1993 and the optimal combination of PEFT method and ViT backbone. This experiment design and the choice of random seed are consistent with previous studies (McDonnell et al., 2023; Zhou et al., 2024). Furthermore, we present the average accuracy and forgetting after each task t in Section 5.3.5.

We compare LayUP with **L2P** (Wang et al., 2022f), **DualPrompt** (Wang et al., 2022e), and **CODA-Prompt** (Smith et al., 2023a) as representative prompt learning methods, **SLCA** (Zhang et al., 2023) as full-body adaptation method, and **APER** (Zhou et al., 2024), **NMC+FSA** (Panos et al., 2023), and **RanPAC** (McDonnell et al., 2023) as class-prototype methods for CL. For more information about the prompt learning, full-body adaptation, and class-prototype baselines used in this chapter, refer to Sections 3.3.1 to 3.3.3.

5.3.3 Baseline Comparison

In the CIL setting, we compare LayUP with several prompt learning methods, fine-tuning approaches, and class-prototype methods for CL. We also report results for Joint Fine-Tuning (**JFT**) of the full backbone (PTM+PL) as well as of the PL only. It is noteworthy that both JFT baselines do not necessarily serve as upper bound for CL model performance in this case, as LayUP is a class-prototype method and does not update the backbone nor uses the linear probe for prediction during inference. As shown in Table 5.2, LayUP surpasses all baselines on four out of the seven split datasets.

The four datasets in which LayUP consistently outperforms all baseline models—IN-R, IN-A, VTAB, and Cars—exhibit two distinct characteristics. First, they have a high domain gap relative to the pretraining ImageNet domain, as detailed in Section 5.3.6. Second, with the exception of IN-R, they contain significantly less

Method	CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
JFT (PTM+PL)	93.6	86.6	71.0	91.1	80.3	92.5	83.7
JFT (PL)	87.9	71.2	56.4	89.1	78.8	90.4	66.4
L2P	84.6	72.5	42.5	65.2	64.7	77.1	38.2
DualPrompt	81.3	71.0	45.4	68.5	65.5	81.2	40.1
CODA-Prompt	86.3	75.5	74.5	79.5	68.7	87.4	43.2
NMC+FSA	87.8	70.1	49.7	85.4	73.4	88.2	40.5
APER	87.6	72.3	52.6	87.1	74.3	84.3	41.4
SLCA	91.5	77.0	59.8*	84.7	73.1*	89.2*	67.7
RanPAC	**92.2**	78.1	61.8	**90.3**	**79.9**	92.6	77.7
LayUP	92.0	**81.4**	**62.2**	88.1	77.6	**93.3**	**82.5**
Ablations							
$k = 1$	90.8	78.8	60.4	86.7	72.0	92.4	74.9
w/o FSA	88.7	73.1	57.7	86.2	77.0	92.6	78.6
$k = 1$, w/o FSA	86.5	69.4	55.4	85.4	70.7	91.6	69.1
LayNMC	88.1	59.3	50.0	82.5	68.9	86.5	40.0
NMC	83.4	61.2	49.3	85.1	73.1	88.4	37.7

Table 5.2: Performance comparison of prompt learning, backbone fine-tuning, and class-prototype methods in the CIL setting. Results are taken from McDonnell et al. (2023) except results for SLCA marked with (*), which are reproduced using the officially released code.

training data compared with CIFAR, OB, and CUB (*cf*. Section 5.3.1). The first characteristic confirms our initial hypothesis that intermediate representations are more domain-invariant and consequently more robust to large distributional shifts from the source to the target domain. The second indicates that, especially in the low-data regime, LayUP constructs class prototypes and decision boundaries that generalize well, even when the amount of training data is scarce compared with the data used for ViT pretraining. It should be noted that RanPAC, which is the only baseline that LayUP does not consistently outperform, is considerably more expensive in terms of both memory and computation (*cf*. Section 5.2.3).

The results for the DIL setting are reported in Table 5.3. We compare our method with several strong class-prototype methods for CL and, similar to the CIL comparison, we also report results for the joint fine-tuning of the full backbone or only the linear probe. LayUP consistently outperforms all baselines. These results confirm that our method effectively utilizes the domain-invariant information contained in the low- and mid-level intermediate features of the PTM, thereby enhancing its robustness to domain shifts.

Additionally, we are interested in evaluating the performance of our method in the challenging OCL setting, where only a single pass over the continual data stream is allowed ($E = 1$) and updates have to be performed on a per-sample basis,

Method	**CDDB-H**	**S-DomainNet**	**IN-R (D)**
JFT (PTM+PL)	92.9	62.8	86.6
JFT (PL)	74.0	57.2	71.2
NMC+FSA	49.8	44.0	76.3
APER	70.7	–	–
RanPAC	86.2	58.8	77.2
LayUP	**88.1**	**59.4**	**80.0**
Ablations			
$k = 1$	84.0	54.2	77.8
w/o FSA	86.7	58.3	73.3
$k = 1$, w/o FSA	77.7	53.7	69.0
LayNMC	61.4	34.4	59.1
NMC	57.6	43.5	64.1

Table 5.3: Performance comparison of class-prototype methods in the DIL setting. Results for APER and RanPAC on CDDB-H are taken from McDonnell et al. (2023).

thus $|b| = 1$. All baselines are class-prototype methods for CL on a frozen PTM (thus we omit FSA stages for NMC, RanPAC, and LayUP), except for Sequential Fine-Tuning (**SFT**), where we update the parameters of the pretrained ViT via cross-entropy loss and the Adam optimizer during a single epoch. Note that NMC exhibits identical behavior in both the CIL and OCL settings, as it uses each training sample only once for running-mean calculation, consequently producing the same outcomes as the ablation baseline in Table 5.2.

Given that the choice of the ridge regression parameter λ, as utilized in RanPAC and our method, necessitates prior knowledge about the downstream continual data—a requirement unmet in realistic streaming learning settings—we opt for comparison with two simplified versions of Gram matrix inversion $(\boldsymbol{G} + \lambda \boldsymbol{I})^{-1}$ (*cf.* Equations (3.23) and (5.2)). In the first variant, we completely omit λ-based regularization, setting $\lambda = 0$, so the Gram matrix is inverted without regularization (*i.e.*, $\boldsymbol{G}^{-1}$). In the second variant, as used in Panos et al. (2023), we set $\lambda = 1$, resulting in the inversion of the identity-regularized Gram matrix (*i.e.*, $(\boldsymbol{G} + \boldsymbol{I})^{-1}$).

As indicated in Table 5.4, unrestricted fine-tuning of the backbone in the OCL setting is detrimental to the generalizability of the pretrained embeddings and leads to forgetting and low performance. Consequently, the SFT baseline is outperformed by class-prototype methods for most benchmarks. With the exception of VTAB, LayUP is largely robust to missing regularization ($\lambda = 0$) and maintains high performance on all benchmarks when regularization is set to $\lambda = 1$. In contrast, RanPAC exhibits high variability in performance across benchmarks, occasionally resulting in near-zero accuracy. This outcome is likely due to the high-dimensional

Method	CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
SFT	12.8	5.3	6.9	5.1	3.3	7.3	10.9
NMC	83.4	61.2	49.3	85.1	73.1	88.4	37.7
RanPAC$_{\lambda=0}$	**88.7**	70.6	1.4	0.9	76.9	9.3	0.6
RanPAC$_{\lambda=1}$	**88.7**	70.6	0.9	0.7	76.9	9.0	0.6
LayUP$_{\lambda=0}$	**88.7**	**73.4**	40.9	86.4	**77.0**	10.4	66.3
LayUP$_{\lambda=1}$	**88.7**	**73.4**	51.6	**86.9**	**77.0**	**92.9**	**77.5**
Ablations							
$k = 1, \lambda = 0$	86.5	69.6	**55.6**	85.4	72.8	92.2	69.2
$k = 1, \lambda = 1$	86.5	69.6	**55.6**	85.4	76.3	92.3	69.2

Table 5.4: Performance comparison of class-prototype methods in the OCL setting. Results for RanPAC and NMC are reproduced according to the officially released code in McDonnell et al. (2023).

prototypes and Gram matrix in RanPAC, obtained after random projections, overfitting the training data, which consequently hampers their ability to generalize. LayUP's superior robustness in the online learning setting makes it a more favorable approach for CL scenarios where the length of the input stream and the nature of the data might not be known in advance.

5.3.4 Ablation Study

The purpose of the following ablation study is to demonstrate the advantages of leveraging multi-layer representations over solely utilizing features from the final representation layer (*i.e.*, choosing $k = 1$) for class-prototype construction in CL. As LayUP integrates multi-layer representations with second-order statistics via Gram matrix inversion as well as FSA in the CIL and DIL settings, we consider three configurations for ablating multi-layer representations: (i) in the standard setting with both FSA and second-order statistics (LayUP vs. $k = 1$), (ii) excluding the FSA stage, *i.e.*, omitting phase A in Algorithm 3 (w/o FSA vs. $k = 1$, w/o FSA), and (iii) excluding FSA and second-order statistics, which corresponds to traditional nearest-mean classifiers (LayNMC vs. NMC). Results for the baselines constructed for settings (i)-(iii) can be found in the bottom sections of Tables 5.2 and 5.3. For (i), enriching the final layer with intermediate features consistently results in absolute performance gains ranging from 1.2% to 7.6% (CIL) and 2.2% to 5.2% (DIL). For (ii), leveraging multi-layer features yields absolute accuracy improvements of between 0.8% and 9.5% (CIL) and between 4.3% and 9.0% (DIL).

In contrast, configuration (iii) fails to provide consistent evidence that LayNMC—conceptually analogous to averaging predictions from individual per-layer classifiers—outperforms the NMC baseline. These findings corroborate our preliminary results in Section 5.2.1 and likely stem from the fact that intermediate representations exhibit reduced discriminative capacity compared to the final-layer features while receiving equivalent weighting in the final prediction due to the lack of a decorrelation mechanism. Consequently, incorporating intermediate features yields minimal benefit when Gram matrix inversion—designed to decorrelate the feature representations—is omitted.

In the lower section of Table 5.4, additional comparisons are presented between ablated versions of LayUP and prototypes that use only final-layer representations ($k = 1$) in the OCL setting, evaluated at two selected values of the regularization parameter λ. In general, LayUP outperforms its ablated variants; however, it exhibits reduced performance on low-data benchmarks such as IN-A and VTAB, particularly when regularization is omitted (*i.e.*, $\lambda = 0$). This decline likely results from the increased dimensionality of both the class prototypes and the Gram matrix—resulting from the incorporation of intermediate representations—which elevates the risk of overfitting. Consequently, in the absence of regularization and with limited training data, the inclusion of multi-layer features may prove counterproductive, whereas in all other cases, they enhance classification accuracy by between 0.6% and 8.3%.

5.3.5 Variability Analysis

In the following, we perform a comprehensive sensitivity analysis of LayUP within the context of CL. Specifically, we examine the impact of stochastic variability in parameter initialization and task ordering, the selection of PEFT strategies and PTMs, the number of intermediate layers utilized for feature extraction during prototype construction, and the influence of varying task counts.

Effect of random initialization. Since the choice of random seed influences both the task order and the initialization of the PEFT and PL parameters, we compute the average accuracy and forgetting, along with the standard error, for all seven datasets in the CIL setting, using seeds 1993 to 1997. The results are shown in Figure 5.4. We observe negligible differences in final performance after the last task across different seeds, indicating that LayUP is robust to variations in task order and PEFT parameter initialization. The greatest variability is observed in the VTAB benchmark, which comprises $T = 5$ tasks from five different datasets with high domain variability. Consequently, the task order—and hence the specific set of classes to which the representations are adapted during the initial session—has a more pronounced effect on the model's generalization capabilities during CL. As anticipated, the average forgetting increases with the introduction of more tasks, since the model must discriminate among a larger number of classes during inference.

Effect of PEFT methods and PTMs. Building upon previous studies (Zhou et al., 2024; McDonnell et al., 2023), we investigate various PEFT methods and ViT-B/16 models as PTMs within the LayUP framework. Since the benefits of

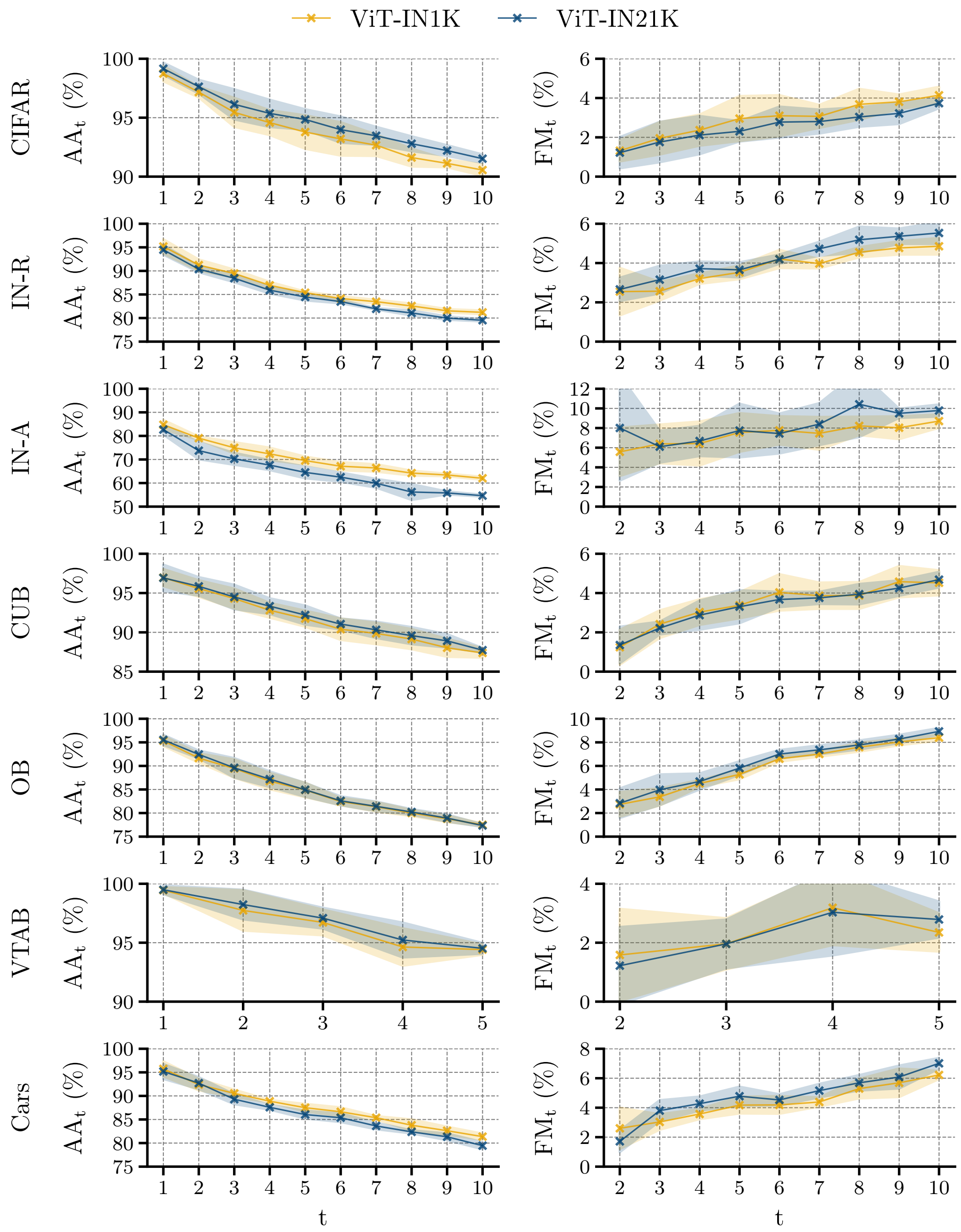


Figure 5.4: Variability of LayUP across random seeds. Average accuracy (*left*) and forgetting measure (*right*) in the CIL setting are reported for each ViT-B/16 model after FSA with AdaptFormer as the PEFT method. Results are reported for seeds 1993 to 1997 to ensure reproducibility, with the resulting standard error indicated by the shaded area.

additional fine-tuning can vary depending on the characteristics of the downstream domain (Panos et al., 2023), our experiments aim to elucidate these effects. The results are presented in Figure 5.5.

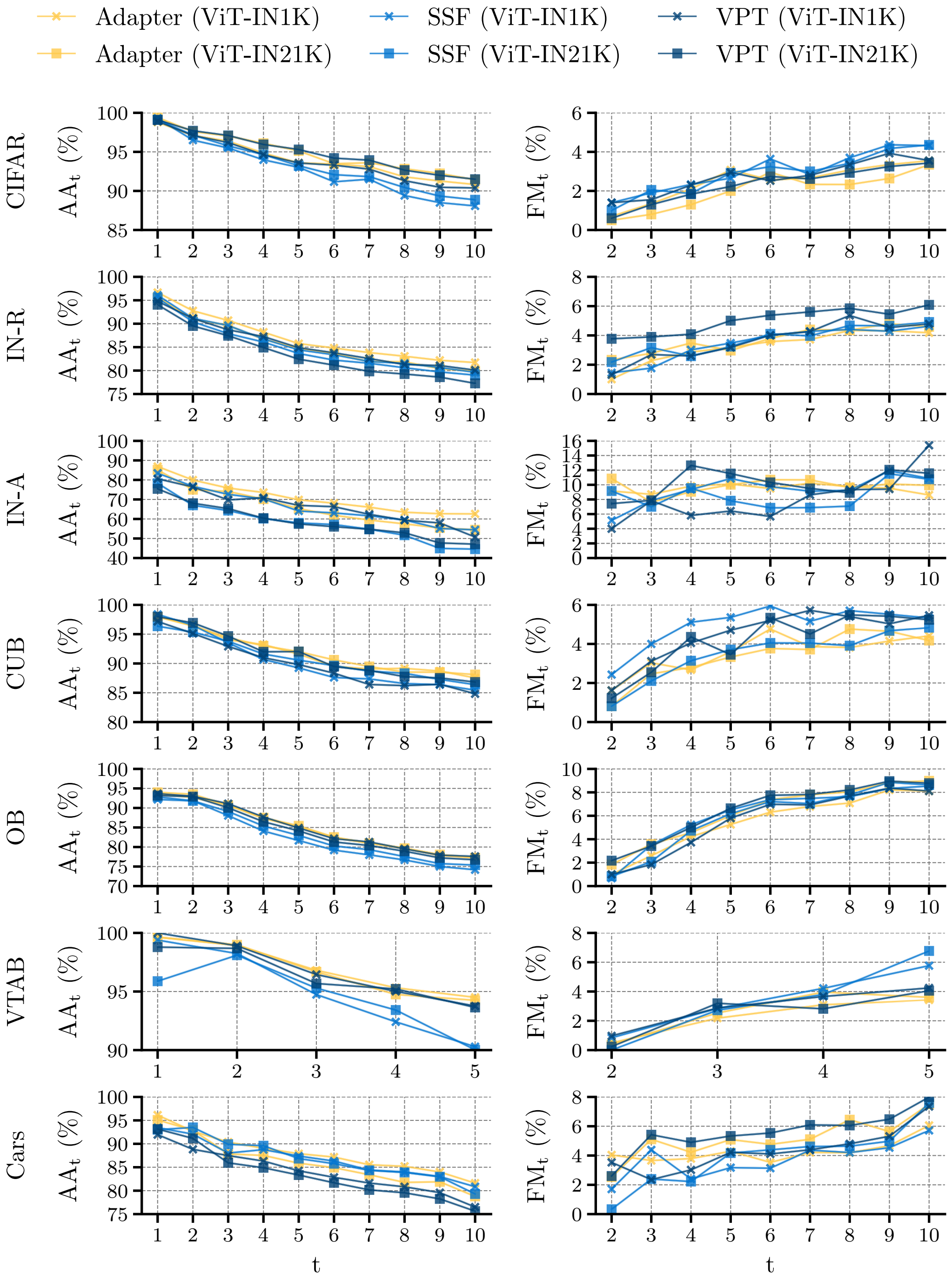


Figure 5.5: Variability of LayUP across parameter-efficient fine-tuning methods. Performance comparison of different PEFT methods in the CIL setting (AdaptFormer, SSF, and VPT) and ViT-B/16 models.

Our findings indicate that LayUP performance shows overall high robustness towards different PEFT methods and ViT models, expect for LayUP with SSF as PEFT method fine-tuned on VTAB, which depicts a relatively steep drop in performance and high forgetting rates during CL with both ViT models. Moreover, we observe that AdaptFormer consistently demonstrates stable performance regardless of the pretraining strategy and generally outperforms both VPT and SSF across all datasets, except for the split OB dataset. No clear pattern emerges regarding whether VPT is superior to SSF or vice versa. Furthermore, the choice of the ViT pretraining strategy does not significantly influence the effectiveness of any PEFT method. Finally, there is no single combination of PEFT method and ViT model that universally outperforms all other variants. These results corroborate the findings of Panos et al. (2023) and underscore the importance of considering a broad range of pretraining schemes for PTMs and fine-tuning strategies to develop high-performing class-prototype methods for CL.

Effect of representation depth k. To analyze the learning behavior over time in the CIL setting among different choices of k for prototype construction, we plot average accuracy and forgetting for $k = 1$, $k = 6$, and $k = 12$. Specifically, $k = 1$ corresponds to classification based solely on the final layer (here, layer 12) representations of the PTM, as done in prior work, $k = 6$ involves concatenating multi-layer features from the latter half of the network layers (layers 7 through 12), and $k = 12$ uses concatenated features from all layers (layers 1 through 12) of the pretrained ViT model for classification.

As shown in Figure 5.6, classification performance based solely on the last layer representations ($k = 1$) is inferior to that based on multi-layer representations utilized in LayUP, resulting in both lower accuracy and higher forgetting rates. Notably, there is no significant performance difference between $k = 6$ and $k = 12$, suggesting that the early layers (layers 1 through 6) of the model do not contribute substantial information to the classification process, nor do they adversely affect it. Considering the trade-off between performance gains and the associated memory and computational costs, these results corroborate the findings of the preliminary analysis in Section 5.2.1 and support the selection of $k = 6$—as used in our baseline comparison in Section 5.3.3—as an effective and efficient choice.[4]

Effect of task count T. To evaluate whether the benefits of multi-layer representations remain consistent across different numbers of tasks, we compare the average accuracy after training (AA_T) across six datasets. We use three different task counts, $T \in \{5, 10, 20\}$, and three different choices for the number of last layers used in class prototype generation, $k \in \{1, 6, 12\}$. The results are presented in Table 5.5.

We hypothesize that increasing the number of tasks leads to decreased performance because fewer data points and classes can be learned during FSA and thus be used to bridge the gap between the source and target domain. This trend is evident in our results: for nearly all datasets and values of k, performance improves as the task count decreases (*e.g.*, a 4.4% improvement when reducing the number of tasks from $T = 20$ to $T = 5$ on IN-A with $k = 6$). Furthermore, across all task counts

[4] We further deepen the analysis of the choice of k on LayUP performance in Section 5.3.6.

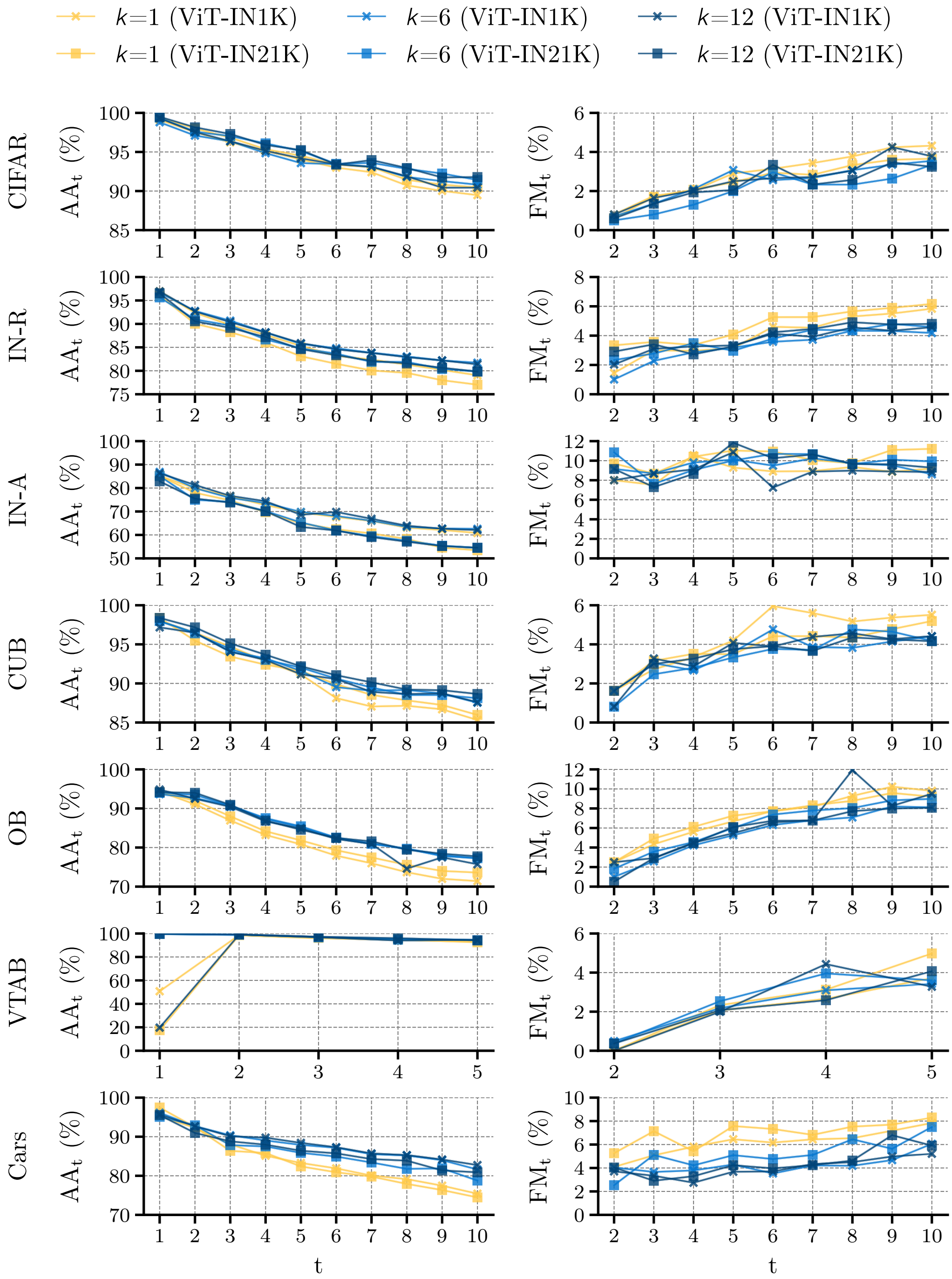


Figure 5.6: Variability of LayUP across maximum representation depths. Comparison of different choices of k last layers for prototype construction ($k \in \{1, 6, 12\}$) and ViT-B/16 models in the CIL setting after first session training with AdaptFormer as PEFT method.

and datasets, using multi-layer representations ($k = 6$ and $k = 12$) yields higher performance than using only the final-layer representations ($k = 1$), underscoring

k	T	CIFAR	IN-R	IN-A	CUB	OB	Cars
1	5	89.5	80.4	61.9	85.0	72.4	75.1
	10	89.2	79.2	60.8	85.3	71.6	75.5
	20	88.4	77.3	57.1	83.6	72.2	75.6
6	5	91.8	82.8	64.0	87.1	77.4	81.7
	10	91.0	81.2	62.2	87.3	77.5	82.5
	20	88.8	79.9	59.6	85.9	77.4	82.2
12	5	91.6	83.2	64.0	87.4	78.3	82.1
	10	90.8	80.8	62.8	88.0	77.8	82.3
	20	88.9	80.2	60.0	86.0	78.1	82.8

Table 5.5: Variability of LayUP across task counts. Performance comparison of different task counts T for $k = 1$ (prototype construction from last layer only), $k = 6$ (prototype construction from the second half of the network layers, as utilized in Section 5.3), and $k = 12$ (prototype construction from all network layers). Reported scores are for the CIL setting, FSA with AdaptFormer as PEFT method, and ViT-B/16-IN1K as PTM. The VTAB benchmark is omitted from the comparison due to its fixed number of datasets from which tasks are constructed ($T = 5$).

the efficacy of incorporating multi-layer representations in prototype construction with LayUP.

5.3.6 Analysis of Multi-layer Representation Depth

To provide more detailed insights into the impact of multi-layer representation depth on LayUP performance, we plot in Table 5.6 the average accuracy after training for all possible choices of k using a ViT-B/16-IN1K backbone and AdaptFormer as PEFT method. The final-layer classifier ($k = 1$) exhibits the lowest performance compared to all other k values across all datasets, except for the split VTAB dataset. Conversely, LayUP configurations with medium or large k values almost consistently outperform those with small k values, demonstrating the broad advantages of leveraging representations from multiple intermediate layers for class-prototype construction.

While the optimal choice of k varies between datasets, performance gains generally diminish with increasing k values. For instance, the average absolute accuracy gain from choosing $k = 6$ over $k = 1$ is 2.7%, whereas it is only 0.4% when choosing $k = 12$ over $k = 6$. This confirms the findings made in Section 5.2.1 and Section 5.3.5, suggesting that a value of $k = 6$ generally suffices to obtain substantial performance gains over conventional last-layer class-prototype methods, while simultaneously incurring considerably lower computational and memory overhead compared with $k = 12$.

k	**CIFAR**	**IN-R**	**IN-A**	**CUB**	**OB**	**VTAB**	**Cars**
1	89.2	79.2	60.8	85.3	71.6	92.7	75.5
2	90.1	79.9	62.2	86.9	73.9	90.7	78.7
3	90.2	80.7	62.5	87.3	74.5	93.2	81.4
4	90.4	81.1	62.0	87.6	76.7	93.1	81.7
5	90.7	81.5	62.8	87.3	77.0	93.0	82.4
6	91.0	81.2	62.2	87.3	77.5	92.2	82.5
7	90.9	**81.6**	**63.4**	87.5	77.7	93.4	**82.6**
8	90.8	81.4	62.8	87.7	77.2	92.1	82.3
9	91.0	81.5	62.1	87.4	**78.3**	92.0	**82.6**
10	90.9	81.4	**63.4**	87.9	77.4	**94.0**	81.5
11	**91.1**	81.5	62.0	87.6	78.1	93.7	81.4
12	90.8	80.8	62.8	**88.0**	77.8	93.9	82.3

Table 5.6: Comparison of maximum representation depths. LayUP performance for different values of k for a pretrained ViT-B/16-IN1K and FSA with AdaptFormer. The 1st, 2nd, and 3rd highest scores and the 1st, 2nd, and 3rd lowest scores are highlighted.

We further aim to determine whether prior knowledge about the characteristics of downstream CL data can inform the optimal choice of maximum multi-layer representation depth with respect to overall performance. Specifically, we identify the value of k that maximizes accuracy on the test set after running Algorithm 3 with AdaptFormer as PEFT method for each split dataset used in Section 5.3.3 and for each pretrained ViT model. We plot the highest performing k per dataset against three measures: (i) the degree of domain gap to the pretraining ImageNet domain, measured by *Maximum Mean Discrepancy* (MMD); (ii) the intra-class similarity, measured by the average pairwise cosine similarity between training image feature vectors of the same class; and (iii) the inter-class similarity, measured by the average pairwise cosine similarity between image feature vectors of disjoint classes.

For (i), we follow Panos et al. (2023) and use the *mini*ImageNet dataset (60 000 images) as a proxy for the ImageNet-1K (1.3 million images) and ImageNet-21K (14 million images) pretraining datasets to reduce computational overhead when calculating pairwise distances. A large MMD value indicates that two datasets have significantly different statistical properties, thus signifying a large domain gap between the source and target domains.

As shown in Figure 5.7, there is no clear relationship between MMD and the optimal choice of k. This suggests that the degree of domain gap from the source domain of a pretrained model is not a reliable indicator for determining the multi-layer representation depth that maximizes performance. However, a positive trend is observed between the optimal k and both intra-class and inter-class similarity.

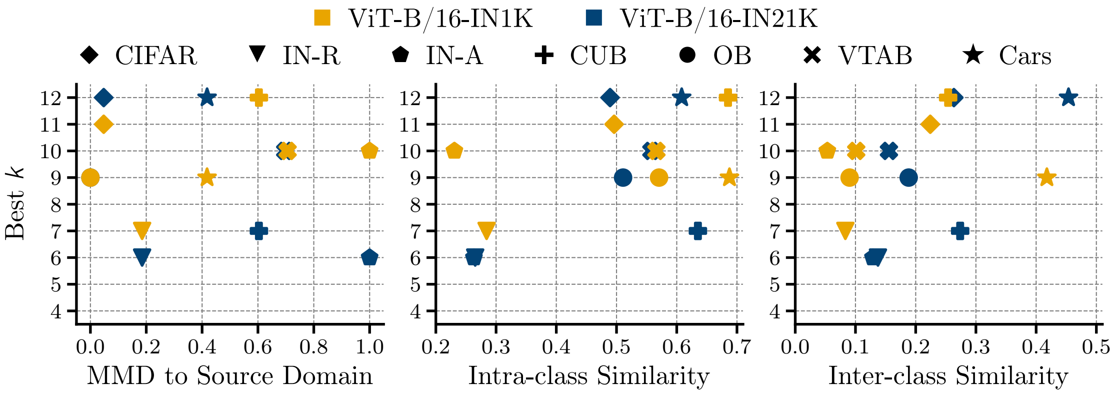


Figure 5.7: Choice of maximum representation depth for different dataset characteristics. Choice of k that yields highest accuracy vs. normalized MMD between each dataset and the ImageNet pretraining domain, represented by *mini*ImageNet dataset (*left*), intra-class similarity (*center*), and inter-class similarity (*right*).

Firstly, datasets with high intra-class and inter-class similarity (*e.g.*, Cars or CUB) tend to benefit from employing higher values of k. In these fine-grained classification tasks, instances within the same class and those across different classes share substantial visual similarity. As a result, integrating features from a larger number of layers allows the model to capture the subtle distinctions necessary for accurate discrimination.

Secondly, datasets characterized by medium intra-class similarity and low inter-class similarity (*e.g.*, VTAB, OB, or CIFAR) benefit from medium to high values of k. These datasets often comprise multiple specialized natural image datasets from distinct domains. The low inter-class similarity arises because classes are drawn from diverse domains, making them inherently different from one another, while the intra-class similarity is moderate due to variability within classes. Incorporating features from several layers can enhance performance, although the marginal benefit diminishes compared with fine-grained datasets.

Lastly, datasets with low intra-class and inter-class similarity (*e.g.*, IN-A or IN-R) benefit from medium values of k. These datasets frequently consist of atypical or stylized images that deviate significantly from standard natural images, resulting in an unsystematic distribution in the feature space. The low intra-class similarity indicates considerable variability among instances within the same class, while the low inter-class similarity suggests that distinct classes lack shared, salient features. In such scenarios, aggregating features from a high number of layers—including those capturing low-level information—can introduce redundant or noisy information, potentially degrading the discriminative quality of the representation. Therefore, a moderate value of k strikes an optimal balance between capturing sufficient information for class separation and mitigating noise.

Although Figure 5.7 demonstrates that the highest performance in most configurations is attained by concatenating features from the majority of layers in the PTM, performance disparities between different k values can occasionally be minor (*cf*. Table 5.6) and may fluctuate with varying random initializations. Moreover, as previously noted, performance improvements tend to plateau with larger k. At the same time, increasing k incurs higher memory consumption and greater runtime complexity during inference. Consequently, selecting an optimal k demands a careful trade-off analysis, taking into account the specific requirements and constraints of the application.

5.3.7 Versatility of Multi-layer Prototypes

We aim to assess the versatility of multi-layer prototypes for classification, specifically, the breadth of classes benefiting from LayUP. To this end, we report the proportion of classes for each dataset whose test accuracy scores are higher, equal, or lower with LayUP compared with the final-layer ridge classifier (*cf*. Equation (3.23)). Beyond merely counting the classes that are better predicted by either method, we also examine the extent of the benefit one method has over the other per improved class. Consequently, we calculate the average difference in accuracy scores between LayUP and the final-layer classifier ($k = 1$), and vice versa, for each class that shows improvement.

Figure 5.8 illustrates that a higher percentage of classes (between 18% and 73%) is more effectively classified upon introducing intermediate representations to class-prototype construction. Although the difference in the number of improved classes is not as pronounced for IN-A and CUB compared with all other benchmarks, the relative difference in accuracy per improved class between LayUP and the final representation layer classifier reaches as high as 39% for IN-A and 21% for CUB. This indicates that for these two datasets, the introduction of intermediate representations significantly benefits a few classes rather than slightly benefiting a large number of classes, as observed with the other benchmarks. Despite the variability with respect to the number of classes per dataset that benefit from LayUP and the extent of the benefit, our results demonstrate the universal applicability of multi-layer prototypes across a broad spectrum of tasks, classes, and domains.

5.3.8 Combination with Other Prototyping Methods

We aim to investigate the effectiveness of using intermediate representations, as employed in LayUP, as a plug-in to enhance other prototype-based methods for CL. Specifically, we incorporate multi-layer representations into two class-prototype methods: APER and RanPAC. Details on these methods can be found in Zhou et al. (2024) and McDonnell et al. (2023). For LayUP combined with APER, we aggregate concatenated features from the last k layers of both the pretrained ViT and the "first session adapted" ViT to construct prototypes. During inference, instead of using cosine similarity matching as done in APER (*cf*. Equation (3.21)),

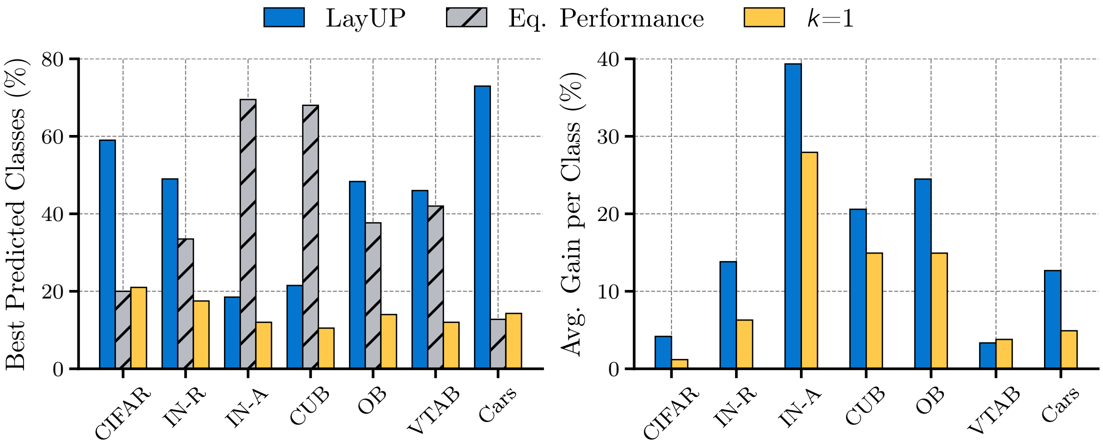


Figure 5.8: LayUP vs. classification from the final representation layer. *Left:* Percentage of classes per dataset that are better classified with LayUP (with $k = 6$) compared with the last layer ridge classifier (equivalent to LayUP with $k = 1$). *Right:* Difference of absolute accuracy per improved class between LayUP and the final-layer classifier ($k = 1$), and vice versa.

we apply Gram matrix inversion following the approach in LayUP (*cf*. Equation (5.2)). This decision is informed by the results from Section 5.3.4, which suggest that multi-layer representations are insufficiently effective for prototype matching when relying solely on first-order feature statistics. For LayUP combined with RanPAC, concatenated features from the last k layers of the first session adapted ViT are fed to the random projection layer, which has an output dimensionality of $M = 10\,000$.

The results for the combinations with APER and RanPAC are depicted in Tables 5.7a and 5.7b, respectively. Integrating LayUP with other class-prototype methods for CL consistently improves performance across all datasets. However, the magnitude of these improvements varies, with the CUB and CIFAR datasets showing the least pronounced enhancements ($\uparrow$ 0.4% for LayUP integrated with RanPAC) and the Cars dataset exhibiting the most substantial gains ($\uparrow$ 31.1% for LayUP integrated with APER). Although the optimal value of k for maximizing performance differs between datasets, overall performance does not vary significantly with different values of k. This suggests that a small, resource-efficient choice of k is sufficient to enhance existing class-prototype methods when integrated with LayUP.

5.4 Discussion

Our empirical results across CIL, DIL, and OCL settings consistently demonstrate that LayUP, our novel class-prototype method that augments last-layer features with first- and second-order intermediate features extracted from a PTM, significantly improves performance in prototype-based CL across diverse datasets.

Method	**CIFAR**	**IN-R**	**IN-A**	**CUB**	**OB**	**VTAB**	**Cars**
APER	87.6	72.3	50.5	87.1	74.3	84.3	51.3
w/ LayUP ($k = 4$)	91.0	81.8	63.3	87.5	78.1	93.9	**82.4**
w/ LayUP ($k = 6$)	**91.4**	81.7	**63.8**	**87.8**	**79.0**	**94.7**	82.2
w/ LayUP ($k = 12$)	91.3	**82.0**	**63.8**	87.5	78.1	94.6	81.7

(a) LayUP + APER (Zhou et al., 2024)

Method	**CIFAR**	**IN-R**	**IN-A**	**CUB**	**OB**	**VTAB**	**Cars**
RanPAC	90.7	78.0	58.2	88.5	76.9	92.6	67.5
w/ LayUP ($k = 4$)	**91.1**	82.4	61.5	88.7	**79.0**	94.0	81.9
w/ LayUP ($k = 6$)	**91.1**	**82.8**	63.4	88.6	78.3	93.4	**82.3**
w/ LayUP ($k = 12$)	**91.1**	82.4	**64.1**	**88.9**	78.3	**94.1**	81.7

(b) LayUP + RanPAC (McDonnell et al., 2023)

Table 5.7: Combination of LayUP with other class-prototype methods. The reported average accuracy scores (%) pertain to the CIL setting as detailed in Section 5.3.2. All baselines were trained using ViT-B/16-IN1K as the PTM and AdaptFormer as the PEFT method for FSA. Results for RanPAC and APER, which were not reported for this specific configuration, have been reproduced using their officially released code repositories. Note that the results for RanPAC and APER may differ from those presented in Table 5.2, as the latter reports the best among *all* configurations of PEFT methods and ViT models.

Incorporating intermediate representations into prototype construction offers multifaceted benefits; by aggregating features from multiple layers, LayUP captures a richer hierarchy of representations that encompass both low-level and high-level features. This holistic approach enhances class separability, particularly in scenarios involving significant domain shifts between pretraining and fine-tuning tasks. Our findings indicate that LayUP consistently outperforms methods relying exclusively on last-layer features, especially on datasets characterized by substantial distributional gaps and in low-data regimes. This suggests that intermediate features are more invariant to domain shifts and contribute to better generalization across tasks.

Furthermore, our analysis reveals that LayUP is universally beneficial across a wide spectrum of classes and datasets. The method effectively improves classification performance for a significant proportion of classes. Its ability to integrate intermediate features into existing class-prototype methods underscores LayUP's versatility. When combined with methods such as APER and RanPAC, LayUP yields substantial performance gains without incurring significant computational overhead, highlighting its potential as a plug-in enhancement for various CL approaches.

However, LayUP is not without limitations. One of the primary challenges lies in determining the optimal number of layers (denoted by k) to include in the prototype construction. Although increasing k generally leads to performance improvements, it also results in higher memory consumption and computational complexity during inference. This trade-off necessitates a careful balance between resource efficiency and performance gains. Moreover, in the OCL setting, the absence of regularization when using multi-layer features can lead to overfitting, particularly when training data are limited. This underscores the need for appropriate regularization strategies to mitigate the risks associated with higher-dimensional feature spaces.

Furthermore, the optimal choice of k may depend on the specific characteristics of the dataset, such as intra-class and inter-class similarity. Our analysis suggests that while higher values of k are beneficial for datasets with high intra-class similarity, they may not confer the same advantages for datasets with low intra-class similarity. This variability indicates that a one-size-fits-all approach may not be optimal, and adaptive strategies for selecting k based on dataset properties could enhance LayUP's effectiveness.

Finally, LayUP is currently tailored for classification problems where tasks involve comparing inputs to class prototypes. In multimodal learning tasks, which require the integration of data from different modalities such as text, images, and audio, the challenges extend beyond comparative classification. These tasks often involve implicit reasoning, complex associations, and the fusion of heterogeneous information sources. The direct application of LayUP's prototype-based approach may not suffice in scenarios where understanding context, semantics, and multimodal interactions is crucial.

This limitation points to an important avenue for future research: adapting the principles of LayUP to multimodal CL settings by developing methods that can effectively leverage intermediate representations across different modalities while addressing the unique challenges posed by multimodal data fusion. For example, extending LayUP to handle multimodal integration problems would involve creating mechanisms to align and combine multi-layer features from disparate modalities, accounting for their varying statistical properties and representation spaces. This aligns with the broader goal of CL to handle more complex real-world tasks that require advanced reasoning capabilities.

5.5 Chapter Summary

In this chapter, we introduced LayUP, a conceptually straightforward yet highly effective class-prototype method for CL that exploits multi-layer representations from unimodal foundation models. By computing second-order feature statistics across multiple intermediate layers, LayUP enhances class separability and robustness, especially in scenarios marked by significant domain shifts and limited training data. Extensive evaluations on diverse image classification datasets and in various CL settings reveal that LayUP consistently surpasses state-of-the-art baselines while demanding considerably less memory and computational resources. Moreover, LayUP can be seamlessly integrated as a complementary plug-in with existing class-prototype methods, underscoring the rich, hierarchical information encoded in PTMs that can be leveraged when *"reading between the layers."*

Despite these promising results, the current design of LayUP does not explicitly address the complexities inherent in multimodal learning tasks, which involve implicit reasoning and the integration of heterogeneous data sources. Overcoming these challenges will require the development of novel approaches that extend beyond conventional comparative classification frameworks to advance the capabilities of CL methods. Accordingly, the next chapter will investigate the layer-wise learning dynamics within a modality interaction network to inform the design of robust and well-informed multimodal CL strategies.

Part III

Multimodal Representation Integration

Chapter 6

Vision-Language Feature Fusion with Selective Specialization

An effective artificial agent should continually learn language-informed visual tasks while balancing task-specific assimilation with the development of transferable general knowledge. *Selective specialization*, in other words, carefully determining which components of the underlying model to adapt for each task, emerges as a key strategy to achieve this balance. In multimodal CL with PTMs, where modality-specific encoders are pretrained and modality fusion parameters must be trained on a sequence of multimodal tasks, these components may include layers or sublayers within the modality interaction network. However, designing effective selection strategies requires a deep understanding of how each model component contributes to specialized or generalizable representations, a challenge that remains inadequately addressed in current research.

In this chapter, we rigorously analyze selective specialization strategies for multimodal CL. To facilitate this analysis, we introduce two diagnostic datasets that offer sufficient control and flexibility for a detailed examination of model behavior. We then evaluate a range of heuristics as well as quantified measures to inform module specialization strategies and measure their effects using three different model architectures. Building on our findings, we propose SMS, a conceptually simple selective specialization approach that surpasses common CL baselines. Overall, our results highlight the necessity of more nuanced multimodal CL algorithms that account for the distinct learning patterns of individual model components.[1]

6.1 Motivation

Grounding language in visual perception represents a critical step toward enabling agents to effectively understand and interact with the physical world (Bisk et al., 2020). This objective lies at the core of multimodal (or crossmodal) learning (*cf.*

[1] The source code is made available at `https://github.com/ky-ah/SMS`.

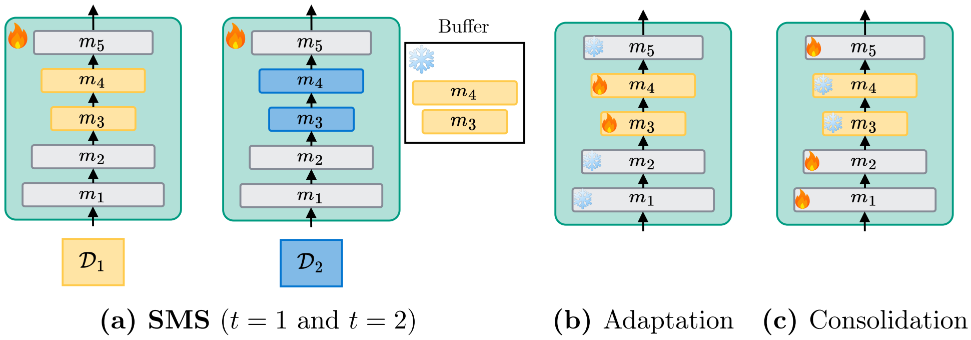


Figure 6.1: Overview of SMS. (a) From a set of *modules* (*i.e.*, self-contained parametric functions) $\mathcal{M} = \{m_1, m_2, \ldots\}$, a subset is chosen for each new task (*selection strategy*), while the remainder of modules is shared across tasks. For every new task, a new set of specialized modules' parameters is initialized, while specialized modules' parameters from previous tasks remain stored in a buffer. Ideally, the randomly initialized task-specific parameters capture only the unique aspects of each task, while broader knowledge transfer occurs through the shared parameters. **(b)+(c)** SMS alternates between multiple adaptation steps, in which only task-specific parameters are updated, and a single consolidation step, in which shared parameters are refined.

Section 2.2). Under realistic conditions, such an agent must continuously integrate new experiences and acquired skills with pre-existing knowledge in an open-ended manner. Multimodal CL thus seeks to achieve this capability. Ideally, the underlying neural model would excel in solving each individual task (*i.e.*, specialization) while simultaneously leveraging common structures and subproblems across tasks to build generalized representations that support knowledge transfer (*i.e.*, generalization). One strategy to manage this trade-off is to introduce task-specific parameters into a judiciously selected subset of model components, as illustrated in Figure 6.1a.

To address this challenge, we propose **S**elective **M**odule **S**pecialization (**SMS**), a conceptually simple approach for CL with PTMs that partitions the components of a modality interaction (or multimodal fusion) network into those that are task-specific and those that are task-agnostic. Inspired by the intellectual development processes described by Piaget (1976), SMS divides training into two interleaved stages, which are presented in Figures 6.1b and 6.1c: *task adaptation*, during which only task-specific parameters are updated, and *knowledge consolidation*, during which only task-agnostic parameters are refined based on the representations learned in the task-specific components. The purpose of the adaptation stage is to allow the model to learn the unique aspects of new tasks by specializing a subset of parameters without overwriting shared knowledge, while the consolidation stage integrates these newly acquired representations into the common parameters to maintain a robust and generalized understanding across tasks.

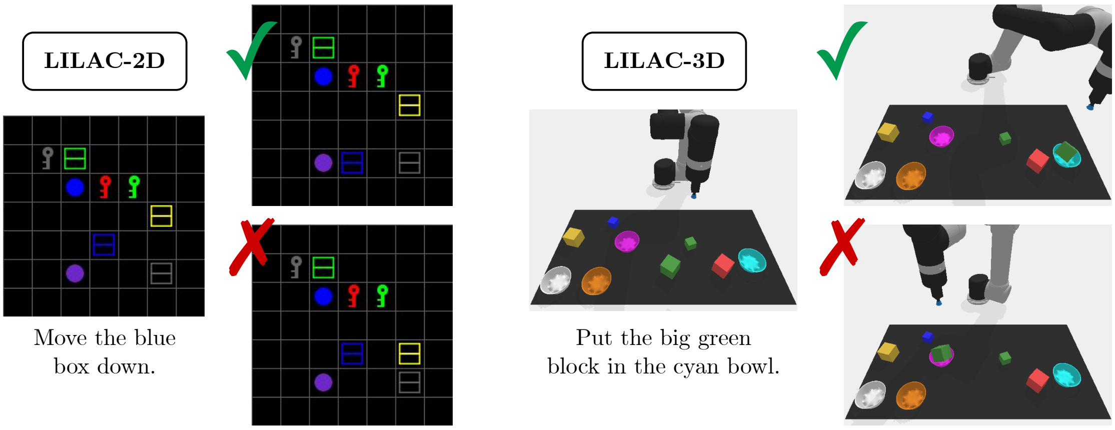


Figure 6.2: Examples of the LILAC datasets. Based on the instruction and the visual premise, a model learns to separate the two visual hypotheses (true target image, false target image) corresponding to a right and a wrong understanding of the instruction, respectively.

Determining which model components (henceforth, *modules*) benefit from task specialization requires information on their role in solving particular tasks, an aspect that prior multimodal CL research has largely overlooked. Nevertheless, current CL benchmarks for VL grounding remain limited. Those relying on synthetic images are intentionally simplistic, such as including only a single type of distributional shift (Greco et al., 2019) or restricting language to trivial phrases over single-object scenes (Skantze and Willemsen, 2022). Others employ real-world imagery, yet allow models to exploit shortcut strategies rather than engage in genuine multimodal reasoning (*e.g.*, Srinivasan et al., 2022; Jin et al., 2020). Such limitations challenge the generalizability of findings from model analyses.[2]

In response, we present the **Li**felong **La**nguage **C**ompositions (**LILAC**) benchmark suite, comprising two diagnostic VL datasets. LILAC-2D and LILAC-3D are specifically designed to enable fine-grained evaluations of continual VL learning. These datasets balance flexibility, control, and complexity and necessitate the acquisition of a broad range of skills such as object localization, spatial reasoning, concept learning, and language grounding. Illustrative examples of these datasets are provided in Figure 6.2.

Building on the proposed SMS method and the LILAC benchmark, we conduct a fine-grained analysis of SMS across three key steps: (i) we derive and evaluate heuristics, inspired by the literature, for introducing task-specific parameters at different network depths and in distinct modality interaction network modules, (ii) we assess the suitability of parameter importance measures, originating from pruning research, as indicators for effective module selection strategies, and (iii) we demonstrate that module selection strategies within our SMS framework that

[2] Nevertheless, bridging the gap to real-world images remains an important direction, which we address in Chapter 7.

we identified through this analysis outperform established CL baselines and even can make the difference between learning or not learning the VL tasks.

Our contributions can be summarized as follows:

(1) We propose SMS, a conceptually simple CL method for multimodal fusion of PTM-derived features, which combines selective module specialization with an iterative adaptation-consolidation training schedule.

(2) We introduce two VL datasets, LILAC-2D and LILAC-3D, each with a well-defined shared structure across tasks. This inherent structure supports a principled examination of module specialization strategies.

(3) We present a comprehensive analysis of diverse selective specialization strategies within the SMS framework, evaluating their efficacy across two different representative modality interaction architectures and two different PTMs.

(4) Finally, we show that SMS, by balancing the trade-off between generalization and specialization, outperforms established CL baselines in the multimodal TIL setting. Our findings underscore the importance of aligning CL techniques with the intrinsic learning dynamics of individual model components.

This chapter is organized as follows: in Section 6.2, we introduce the foundational principles of the SMS approach and present details of the two newly proposed LILAC datasets. In Section 6.3, we describe the experimental setup and provide a comprehensive analysis of the heuristic- and measure-based strategies for module specialization. In Section 6.4, we discuss the implications of our findings and their relevance within the broader landscape of multimodal CL. Finally, in Section 6.5, we summarize the key insights and contributions made in this chapter.

6.2 Proposed Method: SMS

Figure 6.3 provides an overview of the proposed SMS learning setup. We consider a pretrained VLM (referred to as a PTM h for consistency with previous chapters), which comprises a text encoder $h^{(L)}$ and a visual feature extractor $h^{(V)}$, a modality interaction network d_φ, and a PL g_ψ. The CL problem is formulated as described in Section 3.2.1, and the model architecture follows category (e) in the VLM taxonomy described in Section 2.2.3. Consequently, the PTM employs a dual encoder architecture, such as CLIP, trained to align multimodal representations, as detailed in Section 2.2.2. Following the common definition of a TIL setting (*cf.* Table 3.1), we assume the task identifier t to be available during training and inference. In the TIL setting, models are trained in a *multi-headed* fashion, where PL parameters ψ are task-specific, thus $\psi = \{\psi_1, \ldots, \psi_T\}$ and $\psi_i \cap \psi_j = \emptyset$ for all $i \neq j$ (Van De Ven et al., 2022).

The n^{th} input of the t^{th} task, denoted as $(\boldsymbol{x}_{t,n}, \boldsymbol{y}^+_{t,n}, \boldsymbol{y}^-_{t,n}, t) \in \mathcal{D}_t$, includes textual instructions and visual observations $\boldsymbol{x}_{t,n} = (\boldsymbol{l}_{t,n}, \boldsymbol{o}_{t,n})$, $\boldsymbol{y}^+_{t,n} = \boldsymbol{o}^+_{t,n}$, and $\boldsymbol{y}^-_{t,n} = \boldsymbol{o}^-_{t,n}$,

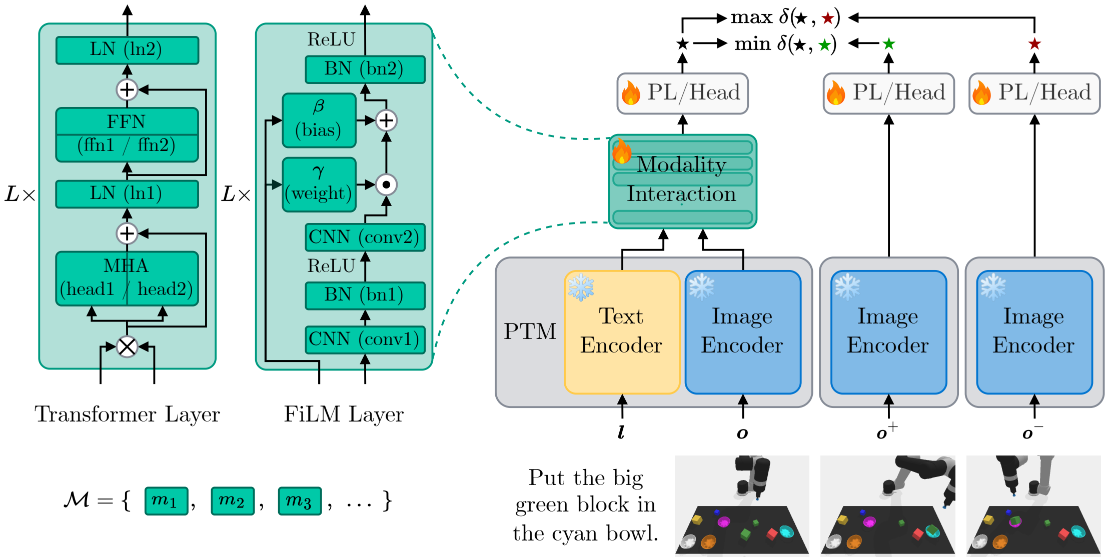


Figure 6.3: Training pipeline on a LILAC-3D example. The training objective is to minimize the distance between encoded instruction-image pair $(\boldsymbol{l}, \boldsymbol{o})$ and encoded positive goal image $\boldsymbol{o}^+$ while maximizing the distance to the negative goal image $\boldsymbol{o}^-$. Each of the L modality interaction network layers—either transformer encoder layers or FiLM layers—contains modules that are candidates for specialization. Each attention head and each of the two feed-forward layers are considered separate modules. δ is a distance measure to quantify the (dis-)similarity between encoded representations of the image-text premise and each visual hypothesis. **LN** = Layer Normalization, **FFN** = Feed-Forward Network, **MHA** = Multi-Head Attention, **BN** = Batch Normalization, **CNN** = Convolutional Neural Network, **ReLU** = Rectified Linear Unit.

where $\boldsymbol{o}^+_{t,n}$ and $\boldsymbol{o}^-_{t,n}$ denote visual scenes corresponding to correct and incorrect executions of the instruction $\boldsymbol{l}_{t,n}$ given the observed scene $\boldsymbol{o}_{t,n}$. This setup can be viewed as an extension of natural language inference (Bowman et al., 2015) into a VL grounding context. Here, $(\boldsymbol{o}_{t,n}, \boldsymbol{l}_{t,n})$ serves as an image-text premise, and $(\boldsymbol{o}^+_{t,n}, \boldsymbol{o}^-_{t,n})$ are visual hypotheses. This formulation can be approached through contrastive learning (Li et al., 2023b), where the model must distinguish correct from incorrect visual outcomes conditioned on textual instructions.

For the modality interaction network d_φ, we employ two well-established architectures in visual language grounding research: (i) a transformer encoder and (ii) a FiLM encoder. These architectures have shown effectiveness in both supervised and imitation learning settings (*e.g.*, Tan and Bansal, 2019; Chen et al., 2020; Panos et al., 2023; Chevalier-Boisvert et al., 2019; Lee et al., 2022; Perez et al., 2018). The left portion of Figure 6.3 outlines these two architectures, highlighting the individual modules within each modality interaction network.

Transformer-based modality interaction. Each transformer encoder layer adopts the original design proposed by Vaswani et al. (2017), as elaborated in Section 2.1. A single encoder layer includes a multi-head attention operation with two attention heads (modules *head1* and *head2*), two normalization layers (*norm1* and *norm2*), and two feed-forward sublayers (*ffn1* and *ffn2*). Text embeddings and visual features are concatenated and fed into the multi-head attention layer, followed by a residual connection and layer normalization. The resulting multimodal latent features are then passed through two fully connected layers, followed by another residual connection and layer normalization. These outputs are then propagated to the next transformer encoder layer.

FiLM-based modality interaction. The FiLM fusion architecture largely follows Chevalier-Boisvert et al. (2019). Each FiLM layer consists of two convolutional layers (modules *conv1* and *conv2*), two batch normalization operations (*bn1* and *bn2*), and two fully connected γ and β layers (*weight* and *bias*) for modulation. The layers γ and β reweight and shift convolutional feature maps based on textual information, following the FiLM paradigm (*cf*. Perez et al., 2018, and Section 2.1.4). Specifically, we first process the visual features of the input image through the first CNN block, followed by batch normalization, a ReLU activation, and then the second CNN block. The sequence embedding of the input instruction is passed through the fully connected layers β and γ, whose output modulates the visual features of the second CNN block via element-wise multiplication and addition. The resulting modulated features are then subjected to a second batch normalization and ReLU activation and subsequently fed to the next encoder layer.

6.2.1 Addressing the Specialization-Generalization Trade-off

Let φ_t denote the set of parameters for the modality interaction network d_φ when processing inputs from task t. Our objective is to determine parameter values $\varphi_1, \ldots, \varphi_T$ that maximize the average test set accuracy across tasks. This objective induces a trade-off between designating parameters for certain tasks for specialization and allowing them to be shared across tasks for generalization, which can be addressed through various design choices for d_φ (Ostapenko et al., 2021).

A straightforward approach is to employ a monolithic network, using the same parameters for all tasks, *i.e.*, $\varphi = \varphi_t$ for all $t \in \{1, \ldots, T\}$ (*e.g.*, Kirkpatrick et al., 2017; Chaudhry et al., 2019). This strategy promotes intertask knowledge transfer, but increases the risk of catastrophic forgetting. At the opposite extreme, one can train task-specific *expert* networks, ensuring that the parameter sets are disjoint ($\varphi_i \cap \varphi_j = \emptyset$ for all $i \neq j$) (*e.g.*, Aljundi et al., 2017; Rusu et al., 2016). This method effectively mitigates forgetting, but requires substantial additional memory, and forgoes opportunities for transfer.

Drawing inspiration from modular network approaches to CL (Ostapenko et al., 2021; Mendez and Eaton, 2021; Mendez et al., 2022), we model d as a collection of modules $\mathcal{M}$. Here, a module $m \in \mathcal{M}$ represents any self-contained parametric function (*e.g.*, a convolutional layer or a batch normalization layer), a definition that is broader than that of a *module* in neural module networks (Andreas et al., 2016). Our goal is to devise a strategy that, for each task t, selects a subset of modules $\mathcal{S} \subset \mathcal{M}$ whose parameters are dedicated to that task, indicated by $\varphi_t^{\mathcal{S}}$, while the parameters of the remaining modules, $\varphi^{\mathcal{M} \setminus \mathcal{S}}$, are shared between tasks. The resulting set of parameters for the task t is therefore $\varphi_t = \varphi_t^{\mathcal{S}} \cup \varphi^{\mathcal{M} \setminus \mathcal{S}}$. In particular, choosing $\mathcal{S} = \emptyset$ recovers the monolithic setting of full parameter sharing, while choosing $\mathcal{S} = \mathcal{M}$ resorts to the aforementioned expert networks with no parameter sharing. In this sense, SMS operates as a model decomposition method situated within the broader category of architecture-based CL approaches (*cf.* Section 3.2.5).

Given a test sample $(\boldsymbol{l}, \boldsymbol{o}, \boldsymbol{o}^+, \boldsymbol{o}^-, t) \in \mathcal{D}_{\text{test}}$, let the pretrained Language (L) and Vision (V) encoders produce representations $h^{(L)}(\boldsymbol{l})$ and $h^{(V)}(\boldsymbol{o})$, respectively. The joint multimodal representation extracted by the PL is

$$\boldsymbol{h} = g_{\psi_t}\left(d_{\varphi_t}\left(h^{(L)}\left(\boldsymbol{l}\right), h^{(V)}\left(\boldsymbol{o}\right)\right)\right) \tag{6.1}$$

The representations of the positive and negative visual hypotheses $\boldsymbol{o}^+$ and $\boldsymbol{o}^-$ are similarly defined:

$$\boldsymbol{h}^+ = g_{\psi_t}\left(h^{(V)}\left(\boldsymbol{o}^+\right)\right) \qquad \boldsymbol{h}^- = g_{\psi_t}\left(h^{(V)}\left(\boldsymbol{o}^-\right)\right) \tag{6.2}$$

Based on these latent representations, the model predicts the correct visual hypothesis by choosing:

$$\hat{\boldsymbol{y}}^+ = \underset{\boldsymbol{o}^+, \boldsymbol{o}^-}{\arg\min} \left\{\delta(\boldsymbol{h}, \boldsymbol{h}^+), \delta(\boldsymbol{h}, \boldsymbol{h}^-)\right\}, \tag{6.3}$$

where δ is a distance measure used to determine which visual hypothesis is more (or less) closely aligned with the given instruction and visual premise. It is important to note that during inference the model does not know which of the visual hypotheses presented is correct; it must rely solely on the learned representations and distance measures.

6.2.2 Interplay Between Adaptation and Consolidation

In Section 3.1, we introduced adaptation and consolidation as integral processes that govern how individuals continually adjust to their environments and stabilize newly acquired knowledge over time. *Adaptation* in human development involves adjusting behaviors, cognitive strategies, and thinking patterns in response to external feedback or environmental demands—an idea central to Piaget's theory of cognitive development (Piaget, 1976). In contrast, *consolidation*, which is detailed in Section 3.1.3, refers to the process through which recently acquired skills or information are integrated into preexisting cognitive frameworks, a process which is reinforced during states of reduced external interference such as sleep.

Building on these concepts, we implement a simplified variant of the lifelong compositional learning approach proposed by Mendez and Eaton (2021), which has

Algorithm 5: SMS Training

```
Input: modality interaction parameters $\bigcup_{t=1}^{T} \varphi_t$, PL parameters $\bigcup_{t=1}^{T} \psi_t$,
       adaptation step count $s$, selection strategy $\mathcal{S}$
# Continual Learning with SMS
for task $t = 1, \dots, T$ do
    for iteration $i = 1, \dots, |\mathcal{B}_t|$ do
        $b_i \leftarrow$ next batch from $\mathcal{B}_t$
        for every sample $(\boldsymbol{l}, \boldsymbol{o}, \boldsymbol{o}^+, \boldsymbol{o}^-, t) \in b_i$ do
            get $\varphi_t = \varphi_t^{\mathcal{S}} \cup \varphi^{\mathcal{M} \setminus \mathcal{S}}$ and $\psi_t$
            collect $\boldsymbol{h}$ using Equation (6.1)
            collect $\boldsymbol{h}^+$ and $\boldsymbol{h}^-$ using Equation (6.2)
            if $i$ mod $s = 0$ then
                # Consolidation Step
                update $\varphi^{\mathcal{M} \setminus \mathcal{S}}$ and $\psi_t$ using AdamW
            end
            else
                # Adaptation Step
                update $\varphi_t^{\mathcal{S}}$ and $\psi_t$ using AdamW
            end
        end
    end
end
```

demonstrated effectiveness in modular CL. Rather than updating all parameters φ_t concurrently during training of each task t, we adopt an alternating procedure consisting of s adaptation steps (fast learning or task assimilation) for the task-specific parameters $\varphi_t^{\mathcal{S}}$, followed by a single consolidation step (slow learning or knowledge accommodation) for shared parameters $\varphi^{\mathcal{M} \setminus \mathcal{S}}$. We refer to this process as *Adaptation-and-Consolidation* (A&C). The pseudocode for the SMS training procedure is provided in Algorithm 5, and the corresponding testing procedure is detailed in Algorithm 6.

6.3 Experiments

In this section, we present a comprehensive series of experiments to evaluate the effectiveness of the SMS framework across a wide range of specialization strategies and model configurations. We begin by introducing the two newly proposed LILAC datasets, detailing the corresponding training protocols and model configuration specifics. Subsequently, we analyze various heuristic-based selection strategies, incorporating insights from parameter importance measures commonly studied in model pruning research. Based on this analysis, we establish baseline selection strategies for comparison with both rehearsal-based and regularization-based CL approaches in the TIL setting. Furthermore, we perform an in-depth variability

Algorithm 6: SMS Testing

```
Input: trained parameters $\bigcup_{t=1}^{T} \varphi_t \cup \psi_t$
for every sample $(\boldsymbol{l}, \boldsymbol{o}, \boldsymbol{o}^+, \boldsymbol{o}^-, t) \in \mathcal{D}_{\text{test}}$ do
    collect $\boldsymbol{h}$ using Equation (6.1)
    collect $\boldsymbol{h}^+$ and $\boldsymbol{h}^-$ using Equation (6.2)
    predict $\hat{\boldsymbol{y}}^+$ using Equation (6.3)
end
```

analysis and conclude this section by assessing the impact of combining SMS orthogonally with replay-based and regularization-based CL baselines.

6.3.1 Benchmarks

In what follows, we introduce two benchmark datasets—LILAC-2D and LILAC-3D—that are designed to explicitly capture the compositional nature of the underlying tasks. By maintaining a high degree of overlap across tasks, these datasets encourage models to minimize the use of task-specific parameters and facilitate the development of effective module selection strategies. During training, each model instance receives three input images that depict objects in a simulated environment, coupled with a template-based language instruction. A summary of both LILAC datasets is provided in Table 6.1.

LILAC-2D tasks. The LILAC-2D dataset is derived from the `minigrid` environments (Chevalier-Boisvert et al., 2023). Each sample consists of a 7 × 7 grid containing between three and nine objects placed at random positions. Each instruction specifies the desired interaction with a target object and follows the template: *"move the <attribute1> <attribute2> <attribute3>"*. To generate an instruction from the given template, *attribute1* is chosen from six possible colors, *attribute2* from three object types, and *attribute3* from four directions, yielding $6 \times 3 \times 4 = 72$ unique instructions. To generate negative (false target) examples, one of the three specified attributes is deliberately violated. For instance, when the correct instruction is "move the green key down," a false target might depict a scenario where a *blue* key is moved down instead. The dataset comprises 500 training, 100 validation, and 100 test samples per instruction, resulting in a total of 36 000 training, 7 200 validation, and 7 200 test samples.

LILAC-3D tasks. To approximate real-world complexity while preserving the ability to control and analyze the scenario, we also propose LILAC-3D. This dataset is built on the simulated Ravens environment (Zeng et al., 2021) and its language-based extension (Shridhar et al., 2022). Scenes depict a tabletop setting where a robotic arm can manipulate objects. Each image contains between five and eight blocks and between three and four bowls placed at random. Instructions specify a pick-and-place action using the template: *"put the <attribute1> <attribute2>*

Dataset	**Attribute**			T	N_{train}	N_{test}
	I	**II**	**III**			
LILAC-2D	blue green gray purple red yellow	ball box key	up down to the left to the right	12	36 000	7 200
LILAC-3D	big small	blue green gray purple red yellow	brown cyan orange petrol pink white	12	36 000	7 200

Table 6.1: Summary of the LILAC datasets. Provided is the range of all three attributes used in the goal instructions, the number of tasks (T) employed in the TIL setting for the main experiments, and the number of training samples (N_{train}) and test samples (N_{test}). Each validation set, used for hyperparameter optimization, is the same size as its corresponding test set.

block in the <attribute3> bowl". Here, *attribute1* indicates size (two possibilities), *attribute2* denotes block color (six possibilities), and *attribute3* specifies bowl color (six possibilities), totaling $2 \times 6 \times 6 = 72$ unique instructions. The sets of block and bowl colors do not overlap to facilitate a clear visual distinction between objects. To create false targets, either the incorrect block or the incorrect bowl is selected. LILAC-3D follows similar data statistics as LILAC-2D, providing an equally large compositional testbed for model evaluation.

6.3.2 Training Details

We conduct our experiments using two variants of a dual-encoder CLIP model as the PTM h, namely CLIP-RN and CLIP-ViT. The former employs a ResNet-50 as image encoder $h^{(V)}$, the latter a ViT-B/16. In accordance with the original CLIP configurations, both models incorporate a 12-layer transformer text encoder $h^{(L)}$ with a 512-dimensional token embedding space. Although the two text encoders share the same architectural design, their parameters differ due to the contrastive pretraining process that aligns each text encoder with its respective image encoder. The PL g_ψ is implemented as a fully connected linear layer that maps the fused multimodal representations to the same target dimensionality. The distance metric, δ, is the cosine distance.

We integrate two distinct modality interaction networks d_φ (as detailed in Section 6.2) with the two aforementioned PTM variants to form three model configurations. The first configuration, denoted as **CLIP-RN/FiLM**, uses the ResNet-50 vision backbone alongside a FiLM-based modality interaction network. The second, **CLIP-RN/TE**, employs the same ResNet-50 vision backbone, but uses a transformer encoder as interaction network. The third configuration, **CLIP-ViT/TE**, leverages the ViT-B/16 vision backbone in conjunction with the transformer encoder interaction network. We do not explore a ViT-based vision encoder with FiLM because FiLM relies on modulating 2D convolutional feature maps, making it less suitable for the 1D patch embeddings produced by a ViT. Throughout all experiments, we fix the depth of the modality interaction network at $L = 4$, which yields a compact yet sufficiently expressive architecture that facilitates detailed model evaluation. We set the adaptation step count to $s = 10$ for all model configurations, guided by preliminary empirical results indicating that these settings consistently support robust performance across various specialization strategies.[3]

In the CLIP-RN/FiLM and CLIP-RN/TE configurations, the ResNet-50 encoder $h^{(V)}$ generates a spatial feature map of size 14×14 with 512 channels. In CLIP-RN/FiLM, the 512-dimensional textual instruction embedding produced by $h^{(L)}$ is used to provide FiLM-specific weight and bias parameters that modulate the visual features. By contrast, in CLIP-RN/TE, the feature maps of the $(14 \times 14 \times 512)$-dimensional image features are first flattened into 196×512 dimensions and then concatenated with the 512-dimensional textual instruction embedding from $h^{(L)}$ along the channel dimension. For the CLIP-ViT/TE variant, the ViT-B/16 encoder outputs a sequence of 197 token embeddings (196 patch tokens plus a [CLS] token), each 768-dimensional. Here, the original 512-dimensional textual embedding from $h^{(L)}$ is zero-padded to 768 dimensions before being concatenated with the ViT token embeddings. In both CLIP-RN/FiLM and CLIP-RN/TE, the combined VL features undergo max pooling across the spatial dimension prior to entering the PL. In the CLIP-ViT/TE configuration, only the fused [CLS] token embedding is passed to the PL.

The parameters of the modality interaction network, d_φ, are optimized over ten epochs ($E = 10$) using batches of size $|b| = 128$. We utilize the InfoNCE loss function (Oord et al., 2019) in conjunction with the AdamW optimizer (Loshchilov and Hutter, 2019) configured with default hyperparameters. For all baselines that employ a monolithic network architecture, the learning rate was determined via hyperparameter tuning on the SFT baseline under the TIL setting, resulting in a value of 2.5×10^{-3} that consistently yielded robust performance across the evaluated datasets and configurations. In contrast, for all SMS baselines, a higher learning rate of 2.5×10^{-2} was selected. This value was identified through hyperparameter optimization using the CLIP-ViT/TE model with task-specific self-attention parameters as representative configuration and trained under the TIL setting on the LILAC-3D dataset. This learning rate was uniformly applied in all SMS base-

[3] An empirical evaluation of alternative step counts is provided in Section 6.3.7.

lines, regardless of the model configuration, CL benchmarks, and specialization strategies.

We report the average accuracy after training on the final task, denoted by AA_T (*cf.* Equation (3.10)), as the primary evaluation metric, where $a_{T,t}$ represents the accuracy of correctly identifying the true visual hypothesis of a sample from the t^{th} task following training on the T^{th} (*i.e.*, last) task. To ensure reproducibility, we repeat *all* experiments conducted in this chapter using five fixed random seeds (1993–1997), which influence both the task order and the random initialization of trainable parameters. In our analysis, we measure the accuracy improvement provided by a specialization strategy $\mathcal{S}$ relative to a monolithic network as

$$\Delta\mathrm{AA}_T(\mathcal{S}) = \mathrm{AA}_T^{(\varphi_{1:T})} - \mathrm{AA}_T^{(\varphi)}, \tag{6.4}$$

where the total set of shared and specialized modality interaction parameters after training on the last task is defined as

$$\varphi_{1:T} = \bigcup_{t=1}^{T} \varphi_t = \underbrace{\varphi^{\mathcal{M}\setminus\mathcal{S}}}_{\text{task-sharing}} \cup \underbrace{\bigcup_{t=1}^{T} \varphi_t^{\mathcal{S}}}_{\text{task-specific}} \tag{6.5}$$

We compare our proposed approach with the following baseline methods. Sequential Fine-Tuning (**SFT**) performs unrestricted updates to a single monolithic model in a sequential learning setting. Experience Replay (**ER**; *cf.* Equation (3.19)) mitigates catastrophic forgetting by maintaining a fixed-size memory buffer to store a subset of previously observed data samples using reservoir sampling. In our experiments, we use a buffer size of 3 000 and draw a mini-batch from the replay buffer for rehearsal after every training epoch, which corresponds to a replay frequency of 24. Online Elastic Weight Consolidation (**O-EWC**; *cf.* Equation (3.13)) builds upon EWC by continuously updating a running Fisher information matrix approximation, thereby regularizing parameter changes deemed important for earlier tasks. We employ O-EWC with regularization strength $\lambda_{\text{O-EWC}} = 10$ and decay factor $\gamma_{\text{O-EWC}} = 0.5$.[4] Joint Fine-Tuning (**JFT**) trains a single model on all tasks simultaneously under an i.i.d. data distribution. Finally, the **Expert** baseline involves training a separate model for each task independently, each starting from a randomly initialized set of modules, which ensures no interference among tasks at the expense of increased model complexity and lack of knowledge transfer.

6.3.3 Evaluation of Alternating Adaptation and Consolidation

We hypothesize that the biologically inspired A&C optimization scheme can enhance SMS training by mitigating catastrophic forgetting in a CL context. Specifi-

[4] The search spaces for the hyperparameters were defined as follows: $\lambda_{\text{O-EWC}} = 10^x$, $x \in \{-4,-3,-2,-1,0,1,2,3,4\}$ and $\gamma_{\text{O-EWC}} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Hyperparameter configurations were determined through Bayesian optimization.

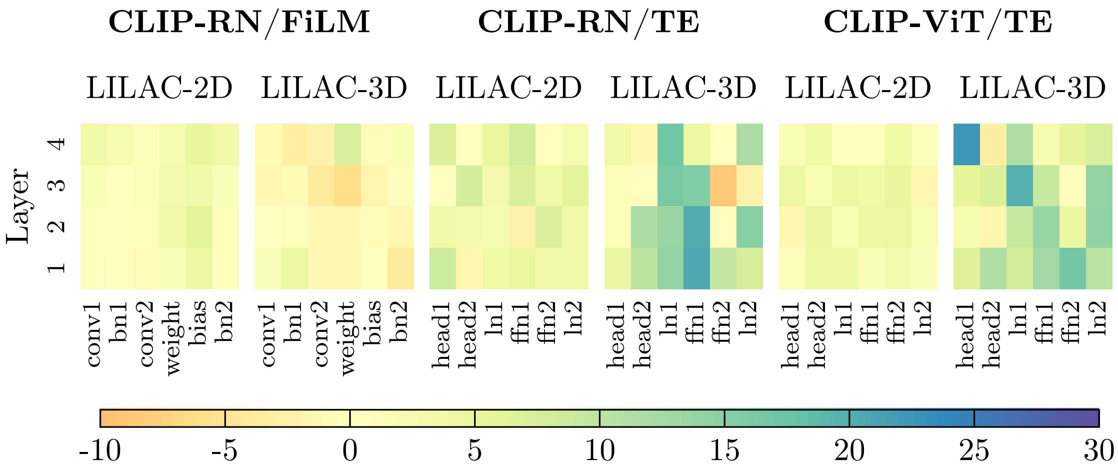


Figure 6.4: A&C vs. joint training of all shared and task-specific modules. Each cell corresponds to a separate experiment defined by a particular selection strategy $\mathcal{S} = \{m\}$ for each model configuration and CL benchmark. The values shown indicate the difference in accuracy gains, $\Delta\mathrm{AA}_T(\mathcal{S})$ (*cf*. Equation (6.4)), achieved by SMS using the A&C strategy relative to the baseline that jointly trains both shared ($\mathcal{M}\backslash\mathcal{S}$) and task-specific ($\mathcal{S}$) modules. Positive (green and blue) entries highlight scenarios where A&C surpasses the joint training approach.

cally, by updating the shared modules less frequently, we anticipate that the parameters responsible for foundational cross-task representations remain more stable. Concurrently, each task's distinct features are captured by a small set of task-specific parameters, thereby reducing interference across tasks.

To examine whether A&C surpasses the joint optimization of φ_t, we measure the difference in accuracy gains for each module $m \in \mathcal{M}$ using SMS with the selection strategy $\mathcal{S} = \{m\}$, both with and without A&C optimization. As illustrated in Figure 6.4, A&C training consistently yields accuracy improvements for transformer-based modality interaction when CLIP-ViT-B/16 is used as the PTM, and predominantly provides favorable gains for transformer-based modality interaction with CLIP-ResNet-50 as the PTM. Notably, the impact of A&C appears especially pronounced for SMS training on LILAC-3D compared with LILAC-2D. In contrast, for FiLM-based modality interaction, the improvements are generally positive for LILAC-2D but predominantly negative for LILAC-3D.

Overall, these findings indicate that A&C proves particularly advantageous for transformer-based modality interaction, yet it also yields benefits across various model configurations and datasets. Despite the shared modules receiving fewer updates, they still learn robust representations of common subproblems across tasks. Thus, ultimately A&C can improve the overall performance while reducing the risk of forgetting.

6.3.4 Heuristics for Specialization Strategies

In real-world CL scenarios, exhaustive experimentation to identify the optimal module selection strategy for SMS is typically infeasible. Consequently, it is critical to develop robust heuristics that, for a given multimodal CL task, can guide decisions about which types of modality interaction network modules—and at which layer depths—should be shared across tasks versus specialized for individual tasks.

Specialization at different layer depths. To investigate how isolating task-specific parameters at varying layer depths influences specialization, we conduct a separate experiment for each model configuration, dataset, and layer of the modality interaction network. We then quantify the resulting accuracy gain (ΔAA_T; *cf.* Equation (6.4)) over a sequential fine-tuning procedure on a monolithic network (*i.e.*, SFT baseline), forward transfer (FWT_T; *cf.* Equation (3.7)), and forgetting measure (FM_T; *cf.* Equation (3.8)). These results, summarized in Figure 6.5, reveal notable performance differences across layers, which underscores the importance of identifying the optimal depth at which to specialize.

Previous research on transformers in NLP (Hao et al., 2019; Tenney et al., 2019) generally concludes that earlier layers capture local syntactic information, whereas later layers tend to encode higher-level semantic or conceptual representations. Likewise, in their investigation of FiLMed networks, Perez et al. (2018) report that early layers facilitate low-level operations (*e.g.*, querying attributes of an object), while later layers handle more abstract reasoning (*e.g.*, comparing objects). In the LILAC tasks, solving 2D or 3D problems requires not only object object identification, but local spatial reasoning, and—in the 3D setting—placing multiple objects within a shared context. Furthermore, these tasks differ in the frequency with which high-level concepts must be recombined (*e.g.*, learning to move a blue key *downward* in one scenario versus *upward* in another).

Building on these observations, we hypothesize that isolating later layers in the modality interaction network, which are more likely to capture complex conceptual information, will yield superior performance when learning tasks sequentially. Our empirical findings support this hypothesis. In configurations employing FiLM for modality interaction, specializing the final layer of the modality interaction network (*i.e.*, the last reasoning layer prior to the PL) yields the largest accuracy improvements. By contrast, in transformer-based configurations, specializing the penultimate layer typically provides the greatest gains, with one exception (CLIP-RN/TE on LILAC-2D) in which the final layer also proves optimal. This outcome underscores that the later stages of the network are especially critical for mitigating catastrophic forgetting and improve overall CL performance.

Interestingly, while the degree of forgetting increases when later layers are progressively specialized, the penultimate or final layer still typically yields the highest overall accuracy gain. Moreover, in nearly all configurations, specializing the last

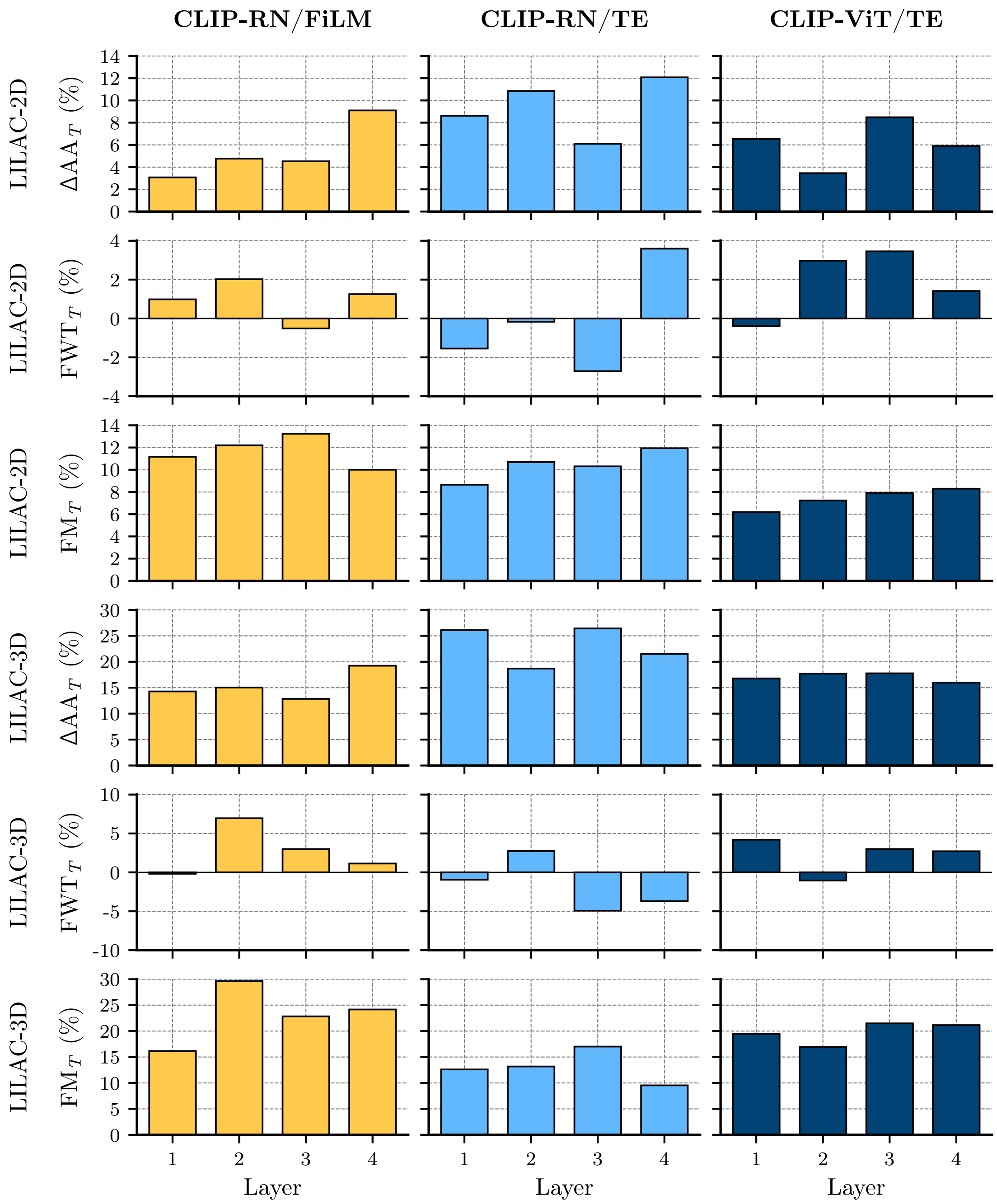


Figure 6.5: Effect of specialization at different layer depths. For each model configuration, dataset, and designated layer in the modality interaction network, we report the accuracy gain ($\Delta\mathrm{AA}_T$) compared with the SFT baseline, average forward transfer (FWT_T), and average forgetting (FM_T) measured after training on the final task. Each bar represents an experiment in which all modules within a single layer of the modality interaction network are specialized.

layer leads to moderately to strongly positive forward transfer, suggesting that specializing late layers not only retains performance on previously encountered tasks but also enhances zero-shot adaptation to new tasks, which are in our setting new compositional variations of familiar concepts. These observations underscore that prioritizing forward transfer or forgetting alone is not suitable as the sole metric to assess a model's CL competence, as argued in Section 3.2.3.

Transformer feed-forward and linear modulation parameters. Following Geva et al. (2021), we interpret the feed-forward networks in transformer architectures as *key-value memories*. Their analysis shows that early-layer feed-forward blocks encode salient semantic patterns, whereas late-layer feed-forward blocks capture more surface-level patterns. In our LILAC tasks, the language instructions are templated and combine various concepts, so they share the same overall semantics. However, these instructions are grounded in diverse visual scenes. Accordingly, we investigate whether specialized feed-forward modules (*i.e.*, distinct key-value memories) might serve as strong candidates for CL with SMS.

Figure 6.6 (blue bars) indicates that task-specific adaptation of both feed-forward modules (ffn1+ffn2) in a transformer-based modality interaction layer does not surpass the performance achieved by specializing only one of the two modules. Moreover, restricting specialization to a single feed-forward module cuts in half the number of task-specific parameters to store, suggesting a promising strategy for balancing performance and memory demands. We also observe that dedicating both feed-forward blocks to specialization yields the greatest improvements when it occurs in either the penultimate or final layer, although the trend is slightly less pronounced when only one feed-forward block is specialized. This pattern implies that the model primarily needs to distinguish shallow task differences in order to succeed in LILAC tasks, where "late specialization" of feed-forward blocks tends to be most helpful. Nevertheless, as the linguistic and visual variety of inputs grows with more complex VL problems or datasets, specializing earlier feed-forward modules may offer increasing benefits. Despite that possibility, our overall analysis shows that later layers deliver the largest performance gains under specialization. Consequently, focusing on late-layer specialization—either for both feed-forward blocks or for a feed-forward block—emerges as an effective approach for CL.

Because LILAC's textual instructions direct the modulation of visual representations, and because new tasks can recombine these same instructions in novel ways, we also examine whether and which FiLM parameters (γ for weights and β for biases) merit task-specific specialization. As shown in Figure 6.6 (yellow bars), the accuracy gains and forward-transfer from late-layer specialization of FiLM parameters decrease, while forgetting increases, particularly when both FiLM weights and biases are specialized. At the same time, dedicating a single layer for task-specific FiLM parameters outperforms specialization across all layers, while substantially reducing parameter storage. Furthermore, specializing the FiLM bias (which shifts visual features) generally surpasses specializing the weight (which scales visual features).

A plausible explanation for these divergent findings is that FiLM's γ and β parameters exert direct control over the modulation of visual feature representations by

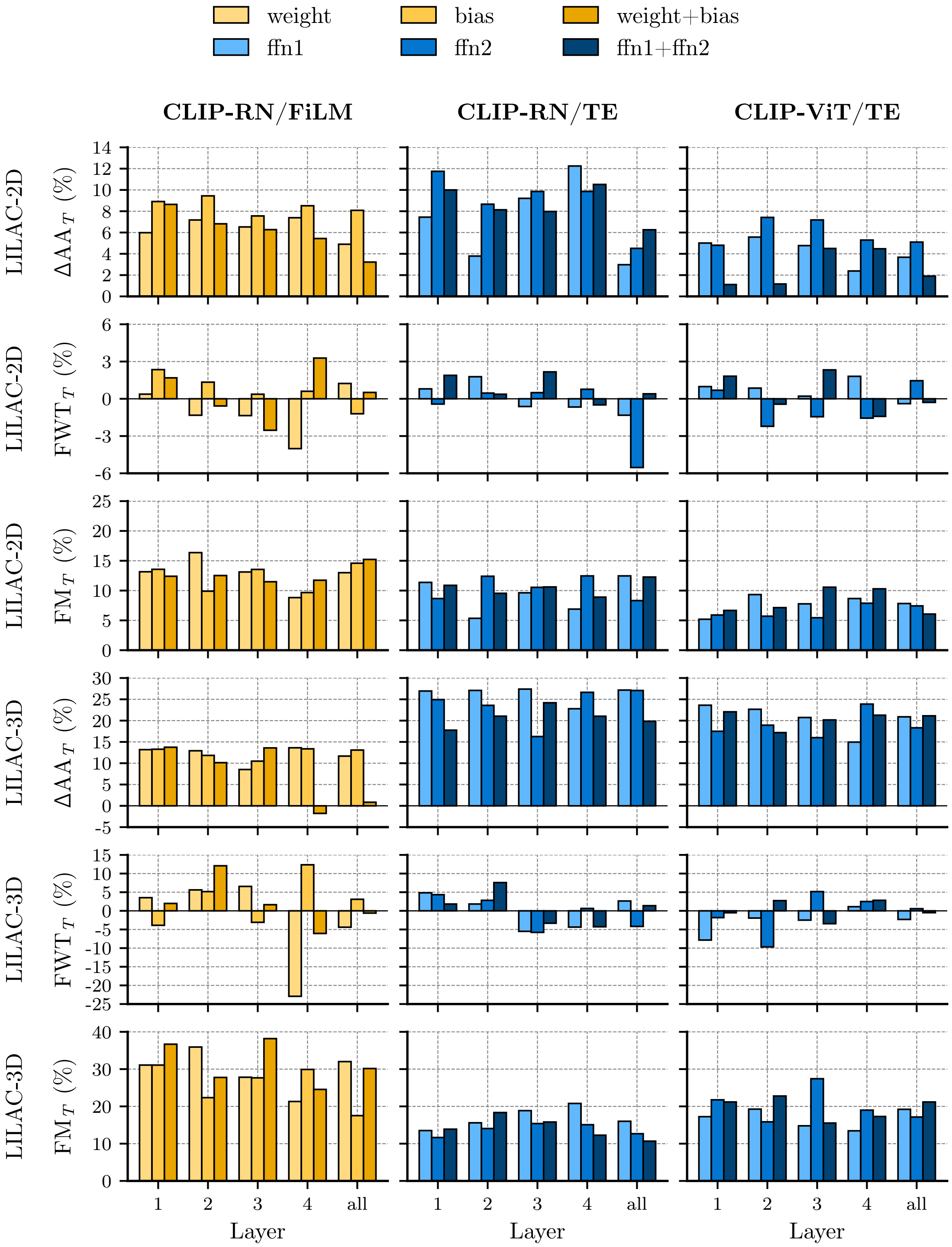


Figure 6.6: Effect of specializing feed-forward and modulation parameters. In the FiLM-based fusion setting, each bar represents an experiment where the weight, the bias, or both (weight+bias) modules are specialized. In the transformer-based fusion setting, each bar indicates specialization of the first feed-forward layer (ffn1), the second feed-forward layer (ffn2), or both (ffn1+ffn2) within a transformer encoder layer.

textual cues. This process yields more pronounced feature enhancement in early and intermediate network layers, where robust multimodal integration is critical. In this context, the bias term (β) is particularly pivotal for adapting to novel visuo-linguistic compositions in new tasks, likely by systematically shifting the latent feature space. As these modulated features propagate to the final layers—where the model more effectively captures task-specific nuances—continued refinement of FiLM parameters may introduce computational overhead and increase the risk of overfitting or cross-task interference. Therefore, prioritizing the specialization of FiLM modulation parameters in the early layers, especially the bias components responsible for shifting visual features, appears to be the most effective strategy.

Multi-head attention and convolution parameters. In their study on PEFT methods (*cf*. Section 3.3.2), Smith et al. (2023b) demonstrate the efficacy of adapting self-attention blocks in a ViT for downstream image classification tasks, while keeping the remaining transformer parameters shared across tasks. To determine whether this insight holds under our multimodal setting, we experiment with specializing individual heads in the multi-head attention mechanism of the transformer modality interaction layers. As shown in Figure 6.7 (blue bars), this approach yields considerable accuracy improvements of up to 14% for LILAC-2D and 27% for LILAC-3D compared with the SFT baseline.

Despite these improvements, self-attention parameters in both the CLIP-RN/TE and CLIP-ViT/TE configurations make up approximately 98% of the total modality interaction network parameters—largely due to the high dimensionality of the multimodal inputs. Consequently, fully specializing these self-attention parameters for each task demands a memory footprint approaching that of expert-based CL methods. Meanwhile, other fine-tuning strategies (*e.g.*, specializing certain feed-forward blocks; *cf*. Figure 6.6) can achieve comparable accuracy gains with far fewer task-specific parameters. Notably, a more conservative (though still memory-intensive) approach that specializes only one attention head in each layer outperforms a strategy that includes *all* attention parameters in three out of four model–dataset configurations. Thus, a strategy requiring roughly half the parameter budget of the one proposed by Smith et al. (2023b) can match or exceed their reported results.

We also examine the convolutional blocks within the FiLM layers of the CLIP-RN/FiLM configuration and hypothesize that task-specific adaptation of convolutional blocks may be less critical for boosting overall task performance, since they primarily handle visual processing and the difference between tasks lies predominantly in different textual instructions. Indeed, Figure 6.7 (yellow bars) confirms that specializing convolutional parameters yields smaller overall accuracy gains than specializing other types of modules. Nonetheless, it is noteworthy that convolutional modules account for only about 8% of all modality interaction parameters

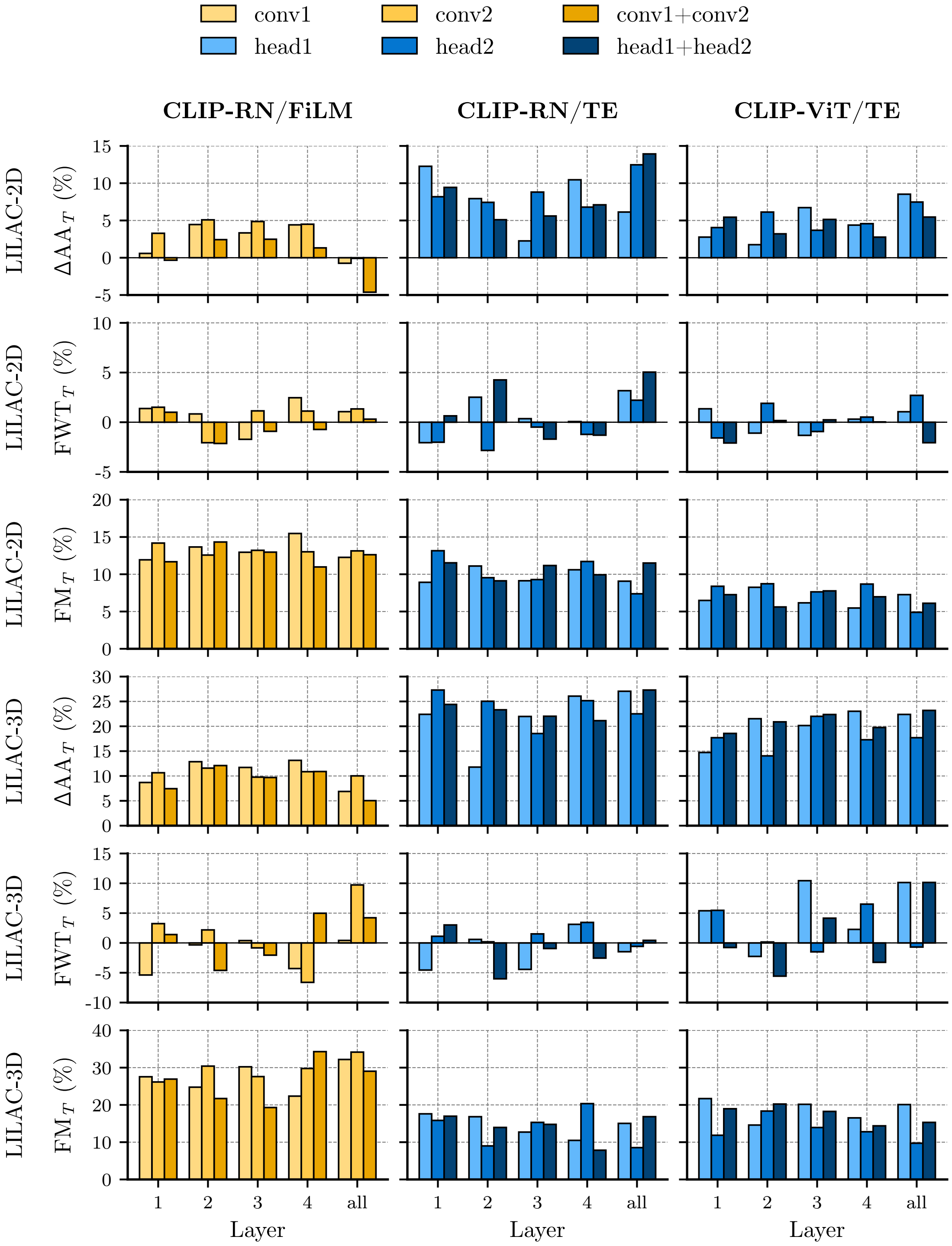


Figure 6.7: Effect of specializing multi-head attention and convolutional neural network parameters. In the FiLM-based fusion setting, each bar represents an experiment where the first CNN block (conv1), the second CNN block (conv2), or both (conv1+conv2) modules are specialized. In the transformer-based fusion setting, each bar indicates specialization of the first attention head (head1), the second attention head (head2), or both (head1+head2) within a transformer encoder layer.

in the model configuration used in our experiments, which makes corresponding specialization strategies highly memory-efficient and thus more scalable.

We further observe that task-specific convolutional parameters in the first FiLM layer confer substantially smaller accuracy gains than in deeper layers. This diminished impact likely reflects the first layer's focus on low-level features (*e.g.*, edges, color gradients), which are less task-specific. As a result, adapting the first convolutional block provides limited benefit for downstream tasks. Instead, we find that specializing the second convolutional block (conv2) in an intermediate layer (either the second or third layer overall) yields stronger improvements. By specializing these mid-level parameters, the model learns a richer set of task-specific abstractions while retaining the more general, low-level representations in the earliest layers. This balance between general and specialized information is consistent with our analysis of specialization at different layer depths.

Normalization scaling parameters. Bilen and Vedaldi (2017) propose leveraging specialized instance-, layer-, or batch-normalization scaling factors to address task-specific biases with minimal additional memory overhead. We investigate how specialization of Batch Normalization (BN) parameters influences performance within the SMS framework. As illustrated in Figure 6.8 (yellow bars), specializing BN scaling factors—especially in the final layers—consistently surpasses the standard SFT baseline. In particular, specializing the second BN block, *i.e.*, the one positioned directly after the linear modulation of language inputs and immediately before the fusion of multimodal features, yields greater improvements than specializing the BN block situated between CNN layers. A plausible explanation is that task-specific nuances are strongly encoded in the language instructions, which often recombine complex concepts in domain-specific ways. Because the second BN block is the first set of trainable parameters following multimodal fusion, it is well-placed to capture these subtle distinctions, thus preserving more of the underlying task-driven variability.

We extend this analysis to Layer Normalization (LN) in a transformer-based modality interaction network, where LN is applied before *and* after key sublayers, such as multi-head attention and feed-forward modules, to stabilize training and promote more robust feature representations. As depicted in Figure 6.8 (blue bars), specializing LN generally leads to larger accuracy gains than BN under comparable configurations, particularly when CLIP with a ResNet-50 backbone is used as the PTM. There is no clear pattern favoring early-layer over late-layer specialization. However, dedicating task-specific parameters to one of the LN modules in all layers appears especially beneficial. For instance, in the CLIP-RN/TE architecture, specializing only the first LN operation in every layer results in a marked performance improvement over the SFT baseline. In contrast, in the CLIP-ViT/TE architecture, the most substantial gains arise when the second LN operation in each layer is specialized.

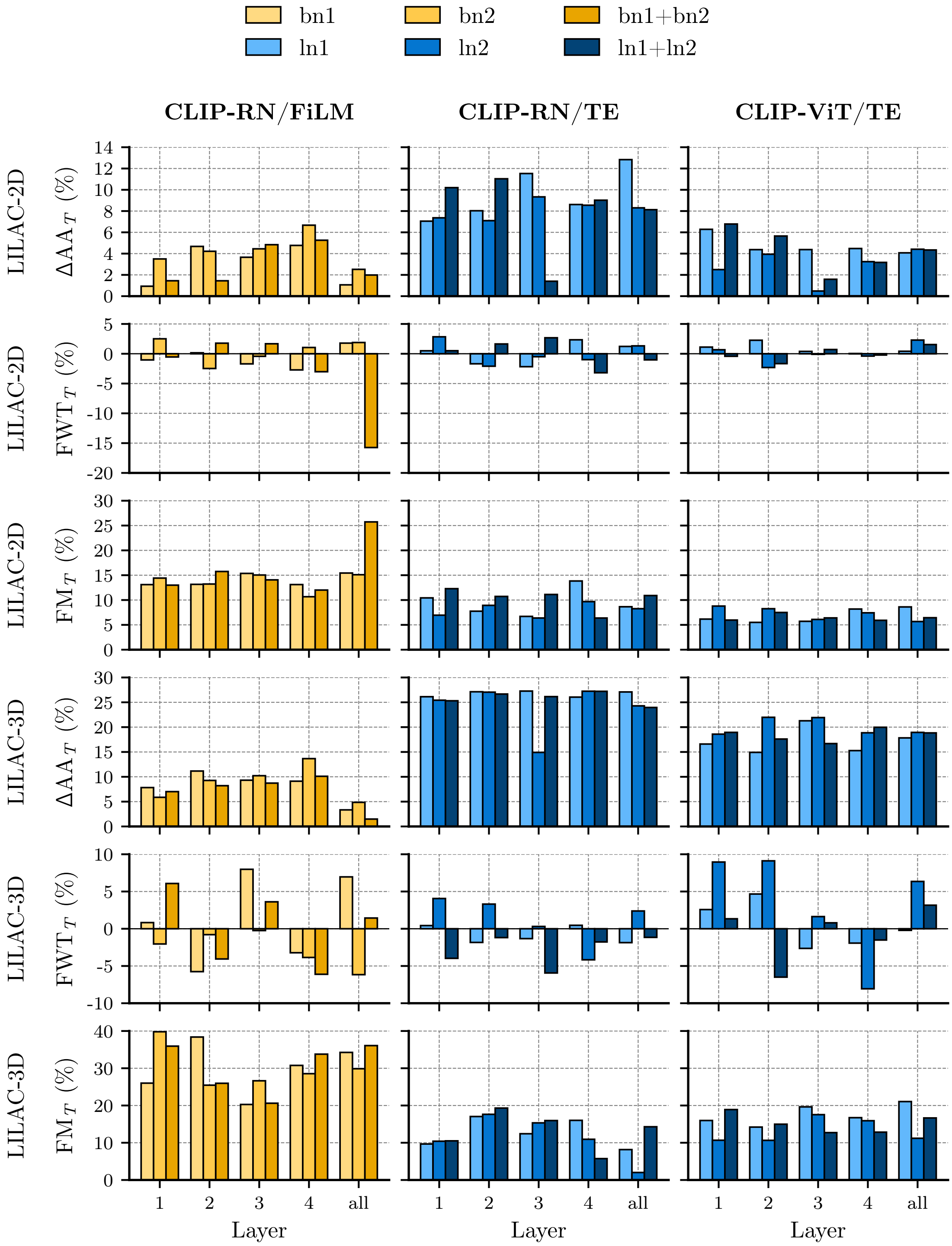


Figure 6.8: Effect of specializing normalization scaling parameters. In FiLM-based fusion, each bar shows an experiment where the first batch normalization (bn1), the second batch normalization (bn2), or both operations (bn1+bn2) are task-specific. In transformer-based fusion, each bar indicates the first layer normalization (ln1), the second layer normalization (ln2), or both operations (ln1+ln2) being specialized within a transformer encoder layer.

These findings can be partially attributed to the integration of residual connections and the manner in which LN normalizes activations across feature dimensions in the standard transformer encoder. The LN layers, which operate on a per-feature basis, appear to capture salient domain- or task-specific information by adjusting normalized activations immediately before or after major transformations (*e.g.*, multi-head attention or feed-forward modules). Consequently, small modifications to LN scaling parameters can yield disproportionately large impacts on model performance. By specializing LN parameters in multiple layers, the network can better adapt residual pathways to task-specific or domain-specific distributions.

6.3.5 Importance Metrics for Specialization Strategies

Measuring the importance of individual neural connections is commonplace in research on neural pruning for CL (Molchanov et al., 2019), as surveyed in Section 3.2.5. We build upon this concept to assess the importance of entire network *modules*, rather than individual parameters, aiming to determine whether these importance measures can guide module-level specialization. Specifically, we employ two widely adopted importance scoring techniques—gradient-based (*e.g.*, Wang et al., 2022d; Gurbuz and Dovrolis, 2022) and activation-based methods (*e.g.*, Jung et al., 2020)—and evaluate their suitability for identifying modules that are beneficial to specialize.

Let φ^m be the parameters of module m during the training of the SFT baseline with modality interaction network d_φ, and let $\varphi^{t,m}$ denote the parameters of the same module m after training on the t^{th} task. Note that φ^m and $\varphi^{t,m}$ should not be confused with $\varphi_t^{\{m\}}$, which refers to the parameters of module m specialized to the t^{th} task when it is trained in the SMS framework under the selection strategy $\mathcal{S} = \{m\}$.

Gradient-based importance score. The gradient-based module importance score $\Omega_{\text{grad}}(m)$ is derived by combining the magnitudes of parameters with the magnitudes of their gradients during training. Formally, we define

$$\Omega_{\text{grad}}(m) := \alpha \sum_{t=1}^{T} \sum_{w \in \varphi^{t,m}} |w| + \frac{1}{2} \left| \frac{\partial \mathcal{L}(\mathcal{D}_t; \varphi)}{\partial w} \right|, \tag{6.6}$$

where $\mathcal{L}(\mathcal{D}_t; \varphi)$ represents the cumulative loss on task t, and

$$\alpha = \frac{1}{T \cdot \sqrt{|\varphi^m|}} \tag{6.7}$$

is a normalization constant designed to mitigate bias toward selecting modules with more parameters for task-specialization.

Activation-based importance score. The activation-based module importance score $\Omega_{\text{act}}(m)$ quantifies how strongly a module responds to input data across

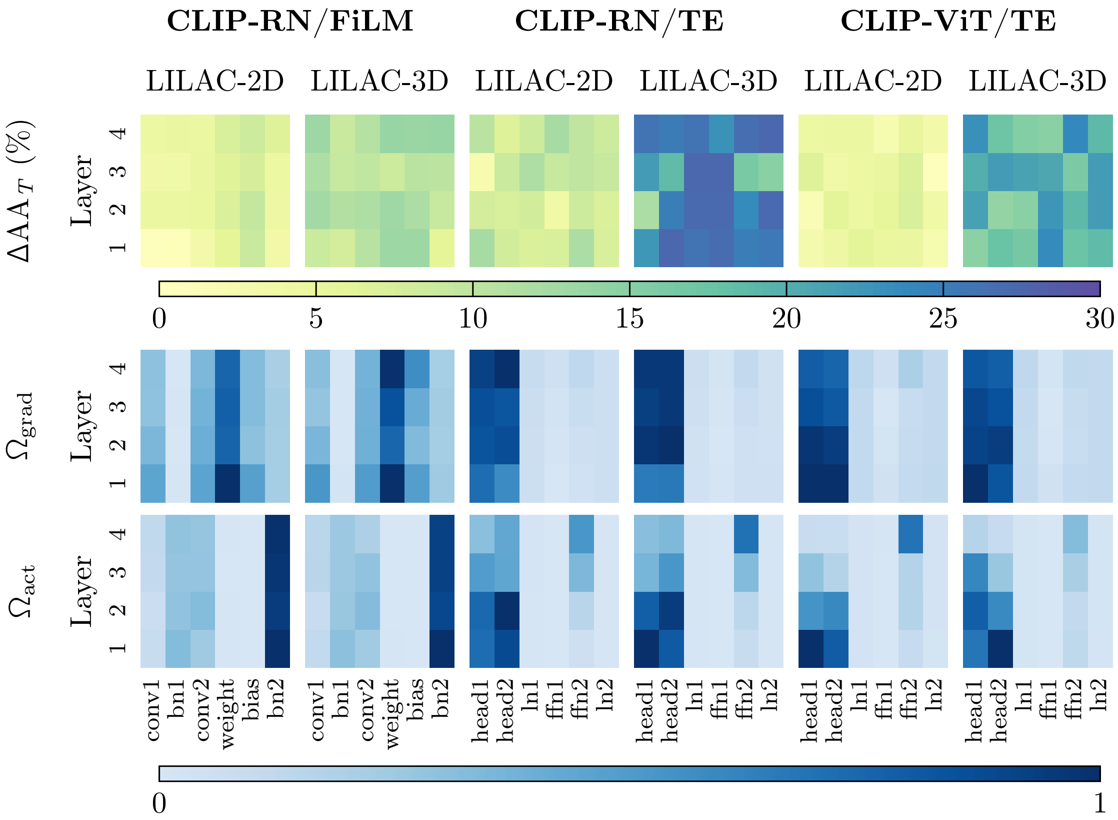


Figure 6.9: Effectiveness of per-module specialization and module importance measures. *1st row:* Accuracy gain of each specialized module $m \in \mathcal{M}$ over fine-tuning a monolithic network. Each field represents an experiment with $\mathcal{S} = \{m\}$. *2nd row:* Min-max normalized gradient-based importance score $\Omega_{\text{grad}}(m)$. *3rd row:* Min-max normalized activation-based importance score $\Omega_{\text{act}}(m)$.

tasks. Employing the same normalization constant α defined in Equation (6.7), we compute

$$\Omega_{\text{act}}(m) := \alpha \sum_{t=1}^{T} \sum_{(\boldsymbol{l}_t, \boldsymbol{o}_t) \in \mathcal{D}_t} \left| d_{\varphi^{t,m}} \left(h^{(L)}(\boldsymbol{l}_t), h^{(V)}(\boldsymbol{o}_t) \right) \right|, \tag{6.8}$$

where $d_{\varphi^{t,m}}(\cdot, \cdot)$ denotes the activation produced by module m of the modality interaction network d after training on task t, and $h^{(L)}(\cdot)$ and $h^{(V)}(\cdot)$ represent the outputs of the text and image encoders, respectively.

To evaluate whether these importance scores reliably predict the effectiveness of specializing a single module, we calculate the Pearson Correlation Coefficient (PCC; Pearson, 1895) between each module's gradient-based and activation-based importance scores ($\Omega_{\text{grad}}(m)$ and $\Omega_{\text{act}}(m)$; second and third row in Figure 6.9) and its corresponding accuracy gain from specialization ($\Delta\text{AA}_T(\{m\})$; first row in Figure 6.9). The resulting PCC values, provided in Table 6.2, illuminate the degree to

Importance Score	Model	LILAC-2D	LILAC-3D
Ω_{grad}	CLIP-RN/FiLM	+0.36	+0.41
	CLIP-RN/TE	−0.17	−0.32
	CLIP-ViT/TE	−0.10	−0.11
Ω_{act}	CLIP-RN/FiLM	−0.31	−0.41
	CLIP-RN/TE	−0.05	−0.30
	CLIP-ViT/TE	−0.03	−0.14

Table 6.2: Correlation between importance measures and module specialization efficacy. Reported values are Pearson correlation coefficients $\text{PCC}(\Omega(m), \Delta\text{AA}_T(\{m\}))$ between each module's importance score and the corresponding accuracy improvement (*cf*. Equation (6.4)) obtained by specializing that module individually (*i.e.*, $\mathcal{S} = \{m\}$) during SMS training. All coefficients are computed separately for each dataset and each modality interaction network configuration.

which each module's importance score aligns with the performance improvements obtained by making that module task-specific.

For FiLM-based modality interaction, the gradient-based importance score shows a moderate positive correlation with accuracy gains, suggesting that modules with higher parameter and gradient magnitudes are more likely to benefit from specialization in FiLM-based architectures. In contrast, for transformer-based modality interaction, the same gradient-based measure exhibits weak negative correlations, indicating that it does not reliably predict performance improvements from task-specialization.

Similarly, the activation-based importance score consistently yields negative correlation coefficients. Although the magnitudes of these correlations are relatively small in some cases, this overall negative trend suggests an inverse relationship between the typical activation level of a module and the performance gain from making it task-specific. In other words, modules that exhibit lower average activation magnitudes across tasks may, somewhat counterintuitively, offer greater specialization benefits. These findings deviate from the results reported by Jung et al. (2020), who observe a positive association between activation strength and module significance when training from scratch. A plausible explanation is that SMS trains modality interaction networks built on top of large-scale PTMs, whereas Jung et al. (2020) investigate a unimodal setup with small-scale models trained from scratch. In our scenario, certain modules may already be highly optimized and less adaptable under the influence of pretraining. Consequently, modules with lower activations—which have more "capacity to adapt"—may demonstrate greater improvements when specialized.

6.3.6 Baseline Comparison

Drawing on the analyses in Sections 6.3.4 and 6.3.5, we investigate whether the proposed SMS method can surpass common CL baselines on the LILAC datasets. Because the space of all possible module selection strategies grows exponentially with the number of modules in a network, we restrict our comparison to the following representative baselines: in alignment with our findings on layer-depth specialization, we compare to (i) specialization of the penultimate layer ($\mathcal{S}_{\text{third-layer}}$) for transformer-based modality interaction encoders (*i.e.*, CLIP-RN/TE and CLIP-ViT/TE) and (ii) specialization of the last layer ($\mathcal{S}_{\text{last-layer}}$) for FiLM-based and transformer-based modality interaction with ResNet-50 as vision encoder (*i.e.*, CLIP-RN/FiLM and CLIP-RN/TE). Since we found in Section 6.3.4 that specializing the second convolutional block of the FiLM-based modality interaction encoder is especially beneficial while being cheap in additional memory, we also compare to specialization of that intermediate block ($\mathcal{S}_{\text{conv2-second-layer}}$). Finally, we include two memory-efficient strategies for transformer-based modality interaction: specializing the first feed-forward block in the third layer of the transformer encoder ($\mathcal{S}_{\text{ffn1-third-layer}}$) and specializing one of the two layer normalization operations in each layer ($\mathcal{S}_{\text{ln1-all-layers}}$ for CLIP-RN/TE and $\mathcal{S}_{\text{ln2-all-layers}}$ for CLIP-ViT/TE), following the heuristics in Section 6.3.4.

Table 6.3 reports the average accuracies on both LILAC datasets, alongside the memory requirements for each chosen module selection strategy. We derive several key observations from these experiments. First, regularization- and replay-based baselines, exemplified by O-EWC and ER, often fail to outperform the vanilla SFT baseline. In some cases, the SFT accuracy hovers around 50%, which is equivalent to not learning the tasks, as this performance is equivalent to random guessing when choosing between two potential visual hypotheses. Even in scenarios where some learning takes place (*e.g.*, with CLIP-RN/FiLM), the additional performance gains from replay or regularization remain mostly modest.

In contrast, SMS consistently outperforms both regularization-based and replay-based methods, regardless of the specific module selection strategy used. Notably, although specializing an entire layer of the modality interaction network frequently proves as the strongest baseline, experiments involving transformer-based modality interactions reveal that less memory-intensive specialization strategies yield comparable or even superior performance. This phenomenon may be explained by the transformer's intrinsic ability to capture long-range dependencies across modalities, thereby facilitating effective task-specific adaptation with minimal module specialization.

Interestingly, we observe that for configurations employing a transformer-based modality interaction, the expert strategy outperforms the JFT baseline. Although JFT trains on all tasks simultaneously and can thus exploit task-sharing features, the specialized modules in the expert strategy appear to offer more targeted capacity for each task, ultimately leading to higher accuracy. This highlights a poten-

Model	Method	$\frac{\lvert\varphi^{\mathcal{S}}\rvert}{\lvert\varphi^{\mathcal{M}}\rvert}$ (%)	LILAC-2D	LILAC-3D
CLIP-RN/FiLM	JFT	–	66.5 ± 0.7	96.3 ± 0.4
	Expert	100	67.8 ± 0.2	92.6 ± 0.0
	SFT	–	52.7 ± 0.1	74.2 ± 1.8
	ER	–	56.7 ± 0.1	87.3 ± 0.2
	O-EWC	–	53.0 ± 0.2	77.5 ± 2.1
	SMS ($\mathcal{S}_{\text{last-layer}}$)	25	**61.8 ± 0.1**	**93.5 ± 0.1**
	SMS ($\mathcal{S}_{\text{bias-first-layer}}$)	11	61.7 ± 0.2	87.5 ± 0.3
	SMS ($\mathcal{S}_{\text{conv2-second-layer}}$)	1	57.8 ± 0.2	85.8 ± 0.2
CLIP-RN/TE	JFT	–	53.1 ± 0.5	65.5 ± 0.6
	Expert	100	63.1 ± 0.1	77.6 ± 0.1
	SFT	–	51.6 ± 0.3	53.6 ± 1.0
	ER	–	50.2 ± 0.1	54.3 ± 0.3
	O-EWC	–	51.3 ± 0.5	55.1 ± 0.9
	SMS ($\mathcal{S}_{\text{third-layer}}$)	25	57.7 ± 0.3	80.1 ± 0.3
	SMS ($\mathcal{S}_{\text{last-layer}}$)	25	63.6 ± 0.2	75.2 ± 0.6
	SMS ($\mathcal{S}_{\text{ffn1-third-layer}}$)	1	60.8 ± 0.3	**81.0 ± 0.2**
	SMS ($\mathcal{S}_{\text{ln1-all-layers}}$)	0.04	**64.4 ± 0.2**	80.7 ± 0.2
CLIP-ViT/TE	JFT	–	50.0 ± 0.2	54.2 ± 1.8
	Expert	100	55.5 ± 0.2	77.0 ± 0.1
	SFT	–	50.3 ± 0.1	52.0 ± 1.5
	ER	–	49.9 ± 0.1	50.0 ± 0.1
	O-EWC	–	50.1 ± 0.2	52.9 ± 2.0
	SMS ($\mathcal{S}_{\text{third-layer}}$)	25	**58.8 ± 0.2**	69.8 ± 0.4
	SMS ($\mathcal{S}_{\text{ffn1-third-layer}}$)	1	55.0 ± 0.3	**72.7 ± 0.4**
	SMS ($\mathcal{S}_{\text{ln2-all-layers}}$)	0.04	54.7 ± 0.3	70.9 ± 0.6

Table 6.3: Performance comparison of replay-based, regularization, and model decomposition methods in the TIL setting. Scores are reported for all three model configurations of CLIP as pretrained VLM with ResNet-50 (RN) or ViT-B/16 (ViT) as image encoder and with FiLM layers (FiLM) or transformer encoder layers (TE) as trainable modality interaction mechansisms. The reported evaluation metric is average accuracy after training AA_T (%). $|\varphi^{\mathcal{S}}|/|\varphi^{\mathcal{M}}|$ is the fraction of parameters in the modality interaction network that is task-specific. SMS baselines with TE are trained with A&C and adaptation step count $s = 10$. Best scores for each model and dataset are highlighted in bold. Standard errors after ±.

tial advantage of model decomposition strategies for CL: by isolating task-specific parameters, they reduce the negative effects of over-parameterization and interference, while allocating representational power where it is most needed.

Finally, across four of the six evaluated model-dataset configurations, SMS outperforms the expert strategy by specializing as few as 0.04% (for CLIP-RN/TE on

LILAC-2D) or 1% (for CLIP-RN/TE on LILAC-3D) of the total parameters. In some cases, such as with the CLIP-ViT/TE architecture, SMS effectively enables learning the multimodal LILAC tasks in the first place, whereas other baselines fail to do so. In summary, these results illustrate that by identifying and fine-tuning a small subset of parameters most critical for the target tasks, SMS avoids catastrophic interference, substantially reduces memory overhead, and can make the difference between learning and not learning at all.

6.3.7 Variability Analysis

In this section, we examine (i) the sensitivity of the proposed SMS method to variations in consolidation frequency within the A&C optimization framework, and (ii) the performance of SMS under different numbers of tasks to assess its robustness across varied CL contexts. This investigation is motivated by the inherent uncertainty in practical CL scenarios, where both the total number of tasks and the optimal number of adaptation steps are typically unknown beforehand.

Effect of consolidation frequency s. We seek to determine how often A&C training should alternate between task-specific adaptation and cross-task knowledge consolidation, and how this alternation impacts final performance. To this end, we measure the SMS performance with five different values of s in the TIL setting for all three model configurations, using the two LILAC datasets and the last-layer module selection strategy ($\mathcal{S}_{\text{last-layer}}$) as a representative SMS baseline. The results are provided in Table 6.4.

Among $s = 6$, $s = 8$, and $s = 10$—which correspond to a consolidation step every 6, 8, or 10 training iterations—no single setting emerges as consistently superior in all conditions. However, as s approaches its smallest possible value of 2 (*i.e.*, alternating one adaptation step with one consolidation step), the performance starts to decline. Despite this slight drop, SMS still outperforms many standard CL baselines (*cf*. Section 6.3.6). These findings suggest that overly frequent consolidation may lead to repeated overwriting of shared parameters, thereby inducing mild but noticeable forgetting. Overall, while performance remains relatively robust for a range of s values, setting s at a moderately higher value appears to provide more stable results.

Effect of task count T. We also aim to evaluate how sensitive SMS performance is when varying the number of tasks learned sequentially. In principle, as T grows, a CL model encounters more distributional shifts, which can pose additional challenges for knowledge retention and transfer. Similarly to our previous analysis of adaptation step counts, we measure SMS performance in the TIL setting—using $T \in \{6, 12, 24\}$—for all three model configurations, with the two LILAC datasets and the last-layer module selection strategy ($\mathcal{S}_{\text{last-layer}}$) serving as our representative SMS baseline. The results are presented in Table 6.5.

Model	Method	s	LILAC-2D	LILAC-3D
CLIP-RN/FiLM	SMS ($\mathcal{S}_{\text{last-layer}}$)	10	**61.8** ± **0.1**	**93.5** ± **0.1**
		8	61.3 ± 0.8	**93.5** ± **0.1**
		6	61.3 ± 0.4	93.2 ± 0.4
		4	59.8 ± 0.5	93.3 ± 0.2
		2	57.7 ± 0.4	93.0 ± 0.3
CLIP-RN/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	10	63.6 ± 0.2	75.2 ± 0.6
		8	61.6 ± 1.2	81.2 ± 0.1
		6	**65.3** ± **1.0**	**81.4** ± **0.4**
		4	64.3 ± 1.1	79.5 ± 1.5
		2	58.5 ± 1.1	77.5 ± 1.3
CLIP-ViT/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	10	56.2 ± 0.2	68.0 ± 0.9
		8	**57.5** ± **1.3**	**73.8** ± **0.6**
		6	55.5 ± 1.6	69.6 ± 4.1
		4	51.9 ± 1.2	68.5 ± 2.5
		2	52.1 ± 1.7	73.0 ± 0.3

Table 6.4: Variability of SMS across consolidation frequencies. Average accuracy after training AA$_T$ (%) is reported for model configurations that have transformer layers as trainable modality interaction mechanisms. For each PTM configuration, two specialization strategies $\mathcal{S}$ and five different adaptation step counts s are reported. Best scores for each model, specialization strategy, and dataset are highlighted in bold. The primary configuration used in the main experiments is shaded in gray.

For LILAC-2D, the overall final accuracies show relatively minor fluctuations across different task counts, whereas for LILAC-3D, the accuracies exhibit more substantial variation. Counterintuitively, performance may decline when training on fewer tasks. A plausible explanation is that a smaller number of tasks can lead to each task encompassing a broader and more heterogeneous distribution. In such cases, the model must address a wide range of subtasks or data clusters simultaneously, increasing the risk of internal interference and exposing the limitations of the small set of task-specific parameters. Conversely, distributing the same data across a larger number of tasks results in narrower, more homogeneous task distributions. This allows the model to specialize incrementally and consolidate knowledge more effectively. Consequently, more frequent but subtler distributional shifts, as observed with larger T, provide the model with additional opportunities to adapt and refine its learned representations, thus mitigating severe performance drops.

6.3.8 Combination with Other Continual Learning Methods

An advantage of SMS is that it can be orthogonally combined with CL methods relying on replay or regularization. Because the shared parameters updated dur-

Model	Method	T	LILAC-2D	LILAC-3D
CLIP-RN/FiLM	SMS ($\mathcal{S}_{\text{last-layer}}$)	24	63.2 ± 0.4	92.0 ± 0.6
		12	61.8 ± 0.1	93.5 ± 0.1
		6	59.6 ± 0.4	93.2 ± 0.2
CLIP-RN/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	24	66.0 ± 1.0	86.4 ± 0.4
		12	63.6 ± 0.2	75.2 ± 0.6
		6	59.6 ± 1.4	71.8 ± 1.9
CLIP-ViT/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	24	60.9 ± 0.3	79.0 ± 0.7
		12	56.2 ± 0.2	68.0 ± 0.9
		6	52.2 ± 0.7	67.1 ± 0.6

Table 6.5: Variability of SMS across task counts. Average accuracy after training AA_T (%) is shown for all model configurations trained with SMS with the final (fourth) layer of the modality interaction network specialized as representative module selection strategy. Results are reported for three different numbers of tasks. The primary configuration used in the main experiments is shaded in gray.

ing the consolidation phase in SMS remain susceptible to forgetting, we explore whether regularizing those parameters via O-EWC or performing replay during consolidation can further boost overall performance. In particular, we evaluate SMS in combination with ER or O-EWC in the TIL setting, using three model configurations, two LILAC datasets, and $\mathcal{S}_{\text{last-layer}}$ as a representative selection strategy. All parameter choices for ER and EWC match those in the main baseline comparisons.

As shown in Table 6.6, integrating SMS with ER or EWC enhances performance exclusively for transformer-based modality interaction configurations on LILAC-3D. In all other scenarios, performance remains unchanged or slightly declines. In complex multimodal settings—such as those in LILAC-3D—supplementing SMS with replay or regularization helps mitigate forgetting more effectively by reinforcing shared parameters during consolidation. Transformer-based modality interaction networks, with their reliance on fine-grained multimodal representations, profit most from these additional constraints because replay buffers or regularizers more effectively preserve feature alignments across sequential tasks.

By contrast, for simpler datasets or architectures (*e.g.*, LILAC-2D, FiLM-based fusion), SMS alone appears sufficient to consolidate knowledge both within and across tasks, making additional replay or regularization mechanisms less necessary or even counterproductive. Consequently, while combining SMS with replay or regularization can yield performance improvements under specific conditions, its effectiveness is highly sensitive to the complexity of the underlying data and model architecture. Nonetheless, as models continue to grow in size and VL datasets become increasingly diverse and complex, integrating SMS with other CL methods represents a promising direction for future research.

Model	Method	LILAC-2D	LILAC-3D
CLIP-RN/FiLM	SMS ($\mathcal{S}_{\text{last-layer}}$)	61.8 ± 0.1	93.5 ± 0.1
	w/ ER	61.8 ± 0.5	92.6 ± 0.5
	w/ O-EWC	62.0 ± 0.5	93.2 ± 0.4
CLIP-RN/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	63.6 ± 0.2	75.2 ± 0.6
	w/ ER	61.3 ± 1.9	80.0 ± 0.8
	w/ O-EWC	59.1 ± 1.6	80.2 ± 0.8
CLIP-ViT/TE	SMS ($\mathcal{S}_{\text{last-layer}}$)	56.2 ± 0.2	68.0 ± 0.9
	w/ ER	53.8 ± 1.2	74.8 ± 0.8
	w/ O-EWC	55.7 ± 1.3	74.9 ± 0.8

Table 6.6: Combination of SMS with other continual learning methods. Average accuracy after training AA_T (%) is reported for the two model configurations that are optimized using the A&C scheme. The SMS baselines are combined so that either rehearsal (ER) or parameter regularization (O-EWC) are applied during the consolidation phase of A&C. The final modality interaction network layer is employed as representative specialization strategies. The primary configuration used in the main experiments is shaded in gray.

6.4 Discussion

Our findings demonstrate that SMS is a powerful and flexible method for addressing the challenges of multimodal CL with foundation models. By selectively allocating task-specific parameters in only a small subset of the network, SMS strikes a balance between leveraging pretrained knowledge and accommodating novel task demands. Notably, this approach significantly outperforms standard CL baselines, including replay- and regularization-based methods, in a variety of experimental settings. Our results thus underscore the importance of carefully designing how and where to specialize parameters within a VL interaction network that integrates pretrained unimodal features from different sources.

A critical insight is that late-layer specialization tends to yield the greatest performance gains for most of the tested architectures and datasets. These observations align with prior research highlighting that late-stage representations capture more contextual and semantic details, making them most suited for task-specific adaptation. Moreover, we found that certain module types—such as biases in FiLM-based architectures—play disproportionately large roles in bridging the gap between general and task-specific representations, suggesting that a focus on task-specific shifting parameters and task-agnostic scaling parameters can be highly advantageous.

Our analyses further revealed that module importance metrics, commonly employed in pruning-based CL strategies for models trained from scratch, do not consistently exhibit strong correlations with the benefits of specialization in large PTMs. While gradient-based metrics demonstrated moderate predictive power for

FiLM-based configurations, activation-based scores generally showed negative correlations with actual performance gains. These findings suggest that pretraining alters the distribution and scale of layer activations, potentially obscuring the conventional signals used to guide parameter pruning or module reallocation. As a result, direct adaptation of pruning-based techniques may be inadequate to identify optimal specialization strategies in multimodal CL settings that leverage large-scale pretraining.

Despite its broad applicability, SMS exhibits several limitations. One key challenge is that, as the number of tasks increases, the risk of parameter fragmentation becomes more pronounced if too many modules are exclusively allocated to individual tasks, thereby limiting opportunities for effective parameter sharing. Although our experiments demonstrate that memory overhead remains manageable for a moderate number of tasks, scaling to dozens or even hundreds of tasks will require more sophisticated strategies for dynamic module reallocation and fine-grained resource sharing. Additionally, SMS currently assumes that task-specific information is available at inference time, a requirement that may not be feasible in many real-world CL scenarios.

6.5 Chapter Summary

Achieving an optimal balance between specialization and generalization remains a significant challenge when training agents on sequential language-conditioned tasks within visual environments. In this chapter, we addressed this challenge by reconceptualizing VL interaction networks as modular compositions to process and integrate modality-specific features extracted from foundation models. To facilitate a systematic evaluation, we introduced two diagnostic datasets which we used to assess various strategies for selectively specializing module types and layer depths across sequentially learned tasks. Our experiments revealed the limitations of relying solely on pruning-based importance metrics to integrate unimodal representations from diverse pretrained models, while demonstrating that even a small set of task-specific parameters from lightweight modules can significantly enhance CL performance—sometimes marking the difference between learning and not learning at all. These findings underscore the efficacy of selective specialization in preserving and extending task-specific knowledge.

Looking ahead, it is imperative to balance task-specific adaptation and cross-task generalization while minimizing memory overhead and relaxing the dependence on explicit task boundaries. There remains a critical need for more computationally efficient methods that achieve similarly promising results without relying on large-scale parameter storage. Therefore, in the next chapter, we introduce a task-agnostic, exemplar-free CL method based on noise augmentation in the latent feature space. By injecting carefully calibrated noise into pretrained representations, our method consolidates knowledge from previous tasks and adapts to new ones in real time, thereby meeting the practical requirements for scalable and effective multimodal CL in increasingly complex VL problems.

Chapter 7

Noise-augmented Multimodal Latent Replay

An artificial agent operating in a visual environment must flexibly identify and locate an ever-growing set of objects based on natural language instructions—a capability essential for many human-agent interaction applications. However, current CL research on object localization and segmentation is predominantly confined to vision-only inputs, limiting its effectiveness in multimodal contexts. At the same time, methods that integrate language grounding into a CL framework, such as the SMS method introduced in Chapter 6, often assume prior knowledge of task identities or rely on memory-intensive strategies, limiting their scalability.

In this chapter, we explore latent replay with targeted noise injection and introduce NLR, a novel replay-based multimodal CL method that augments latent representations with spatially aware statistical noise. By storing only a minimal set of feature vectors for each newly encountered object class, NLR preserves conceptual and object-specific knowledge without incurring prohibitive memory costs. In contrast to many existing baselines, NLR is exemplar-free, fully task-agnostic, supports online learning, and accommodates open-world free-form language and image inputs. Extensive experiments demonstrate that NLR outperforms architecture-based, regularization-based, and latent replay-based approaches across various CL scenarios while making less strict assumptions. Ultimately, NLR lays the groundwork for robust, scalable CL in open-world multimodal settings.[1]

7.1 Motivation

Integrating linguistic processing with visual perception challenges agents to harmonize multiple sensory modalities for effective real-world interaction. By combining language cues with visual data, agents can form robust adaptive representations that evolve as new observations arise. There is a growing consensus that CL must

[1] The source code is made available at `https://github.com/ky-ah/NLR`.

advance beyond simple label prediction problems, such as incremental classification, to address more intricate multimodal tasks and structured prediction problems that integrate diverse input sources (Mitchell et al., 2025). In this chapter, we therefore focus on *Continual Referring Expression Segmentation* (CRES), which segments image regions based on natural language descriptions and requires continual adaptation to diverse visual scenes and arbitrary text inputs referencing an ever-growing set of objects and attributes.[2]

As discussed in Section 3.2.2, effective operation in dynamic, unpredictable environments often means that the agent lacks explicit information about the current task, rendering *task-agnostic* learning indispensable for CRES. In addition, *exemplar-free* approaches are highly relevant in household and other sensitive contexts, where storing past exemplars is either infeasible or poses privacy concerns. Our SMS approach (*cf.* Chapter 6) circumvents the need to store historical data by isolating critical parameters, albeit under the strong assumption that explicit task information is available at all times. Parameter allocation methods such as WSN (*cf.* Section 3.2.5) and its soft-masking, regularization-based variant SPG (Konishi et al., 2023) refrain from storing complete parameter sets for every task but still require predefined task boundaries and task-specific information. Moreover, as the task count increases, these methods must protect an expanding portion of the parameter space, which can lead to model saturation. Similarly, prompt learning methods for foundation models (*cf.* Section 3.3.2), including approaches to multimodal CL (Jin et al., 2024), show promise yet typically need separate prompts per task, thereby precluding a fully task-agnostic approach.

We outlined in Section 3.2.5 two replay-based strategies to mitigate catastrophic forgetting in CL without storing raw data or exemplars: generative replay and latent replay. Generative replay (*e.g.*, Shin et al., 2017; Sun et al., 2020a) typically operates in a task-aware manner by using an auxiliary generative model that demands considerable computational and memory resources. Moreover, this approach is prone to adversarial attacks (Kang et al., 2023). In contrast, latent replay methods can be designed to be both exemplar-free and task-agnostic. However, existing latent replay approaches in CL often rely on statistical summaries of past data, and were originally developed for classification tasks (*e.g.*, Zhang et al., 2023). Their adaptation to continual image segmentation, as proposed by Chen et al. (2023), involves a simplification of the target by employing all-zero segmentation masks. This strategy leads to an underestimation of object boundaries and causes models to generate minimal or empty masks, a phenomenon commonly referred to as *background bias*.

Drawing on insights from Chapter 5 regarding the effectiveness of learned prototypes in pretrained latent spaces, we introduce **N**oise-augmented **L**atent **R**eplay (**NLR**), a versatile and efficient multimodal CL method that extends beyond stan-

[2] In the continual multi-task setup of CRES, each task features a different set of target object classes.

dard classification tasks.[3] Our method leverages prototypical, pretrained multimodal features (*e.g.*, from CLIP) enhanced by controlled noise injection. We apply *spatially aware* noise that minimally perturbs features critical to the target object or region of interest while more heavily perturbing background features. This dual-purpose strategy both mitigates overfitting to the limited set of features in the replay buffer and counteracts the model's background bias, *i.e.*, its tendency to underestimate salient target regions. Consequently, NLR expands latent space coverage and enables more precise segmentation while preserving its ability to recognize and segment previously encountered object categories.

The contributions made in this chapter are threefold:

(1) We introduce NLR, a versatile and efficient method for multimodal CL that uses a novel spatially aware noise injection mechanism to combat background bias in latent replay.

(2) We provide a detailed investigation of how replay frequency, buffer size, and background bias influence model performance in CRES, demonstrating that targeted noise boosts segmentation accuracy by guiding the model to more confidently predict foreground areas.

(3) We demonstrate that NLR achieves state-of-the-art performance on diverse CRES benchmarks in both CIL and OCL settings, all while being exemplar-free, resource-efficient, and fully task-agnostic.

The remainder of this chapter is structured as follows: in Section 7.2, we describe the theoretical foundation and implementation details of NLR. Section 7.3 outlines our training protocols, experimental results, and analyses that underscore the effectiveness of targeted noise in latent replay. We discuss broader implications in Section 7.4 and conclude the chapter in Section 7.5.

7.2 Proposed Method: NLR

An overview of our proposed NLR method is provided in Figure 7.1. The training pipeline comprises a pretrained VLM (denoted as PTM h), which contains both a text encoder $h^{(L)}$ and a visual feature extractor $h^{(V)}$, alongside a modality interaction network d_φ with trainable parameters φ and a PL g_ψ with trainable parameters ψ. We adopt the CL problem formulation outlined in Section 3.2.1. Consistent with Chapter 6, the VLM model architecture follows category (e) of

[3] It is noteworthy that while we demonstrate the effectiveness of NLR on the CRES problem, it is to a certain extent applicable to any unimodal or multimodal CL problem, including classification and regression, since it only requires a strong foundation model for latent feature extraction. More details on this are provided in Section 7.2.2.

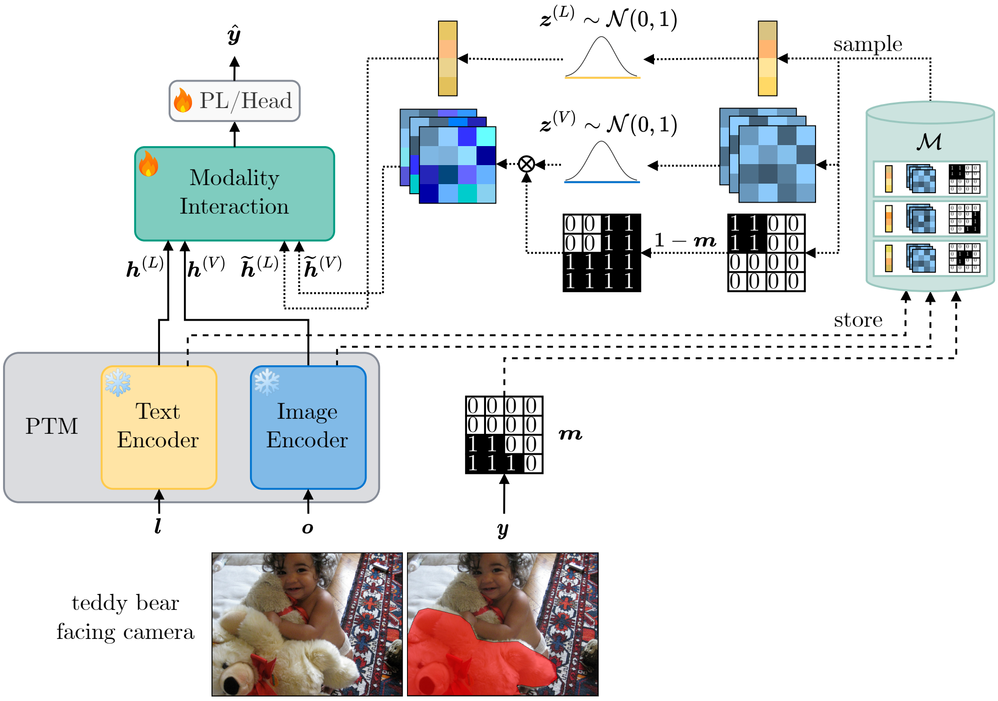


Figure 7.1: Overview of NLR. Given a textual referring expression $\boldsymbol{l}$ and an image $\boldsymbol{o}$, the objective in CRES is to produce a binary pixel mask $\hat{\boldsymbol{y}}$ that localizes the referenced object(s). The binary feature map mask $\boldsymbol{m}$ is derived via mean pooling over non-overlapping subregions in the ground-truth segmentation map $\boldsymbol{y}$. During the replay phase (dotted arrows), a batch is randomly sampled from $\mathcal{M}$, and statistical noise (indicated by $\boldsymbol{z}^{(V)}$ and $\boldsymbol{z}^{(L)}$) is applied to image and text features $\boldsymbol{h}^{(V)}$ and $\boldsymbol{h}^{(L)}$ to yield $\tilde{\boldsymbol{h}}^{(V)}$ and $\tilde{\boldsymbol{h}}^{(L)}$, respectively. To avoid excessive distortion of target object features, higher noise is assigned to the background and lower noise to the object/foreground, guided by the binary feature map mask $\boldsymbol{m}$.

the VLM taxonomy described in Section 2.2.3 in a way that h is a dual encoder and d_φ is a multi-layer modality interaction network.[4]

The n^{th} training input of the t^{th} task, denoted as $(\boldsymbol{x}_{t,n}, \boldsymbol{y}_{t,n}) \in \mathcal{D}_t$, consists of a textual referring expression and a visual scene $\boldsymbol{x}_{t,n} = (\boldsymbol{l}_{t,n}, \boldsymbol{o}_{t,n})$ with $\boldsymbol{o}_{t,n} \in \mathbb{R}^{H \times W \times 3}$, as well as a binary segmentation mask $\boldsymbol{y}_{t,n} \in \{0,1\}^{H \times W}$ as ground truth label. Here, $H \times W$ indicates the height and width of the image in pixels. For some test sample $(\boldsymbol{l}, \boldsymbol{o}) \in \mathcal{D}_{\text{test}}$, the text encoder $h^{(L)}$ processes $\boldsymbol{l}$ to produce latent textual features $\boldsymbol{h}^{(L)} \in \mathbb{R}^{d^{(L)}}$, and the visual feature extractor $h^{(V)}$ processes $\boldsymbol{o}$ to yield latent visual features $\boldsymbol{h}^{(V)} \in \mathbb{R}^{d^{(V)}}$:

$$\boldsymbol{h}^{(L)} = h^{(L)}(\boldsymbol{l}), \quad \boldsymbol{h}^{(V)} = h^{(V)}(\boldsymbol{o}). \tag{7.1}$$

[4] Unlike in Chapter 6, we assume that all parameters, particularly those trained in the CL setting, are shared across tasks and therefore omit any task-specific subscript or superscript notation.

Subsequently, the modality interaction network fuses the language and vision latents into a joint multimodal representation $\boldsymbol{h}$, calculated as

$$\boldsymbol{h} = d_{\varphi}\left(\boldsymbol{h}^{(L)}, \boldsymbol{h}^{(V)}\right). \tag{7.2}$$

Finally, the predicted segmentation mask containing the object referred to in the textual input is expressed as

$$\hat{\boldsymbol{y}} = g_{\psi}(\boldsymbol{h}) \tag{7.3}$$

with $\hat{\boldsymbol{y}} \in \mathbb{R}^{H \times W}$. This output is then used to compute a loss function that updates the trainable parameters φ and ψ. Although the dual encoders operate in a unified VL feature space and produce dense embeddings of the same dimensionality, we do not assume that $d^{(V)}$ is equal to $d^{(L)}$. In practice, when training modality interaction networks on multimodal features extracted from dual encoders (*e.g.*, Lüddecke and Ecker, 2022; Shridhar et al., 2022), image features passed to the interaction network are extracted prior to the final projection layer of the image encoder to preserve spatial information. Consequently, $d^{(V)}$ is typically larger than $d^{(L)}$.

7.2.1 Latent Replay for Continual Learning

Latent replay is not a novel technique for mitigating catastrophic forgetting in CL. However, its application has been predominantly investigated within generative networks or in the context of classification problems, the limitations of which we discussed in Section 7.1.

Essentially, methods for latent replay maintain a per-class buffer $\mathcal{M}_c$ for each class $c \in \{1, \ldots, C\}$, each with a fixed capacity B. Collectively, these form the overall memory buffer $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_C\}$, containing a maximum of $B \cdot C$ samples. Since the object class identifier is directly derivable from the input text or referring expression—or, in classification problems, from the label itself—additional class labels are not necessary. Maintaining distinct buffers for each class facilitates a more balanced replay, which is essential given the common class imbalances in real-world VL datasets (*e.g.*, the class *person* often appears more frequently than others).

Each per-class buffer $\mathcal{M}_c$ is populated using reservoir sampling, as detailed in Section 3.2.5. Furthermore, a replay frequency R_F is defined in a way that every R_F iterations (or update steps), a batch b_R of size $|b_R|$ is randomly sampled from $\mathcal{M}$ for replay. In the case of our NLR method, b_R consists of tuples $(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{m})$ that contain image and text *features* rather than the original input, along with the mask indicating the target region in the feature space (*cf.* Section 7.2.2). Importantly, by storing $\boldsymbol{m}$ as a binary mask instead of raw image data, we minimize privacy risks by retaining only essential segmentation details without exposing sensitive visual content, thereby complying with privacy storage restrictions.[5]

[5] In object localization/detection tasks, $\boldsymbol{m}$ may also be derived from a bounding box instead of a pixel mask.

7.2.2 Spatially Aware Noise Augmentation

We propose a principled procedure for injecting *spatially aware* noise into image features extracted by both CNN-based and transformer-based PTMs, the prevailing backbones for contemporary foundation models in CV and VL grounding (*cf.* Sections 2.1.3 and 2.2.1). This procedure requires a consistent mapping between pixels in the ground-truth segmentation mask $\boldsymbol{y}$ and their associated regions in feature space.

Notably, when such mapping is unavailable—*e.g.*, in image classification or regression problems, where labels convey no spatial information—we consider two adaptations. First, we dispense with spatially aware noise and instead inject a small, feature-wise uniform noise. In the ablation study (*cf.* Section 7.3.7), we will demonstrate that even this naive variant improves the performance of latent replay in multimodal CL settings. Second, to retain spatially aware noise as a central contribution of NLR, we employ an auxiliary image segmentation network that partitions the input image $\boldsymbol{o}$ into foreground and background.[6] Denoting the resulting segmentation mask by $\hat{\boldsymbol{o}}$, we compute the binary feature map mask $\boldsymbol{m}$ via Equations (7.8) and (7.9), substituting $\hat{\boldsymbol{o}}$ for $\boldsymbol{y}$. This strategy makes NLR fully compatible with standard image classification or regression pipelines, albeit at the cost of the computational and memory overhead introduced by the auxiliary model.

Incorporating noise directly into the feature space counteracts overfitting by preventing the model from memorizing specific latent representations. Conceptually, this is akin to established augmentation techniques such as color jittering applied in the original input space. However, applying too much noise to the “foreground pixels” in the latent space—those encoding the target object—can degrade performance by obscuring critical information.

Consequently, our aim is twofold: (i) introduce minimal perturbations to the target object's latent features in order to preserve their discriminative power, and (ii) add stronger noise to background pixels, which are less crucial for localizing the target object. To realize this, we propose a spatially aware noise augmentation strategy that relies on the ground-truth segmentation mask stored in the memory buffer. Using this mask, we selectively target background features for noise injection, thereby preserving the integrity of the target object representation and promoting improved generalization.

Spatial alignment and mask computation. For CNN-based feature extractors, we benefit from the inductive bias that preserves approximate spatial alignment between an input image and its feature maps. Let the feature map have dimensions $d^{(V)} = h \times w \times c$, where h and w denote the spatial dimensions, and c

[6] The network operates in a zero-shot manner and thus requires neither fine-tuning nor domain adaptation; it merely needs to distinguish foreground objects from background.

is the number of channels. Suppose

$$\boldsymbol{y} : \{0, \ldots, H-1\} \times \{0, \ldots, W-1\} \to \{0, 1\} \tag{7.4}$$

is the binary segmentation map associated with a replay sample of features $\boldsymbol{h}^{(V)}$ and $\boldsymbol{h}^{(L)}$. We construct a binary feature map mask

$$\boldsymbol{m} : \{0, \ldots, h-1\} \times \{0, \ldots, w-1\} \times \{0, \ldots, c-1\} \to \{0, 1\} \tag{7.5}$$

by performing mean pooling over non-overlapping subregions of $\boldsymbol{y}$ that correspond to positions (i, j) in the downsampled feature space. Concretely, let

$$s_H = \frac{H}{h} \quad \text{and} \quad s_W = \frac{W}{w}. \tag{7.6}$$

and define the region

$$\mathcal{R}_{i,j} \;=\; \big\{(p, q) \mid p \in \big[\lfloor s_H \cdot i \rfloor, \lfloor s_H \cdot (i+1) \rfloor\big],\ q \in \big[\lfloor s_W \cdot j \rfloor, \lfloor s_W \cdot (j+1) \rfloor\big]\big\} \tag{7.7}$$

within $\boldsymbol{y}$. The mean value of $\boldsymbol{y}$ over $\mathcal{R}_{i,j}$ is

$$\operatorname{mean}\big(\boldsymbol{y}, \mathcal{R}_{i,j}\big) = \frac{1}{\big|\mathcal{R}_{i,j}\big|} \sum_{(p,q) \in \mathcal{R}_{i,j}} \boldsymbol{y}(p, q). \tag{7.8}$$

and the mask $\boldsymbol{m}$ at position (i, j, k) is defined by

$$\boldsymbol{m}_{i,j,k} \;=\; \begin{cases} 1 & \text{if } \operatorname{mean}\big(\boldsymbol{y}, \mathcal{R}_{i,j}\big) \geq \epsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{7.9}$$

Notably, the mask remains consistent across all channels so that $\boldsymbol{m}_{i,j,k} = \boldsymbol{m}_{i,j,l}$ for $k \neq l$. The threshold parameter ϵ regulates how cautiously subregions of the feature map are classified as belonging to the target object. Throughout our experiments, we set $\epsilon = 0.5$, as preliminary analyses showed that performance remains robust for moderate variations in this parameter and that $\epsilon = 0.5$ strikes a useful balance between sensitivity and specificity.

In a transformer-based image encoder, the original image of size $H \times W$ is divided into M patches (for a typical ViT, via a non-overlapping grid of patch size, *e.g.*, 16×16), and each patch is linearly projected to a flattened embedding vector. Let M be the total number of patch embeddings extracted from a ViT encoder for an input image of size $H \times W$, arranged in an $h \times w$ grid such that $M = h \times w$. We assume these patch embeddings maintain approximate spatial correspondence. Consequently, we can apply Equations (7.6) to (7.8) in the same way as for CNN-extracted feature maps. Then, for a threshold $\epsilon \in [0, 1]$, the mask value for each patch is set to

$$m_{i,j} \;=\; \begin{cases} 1 & \text{if } \operatorname{mean}\big(\boldsymbol{y}, \mathcal{R}_{i,j}\big) \geq \epsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{7.10}$$

Rewriting this as a single index $l \in \{0, \ldots, M-1\}$, $l = i \cdot w + j$, we obtain a length-M binary mask $\boldsymbol{m}$ over image patches:

$$\boldsymbol{m} = (m_0, m_1, \ldots, m_{M-1}) \tag{7.11}$$

Feature variability and noise scaling. A key consideration is the appropriate level of noise to inject along each feature dimension. To achieve this, we scale the noise magnitude by the empirical standard deviation of each feature, which reflects the overall variability across the dataset. This strategy preserves essential information by ensuring that features with high variability are perturbed more significantly, while more stable features experience only slight perturbations.

At each iteration i, we sample a batch $b_{R,i} \sim \mathcal{M}$ consisting of tuples $(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{m})$. From this batch, we compute the mean image and text feature vectors, $\boldsymbol{\mu}_i^{(V)} \in \mathbb{R}^{d^{(V)}}$ and $\boldsymbol{\mu}_i^{(L)} \in \mathbb{R}^{d^{(L)}}$, as

$$\begin{aligned} \boldsymbol{\mu}_i^{(V)} &= \frac{1}{|b_R|} \sum_{\boldsymbol{h}^{(V)} \in b_{R,i}} \boldsymbol{h}^{(V)}, \\ \boldsymbol{\mu}_i^{(L)} &= \frac{1}{|b_R|} \sum_{\boldsymbol{h}^{(L)} \in b_{R,i}} \boldsymbol{h}^{(L)}. \end{aligned} \tag{7.12}$$

Using these mean vectors, we then compute the standard deviations for the image and text features, $\boldsymbol{\sigma}_i^{(V)} \in \mathbb{R}^{d^{(V)}}$ and $\boldsymbol{\sigma}_i^{(L)} \in \mathbb{R}^{d^{(L)}}$, as follows:

$$\begin{aligned} \boldsymbol{\sigma}_i^{(V)} &= \sqrt{\frac{1}{|b_R|-1} \sum_{\boldsymbol{h}^{(V)} \in b_{R,i}} \left(\boldsymbol{h}^{(V)} - \boldsymbol{\mu}_i^{(V)}\right)^2}, \\ \boldsymbol{\sigma}_i^{(L)} &= \sqrt{\frac{1}{|b_R|-1} \sum_{\boldsymbol{h}^{(L)} \in b_{R,i}} \left(\boldsymbol{h}^{(L)} - \boldsymbol{\mu}_i^{(L)}\right)^2}. \end{aligned} \tag{7.13}$$

Larger replay batch sizes $|b_R|$ yield more accurate estimates of these statistics. In practice, we observed that using $|b_R| = 16$ or more provides sufficiently accurate estimates. When batch sizes are too small, the mean and standard deviation can be computed over the entire observed dataset—omitting the iteration subscript. In this case, the statistics $\boldsymbol{\mu}^{(V)}$, $\boldsymbol{\mu}^{(L)}$, $\boldsymbol{\sigma}^{(V)}$, and $\boldsymbol{\sigma}^{(L)}$ are calculated as

$$\begin{aligned} \boldsymbol{\mu}^{(V)} &= \frac{1}{N} \sum_{t=1}^{T} \sum_{n=1}^{N_t} \boldsymbol{h}_{t,n}^{(V)}, \\ \boldsymbol{\mu}^{(L)} &= \frac{1}{N} \sum_{t=1}^{T} \sum_{n=1}^{N_t} \boldsymbol{h}_{t,n}^{(L)}, \\ \boldsymbol{\sigma}^{(V)} &= \sqrt{\frac{1}{N-1} \sum_{t=1}^{T} \sum_{n=1}^{N_t} \left(\boldsymbol{h}_{t,n}^{(V)} - \boldsymbol{\mu}^{(V)}\right)^2}, \\ \boldsymbol{\sigma}^{(L)} &= \sqrt{\frac{1}{N-1} \sum_{t=1}^{T} \sum_{n=1}^{N_t} \left(\boldsymbol{h}_{t,n}^{(L)} - \boldsymbol{\mu}^{(L)}\right)^2}, \end{aligned} \tag{7.14}$$

where $N = \sum_{t=1}^{T} N_t$ denotes the total number of training instances up to task T. These statistics can be updated incrementally on a per-sample basis. In particular, Welford's method (Welford, 1962) is commonly used to compute the standard deviation in a numerically stable manner without retaining all previous samples.[7]

Noise injection mechanism. To introduce noise into the latent representation, let $\boldsymbol{z}^{(V)} \in \mathbb{R}^{d^{(V)}}$ and $\boldsymbol{z}^{(L)} \in \mathbb{R}^{d^{(L)}}$ be vectors whose individual components are independently sampled from a standard normal distribution:

$$\boldsymbol{z}_i^{(V)} \sim \mathcal{N}(0,1) \quad \text{and} \quad \boldsymbol{z}_j^{(L)} \sim \mathcal{N}(0,1) \tag{7.15}$$

for $i \in \{1, \dots, d^{(V)}\}$ and $j \in \{1, \dots, d^{(L)}\}$. For some tuple $(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{m})$ of a replay batch $b_{R,i}$, both spatially aware and global noise are then added to the image features:

$$\tilde{\boldsymbol{h}}^{(V)} = \boldsymbol{h}^{(V)} + \alpha\, \boldsymbol{\sigma}_i^{(V)} \big(\boldsymbol{z}^{(V)} \otimes (\boldsymbol{1} - \boldsymbol{m})\big) + \beta\, \boldsymbol{\sigma}_i^{(V)} \boldsymbol{z}^{(V)}, \tag{7.16}$$

where α and β respectively control the magnitude of the spatially aware and global noise components. The term $\boldsymbol{z}^{(V)} \otimes (\boldsymbol{1} - \boldsymbol{m})$ ensures that stronger noise is predominantly injected into background regions. By contrast, text features do not encode explicit spatial structure, so we only apply global noise, yielding

$$\tilde{\boldsymbol{h}}^{(L)} = \boldsymbol{h}^{(L)} + \gamma\, \boldsymbol{\sigma}_i^{(L)} \boldsymbol{z}^{(L)}, \tag{7.17}$$

with γ specifying the overall noise level.[8] Ultimately, rather than repeatedly retrieving the same stored features $(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)})$ from the memory buffer, we introduce a stochastic augmentation step in each replay iteration by sampling statistical noise vectors to produce new pseudo-realistic features $(\tilde{\boldsymbol{h}}^{(V)}, \tilde{\boldsymbol{h}}^{(L)})$. As demonstrated later in this chapter, this augmentation strategy enhances the model's robustness to variations in the feature space, thereby reducing the likelihood of overfitting to the limited replay samples. In addition, it mitigates the background bias inherent in localizing only a small region within an image, as the model is encouraged to expand the predicted foreground region. The training and testing procedures for the NLR method are summarized in Algorithms 7 and 8, respectively.

7.3 Experiments

In this section, we present a series of experiments evaluating the performance of our NLR method for CRES. We begin by describing the benchmarks and training details, followed by an investigation of how memory buffer size and replay frequency

[7] For our experiments in Section 7.3, we employ the per-batch computation of feature statistics as described in Equation (7.12) and Equation (7.13).

[8] Although it is conceivable to apply targeted noise to text features that takes into account tokens that do or do not include a reference to the target object, we leave this investigation for future work.

Algorithm 7: NLR Training

```
Input: modality interaction parameters $\varphi$, PL parameters $\psi$, per-class
       buffer capacity $B$, replay frequency $R_F$, noise parameters $\alpha$, $\beta$, $\gamma$
# Initialization
$\mathcal{M} \leftarrow \emptyset$
# Continual Learning with NLR
for iteration $i = 1, 2, \ldots$ do
    if $i$ mod $R_F = 0$ then
        $b_{R,i} \leftarrow$ sample($\mathcal{M}$)
        compute $\boldsymbol{\mu}_i^{(V)}$ and $\boldsymbol{\mu}_i^{(L)}$ using Equation (7.12)
        compute $\boldsymbol{\sigma}_i^{(V)}$ and $\boldsymbol{\sigma}_i^{(L)}$ using Equation (7.13)
        for every sample $(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{m}) \in b_{R,i}$ do
            collect $\tilde{\boldsymbol{h}}^{(V)}$ and $\tilde{\boldsymbol{h}}^{(L)}$ using Equations (7.16) and (7.17)
            compute $\mathcal{L}(\tilde{\boldsymbol{h}}^{(V)}, \tilde{\boldsymbol{h}}^{(L)}, \boldsymbol{y})$
        end
        update $\varphi$ and $\psi_t$ using Adam
    end
    else
        $b_i \leftarrow$ next batch from the stream
        for every sample $(\boldsymbol{l}, \boldsymbol{o}, \boldsymbol{y}) \in b_i$ do
            collect $\boldsymbol{h}^{(V)}$ and $\boldsymbol{h}^{(L)}$ using Equation (7.1)
            compute $\mathcal{L}(\boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{y})$
            compute $\boldsymbol{m}$ using Equation (7.9)/Equation (7.10)
            write($\mathcal{M}, \boldsymbol{h}^{(V)}, \boldsymbol{h}^{(L)}, \boldsymbol{m}, B$)
        end
        update $\varphi$ and $\psi_t$ using Adam
    end
end
```

Algorithm 8: NLR Testing

```
Input: trained parameters $\varphi \cup \psi$
for every sample $(\boldsymbol{l}, \boldsymbol{o}) \in \mathcal{D}_{\text{test}}$ do
    collect $\boldsymbol{h}(\boldsymbol{x})$ using Equation (7.2)
    predict $\hat{\boldsymbol{y}}$ using Equation (7.3)
end
```

affect the latent replay strategy. Next, we assess how much spatially aware and global noise to inject in image features and demonstrate how noise augmentation alleviates background bias, a common challenge in sequentially learned segmentation tasks. Building on these findings, we compare NLR with recent state-of-the-art baselines in both CIL and OCL settings. Finally, we conduct ablation studies and analyze the robustness of our method to varying replay frequencies and numbers of tasks.

Dataset	T	N_{train}	N_{test}	C
PhraseCut100	20	170 095	10 884	100
ADE20K	20	89 406	9 307	150
RefCOCO	12	120 624	10 834	80

Table 7.1: Summary of referring expression segmentation datasets. Reported is the number of tasks (T) used in the all experiments in this chapter unless otherwise specified, along with the number of training samples (N_{train}), validation/test samples (N_{test}), and distinct object classes (C). The datasets are partitioned by object classes.

7.3.1 Benchmarks

In this chapter, we evaluate CL methods using three distinct split datasets for CRES. In each of the datasets, an example includes an input image and an accompanying text referring to one or more objects. The goal is to produce a binary pixel mask that delineates the location(s) of the referenced object(s) within the image. Since these datasets include only training and test sets, we reserve a randomly sampled 10% subset of the training data for validation. An overview of the key statistics for all datasets used in our baseline comparisons is provided in Table 7.1.

The first dataset, PhraseCut100, is a subsampled version of the original VGPhraseCut dataset (Wu et al., 2020), a large-scale benchmark for referring expression segmentation in natural scenes featuring more than 3 000 distinct object classes. For our experiments, we focus on the 100 most frequent classes, resulting in 66 034 unique training images and 2 775 testing images. The second dataset is a referring expression segmentation variant of the ADE20K image segmentation benchmark (Zhou et al., 2017). In this version, referring expression prompts are generated from CLIP inference templates combined with the corresponding object class names. ADE20K contains 17 277 training images and 1 743 testing images covering 150 object classes. The third dataset, RefCOCO (Kazemzadeh et al., 2014), is widely used in VL grounding for object detection and segmentation tasks. Derived from MSCOCO (Lin et al., 2014), it comprises images spanning 80 object classes paired with free-form natural language referring expressions, with 16 994 training images and 1 500 testing images.

7.3.2 Training Details

Experiments are conducted using a model configuration that employs CLIP with a pretrained ResNet-50 feature extractor as dual encoder VLM and a FiLM-based modality interaction network closely following the semantic stream of the CLIPort architecture (Shridhar et al., 2022). The semantic stream in CLIPort is specifically designed for predicting object affordances, *i.e.*, how objects can be manipulated or interacted with in a scene. By leveraging FiLM, the visual features are modulated according to textual cues, which allows the model to highlight salient regions and effectively capture object-specific properties critical for affordance prediction. This process is closely related to semantic segmentation, since language-based inputs are aligned with the spatial understanding and localization of objects.[9]

The parameters φ of the modality interaction network d_φ are trained for ten epochs ($E = 10$) in the CIL setting and for one epoch ($E = 1$) in the OCL setting. The batch size is $|b| = 32$ for CIL and $|b| = 1$ for OCL (*cf*. Table 3.1). Training on PhraseCut100 and ADE20K is performed using the Tversky loss (Salehi et al., 2017) with $\alpha_\text{T} = 0.2$ and $\beta_\text{T} = 0.9$, while training on RefCOCO is performed using Dice loss (Sudre et al., 2017). Both loss functions are well suited for referring expression segmentation tasks that typically feature a significant imbalance between few foreground pixels and many background pixels. Optimization is performed with Adam (momentum of 0.9) and a cosine annealing scheduler. Additionally, replay batches are drawn with sizes of $|b_R| = 32$ in CIL and $|b_R| = 16$ in OCL.

During training, each input image undergoes a series of preprocessing steps. Specifically, images are randomly flipped horizontally with a probability of 0.5. Subsequently, a random photometric distortion is applied, independently adjusting brightness, contrast, saturation, and hue within a 10% range. Next, each image is randomly rotated by an angle within 10 degrees. After these augmentations, the images undergo normalization to standardize pixel intensity values. Finally, all processed images are uniformly resized to a resolution of 224 × 224 pixels. During testing, images only undergo normalization and resizing to the standard input dimension of 224 × 224 pixels, without applying any augmentation steps.

For performance comparisons across all split CRES datasets, we first tuned the hyperparameters on the SFT baseline trained on PhraseCut100 to determine the optimal learning rates for each CL setting, and then applied these rates to the other baselines. The final learning rate is 3×10^{-3} for both OCL and CIL settings. We set the spatial noise coefficient $\alpha = 1.5$ and the global noise coefficient $\beta = 0.25$ (*cf*. Section 7.3.4) for the noise applied to image features, and use $\gamma = 0.05$ for the noise applied to the text features.

Similarly to Chapters 4 to 6, we compare our method with Sequential Fine-Tuning (**SFT**) and Joint Fine-Tuning (**JFT**), which represent monolithic network training without CL mechanisms in the sequential and i.i.d. training settings, respectively. We also compare with **WSN** (Kang et al., 2022) as representative parameter allocation approach, and with **SPG** (Konishi et al., 2023) and **O-EWC** (*cf*. Section 3.2.5) as regularization-based methods. For O-EWC, we set $\lambda_\text{O-EWC} = 1\,000$ and $\gamma_\text{O-EWC} = 0.9$.[10] Furthermore, we include in our comparison three methods

[9] The model configuration used in this chapter is specifically tailored for segmentation tasks; as a result, its modality interaction network architecture differs from the CLIP-RN/FiLM variant described in Chapter 6. For additional details on the the chosen model configuration, refer to Shridhar et al. (2022).

[10] The search spaces for O-EWC hyperparameters defined as follows: $\lambda_\text{O-EWC} = 10^x$, $x \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$ and $\gamma_\text{O-EWC} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

employing latent replay: the standard **LR** method, which follows the procedure described in Section 7.2.1, as well as modified versions of **DER** and **DER++** (Buzzega et al., 2020) that replay latent features instead of exemplars, which however have similar behavior to the original methods due to frozen encoder networks. The DER/DER++ hyperparameters are set to $\alpha_{\text{DER}} = \alpha_{\text{DER++}} = 0.1$ and $\beta_{\text{DER++}} = 0.9$.[11] Both DER and DER++ incorporate a distillation loss, controlled by α, to align the network's output logits with those stored in the replay buffer. Given our observation that a very small value of α yields the best performance, we suspect that strong knowledge distillation—which forces the model to recover older logits—may not be well-suited for CRES tasks.

We report average accuracy after training on the final task as the primary evaluation metric, denoted by AA_T (*cf*. Equation (3.10)), where $a_{T,t}$ represents the accuracy of correctly identifying the object(s) referred to in the textual input of a sample from the t^{th} task following training on the T^{th} (*i.e.*, last) task. An object is considered correctly located if the mean Intersection-over-Union (IoU) between the predicted pixel mask and the ground-truth mask—computed over both background and foreground pixels—exceeds 0.5. We additionally report the forgetting rate FM_T (*cf*. Equation (3.8)) measured after CL training. For reproducibility, unless stated otherwise, all experiments in this chapter are conducted using five fixed random seeds (1993–1997), which affect both task order and random initialization of trainable parameters.

7.3.3 Memory Buffer Size in Latent Replay

Our primary objective is to systematically investigate how the performance of latent replay methods for CL depends on (i) the number of representative latent features per object class stored in the memory buffer and (ii) the replay frequency, *i.e.*, how often a batch of latents is retrieved to retrain the modality interaction network. To this end, we measure average accuracy after training on $T = 20$ tasks, experimenting with seven different memory buffer sizes (powers of two from 1 to 64) and five replay frequencies. Notably, when $R_F = 2$, a batch is drawn from memory after every batch from the data stream, mirroring the sampling rate used in DER and DER++.

As illustrated in Figure 7.2, performance consistently improves with larger memory buffers, likely because additional stored latent representations per class provide a more comprehensive coverage of the feature distribution. In the case of OCL training on PhraseCut100, performance converges at a buffer size of $B = 16$, beyond which further expansion yields only marginal gains. A similar observation can be made for OCL training on ADE20K, where the convergence for most replay frequency settings occurs already at a buffer size of $B = 8$.

[11] The DER/DER++ hyperparameters were searched over the interval $\{0, 0.1, \ldots, 0.9, 1\}$. Note that β is only defined for DER++.

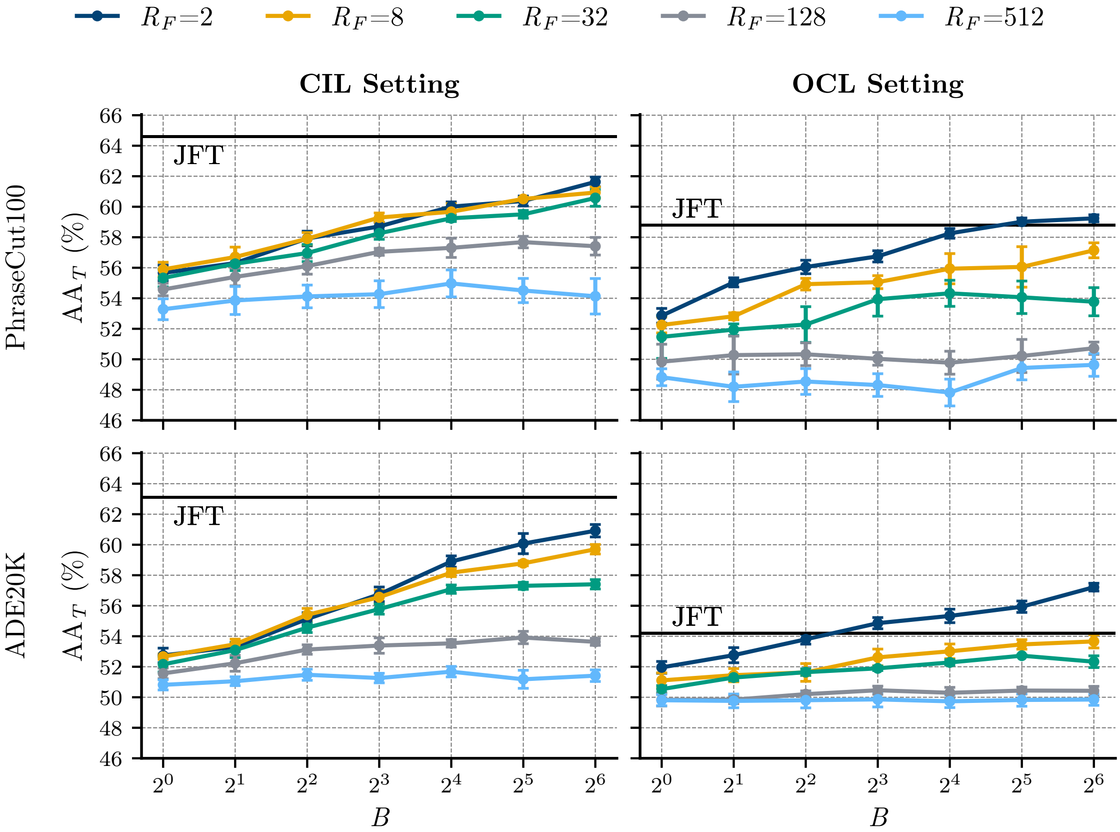


Figure 7.2: Effect of replay frequency and buffer size on the efficacy of latent replay. The plot shows the effects of varying buffer sizes (powers of two) and replay frequencies R_F on two CRES datasets in both the CIL setting ($E = 10$, $|b| = 32$) and the OCL setting (single-epoch, $E = 1$, and per-sample updates, $|b| = 1$). In the OCL setting, latent replay uses $|b_R| = 16$, which explains the superior performance of the LR baseline over JFT, suggesting that the benefits of larger batch sizes can outweigh strict i.i.d. assumptions in online learning. Vertical lines indicate standard error bars.

Furthermore, in the CIL setting, replay frequency exerts limited influence on performance unless it becomes excessively sparse (as at $R_F = 128$ or $R_F = 512$). By contrast, replay frequency has a pronounced effect in the OCL setting, where more frequent replay steps are favorable. Consequently, unless otherwise stated, we adopt $R_F = 2$ for the OCL setting and $R_F = 32$ for the CIL setting in subsequent experiments.

7.3.4 Balancing Spatially Aware and Global Noise

In Section 7.2.2, we propose to augment image features by combining spatially aware noise augmentation with a mild degree of noise applied across *all* feature dimensions. The primary objective is to prevent overfitting to the relatively limited latent representations stored in $\mathcal{M}$, while still preserving crucial information. By

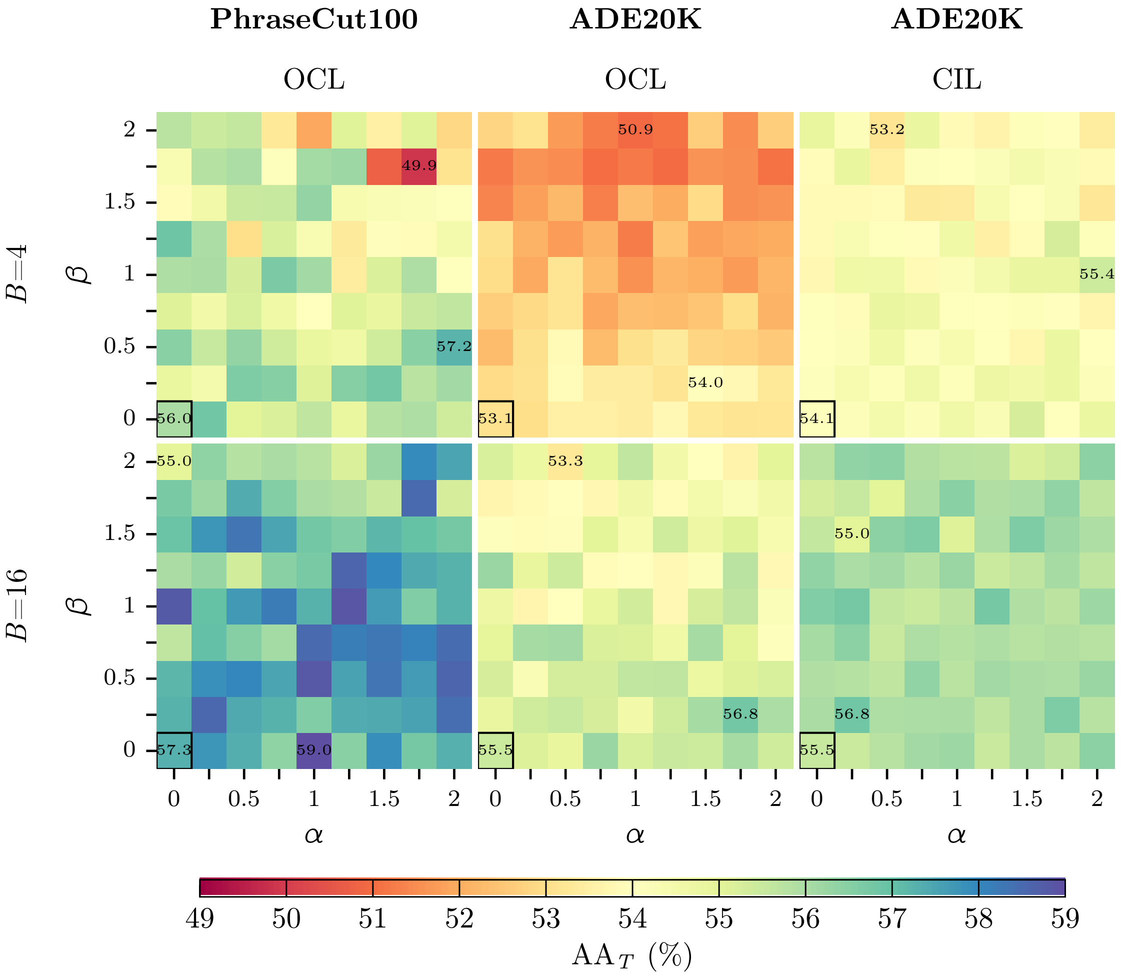


Figure 7.3: Effect of spatially aware vs. global noise on performance. The heatmaps illustrate the effects of the spatially aware noise level α and the global noise level β of NLR without noise applied to the text features (*i.e.*, $\gamma = 0$). Results are shown under both the CIL and OCL settings, using two distinct per-class buffer capacity values B and a random seed of 1993. The LR baseline, which does not incorporate noise, is highlighted with a black box; the highest, lowest, and LR baseline scores are displayed as text.

introducing small perturbations even to those features representing the region of interest, we aim to achieve a broader coverage of the PTM's latent space and ultimately promote better generalization. To this end, our objective is to determine the appropriate levels of both spatially aware and global noise to achieve this enhanced generalization.

Figure 7.3 depicts the average performance of latent replay with noise augmentation applied to image features in both OCL and CIL settings, varying the levels of spatially aware and global noise. Each block of the figure displays configurations where points below the diagonal correspond to a larger spatially aware noise coefficient α relative to the global noise coefficient β. The bottom left corner of each block denotes the LR baseline without noise.

The results indicate that excessive global noise undermines performance, often falling below the LR baseline. In contrast, combining a stronger spatially aware noise coefficient with a small global noise component (as shown in the bottom-right region of each block) yields performance improvements between up to 0.9% and 1.7%, depending on the particular values of α and β. These results suggest that moderate perturbations applied to all image features, together with more pronounced noise on background image features, enhance the effectiveness of stored latents in the buffer. Notably, these gains persist even when the per-class buffer size is very small ($B = 4$), highlighting the robustness of noise-augmented latent replay in data-scarce conditions.

7.3.5 Addressing the Background Bias with Targeted Noise

In referring expression segmentation, the model is tasked with isolating the specific object or region described by a textual query within an image. Because only a small number of pixels typically belong to the target object, with most pixels constituting the background, a model that is not well trained to recognize that object—whether due to distribution imbalances or inadequate examples—may over-rely on predicting background pixels as a "safe" default. This phenomenon is commonly known as *background bias*. In a CL scenario with only a small memory buffer to store latents of past samples for replay, we hypothesize this bias to become more pronounced, since the model gains access to fewer examples from previous tasks as it encounters new ones. As a result, it becomes increasingly difficult for the model to retain or reinforce its capability to distinguish the target from the background, thereby amplifying the tendency to label infrequently seen or more challenging image regions as background.

Consequently, our objective is to determine whether background bias persists under latent replay and whether introducing a targeted combination of spatially aware and global noise into the image features encourages the model to predict a larger foreground area. Figure 7.4 illustrates, for different pairs of noise coefficients α and β (*cf*. Equation (7.16)), the ratio between the average number of pixels classified as foreground (*i.e.*, the number of ones in the predicted segmentation mask) and the ground-truth foreground pixels, computed over all validation samples. A value of one implies that the size of the predicted foreground area exactly matches the size of the ground-truth foreground area.

As shown in Figure 7.4, the ratio of predicted-to-true foreground areas for the LR baseline, depicted in the bottom-left field of each heatmap, spans from 0.4 to 0.7, with lower values when fewer buffer samples per class are available. This range indicates that the model frequently mislabels too many pixels as background and therefore suffers from background bias, which gets more pronounced the fewer latent representations are available for replay to the modality interaction network. In

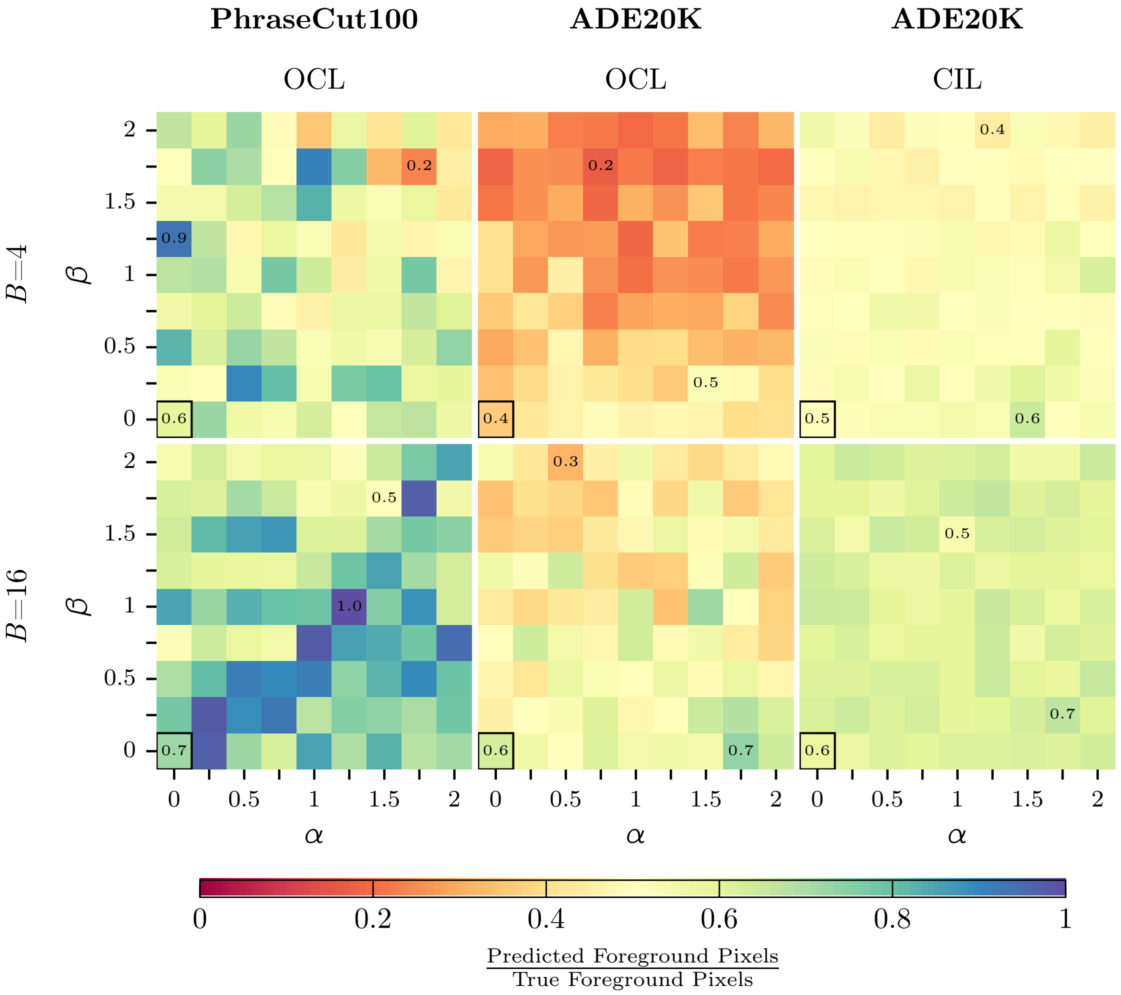


Figure 7.4: Impact of spatially aware vs. global noise on predicted-to-true foreground area ratio. The heatmaps display the average ratio of predicted to ground-truth foreground pixels over the validation set under varying levels of spatially aware noise (α) and global noise (β), while text features remain noise-free ($\gamma = 0$). Ratios exceeding one would indicate a bias toward overestimating the foreground area, although no such bias was observed experimentally.

contrast, for a moderate to high spatially aware noise coefficient α in combination with a low-to-medium global noise coefficient β, this ratio tends to be substantially higher, suggesting that our noise augmentation strategy reduces background bias perturbing background features and prompting the model to focus more on salient foreground regions. Examples that showcase this effect are provided in Figure 7.5. Notably, in none of the configurations shown in Figure 7.4 does the predicted foreground area exceed the true foreground area.

Although applying a high degree of spatially aware noise to background features while using a moderate degree of global noise on all image features brings the ratio of predicted-to-true foreground areas closer to one, this alone does not guar-

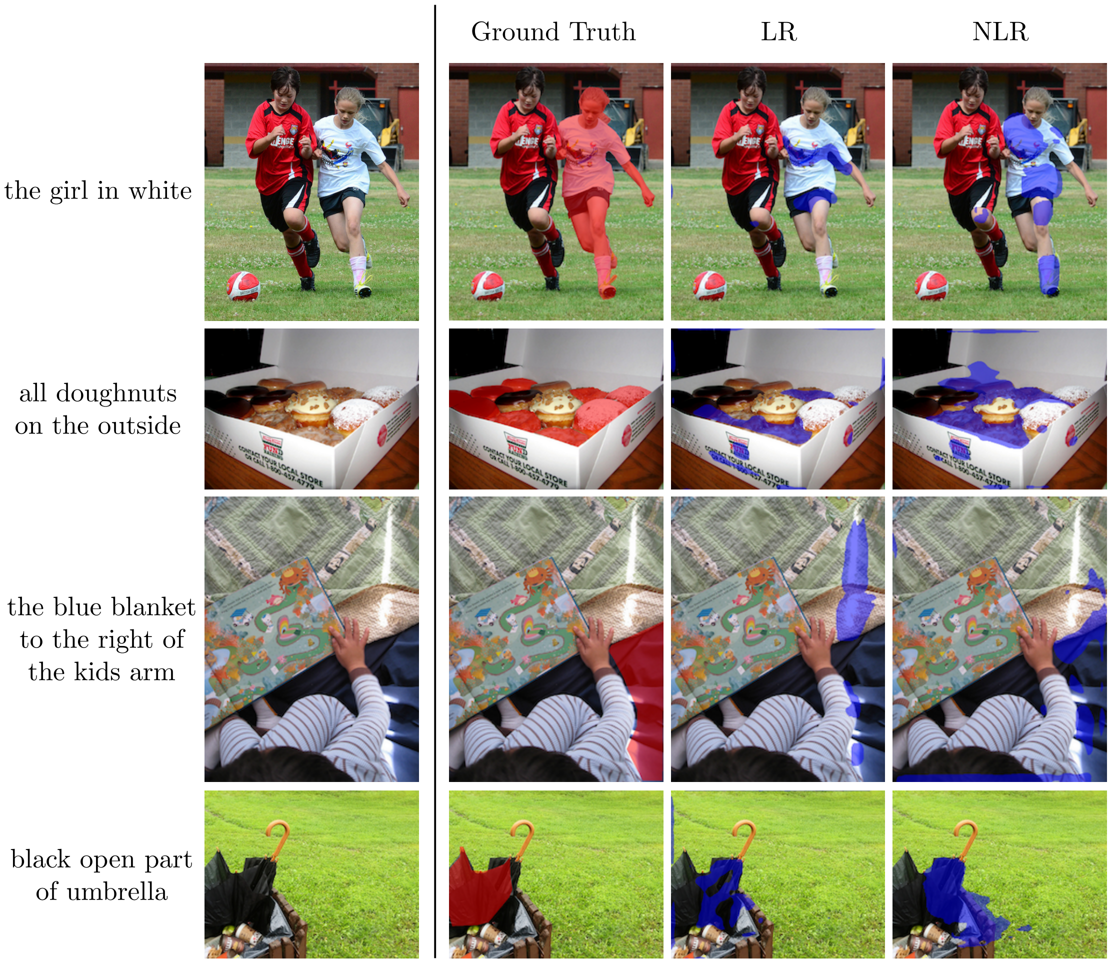

Ground Truth
LR
NLR
the girl in white
all doughnuts on the outside
the blue blanket to the right of the kids arm
black open part of umbrella

Figure 7.5: Illustration of background bias in latent replay methods for continual referring expression segmentation. Shown are failure and edge cases from the RefCOCO dataset. Across every example, the models (somewhat) correctly locate the target object, yet their predicted foreground masks (blue) encompass considerably fewer pixels than the ground-truth masks (red), which can lead to the mIoU evaluation metric judging them as incorrect. Both latent replay baselines exhibit this limitation; however, NLR consistently produces denser masks, indicating that it partially mitigates the background bias. Both models were trained with $B = 4$ and seed 1993.

antee that the predicted and ground-truth foreground regions overlap sufficiently to identify the target object accurately. The question remains whether the noise augmentation approach not only leads the model to predict a larger foreground region but also ensures that this region corresponds to the actual target object. To investigate this, we compute the Pearson correlation coefficient between the predicted-to-true foreground area ratio (*cf*. Figure 7.4) and the average accuracy of each noise coefficient configuration (*cf*. Figure 7.3), evaluated separately for each block in the heatmaps, *i.e.*, per combination of dataset, CL setting, and memory buffer size.

	PhraseCut100 OCL	**ADE20K** OCL	**ADE20K** CIL
$B = 4$	0.95	0.88	0.83
$B = 16$	0.84	0.85	0.77

Table 7.2: Correlation between model performance and predicted-to-true foreground area ratio. Reported values are Pearson correlation coefficients between the overall accuracy and the ratio between predicted and ground-truth foreground area. All coefficients are computed separately for each buffer size per class, dataset, and CL setting. The consistently large positive linear correlation indicates the importance of strategies to address the background bias problem for performance improvement in CRES scenarios, which can be effectively done with targeted noise injection.

As shown in Table 7.2, the Pearson correlation coefficients range from 0.77 to 0.95 across all configurations, demonstrating a strong positive linear association between the predicted-to-true foreground area ratio and overall model performance. This high correlation indicates that the additional pixels included in the predicted foreground following noise augmentation accurately represent components of the target object, rather than spurious background elements, which in turn enhances accuracy. In summary, these results demonstrate that our proposed noise augmentation technique effectively mitigates the background bias typically observed in conventional latent replay approaches, thereby improving performance in CRES problems.

7.3.6 Baseline Comparison

In the CIL setting, we compare NLR against a variety of regularization-based, parameter allocation, and latent replay methods for CL. With SFT and JFT, we additionally report results for sequential and joint fine-tuning of the modality interaction network, respectively.

Table 7.3 demonstrates that NLR attains the highest post-training accuracy across all datasets. It outperforms DER and DER++ while relying on smaller per-class memories. Although WSN, a parameter allocation method, appears more effective than regularization methods O-EWC and SPG, we observed its pool of free parameters rapidly diminish after only a few tasks, showcasing its scalability issues for a large or potentially unbounded number of tasks.

Remarkably, the distillation-based methods DER and DER++ perform only on par with the vanilla SFT baseline and even lag behind it on the split RefCOCO benchmark. These findings indicate that distilling knowledge from few latent representations is not only less effective than periodically replaying them directly, but also inferior to naively fine-tuning model parameters without explicit protection against catastrophic forgetting. This discrepancy can be attributed to the fact that

Method	***B***	**PhraseCut100**	**ADE20K**	**RefCOCO**
JFT	–	64.6 ± 0.2	63.1 ± 0.4	68.5 ± 0.3
SFT	–	52.3 ± 1.2	50.1 ± 0.5	56.2 ± 0.5
$\text{WSN}_{0.3}$	–	58.8 ± 0.1	52.6 ± 0.3	55.2 ± 0.3
$\text{WSN}_{0.5}$	–	57.9 ± 0.2	52.1 ± 0.3	53.3 ± 0.7
$\text{WSN}_{0.7}$	–	57.0 ± 0.2	51.2 ± 0.2	53.2 ± 0.8
O-EWC	–	53.1 ± 0.7	50.7 ± 0.6	56.4 ± 0.4
SPG	–	34.0 ± 3.4	35.1 ± 1.6	40.0 ± 4.1
LR	4	57.5 ± 0.4	54.6 ± 0.2	58.2 ± 0.3
LR	20	59.1 ± 0.2	57.1 ± 0.1	59.5 ± 0.4
DER*	20	53.8 ± 0.7	50.2 ± 0.3	48.3 ± 0.4
DER++*	20	52.8 ± 0.8	50.3 ± 0.3	48.7 ± 0.4
NLR	4	59.6 ± 0.2	56.6 ± 0.3	60.6 ± 0.4
NLR	20	**60.5** ± **0.3**	**57.5** ± **0.3**	**61.0** ± **0.4**
Ablations				
w/o Image Noise ($\alpha = \beta = 0$)	4	58.4 ± 0.1	55.3 ± 0.3	58.9 ± 0.3
w/o Spatial Noise ($\alpha = 0$)	4	59.0 ± 0.3	56.0 ± 0.4	59.3 ± 0.2
w/o Text Noise ($\gamma = 0$)	4	58.6 ± 0.3	55.7 ± 0.3	59.8 ± 0.1

Table 7.3: Performance comparison of parameter allocation, regularization, and latent replay methods in the CIL setting. Average accuracy scores after training on the last task are reported for parameter-allocation, regularization, and replay-based CL approaches. The subscript of WSN indicates the percentage of *masked* (*i.e.*, task-specific) parameters. It is noteworthy that both WSN and SPG require task-specific information during training *and* inference, thus operating in a TIL rather than CIL setting, which poses a simpler challenge.

knowledge distillation from memory regularizes the network only within the local neighborhood of the few replayed activations, leaving the broader feature manifold susceptible to drift when optimized for new tasks. In contrast, latent replay repeatedly injects representative anchor points that preserve the global geometry of the shared representation space. Injected noise, as done with NLR, further expands this manifold, which can additionally boost robustness against representational drift.

Unsurprisingly, the performance of latent replay methods increases with larger memory buffer size B, as more latents enable a model to better capture the distribution of features for each object class. At the same time, the noise augmentation mechanism in NLR surpasses conventional latent replay (*i.e.*, LR baseline) while storing 80% fewer latents in memory or delivering even higher performance with the same memory size. This suggests that noise helps the model generalize more effectively by successfully addressing the background bias common in CRES tasks (*cf.* Section 7.3.5), even when as few as *four* sets of features are stored per observed

Method	***B***	**PhraseCut100**	**ADE20K**	**RefCOCO**
JFT	–	58.8 ± 0.6	54.2 ± 0.8	59.8 ± 0.6
SFT	–	48.4 ± 0.8	50.0 ± 0.5	48.9 ± 0.7
O-EWC	–	48.7 ± 1.0	49.4 ± 0.7	50.6 ± 1.7
LR	4	55.7 ± 0.6	54.1 ± 0.3	54.5 ± 0.5
LR	20	57.9 ± 0.4	56.3 ± 0.5	56.7 ± 0.3
DER*	20	49.5 ± 0.3	48.8 ± 0.3	47.2 ± 0.7
DER++*	20	50.2 ± 0.7	48.9 ± 0.4	46.9 ± 0.8
NLR	4	57.0 ± 0.3	56.0 ± 0.2	57.5 ± 0.3
NLR	20	**58.2 ± 0.2**	**57.2 ± 0.2**	**58.2 ± 0.1**
Ablations				
w/o Image Noise ($\alpha = \beta = 0$)	4	56.3 ± 0.4	55.0 ± 0.1	55.7 ± 0.2
w/o Spatial Noise ($\alpha = 0$)	4	56.5 ± 0.1	55.3 ± 0.3	56.1 ± 0.2
w/o Text Noise ($\gamma = 0$)	4	56.2 ± 0.2	55.3 ± 0.2	56.8 ± 0.1

Table 7.4: Performance comparison of regularization and latent replay methods in the OCL setting. Average accuracy after continual training on the final task achieved by methods that strictly satisfy the OCL protocol. All baselines are trained for a single epoch ($E = 1$) with online, single-sample parameter updates ($|b| = 1$).

object class. Despite concerns about potential variability arising from sampling these four latents, NLR still exhibits relatively low variability in practice.

We additionally compare NLR with regularization-based and latent replay approaches in the OCL setting, while also reporting SFT and JFT performance. Notably, the JFT baseline does not necessarily serve as an upper bound for CL model performance under OCL constraints. Although the i.i.d. assumption holds and no distributional shift is present, updates in the OCL setting for every baseline including JFT are performed one sample at a time in a single-epoch process, which aligns with the common definition of OCL. In contrast, replay-based methods can draw larger batches from memory while still satisfying OCL assumptions.

In Table 7.4, we present the comparative results for OCL. Consistent with the findings from the CIL setting, NLR outperforms all other CL methods and yields an absolute improvement of 3.0% over the JFT baseline on ADE20K. These results indicate that the use of larger replay batches in combination with noise augmentation not only mitigates the adverse effects of non-stationary data distributions but also enhances the model's overall generalization capabilities.

Consistent with our observations in the CIL setting, knowledge distillation employed by DER and DER++ fails to match the performance achieved by periodic latent replay and fails to outperform the vanilla SFT baseline. In the OCL setting, features of *every* individual training sample are concatenated with one replayed set of features drawn from a constrained memory buffer and a distillation loss is computed over this composite input. This behavior exacerbates the aforementioned problem of overly local regularization, which cannot adequately mitigate overall representational drift. This shortcoming is particularly pronouned in segmentation problems: operating within a shared, pixel-wise output space, the distillation loss frequently degenerates—as confirmed by our preliminary hyperparameter searches—preventing the network from converging to precise object boundaries and locations.

By contrast, NLR yields absolute performance gains of 6.8–10.6% over these knowledge distillation baselines while using only 20% of their memory buffer size ($B = 4$ vs. $B = 20$) at a comparable replay frequency. These results demonstrate that NLR effectively makes multimodal integration networks robustly generalize in low-resource, low-data conditions, where new data arrive one instance at a time and the underlying distribution shifts gradually.

7.3.7 Ablation Study

The primary goal of our ablation study is to showcase the effectiveness of injecting noise into both text and image features during latent replay. In particular, we investigate the impact of adding image feature noise in general, as well as applying spatially aware noise. For all ablation experiments, we set $B = 4$. The corresponding results can be found in the lower sections of Tables 7.3 and 7.4.

Across all datasets and CL settings, removing either image or text noise consistently reduces performance, a result that highlights the importance of injecting noise into features from both modalities during replay. Moreover, we observe that omitting the spatially aware image noise described in detail in Section 7.2.2 consistently yields a performance drop of up to 1.4%. In addition, although the global text feature noise coefficient γ introduces slight perturbations to the target object's description, eliminating them altogether decreases performance. This result corroborates our argument that even small amounts of noise, when applied to all features, help the modality interaction network better generalize from the limited set of stored latent representations.

7.3.8 Variability Analysis

In this section, we investigate how sensitive NLR is to (i) the replay frequency, *i.e.*, how often during CL training a batch is sampled from memory for replay, and (ii) different task counts, *i.e.*, how many distributional shifts occur during CL training. The analysis is conducted using the default noise parameter configuration that was used in the main experiments (*cf.* Section 7.3.6), as we found that the trends reported in the following are not greatly affected by varying the noise parameters of NLR within the range examined in Section 7.3.4.

Effect of replay frequency $\boldsymbol{R_F}$. As demonstrated in Section 7.3.3, the effectiveness of latent replay methods is sensitive to the choice of replay frequency, with

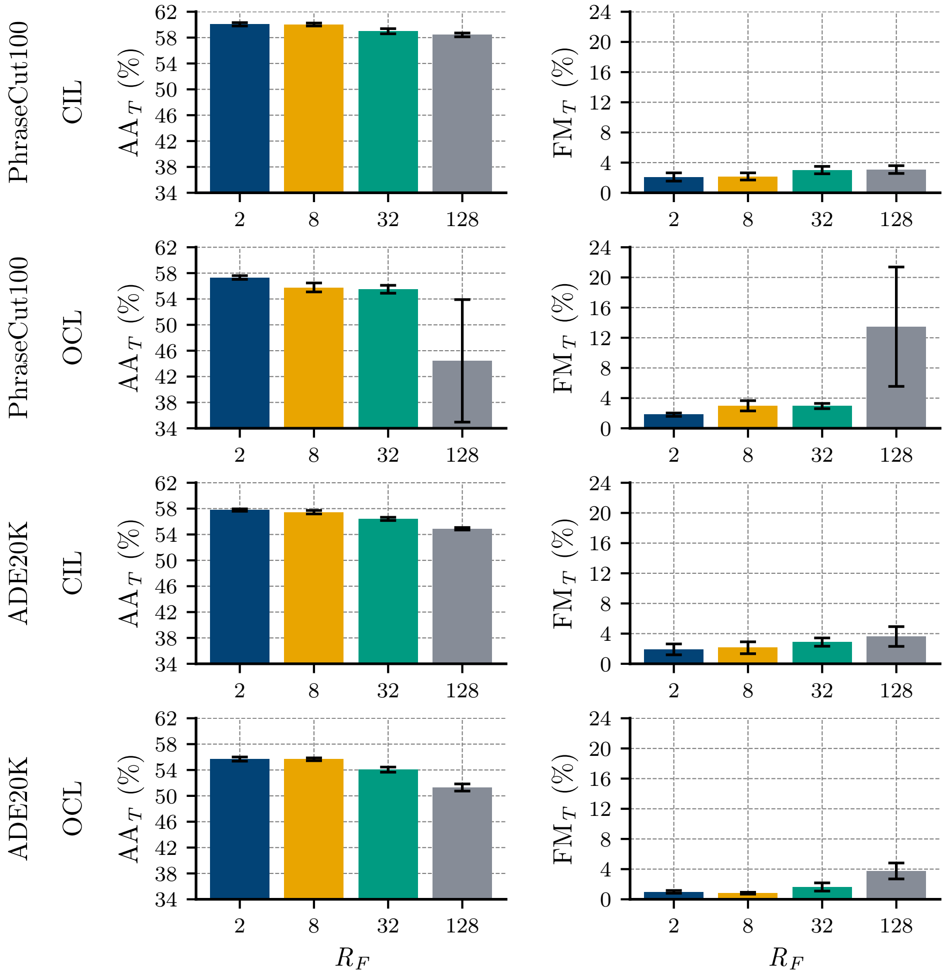


Figure 7.6: Variability of NLR across replay frequencies. Average accuracy (*left*) and forgetting (*right*) after training are reported for two different datasets and CL settings using NLR with per-class buffer size $B = 4$. Vertical lines indicate standard error bars.

higher values typically yielding improved performance. Since these experiments were conducted using the standard LR baseline, we now assess how robust noise augmentation implemented in NLR responds to variations in the replay frequency parameter R_F. Figure 7.6 reports the average accuracy and forgetting metric after continual training on the split datasets PhraseCut100 and ADE20K, evaluated under both CIL and OCL settings for four distinct replay frequencies.[12]

We observe that replay frequencies between 2 and 32 yield relatively similar overall performance and forgetting. However, at $R_F = 128$, performance drops substantially, accompanied by a marked increase in forgetting, an effect which is particularly visible for the OCL setting. In the case of OCL training on PhaseCut100, such a sparse replay can even lead to performance that is worse than sequential

[12] $R_F = 32$ is the default configuration for CIL baseline comparisons, whereas $R_F = 2$ is used as the standard in OCL baseline experiments (*cf*. Section 7.3.6).

online updates without forgetting prevention mechanisms (as with the SFT baseline, *cf*. Table 7.4). A plausible explanation is that excessively infrequent replay, especially when noise is added, fails to stabilize feature representations and may instead degrade their quality. Without consistent reinforcement, the model struggles to differentiate meaningful variations from noise, leading to discrepancies between the pre-output representations of current and previous tasks. This exacerbates the vulnerability to representational drift in the modality interaction network and impairs the recovery of accurate features from earlier tasks.

Effect of task count T. To examine whether the advantages of spatially aware multimodal noise augmentation remain consistent under various task counts, we measure the average accuracy and forgetting metrics after CL training using four different task counts, $T \in \{10, 20, 50, 100\}$. Although $T = 10$ is a common choice in the CL research literature, it can be too low to reveal how methods scale under frequent distributional shifts or how effectively they handle open-ended scenarios. At the other extreme, $T = 100$ constitutes the most demanding setting: in the PhraseCut100 split dataset, it involves learning one object class at a time.[13]

As shown in Figure 7.7, each larger value of T causes the performance of NLR to drop by about 1–3%, while the forgetting metric and its associated standard error consistently increase. This behavior likely arises because learning only one or a few object classes per task increases the model sensitivity to the task sequence in the data stream, an effect that is amplified by the class imbalance in many CRES datasets, as mentioned in Section 7.2.1. Nonetheless, the overall performance reduction from $T = 10$ to $T = 100$ is limited to roughly 2.5–5%, underscoring the robustness of NLR to multiple distributional shifts.

7.4 Discussion

Our results demonstrate that NLR effectively addresses the CRES challenge by selectively replaying latent representations augmented with spatially aware statistical noise. By periodically replaying these augmented multimodal representations, NLR consistently mitigates catastrophic forgetting and enhances generalization across a variety of experimental conditions, including both CIL and the more challenging OCL setting.

A key insight we gained from our experiments is that maintaining a minimal per-class memory—storing as few as four latent representations per object class—outperforms most established CL baselines. This finding suggests that NLR can operate under strict memory and privacy constraints without sacrificing performance. Notably, the strategic application of noise—injecting stronger perturbations into background features while applying milder perturbations to target object features—reduces background bias. Improved alignment between predicted

[13] For baseline comparisons on PhraseCut100 and ADE20K (*cf*. Section 7.3.6), $T = 20$ is the default configuration.

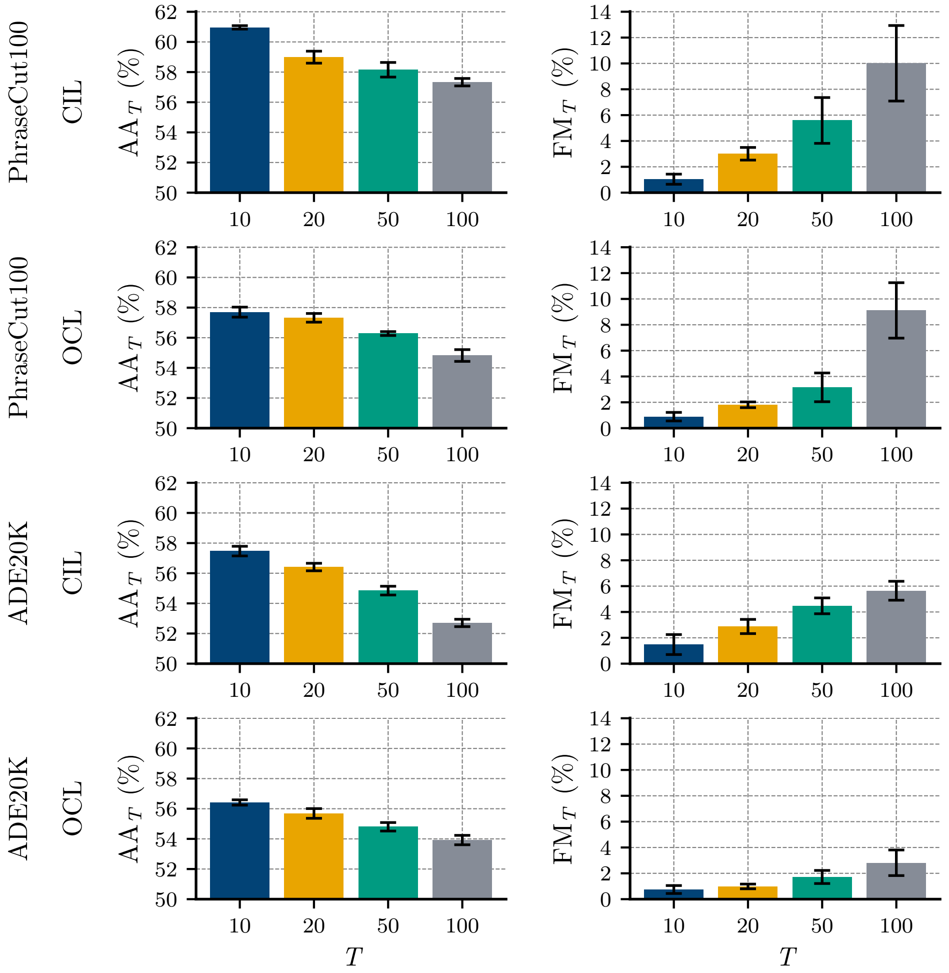


Figure 7.7: Variability of NLR across task counts. This figure compares the performance of NLR for different values of T. In our experiments, $T = 10$ represents the simplest scenario, whereas $T = 100$ corresponds to the most challenging setting—particularly for PhraseCut100, where each object class is learned sequentially. The memory buffer is fixed at $B = 4$ per class.

and ground-truth foreground areas confirms that the model learns to avoid over-reliance on default background labeling, thereby enhancing segmentation accuracy for both new and previously encountered classes. Our ablation study further reveals that the observed performance gains are attributable to the combined application of a small degree of global noise to both text and image features and a larger degree of spatially aware noise to image features.

It is crucial to delineate the scope within which NLR is most effective. Our method is purposely tailored for non-comparative tasks in which all predictions inhabit a shared spatial output space, *e.g.*, as in object localization and semantic segmentation. Its spatially aware noise augmentation presupposes an image-like input modality whose two-dimensional topology guides location-dependent perturbations. When NLR is transposed to simpler classification or regression problems, this spatial prior disappears: either the spatially aware noise component must be omitted and replaced by globally uniform noise injected into the stored latents, or an auxiliary generic image segmentation model must be used to derive pseudo-spatial masks by separating foreground from background pixels in raw images. Although both adaptations are theoretically viable, they have not yet been empirically validated. Therefore, a systematic assessment of these variants constitutes an open avenue for future research.

Nevertheless, from a broader perspective, the strong performance of NLR underscores the potential of further investigation into latent replay-based methods for open-world multimodal tasks, with particular attention to maximizing the utility of stored latents for recovering old data distributions. In contrast to earlier approaches that rely on extensive exemplar sets or task-specific parameter expansion, NLR remains task-agnostic and operates with only a minimal buffer of latent features—so compact that these features effectively serve as prototypes. By combining this efficient memory design with noise-driven, spatially aware feature augmentation, NLR mitigates the drawbacks of excessive storage requirements and rigid task boundaries. Notably, it supports open-ended learning to the extent that only one object class or concept is learned at a time, marking it as a promising strategy for scalable and privacy-preserving CL in real-world multimodal applications.

7.5 Chapter Summary

In this chapter, we introduced NLR, a novel latent replay framework for continual referring expression segmentation. By selectively retaining a small set of multimodal features per object class and augmenting them with targeted noise, NLR mitigates catastrophic forgetting and delivers consistently high performance across diverse experimental settings. Our analysis demonstrates that incorporating spatially aware noise into background features while applying moderate perturbations to foreground and textual features effectively counteracts background bias and enhances segmentation accuracy. Empirical results reveal that NLR consistently outperforms competitive baselines for continual learning, including conventional latent replay and knowledge distillation methods, even when allocated fewer memory resources.

Beyond its empirical merits, NLR operates in a fully task-agnostic manner and mitigates privacy concerns typically associated with storing raw images. Our approach minimizes memory overhead, scales efficiently with an increasing number of tasks, and maintains robust generalization without relying on explicit task delineations. These results highlight the broad potential of targeted augmentation of stored and periodically replayed latent representations—essentially, maximizing the information extracted from prototypical features—for privacy-sensitive real-world applications and the advancement of open-world, multimodal continual learning research.

Part IV

Closing

Chapter 8

Conclusion

8.1 Thesis Summary

In this thesis, we presented a comprehensive investigation and evaluation of multiple approaches for downstream CL with foundation models. To improve real-world applicability, we progressively relaxed the restrictive assumptions that typically constrain models operating on non-stationary data streams. In doing so, we transitioned from reliance on single-source inputs to the integration of multimodal information from diverse sources, thereby taking a critical step toward developing models that are more robust to and capable of CL in challenging open-domain scenarios.

We began by reviewing the rapid emergence of foundation models over recent years and the paradigm shift they have driven in both unimodal and multimodal learning. Afterwards, we highlighted the current progress in CL research, positioned it within this evolving paradigm, and outlined the remaining research gaps.

Subsequently, in the second part of the thesis, we proposed novel methods for CL using foundation models pretrained on unimodal tasks. In particular, we introduced two innovative approaches: DRILL and LayUP. Each method leverages prototypical representations in a distinct fashion. DRILL constructs class prototypes via a topological mapping of the feature space, while LayUP exploits multi-layer features that combine high-level with low-level statistical characteristics to enhance class separability. Remarkably, LayUP achieves performance that exceeds that of fully fine-tuning the foundation model, which shows how it successfully maximizes the utility of unimodal foundation models. Additionally, we provided best practices—tailored to the characteristics of specific datasets—for extracting deep, multi-layer features with minimal memory and computational overhead yet maximum performance.

In the third part of the thesis, our focus shifted to multimodal CL with foundation models, with an emphasis on VL grounding tasks. In light of the relatively sparse research at the intersection of foundation models and multimodal CL, we commenced with a detailed analysis of how various components within modality interaction networks impact task-sequential learning. Based on these findings, we developed straightforward CL baselines, denoted as SMS, which designate only select components of the modality interaction network as task-specific while sharing the remaining components across tasks. Furthermore, we enhanced SMS by integrating a neurocognitively inspired adaptation and consolidation scheme, demonstrating its high efficacy for preserving learned knowledge over sequential tasks.

Recognizing that SMS necessitates task-specific knowledge during inference, we subsequently introduced a successor method that relaxes these requirements while maintaining efficiency and effectiveness. Specifically, we identified latent replay as a promising strategy for downstream CL with multimodal foundation models and proposed our novel noise augmentation approach, NLR, which substantially reduces assumptions used to derive CL desiderata we established in the first part of this thesis. NLR offers a highly efficient and versatile solution for a wide range of real-world multimodal CL applications, as exemplified by in-the-wild continual referring expression segmentation.

8.2 Discussion

Within the scope of this work, we sought to address the research challenge of **developing efficient and versatile CL methods that enable foundation models to be incrementally fine-tuned on sequences of diverse and complex unimodal and multimodal tasks, all while preserving previously acquired capabilities.** We pursued this overarching objective by devising four distinct methodologies for CL, two tailored to unimodal and two designed for multimodal foundation models. A high-level comparison of these methodologies with respect to efficiency and versatility criteria is provided in Table 8.1. These methodologies underpin four primary research objectives detailed in Section 1.2, each of which directly contributes to the achieving of the overarching goal of this thesis.

Integrating topology-aware prototypical feature replay from a self-organizing network within a dual-memory architecture diminishes sensitivity to task order and enhances predictive accuracy in CL scenarios with imbalanced data (*Objective 1*). We have shown in Chapter 4 that sequential fine-tuning of unimodal foundation models on class-imbalanced data—without mechanisms to counteract forgetting—drastically undermines performance, particularly when underrepresented classes occur early in the data stream. Based on these findings, we have demonstrated that integrating a topology mapping of the feature space of the foundation model within a self-organizing network architecture, together with a frequency-based prototype sampling mechanism, provides better control of the representation shift by stabilizing the learning trajectory of the foundation model, preserves the integrity of features particularly for heavily underrepresented classes, and recalibrates the prediction head to mitigate task-recency bias.

CL Desiderata	**DRILL** Chapter 4	**LayUP** Chapter 5	**SMS** Chapter 6	**NLR** Chapter 7
Learning Efficiency	✓	✓	✓	✓
Overcoming Forgetting	✓	✓	✓	✓
Task-agnostic Learning	✓	✓	✗	✓
Open-ended Learning	✓	✓	✓	✓
Resource Efficiency	✗	✓	✓	✓
Exemplar-free Retention	✗	✓	✓	✓
Unimodal Label Prediction	✓	✓	✓	✓
Multimodal Reasoning	✗	✗	✓	✓

Table 8.1: Comparison of the proposed CL methods with respect to versatility and efficiency criteria. Most properties related to efficiency and versatility derive from the established CL desiderata (*cf*. Section 3.2.2). Orange checkmarks denote *partial* compliance—*e.g.*, leveraging task-specific signals during training but not inference (*task-agnostic learning*) or satisfying efficiency goals that hinge on the split between shared and task-specific parameters (*resource* and *learning efficiency*). Although *unimodal label prediction* and *multimodal reasoning* are not canonical desiderata, they capture practical challenges that arise as CL moves beyond traditional classification (Mitchell et al., 2025). While the core ideas behind SMS and NLR—selective specialization of trainable parameters and noise augmentation to stored latent features—are, in principle, applicable to label prediction problems such as classification and regression, they were originally conceived and empirically validated for dense, structured-output problems like object localization and semantic segmentation.

When integrated with meta-learning and episodic exemplar replay, this methodology substantially reduces performance variability in single-epoch class-incremental learning settings, irrespective of task order and heavy data imbalances.

Aggregating first- and second-order feature statistics across multiple layers of a unimodal foundation model greatly enhances class separability, domain invariance, resource and learning efficiency in exemplar-free prototype-based CL (*Objective 2*). We have demonstrated in Chapter 5 that explicitly capturing multiple abstraction levels in foundation models enhances the robustness of prototype-based classifiers against domain shifts, leading to significant improvements in classification performance, particularly when pretraining and fine-tuning domains diverge greatly or labeled data are scarce. Although the computational and memory demands increase with the number of layers used for feature extraction, we have found that selecting features from the latter half of the foundation model offers an optimal balance between accuracy and resource efficiency. Additionally, we have established best practices for layer depth choice tailored to the characteristics of the fine-tuning tasks. Ultimately, integrating multi-layer features of a unimodal foundation model for prototype construction serves as a versatile plug-in applicable to any CL scenario, independent of specific data presentation assumptions.

Selecting a targeted subset of network components for task-specific adaptation and alternating their updates with shared components is a resource-efficient and effective strategy to balance the trade-off between specialization and generalization in multimodal CL with foundation models (*Objective 3*). We have introduced in Chapter 6 two diagnostic datasets with a clearly defined shared task structure, enabling a comprehensive analysis of modality interaction networks to pinpoint components that distinguish subtle task differences. Drawing on these insights, we have devised heuristics as well as importance-based metrics for selective specialization strategies that store a subset of component parameters in a buffer, which can be flexibly integrated into the model during inference. Moreover, we have demonstrated that a developmental psychology-inspired learning scheme—alternating multiple task adaptation steps of task-specific components with a single consolidation step updating task-sharing components—improves performance in task-incremental learning settings by protecting parameters critical for cross-task generalization against frequent disruptive updates. In particular, we have shown that, with a well-designed model decomposition strategy, making less than 1% of the modality interaction parameters task-specific can achieve strong performance and, in some cases, enable learning where previous baselines have failed.

Spatially aware statistical noise on latent features effectively mitigates background bias in latent replay for multimodal CL, substantially reduces storage requirements, and remains exemplar-free, task-agnostic, and universally compatible with any CL setting (*Objective 4*). In Chapter 7, we have demonstrated that latent replay is a promising, privacy-preserving strategy for multimodal CL under minimal assumptions about data presentation. However, our empirical results revealed that storing a too small set of latents leads to overfitting and background bias, causing the model to predict overly large background regions that obscure the region of interest. We have proposed a strategy to automatically delineate language-informed foreground from background image features using ground-truth information about the region of interest in the original image space. Based on this, we have shown that injecting modest noise into text features and foreground image features, while adding stronger noise to background image features, allows a vision-language interaction network to more accurately identify the target object in continual referring expression segmentation tasks. Our approach is applicable to a wide variety of foundation model and modality interaction network architectures, allowing up to 80% reduction in stored latents without sacrificing performance, while meeting the CL desiderata described in this thesis.

8.3 Future Work

In this thesis, we have successfully developed and evaluated methods for downstream continual learning with foundation models, such as BERT, ViT, or CLIP. Although we believe that our proposed approaches are widely applicable across various CL scenarios and challenges, significant hurdles remain in achieving genuinely open-ended and unconstrained continual learning, which we discuss in the following.

Continual upstream pretraining. One central research challenge is how to continuously update the pretrained weights of foundation models, as opposed to continually fine-tuning them for specific downstream tasks, which is the focus of this thesis. Typically, such updates employ unsupervised or self-supervised learning schemes. Cossu et al. (2024) select a subset of pretraining data within a fixed budget using a median-loss alignment criterion, although they assume full access to historical data. Gupta et al. (2023) examine "rewarming" language models, *i.e.*, re-increasing the learning rate on new data, but do not show compelling results. In contrast, the adaptation and consolidation scheme in SMS could improve simple rewarming by splitting training into multiple adaptation and stabilization phases, adjusting the learning rates accordingly. Another promising direction is layer-wise adaptation: while early layers remain relatively fixed to preserve cross-domain knowledge from initial pretraining, later layers adapt to new data from recently introduced corpora. This principle underlies LayUP, which leverages representations at different layer depths. In a similar vein, extending the NLR method could involve storing latents extracted from layers designated to remain fixed, then using noise-augmented replay to retrain subsequent layers.

Continual training of generative LLMs. Generative LLMs such as GPT-4 (OpenAI et al., 2024) and Llama (Dubey et al., 2024) are trained via multi-stage pipelines—pretraining, instruction tuning, and alignment—each of which can benefit from tailored CL strategies (Wu et al., 2024a). For example, while updating factual knowledge, regularization may be used to preserve instruction-following skills. The immense scale of these models demands parameter-efficient approaches and architectures that integrate external memory (*e.g.*, through retrieval-based methods). Such external memory supports dynamic access to previously acquired knowledge, complementing techniques that update only the model weights. However, both retrieval-based and model adaptation methods largely address narrow updates to specific facts or domains. CL, in contrast, could provide a holistic framework to concurrently refine language fluency, multi-step reasoning, and world knowledge. One promising direction involves leveraging LLMs within the CL loop, *e.g.*, by generating pseudo-data or explanations for past tasks, or by maintaining an LLM-based memory module that evolves with experience. Methods like DRILL, which preserves topology-aware feature representations, could be adapted to safeguard critical latent features in LLM hidden states, while noise augmentation methods such as NLR may help rehearse diverse linguistic patterns by replaying noise-augmented pseudo-text sequences resembling prior knowledge.

Leveraging compositionality for continual learning. Real-world tasks often decompose into reusable components, yet standard CL methods typically treat each task independently. Benchmarks such as LILAC-2D (*cf*. Section 6.3.1) systematically recombine concepts like color, object, and direction and thus reveal whether a model truly learns core abstractions or simply memorizes instances. Our multi-layer representation method LayUP could be extended to capture these compositional factors. For example, prototypes at different layers can encode distinct concepts (ranging from low-level attributes to high-level features) and be recombined for rapid adaptation to novel combinations of known concepts. Similarly, specialized intra-layer adapters aligned with specific conceptual dimensions could be reused and recombined when familiar concepts recur, a strategy taking inspiration from both our SMS method and mixture-of-expert approaches for CL (*e.g.*, Yu et al., 2024; Zhu et al., 2024). Alternatively, future directions leveraging compositionality for generation, instruction following, and program synthesis may integrate generative or latent replay—potentially augmented by targeted noise as in NLR—with compositional reasoning to produce synthesized samples that combine learned primitives.

Embodied continual learning. When CL shifts from static datasets to embodied agents, challenges related to partial observability, real-time adaptation, and tight resource constraints emerge. Agents must not only learn incrementally, but also strategically decide how to gather experience—for example, by revisiting tasks or querying a teacher LLM if performance declines (Yoo et al., 2024). The interplay of perception, action, and memory necessitates innovative approaches that take advantage of the agent's ability to interact and probe its environment (Mendez-Mendez et al., 2023). Building on the methodologies put forth in this thesis, an agent could maintain compact latent prototypes or noise-augmented features to rehearse previously acquired skills. DRILL's topological replay, for instance, preserves robust perceptual maps, while SMS-like selective specialization controls which motor or sensor parameters are updated with each new task. Periodic consolidation phases further ensure that older behaviors remain accessible. In robotics, where multiple tasks draw on shared subroutines (grasping, pushing, *etc.*), skill compositionality is crucial. Mechanisms that encode such subskills as recomposable modules can reduce forgetting by refining existing building blocks or generating new ones when needed (Mendez et al., 2022). By integrating these strategies with the resource-efficient, prototype-based techniques presented in this thesis, embodied CL agents could systematically broaden their skill repertoires in complex, ever-evolving real-world settings.

8.4 Final Remarks

In summary, this thesis pushes the boundaries of foundation models by deploying a spectrum of downstream fine-tuning techniques to enable effective sequential learning across both unimodal and multimodal tasks. We have demonstrated that applying unsupervised topology mapping to the feature space effectively stabilizes the learning trajectory of foundation models during continual learning, and extracting features from multiple intermediate layers for prototyping helps bridge distributional gaps between the pretraining and downstream continual fine-tuning domains, while simultaneously improving class separability even in low-data regimes. We have further shown that decomposing modality interaction networks into task-sharing and task-specific components and optimizing them via an intermittent adaptation-consolidation strategy protects critical model parameters from frequent disruptive updates when learning multimodal tasks sequentially. Finally, augmenting a minimal set of prototypical features extracted from a foundation model with spatially aware noise provides a highly efficient and versatile approach to acquiring sequences of challenging multimodal reasoning tasks without imposing restrictive assumptions about data type or presentation. Although open-ended continual learning in real-world scenarios remains challenging, this thesis represents a significant leap toward agents capable of operating in dynamic, real-world environments where information becomes progressively available over time.

Appendices

Appendix A

Nomenclature

Acronyms

AI	Artificial Intelligence
BMU	Best-Matching Unit
BN	Batch Normalization
CIL	Class-Incremental Learning
CL	Continual Learning
CLS	Complementary Learning Systems
CNN	Convolutional Neural Network
CRES	Continual Referring Expression Segmentation
CV	Computer Vision
DIL	Domain-Incremental Learning
ER	Experience Replay
FFN	Feed-Forward Network
FSA	First Session Adaptation
JFT	Joint Fine-Tuning
LLM	Large Language Model
LN	Layer Normalization
MHA	Multi-Head Attention
NLP	Natural Language Processing
NMC	Nearest-Mean Classifier
OCL	Online Continual Learning
PEFT	Parameter-Efficient Fine-Tuning
PL	Prediction Layer
PTM	Pretrained Model
RNN	Recurrent Neural Network
SFT	Sequential Fine-Tuning
SGD	Stochastic Gradient Descent
TIL	Task-Incremental Learning
VL	Vision-Language
VLM	Vision-Language Model

Notations

v	Scalar
$\boldsymbol{v}$	Vector
$\boldsymbol{M}$	Matrix
$f_\theta(\boldsymbol{x})$	Function of $\boldsymbol{x}$ parameterized by θ
$\boldsymbol{h}_l$	Hidden/intermediate representation at the l^{th} network layer
d_l	Dimensionality of $\boldsymbol{h}_l$
$\mathcal{T}$	The space of all tasks
t	Task index
$\mathcal{X}_t$	The space of input samples of the t^{th} task
$\boldsymbol{x}_{t,n}$	n^{th} input sample associated with the t^{th} task
$\mathcal{Y}_t$	The space of output labels of the t^{th} task
$y_{t,n}$	Output label of input sample $\boldsymbol{x}_{t,n}$
$\mathcal{D}_t$	Training dataset of the t^{th} task
$\mathcal{B}_t$	Space of batches of the t^{th} task
$\mathcal{D}_{\text{test}}$	Test dataset
N_t	Number of training samples of the t^{th} task
θ	Trainable neural network parameters
Ω	Importance score of neural network parameters

Symbols and Operations

$(x_1, \ldots, x_n)$	Sequence of n elements
$[\boldsymbol{a}; \boldsymbol{b}]$	Concatenation of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$
$\odot$	Dot product
$\otimes$	Hadamard product (element-wise multiplication)
$\|\|\boldsymbol{v}\|\|$	L^2 norm of $\boldsymbol{v}$
$[\![P]\!]$	Iverson bracket: $[\![P]\!] = \begin{cases} 1 & \text{if } P \text{ is true} \\ 0 & \text{otherwise} \end{cases}$

Measurements and Metrics

AA	Average Accuracy
BWT	Backward Transfer
FM	Forgetting Measure
FWT	Forward Transfer
IoU	Intersection over Union
MMD	Maximum Mean Discrepancy

Appendix B

Publications Originating from this Thesis

Journal Articles

- **Ahrens, K.**, Lehmann, H. H., Lee, J. H., Wermter, S. (2024). Read Between the Layers: Leveraging Multi-Layer Representations for Rehearsal-Free Continual Learning with Pre-Trained Models. In *Transactions on Machine Learning Research* (TMLR).
- Becker, D., Braach, L., Clasmeier, L., Kaufmann, T., Ong, O., **Ahrens, K.**, Gäde, C., Strahl, E., Fu, D., Wermter, S. (2024) Influence of Robots' Voice Naturalness on Trust and Compliance. In *ACM Transactions on Human-Robot Interaction* (THRI), 14(2):1–25.

Conference Papers

- **Ahrens, K.**, Abawi, F., Wermter, S. (2021) DRILL: Dynamic Representations for Imbalanced Lifelong Learning. In *International Conference on Artificial Neural Networks* (ICANN), pages 409–420.
- Lee, J. H., Kerzel, M., **Ahrens, K.**, Weber, C., Wermter, S. (2022) What is Right for Me is Not Yet Right for You: A Dataset for Grounding Relative Directions via Multi-Task Learning. In *International Joint Conference on Artificial Intelligence* (IJCAI), pages 1039–1045.
- Ambsdorf, J., Munir, A., Wei, Y., Degkwitz, K., Harms, H. M., Stannek, S., **Ahrens, K.**, Becker, D. Strahl, E., Weber, T., Wermter, S. (2022) Explain yourself! Effects of Explanations in Human-Robot Interaction. In *IEEE International Conference on Robot and Human Interactive Communication* (RO-MAN), pages 393–400.

- Becker, D., Rueda, D., Beese, F., Gutierrez Torres, B. S., Lafdili, M., **Ahrens, K.**, Fu, D., Strahl, E., Weber, T., Wermter, S. (2023) The Emotional Dilemma: Influence of a Human-like Robot on Trust and Cooperation. In *IEEE International Conference on Robot and Human Interactive Communication* (RO-MAN), pages 1689–1696.
- **Ahrens, K.**, Bengtson, L., Lee, J. H., Wermter, S. (2023) Visually Grounded Continual Language Learning with Selective Specialization. In *Findings of the Association for Computational Linguistics: Empirical Methods in Natural Language Processing* (EMNLP), pages 7037–7054.
- Allgeuer, P., **Ahrens, K.**, Wermter, S. (2025) Unconstrained Open Vocabulary Image Classification: Zero-Shot Transfer from Text to Image via CLIP Inversion. In *IEEE/CVF Winter Conference on Applications of Computer Vision* (WACV), pages 8206–8217.

Other Publications

- **Ahrens, K.**, Kerzel, M., Lee, J. H., Weber, C., Wermter, S. (2022) Knowing Earlier what Right Means to You: A Comprehensive VQA Dataset for Grounding Relative Directions via Multi-Task Learning. In *International Workshop on Spatio-Temporal Reasoning and Learning* (STRL).
- Lee, J. H., Sioutis, M., **Ahrens, K.**, Alirezaie, M., Kerzel, M., Wermter, S. (2023) Neuro-Symbolic Spatio-Temporal Reasoning. In *Compendium of Neurosymbolic Artificial Intelligence*, pages 410–429.

Appendix C

Acknowledgments

First and foremost, I am deeply grateful to Stefan Wermter for giving me the opportunity to pursue this doctoral journey in the first place and granting me the freedom to explore new ideas (and conferences) that have shaped me into the researcher I am today. I owe a great deal to Jae Hee Lee, who never shied away from asking those tricky questions that propelled my thinking forward. Thank you for being an inspiring advisor, co-author and sparring partner.

Thank you, Katja Kösters, for helping me navigate bureaucracy's maze so I could stay focused on wrangling data, not paperwork. Special thanks to Erik Strahl and Matthias Kerzel for tirelessly keeping our GPUs in peak form, and to Reinhard Zierke for being my ever-reliable troubleshooter. Sven Magg, you taught me the craft (and occasional cartoon) of science, and our co-teaching adventures remain among my best memories. A big round of thanks goes to Cornelius Weber, Dennis Becker, and Connor Gäde, who made teaching a genuinely enjoyable experience. To Ozan Özdemir, who started this doctoral journey alongside me, thank you for the great discussions, shared laughter, and memorable stories, and for becoming a cherished friend in the process.

I also want to acknowledge every postdoc and doctoral student in the WTM group for their camaraderie, hilarious conversations, and legendary "WTM nights out," complete with globe-spanning cuisines that fueled both belly and brain. Special recognition goes to Lennart Bengtson and Hans Hergen Lehmann, who proved to me that mentoring can be every bit as rewarding as being mentored.

My deepest gratitude goes to my parents, Regine and Jörg, for their emotional (and financial) support, moral guidance, and for giving me the freedom to pursue every ambition. My siblings, Kolja and Maris, deserve a medal not just for putting up with their occasionally exasperating youngest sister, but for being true friends every step of the way. Finally, to Ricky, who stood by every idea I pursued—both brilliant and not-so-brilliant—with unwavering love and support, thank you for being my partner in all realms of life. I am beyond excited about the adventures our future holds.

Bibliography

Abati, D., Tomczak, J., Blankevoort, T., Calderara, S., Cucchiara, R., and Bejnordi, B. E. (2020). Conditional Channel Gated Networks for Task-Aware Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Abraham, W. C. (2003). How long will long-term potentiation last? *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358(1432):735–744.

Abraham, W. C. (2008). Metaplasticity: tuning synapses and networks for plasticity. *Nature Reviews Neuroscience*, 9(5):387–387.

Abraham, W. C. and Bear, M. F. (1996). Metaplasticity: the plasticity of synaptic plasticity. *Trends in Neurosciences*, 19(4):126–130.

Abraham, W. C. and Robins, A. (2005). Memory retention – the synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 28(2):73–78.

Ahn, H., Cha, S., Lee, D., and Moon, T. (2019). Uncertainty-based Continual Learning with Adaptive Regularization. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Aljundi, R., Chakravarty, P., and Tuytelaars, T. (2017). Expert Gate: Lifelong Learning With a Network of Experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Aljundi, R., Lin, M., Goujaud, B., and Bengio, Y. (2019). Gradient based sample selection for online continual learning. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Allgeuer, P., Ahrens, K., and Wermter, S. (2025). Unconstrained Open Vocabulary Image Classification: Zero-Shot Transfer from Text to Image via CLIP Inversion. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 8206–8217.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016). Neural Module Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Atkinson, C., McCane, B., Szymanski, L., and Robins, A. (2021). Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *Neurocomputing*, 428:291–307.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization. *arXiv:1607.06450*.

Baltrušaitis, T., Ahuja, C., and Morency, L.-P. (2019). Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443.

Banayeeanzade, M., Mirzaiezadeh, R., Hasani, H., and Soleymani, M. (2021). Generative vs. Discriminative: Rethinking The Meta-Continual Learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21592–21604. Curran Associates, Inc.

Bear, M. F. and Abraham, W. C. (1996). Long-Term Depression in Hippocampus. *Annual Review of Neuroscience*, 19(1):437–462.

Beaulieu, S., Frati, L., Miconi, T., Lehman, J., Stanley, K. O., Clune, J., and Cheney, N. (2020). Learning to Continually Learn. In Giacomo, G. D., Catalá, A., Dilkina, B., Milano, M., Barro, S., Bugarín, A., and Lang, J., editors, *European Conference on Artificial Intelligence*, volume 325, pages 992–1001. IOS Press.

Belouadah, E. and Popescu, A. (2019). IL2M: Class Incremental Learning With Dual Memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A Pretrained Language Model for Scientific Text. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Benavides-Prado, D. and Riddle, P. (2022). A Theory for Knowledge Transfer in Continual Learning. In Chandar, S., Pascanu, R., and Precup, D., editors, *Proceedings of The 1st Conference on Lifelong Learning Agents*, volume 199 of *Proceedings of Machine Learning Research*, pages 647–660. PMLR.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA. Association for Computing Machinery.

Bi, G.-q. and Poo, M.-m. (1998). Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type. *The Journal of Neuroscience*, 18(24):10464–10472.

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. (1982). Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *The Journal of Neuroscience*, 2(1):32–48.

Bilen, H. and Vedaldi, A. (2017). Universal representations: The missing link between faces, text, planktons, and cat breeds. *arXiv:1701.07275*.

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N., and Turian, J. (2020). Experience Grounds Language. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Bliss, T. V. P. and Collingridge, G. L. (1993). A synaptic model of memory: long-term potentiation in the hippocampus. *Nature*, 361(6407):31–39.

Bliss, T. V. P. and Lømo, T. (1973). Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path. *The Journal of Physiology*, 232(2):331–356.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., et al. (2020). Language Models are Few-Shot Learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bugliarello, E., Cotterell, R., Okazaki, N., and Elliott, D. (2021). Multimodal Pretraining Unmasked: A Meta-Analysis and a Unified Framework of Vision-and-Language BERTs. *Transactions of the Association for Computational Linguistics*, 9:978–994.

Burke, S. N. and Barnes, C. A. (2006). Neural plasticity in the ageing brain. *Nature Reviews Neuroscience*, 7(1):30–40.

Buzzega, P., Boschini, M., Porrello, A., Abati, D., and Calderara, S. (2020). Dark Experience for General Continual Learning: a Strong, Simple Baseline. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15920–15930. Curran Associates, Inc.

Calvert, G., Spence, C., and Stein, B. E., editors (2004). *The handbook of multisensory processes.* MIT Press, Cambridge, Mass.

Cetin, E., Carta, A., and Celiktutan, O. (2023). A Simple Recipe to Meta-Learn Forward and Backward Transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18732–18742.

Chaudhry, A., Dokania, P. K., Ajanthan, T., and Torr, P. H. S. (2018). Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV).*

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H. S., and Ranzato, M. (2019). Continual learning with tiny episodic memories. In *ICML Workshop on Multi-Task and Lifelong Reinforcement Learning.*

Chen, J., Cong, R., LUO, Y., Ip, H., and Kwong, S. (2023). Saving 100x Storage: Prototype Replay for Reconstructing Training Sample Distribution in Class-Incremental Semantic Segmentation. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 35988–35999. Curran Associates, Inc.

Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., and Luo, P. (2022). AdaptFormer: Adapting Vision Transformers for Scalable Visual Recognition. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16664–16678. Curran Associates, Inc.

Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. (2020). UNITER: UNiversal Image-TExt Representation Learning. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 104–120, Cham. Springer International Publishing.

Chen, Z. and Liu, B. (2018). *Lifelong Machine Learning.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Springer International Publishing, Cham.

Cheng, G., Han, J., and Lu, X. (2017). Remote Sensing Image Scene Classification: Benchmark and State of the Art. *Proceedings of the IEEE*, 105(10):1865–1883.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. (2019). BabyAI: A Platform to Study the Sample Efficiency of Grounded Language Learning. In *International Conference on Learning Representations*.

Chevalier-Boisvert, M., Dai, B., Towers, M., de Lazcano, R., Willems, L., Lahlou, S., Pal, S., Castro, P. S., and Terry, J. (2023). Minigrid & Miniworld: Modular & Customizable Reinforcement Learning Environments for Goal-Oriented Tasks. *arXiv:2306.13831*.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Cholesky, A.-L. (1924). Note sur une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés à un système d'équations linéaires en nombre inférieur à celui des inconnues. Application de la méthode à la résolution d'un système défini d'équations linéaires. *Bulletin Géodésique*, 2(1):67–77.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. (2014). Describing Textures in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised Cross-lingual Representation Learning at Scale. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Cossu, A., Carta, A., Passaro, L., Lomonaco, V., Tuytelaars, T., and Bacciu, D. (2024). Continual pre-training mitigates forgetting in language and vision. *Neural Networks*, 179:106492.

Davis, G. W. (2006). Homeostatic Control of Neural Activity: From Phenomenology to Molecular Design. *Annual Review of Neuroscience*, 29(1):307–323.

de Masson d' Autume, C., Ruder, S., Kong, L., and Yogatama, D. (2019). Episodic Memory in Lifelong Language Learning. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, and Li Fei-Fei (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL. IEEE.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Du, Y., Liu, Z., Li, J., and Zhao, W. X. (2022). A Survey of Vision-Language Pre-Trained Models. In Raedt, L. D., editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5436–5443. International Joint Conferences on Artificial Intelligence Organization.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., et al. (2024). The Llama 3 Herd of Models. *arXiv:2407.21783*.

Dudai, Y. (2004). The Neurobiology of Consolidations, Or, How Stable is the Engram? *Annual Review of Psychology*, 55(1):51–86.

Eichenbaum, H. (2004). Hippocampus: Cognitive Processes and Neural Representations that Underlie Declarative Memory. *Neuron*, 44(1):109–120.

Ermis, B., Zappella, G., Wistuba, M., Rawal, A., and Archambeau, C. (2022). Memory Efficient Continual Learning with Transformers. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 10629–10642. Curran Associates, Inc.

Faghri, F., Fleet, D. J., Kiros, J. R., and Fidler, S. (2018). VSE++: Improving Visual-Semantic Embeddings with Hard Negatives. *Proceedings of the British Machine Vision Conference (BMVC)*.

Farahani, A., Voghoei, S., Rasheed, K., and Arabnia, H. R. (2021). A Brief Review of Domain Adaptation. In Stahlbock, R., Weiss, G. M., Abou-Nasr, M., Yang, C.-Y., Arabnia, H. R., and Deligiannidis, L., editors, *Advances in Data*

Science and Information Engineering, pages 877–894, Cham. Springer International Publishing.

Farquhar, S. and Gal, Y. (2019). Towards Robust Evaluations of Continual Learning. *arXiv:1805.09733*.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. (2017). PathNet: Evolution Channels Gradient Descent in Super Neural Networks. *arXiv:1701.08734*.

Florensa, C., Held, D., Wulfmeier, M., Zhang, M., and Abbeel, P. (2017). Reverse Curriculum Generation for Reinforcement Learning. In Levine, S., Vanhoucke, V., and Goldberg, K., editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 482–495. PMLR.

Frankland, P. W. and Bontempi, B. (2005). The organization of recent and remote memories. *Nature Reviews Neuroscience*, 6(2):119–130.

Gatys, L., Ecker, A., and Bethge, M. (2016). A Neural Algorithm of Artistic Style. *Journal of Vision*, 16(12):326.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional Sequence to Sequence Learning. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Geva, M., Schuster, R., Berant, J., and Levy, O. (2021). Transformer Feed-Forward Layers Are Key-Value Memories. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Grant, W. S., Tanner, J., and Itti, L. (2017). Biologically plausible learning in neural networks with modulatory feedback. *Neural Networks*, 88:32–48.

Greco, C., Plank, B., Fernández, R., and Bernardi, R. (2019). Psycholinguistics Meets Continual Learning: Measuring Catastrophic Forgetting in Visual Question Answering. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3601–3605, Florence, Italy. Association for Computational Linguistics.

Grossberg, S. (1982). How Does a Brain Build a Cognitive Code? In Cohen, R. S. and Wartofsky, M. W., editors, *Studies of Mind and Brain*, volume 70, pages 1–52. Springer Netherlands, Dordrecht.

Gupta, K., Thérien, B., Ibrahim, A., Richter, M. L., Anthony, Q. G., Belilovsky, E., Rish, I., and Lesort, T. (2023). Continual Pre-Training of Large Language Models: How to re-warm your model? In *ICML Workshop on Efficient Systems for Foundation Models*.

Gurbuz, M. B. and Dovrolis, C. (2022). NISPA: Neuro-Inspired Stability-Plasticity Adaptation for Continual Learning in Sparse Networks. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8157–8174. PMLR.

Hadsell, R., Rao, D., Rusu, A. A., and Pascanu, R. (2020). Embracing Change: Continual Learning in Deep Neural Networks. *Trends in Cognitive Sciences*, 24(12):1028–1040.

Hao, Y., Dong, L., Wei, F., and Xu, K. (2019). Visualizing and Understanding the Effectiveness of BERT. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4143–4152, Hong Kong, China. Association for Computational Linguistics.

Hayes, T. L., Cahill, N. D., and Kanan, C. (2019). Memory Efficient Experience Replay for Streaming Learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9769–9776.

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., and Kanan, C. (2020). REMIND Your Neural Network to Prevent Catastrophic Forgetting. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 466–483, Cham. Springer International Publishing.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hebb, D. O. (1949). *The Organization of Behavior*. John Wiley & Sons, New York, 1st edition.

Hedden, T. and Gabrieli, J. D. E. (2004). Insights into the ageing mind: a view from cognitive neuroscience. *Nature Reviews Neuroscience*, 5(2):87–96.

Helber, P., Bischke, B., Dengel, A., and Borth, D. (2019). EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. (2021a). The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8340–8349.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. (2021b). Natural Adversarial Examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15262–15271.

Hensch, T. K. (2005). Critical period plasticity in local cortical circuits. *Nature Reviews Neuroscience*, 6(11):877–888.

Hinton, G. E., Krizhevsky, A., and Wang, S. D. (2011). Transforming Auto-Encoders. In Honkela, T., Duch, W., Girolami, M., and Kaski, S., editors, *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 44–51, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ho, S., Liu, M., Du, L., Gao, L., and Xiang, Y. (2024). Prototype-Guided Memory Replay for Continual Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):10973–10983.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67.

Holla, N., Mishra, P., Yannakoudakis, H., and Shutova, E. (2021). Meta-Learning with Sparse Experience Replay for Lifelong Language Learning. *arXiv:2009.04891*.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. (2019). Parameter-Efficient Transfer Learning for NLP. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Hu, E. J., shen, y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2022). LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Huang, Z., Zeng, Z., Huang, Y., Liu, B., Fu, D., and Fu, J. (2021). Seeing Out of the Box: End-to-End Pre-Training for Vision-Language Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12976–12985.

Huang, Z., Zeng, Z., Liu, B., Fu, D., and Fu, J. (2020). Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers. *arXiv:2004.00849*.

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1):106–154.

Hung, C.-Y., Tu, C.-H., Wu, C.-E., Chen, C.-H., Chan, Y.-M., and Chen, C.-S. (2019). Compacting, Picking and Growing for Unforgetting Continual Learning. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hurtado, J., Raymond, A., and Soto, A. (2021). Optimizing Reusable Knowledge for Continual Learning via Metalearning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 14150–14162. Curran Associates, Inc.

Huttenlocher, P. (2009). *Neural Plasticity: The Effects of Environment on the Development of the Cerebral Cortex.* Perspectives in Cognitive Neuroscience. Harvard University Press.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR.

Iscen, A., Zhang, J., Lazebnik, S., and Schmid, C. (2020). Memory-Efficient Incremental Learning Through Feature Adaptation. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 699–715, Cham. Springer International Publishing.

Iverson, K. E. (1962). A programming language. In *Proceedings of the May 1-3, 1962, Spring Joint Computer Conference*, AIEE-IRE '62 (Spring), pages 345–351, New York, NY, USA. Association for Computing Machinery.

Janson, P., Zhang, W., Aljundi, R., and Elhoseiny, M. (2022). A Simple Baseline that Questions the Use of Pretrained-Models in Continual Learning. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*.

Javed, K. and White, M. (2019). Meta-Learning Representations for Continual Learning. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. (2021). Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4904–4916. PMLR.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B., and Lim, S.-N. (2022). Visual Prompt Tuning. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 709–727, Cham. Springer Nature Switzerland.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. (2020). TinyBERT: Distilling BERT for Natural Language Understanding. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Jin, T., Yan, W., Wang, Y., Cai, S., Shuaiqifan, and Zhao, Z. (2024). Calibrating Prompt from History for Continual Vision-Language Retrieval and Grounding. In *ACM Multimedia 2024*.

Jin, X., Du, J., Sadhu, A., Nevatia, R., and Ren, X. (2020). Visually Grounded Continual Learning of Compositional Phrases. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2018–2029, Online. Association for Computational Linguistics.

Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., and Song, M. (2020). Neural Style Transfer: A Review. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3365–3385.

Jung, D., Han, D., Bang, J., and Song, H. (2023). Generating Instance-level Prompts for Rehearsal-free Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11847–11857.

Jung, S., Ahn, H., Cha, S., and Moon, T. (2020). Continual Learning with Node-Importance based Adaptive Group Sparse Regularization. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3647–3658. Curran Associates, Inc.

Kanakis, M., Bruggemann, D., Saha, S., Georgoulis, S., Obukhov, A., and Van Gool, L. (2020). Reparameterizing Convolutions for Incremental Multi-Task Learning Without Task Interference. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 689–707, Cham. Springer International Publishing.

Kandel, E. R. and Hawkins, R. D. (1992). The Biological Basis of Learning and Individuality. *Scientific American*, 267(3):78–87.

Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S. A., and Hudspeth, A. J. (2013). *Principles of neural science*. McGraw-Hill, New York, 5th edition.

Kang, H., Mina, R. J. L., Madjid, S. R. H., Yoon, J., Hasegawa-Johnson, M., Hwang, S. J., and Yoo, C. D. (2022). Forget-free Continual Learning with Winning Subnetworks. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 10734–10750. PMLR.

Kang, S., Shi, Z., and Zhang, X. (2023). Poisoning Generative Replay in Continual Learning to Promote Forgetting. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15769–15785. PMLR.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv:2001.08361*.

Kazemzadeh, S., Ordonez, V., Matten, M., and Berg, T. (2014). ReferItGame: Referring to Objects in Photographs of Natural Scenes. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, Doha, Qatar. Association for Computational Linguistics.

Ke, Z., Liu, B., Ma, N., Xu, H., and Shu, L. (2021a). Achieving Forgetting Prevention and Knowledge Transfer in Continual Learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 22443–22456. Curran Associates, Inc.

Ke, Z., Xu, H., and Liu, B. (2021b). Adapting BERT for Continual Learning of a Sequence of Aspect Sentiment Classification Tasks. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online. Association for Computational Linguistics.

Kemker, R., McClure, M., Abitino, A., Hayes, T., and Kanan, C. (2018). Measuring Catastrophic Forgetting in Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kim, W., Son, B., and Kim, I. (2021). ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR.

Kingma, D. P. and Ba, J. L. (2015). Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

Kingma, D. P. and Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv:1312.6114*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Knudsen, E. I. (2004). Sensitive Periods in the Development of the Brain and Behavior. *Journal of Cognitive Neuroscience*, 16(8):1412–1425.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks*, 37:52–65.

Konishi, T., Kurokawa, M., Ono, C., Ke, Z., Kim, G., and Liu, B. (2023). Parameter-Level Soft-Masking for Continual Learning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17492–17505. PMLR.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. (2013). 3D Object Representations for Fine-Grained Categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*.

Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images. *Master's thesis, University of Toronto*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F., Burges, C. J., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Kudithipudi, D., Aguilar-Simon, M., Babb, J., Bazhenov, M., Blackiston, D., Bongard, J., Brna, A. P., Chakravarthi Raja, S., Cheney, N., Clune, J., Daram, A., Fusi, S., Helfer, P., Kay, L., Ketz, N., Kira, Z., Kolouri, S., Krichmar, J. L., Kriegman, S., Levin, M., et al. (2022). Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210.

Kumaran, D., Hassabis, D., and McClelland, J. L. (2016). What Learning Systems do Intelligent Agents Need? Complementary Learning Systems Theory Updated. *Trends in Cognitive Sciences*, 20(7):512–534.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1989). Handwritten Digit Recognition with a Back-Propagation Network. In Touretzky, D., editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Lee, J. H., Kerzel, M., Ahrens, K., Weber, C., and Wermter, S. (2022). What is Right for Me is Not Yet Right for You: A Dataset for Grounding Relative Directions via Multi-Task Learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pages 1039–1045, Vienna, Austria. International Joint Conferences on Artificial Intelligence Organization.

Lee, K.-H., Chen, X., Hua, G., Hu, H., and He, X. (2018). Stacked Cross Attention for Image-Text Matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Lester, B., Al-Rfou, R., and Constant, N. (2021). The Power of Scale for Parameter-Efficient Prompt Tuning. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Li, C., Huang, Z., Paudel, D. P., Wang, Y., Shahbazi, M., Hong, X., and Van Gool, L. (2023a). A Continual Deepfake Detection Benchmark: Dataset, Methods, and Essentials. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1339–1349.

Li, J., Selvaraju, R., Gotmare, A., Joty, S., Xiong, C., and Hoi, S. C. H. (2021). Align before Fuse: Vision and Language Representation Learning with Momentum Distillation. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 9694–9705. Curran Associates, Inc.

Li, S., Hu, X., Lin, L., Liu, A., Wen, L., and Yu, P. S. (2023b). A Multi-Level Supervised Contrastive Learning Framework for Low-Resource Natural Language

Inference. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1771–1783.

Li, Y., Wang, N., Liu, J., and Hou, X. (2017). Demystifying neural style transfer. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 2230–2236. AAAI Press.

Li, Z. and Hoiem, D. (2018). Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947.

Lian, D., Zhou, D., Feng, J., and Wang, X. (2022). Scaling & Shifting Your Features: A New Baseline for Efficient Model Tuning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 109–123. Curran Associates, Inc.

Lin, S., Yang, L., Fan, D., and Zhang, J. (2022a). Beyond Not-Forgetting: Continual Learning with Backward Knowledge Transfer. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16165–16177. Curran Associates, Inc.

Lin, T., Wang, Y., Liu, X., and Qiu, X. (2022b). A survey of transformers. *AI Open*, 3:111–132.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, Cham. Springer International Publishing.

Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., and Stone, P. (2023). LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 44776–44791. Curran Associates, Inc.

Liu, X., Ji, K., Fu, Y., Tam, W., Du, Z., Yang, Z., and Tang, J. (2022). P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Liu, X., Wu, C., Menta, M., Herranz, L., Raducanu, B., Bagdanov, A. D., Jui, S., and de Weijer, J. v. (2020a). Generative Feature Replay for Class-Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Liu, X., Yu, H.-F., Dhillon, I., and Hsieh, C.-J. (2020b). Learning to Encode Position for Transformer with Continuous Dynamical Model. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6327–6335. PMLR.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692*.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022.

Logeswaran, L. and Lee, H. (2018). An efficient framework for learning sentence representations. In *International Conference on Learning Representations*.

Loo, N., Swaroop, S., and Turner, R. E. (2021). Generalized Variational Continual Learning. In *International Conference on Learning Representations*.

Lopez-Paz, D. and Ranzato, M. A. (2017). Gradient Episodic Memory for Continual Learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Loshchilov, I. and Hutter, F. (2019). Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Lövdén, M., Bäckman, L., Lindenberger, U., Schaefer, S., and Schmiedek, F. (2010). A theoretical framework for the study of adult cognitive plasticity. *Psychological Bulletin*, 136(4):659–676.

Lüddecke, T. and Ecker, A. (2022). Image Segmentation Using Text and Image Prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7086–7096.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Magee, J. C. and Johnston, D. (1997). A Synaptically Controlled, Associative Signal for Hebbian Plasticity in Hippocampal Neurons. *Science*, 275(5297):209–213.

Mai, Z., Li, R., Kim, H., and Sanner, S. (2021). Supervised Contrastive Replay: Revisiting the Nearest Class Mean Classifier in Online Class-Incremental Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3589–3599.

Malenka, R. C. and Bear, M. F. (2004). LTP and LTD: an embarrassment of riches. *Neuron*, 44(1):5–21.

Mallya, A., Davis, D., and Lazebnik, S. (2018). Piggyback: Adapting a Single Network to Multiple Tasks by Learning to Mask Weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Mallya, A. and Lazebnik, S. (2018). PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Martinetz, T. (1993). Competitive Hebbian Learning Rule Forms Perfectly Topology Preserving Maps. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 427–434, Amsterdam, NL. Springer.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457.

McDonnell, M. D., Gong, D., Parvaneh, A., Abbasnejad, E., and van den Hengel, A. (2023). RanPAC: Random Projections and Pre-trained Models for Continual Learning. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 12022–12053. Curran Associates, Inc.

McGaugh, J. L. (2000). Memory–a Century of Consolidation. *Science*, 287(5451):248–251.

Mendez, J. A. and Eaton, E. (2021). Lifelong Learning of Compositional Structures. In *International Conference on Learning Representations*.

Mendez, J. A., Seijen, H. v., and Eaton, E. (2022). Modular Lifelong Reinforcement Learning via Neural Composition. In *International Conference on Learning Representations*.

Mendez-Mendez, J., Kaelbling, L. P., and Lozano-Pérez, T. (2023). Embodied Lifelong Learning for Task and Motion Planning. In Tan, J., Toussaint, M., and Darvish, K., editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2134–2150. PMLR.

Miller, K. D. (1996). Synaptic Economics: Competition and Cooperation in Synaptic Plasticity. *Neuron*, 17(3):371–374.

Mitchell, R., Alliegro, A., Camoriano, R., Carrión-Ojeda, D., Carta, A., Chalvatzaki, G., Churamani, N., D'Eramo, C., Hamidi, S., Hesse, R., Hinder, F., Kamath, R. R., Lomonaco, V., Paul, S., Pistilli, F., Tuytelaars, T., Ven, G. M. v. d., Kersting, K., Schaub-Meyer, S., and Mundt, M. (2025). Continual Learning Should Move Beyond Incremental Classification. *arXiv:2502.11927*.

Mokady, R., Hertz, A., and Bermano, A. H. (2021). ClipCap: CLIP Prefix for Image Captioning. *arXiv:2111.09734*.

Molchanov, P., Mallya, A., Tyree, S., Frosio, I., and Kautz, J. (2019). Importance Estimation for Neural Network Pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Morrison, J. H. and Baxter, M. G. (2012). The ageing cortical synapse: hallmarks and implications for cognitive decline. *Nature Reviews Neuroscience*, 13(4):240–250.

Mountcastle, V. B. (1957). Modality and Topographic Properties of Single Neurons of Cat's Somatic Sensory Cortex. *Journal of Neurophysiology*, 20(4):408–434.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning series. MIT Press, Cambridge, MA.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. (2011). Multimodal deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 689–696, Madison, WI, USA. Omnipress.

Nilsback, M.-E. and Zisserman, A. (2006). A Visual Vocabulary for Flower Classification. In *2006 IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, New York, NY, USA. IEEE.

Oord, A. v. d., Li, Y., and Vinyals, O. (2019). Representation Learning with Contrastive Predictive Coding. *arXiv:1807.03748*.

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., et al. (2024). GPT-4 Technical Report. *arXiv:2303.08774*.

O'Reilly, R. C. and Rudy, J. W. (2000). Computational principles of learning in the neocortex and hippocampus. *Hippocampus*, 10(4):389–397.

Ostapenko, O., Rodriguez, P., Caccia, M., and Charlin, L. (2021). Continual Learning via Local Module Composition. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30298–30312. Curran Associates, Inc.

Panos, A., Kobe, Y., Reino, D. O., Aljundi, R., and Turner, R. E. (2023). First Session Adaptation: A Strong Replay-Free Baseline for Class-Incremental Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18820–18830.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. V. (2012). Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505, Providence, RI. IEEE.

Payeur, A., Guerguiev, J., Zenke, F., Richards, B. A., and Naud, R. (2021). Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature Neuroscience*, 24(7):1010–1019.

Pearson, K. (1895). Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58(347-352):240–242.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., and Wang, B. (2019). Moment Matching for Multi-Source Domain Adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Perez, E., Strub, F., De Vries, H., Dumoulin, V., and Courville, A. (2018). FiLM: Visual Reasoning with a General Conditioning Layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peters, R. (2006). Ageing and the brain. *Postgraduate Medical Journal*, 82(964):84–88.

Petit, G., Popescu, A., Schindler, H., Picard, D., and Delezoide, B. (2023). FeTrIL: Feature Translation for Exemplar-Free Class-Incremental Learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3911–3920.

Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., and Gurevych, I. (2021). AdapterFusion: Non-Destructive Task Composition for Transfer Learning. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Piaget, J. (1976). Piaget's Theory. In Inhelder, B., Chipman, H. H., and Zwingmann, C., editors, *Piaget and His School*, pages 11–23. Springer Berlin Heidelberg, Berlin, Heidelberg.

Pozo, K. and Goda, Y. (2010). Unraveling Mechanisms of Homeostatic Synaptic Plasticity. *Neuron*, 66(3):337–351.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. *OpenAI Blog*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. *OpenAI Blog*.

Ramapuram, J., Gregorova, M., and Kalousis, A. (2020). Lifelong generative modeling. *Neurocomputing*, 404:381–400.

Ramasesh, V. V., Dyer, E., and Raghu, M. (2021). Anatomy of Catastrophic Forgetting: Hidden Representations and Task Semantics. In *International Conference on Learning Representations*.

Rasch, B. and Born, J. (2013). About Sleep's Role in Memory. *Physiological Reviews*, 93(2):681–766.

Razdaibiedina, A., Mao, Y., Hou, R., Khabsa, M., Lewis, M., and Almahairi, A. (2023). Progressive Prompts: Continual Learning for Language Models. In *The Eleventh International Conference on Learning Representations*.

Rebuffi, S.-A., Bilen, H., and Vedaldi, A. (2017a). Learning multiple visual domains with residual adapters. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017b). iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., and Tesauro, a. G. (2019). Learning to Learn without Forgetting By Maximizing Transfer and Minimizing Interference. In *International Conference on Learning Representations*.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. (2016). Progressive Neural Networks. *arXiv:1606.04671*.

Rymarczyk, D., van de Weijer, J., Zieliński, B., and Twardowski, B. (2023). ICICLE: Interpretable Class Incremental Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1887–1898.

Salehi, S. S. M., Erdogmus, D., and Gholipour, A. (2017). Tversky Loss Function for Image Segmentation Using 3D Fully Convolutional Deep Networks. In Wang, Q., Shi, Y., Suk, H.-I., and Suzuki, K., editors, *Machine Learning in Medical Imaging*, pages 379–387, Cham. Springer International Publishing.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeurIPS 2019 Workshop on Energy Efficient Machine Learning and Cognitive Computing*.

Say, H. and Oztop, E. (2023). A Model for Cognitively Valid Lifelong Learning. In *2023 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1–7.

Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., and Hadsell, R. (2018). Progress & Compress: A scalable framework for continual learning. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4528–4537. PMLR.

Serra, J., Suris, D., Miron, M., and Karatzoglou, A. (2018). Overcoming Catastrophic Forgetting with Hard Attention to the Task. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4548–4557. PMLR.

Shen, F. and Hasegawa, O. (2010). Self-Organizing Incremental Neural Network and Its Application. In Diamantaras, K., Duch, W., and Iliadis, L. S., editors, *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 535–540, Berlin, Heidelberg. Springer Berlin Heidelberg.

Shepherd, J. D. and Huganir, R. L. (2007). The Cell Biology of Synaptic Plasticity: AMPA Receptor Trafficking. *Annual Review of Cell and Developmental Biology*, 23(1):613–643.

Shi, W. and Ye, M. (2023). Prototype Reminiscence and Augmented Asymmetric Knowledge Aggregation for Non-Exemplar Class-Incremental Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1772–1781.

Shin, H., Lee, J. K., Kim, J., and Kim, J. (2017). Continual Learning with Deep Generative Replay. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Shridhar, M., Manuelli, L., and Fox, D. (2022). CLIPort: What and Where Pathways for Robotic Manipulation. In Faust, A., Hsu, D., and Neumann, G., editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 894–906. PMLR.

Shu, Y., Cao, Z., Long, M., and Wang, J. (2019). Transferable Curriculum for Weakly-Supervised Domain Adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4951–4958.

Skantze, G. and Willemsen, B. (2022). CoLLIE: Continual Learning of Language Grounding from Language-Image Embeddings. *Journal of Artificial Intelligence Research*, 74:1201–1223.

Smith, J. S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., and Kira, Z. (2023a). 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)*, pages 11909–11919.

Smith, J. S., Tian, J., Halbe, S., Hsu, Y.-C., and Kira, Z. (2023b). A Closer Look at Rehearsal-Free Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2410–2420.

Srinivasan, T., Chang, T.-Y., Pinto Alva, L., Chochlakis, G., Rostami, M., and Thomason, J. (2022). CLiMB: A Continual Learning Benchmark for Vision-and-Language Tasks. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 29440–29453. Curran Associates, Inc.

Srivastava, N. and Salakhutdinov, R. R. (2012). Multimodal Learning with Deep Boltzmann Machines. In Pereira, F., Burges, C. J., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Stein, B. E. and Stanford, T. R. (2008). Multisensory integration: current issues from the perspective of the single neuron. *Nature Reviews Neuroscience*, 9(4):255–266.

Stern, Y. (2009). Cognitive reserve. *Neuropsychologia*, 47(10):2015–2028.

Su, W., Zhu, X., Cao, Y., Li, B., Lu, L., Wei, F., and Dai, J. (2020). VL-BERT: Pre-training of Generic Visual-Linguistic Representations. In *International Conference on Learning Representations*.

Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., and Jorge Cardoso, M. (2017). Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations. In Cardoso, M. J., Arbel, T., Carneiro, G., Syeda-Mahmood, T., Tavares, J. M. R., Moradi, M., Bradley, A., Greenspan, H., Papa, J. P., Madabhushi, A., Nascimento, J. C., Cardoso, J. S., Belagiannis, V., and Lu, Z., editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248, Cham. Springer International Publishing.

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. (2017). Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Sun, F.-K., Ho, C.-H., and Lee, H.-Y. (2020a). LAMOL: LAnguage MOdeling for Lifelong Language Learning. In *International Conference on Learning Representations*.

Sun, J., Wang, S., Zhang, J., and Zong, C. (2020b). Distill and Replay for Continual Language Learning. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Takashima, A., Nieuwenhuis, I. L., Jensen, O., Talamini, L. M., Rijpkema, M., and Fernández, G. (2009). Shift from Hippocampal to Neocortical Centered Retrieval Network with Consolidation. *The Journal of Neuroscience*, 29(32):10087–10093.

Takeuchi, T., Duszkiewicz, A. J., and Morris, R. G. M. (2014). The synaptic plasticity and memory hypothesis: encoding, storage and persistence. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1633):20130288.

Tan, H. and Bansal, M. (2019). LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Tang, Y., Zhang, C., Xu, H., Chen, S., Cheng, J., Leng, L., Guo, Q., and He, Z. (2023). Neuro-Modulated Hebbian Learning for Fully Test-Time Adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3728–3738.

Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R. T., Kim, N., Durme, B. V., Bowman, S., Das, D., and Pavlick, E. (2019). What do you learn from context? Probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Thrun, S. and Pratt, L. (1998). Learning to Learn: Introduction and Overview. In Thrun, S. and Pratt, L., editors, *Learning to Learn*, pages 3–17. Springer US, Boston, MA.

Toldo, M. and Ozay, M. (2022). Bring Evanescent Representations to Life in Lifelong Class Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16732–16741.

Turrigiano, G. (2012). Homeostatic Synaptic Plasticity: Local and Global Mechanisms for Stabilizing Neuronal Function. *Cold Spring Harbor Perspectives in Biology*, 4(1):a005736–a005736.

Turrigiano, G. G. (1999). Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. *Trends in Neurosciences*, 22(5):221–227.

Turrigiano, G. G. (2008). The Self-Tuning Neuron: Synaptic Scaling of Excitatory Synapses. *Cell*, 135(3):422–435.

Turrigiano, G. G. and Nelson, S. B. (2004). Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, 5(2):97–107.

Van De Ven, G. M., Tuytelaars, T., and Tolias, A. S. (2022). Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is All you Need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vilalta, R. and Drissi, Y. (2002). A Perspective View and Survey of Meta-Learning. *Artificial Intelligence Review*, 18(2):77–95.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57.

Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C., and Gerstner, W. (2011). Inhibitory Plasticity Balances Excitation and Inhibition in Sensory Pathways and Memory Networks. *Science*, 334(6062):1569–1573.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology.

Wang, L., Zhang, X., Su, H., and Zhu, J. (2024). A Comprehensive Survey of Continual Learning: Theory, Method and Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20.

Wang, Q., Wang, R., Wu, Y., Jia, X., and Meng, D. (2023). CBA: Improving Online Continual Learning via Continual Bias Adaptor. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19082–19092.

Wang, X., Chen, Y., and Zhu, W. (2022a). A Survey on Curriculum Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576.

Wang, Y., Gan, W., Yang, J., Wu, W., and Yan, J. (2019). Dynamic Curriculum Learning for Imbalanced Data Classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Wang, Y., Huang, Z., and Hong, X. (2022b). S-Prompts Learning with Pre-trained Transformers: An Occam's Razor for Domain Incremental Learning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 5682–5695. Curran Associates, Inc.

Wang, Z., Yu, J., Yu, A. W., Dai, Z., Tsvetkov, Y., and Cao, Y. (2022c). SimVLM: Simple Visual Language Model Pretraining with Weak Supervision. In *International Conference on Learning Representations*.

Wang, Z., Zhan, Z., Gong, Y., Yuan, G., Niu, W., Jian, T., Ren, B., Ioannidis, S., Wang, Y., and Dy, J. (2022d). SparCL: Sparse Continual Learning on the Edge. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 20366–20380. Curran Associates, Inc.

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J., and Pfister, T. (2022e). DualPrompt: Complementary Prompting for Rehearsal-Free Continual Learning. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, Cham. Springer Nature Switzerland.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., and Pfister, T. (2022f). Learning To Prompt for Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 139–149.

Welford, B. P. (1962). Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, 4(3):419–420.

Werker, J. F. and Hensch, T. K. (2015). Critical Periods in Speech Perception: New Directions. *Annual Review of Psychology*, 66(1):173–196.

Wilson, M. A. and McNaughton, B. L. (1994). Reactivation of Hippocampal Ensemble Memories During Sleep. *Science*, 265(5172):676–679.

Wiwatcharakoses, C. and Berrar, D. (2020). SOINN+, a Self-Organizing Incremental Neural Network for Unsupervised Learning from Noisy Data Streams. *Expert Systems with Applications*, 143:113069.

Wixted, J. T. (2004). The Psychology and Neuroscience of Forgetting. *Annual Review of Psychology*, 55(1):235–269.

Wu, C., Lin, Z., Cohen, S., Bui, T., and Maji, S. (2020). PhraseCut: Language-Based Image Segmentation in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., and Zhang, L. (2021a). CvT: Introducing Convolutions to Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22–31.

Wu, T., Luo, L., Li, Y.-F., Pan, S., Vu, T.-T., and Haffari, G. (2024a). Continual Learning for Large Language Models: A Survey. *arXiv:2402.01364*.

Wu, X., Dyer, E., and Neyshabur, B. (2021b). When Do Curricula Work? In *International Conference on Learning Representations*.

Wu, Y., Huang, L.-K., Wang, R., Meng, D., and Wei, Y. (2024b). Meta Continual Learning Revisited: Implicitly Enhancing Online Hessian Approximation via Variance Reduction. In *The Twelfth International Conference on Learning Representations*.

Wu, Z., Baek, C., You, C., and Ma, Y. (2021c). Incremental Learning via Rate Reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1133.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yoo, M., Jang, J., Park, W.-j., and Woo, H. (2024). Exploratory Retrieval-Augmented Planning For Continual Embodied Instruction Following. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C., editors, *Advances in Neural Information Processing Systems*, volume 37, pages 67034–67060. Curran Associates, Inc.

Yoon, J., Kim, S., Yang, E., and Hwang, S. J. (2020). Scalable and Order-robust Continual Learning with Additive Parameter Decomposition. In *International Conference on Learning Representations*.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. (2018). Lifelong Learning with Dynamically Expandable Networks. In *International Conference on Learning Representations*.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Yu, J., Zhuge, Y., Zhang, L., Hu, P., Wang, D., Lu, H., and He, Y. (2024). Boosting Continual Learning of Vision-Language Models via Mixture-of-Experts Adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23219–23230.

Zeng, A., Florence, P., Tompson, J., Welker, S., Chien, J., Attarian, M., Armstrong, T., Krasin, I., Duong, D., Sindhwani, V., and Lee, J. (2021). Transporter Networks: Rearranging the Visual World for Robotic Manipulation. In Kober, J., Ramos, F., and Tomlin, C., editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 726–747. PMLR.

Zenke, F., Poole, B., and Ganguli, S. (2017). Continual Learning Through Synaptic Intelligence. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR.

Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A. S., Neumann, M., Dosovitskiy, A., Beyer, L., Bachem, O., Tschannen, M., Michalski, M., Bousquet, O., Gelly, S., and Houlsby, N. (2020). A Large-scale Study of Representation Learning with the Visual Task Adaptation Benchmark. *arXiv:1910.04867*.

Zhang, G., Wang, L., Kang, G., Chen, L., and Wei, Y. (2023). SLCA: Slow Learner with Classifier Alignment for Continual Learning on a Pre-trained Model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19148–19158.

Zhang, J. O., Sax, A., Zamir, A., Guibas, L., and Malik, J. (2020). Side-Tuning: A Baseline for Network Adaptation via Additive Side Networks. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 698–714, Cham. Springer International Publishing.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Zhang, Y., Yin, Z., Shao, J., and Liu, Z. (2022). Benchmarking Omni-Vision Representation Through the Lens of Visual Realms. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 594–611, Cham. Springer Nature Switzerland.

Zhao, Y., Zheng, Y., Yu, B., Tian, Z., Lee, D., Sun, J., Li, Y., and Zhang, N. L. (2022). Semi-Supervised Lifelong Language Learning. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3937–3951, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. (2017). Scene Parsing through ADE20K Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, Honolulu, HI. IEEE.

Zhou, D.-W., Cai, Z.-W., Ye, H.-J., Zhan, D.-C., and Liu, Z. (2024). Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need. *International Journal of Computer Vision*.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. (2022a). Conditional Prompt Learning for Vision-Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16816–16825.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. (2022b). Learning to Prompt for Vision-Language Models. *International Journal of Computer Vision*, 130(9):2337–2348.

Zhu, H., Majzoubi, M., Jain, A., and Choromanska, A. (2024). TAME: Task Agnostic Continual Learning using Multiple Experts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4139–4148, Seattle, WA, USA. IEEE.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. (2021). A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1):43–76.

Eidesstattliche Versicherung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Sofern im Zuge der Erstellung der vorliegenden Dissertationsschrift generative Künstliche Intelligenz (gKI) basierte elektronische Hilfsmittel verwendet wurden, versichere ich, dass meine eigene Leistung im Vordergrund stand und dass eine vollständige Dokumentation aller verwendeten Hilfsmittel gemäß der Guten wissenschaftlichen Praxis vorliegt. Ich trage die Verantwortung für eventuell durch die gKI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate.

Hamburg, 02.05.2025	Kyra Ahrens
....	....
Ort, Datum	Unterschrift

Erklärung zur Veröffentlichung

Ich erkläre mein Einverständnis mit der Einstellung dieser Dissertation in den Bestand der Bibliothek.

Hamburg, 02.05.2025	Kyra Ahrens
....	....
Ort, Datum	Unterschrift