

**VIETNAM NATIONAL UNIVERSITY,
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**



PHAM THI QUYNH TRANG

**MITIGATING CATASTROPHIC FORGETTING IN CONTINUAL LEARNING
VIA MEMORY REPLAY, KNOWLEDGE TRANSFER,
AND ADAPTIVE ARCHITECTURE OPTIMIZATION**

**(GIẢM THIỂU QUÊN NGHIÊM TRỌNG TRONG HỌC LIÊN TỤC SỬ DỤNG
CÁC PHƯƠNG PHÁP PHÁT LẠI BỘ NHỚ, CHUYỂN GIAO TRI THỨC
VÀ KIẾN TRÚC ĐỘNG)**

DOCTOR OF PHILOSOPHY IN INFORMATION SYSTEMS DISSERTATION

Major: Information Systems

Code: 9480104

Hanoi - 2026

**VIETNAM NATIONAL UNIVERSITY,
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

Pham Thi Quynh Trang

**MITIGATING CATASTROPHIC FORGETTING IN CONTINUAL LEARNING
VIA MEMORY REPLAY, KNOWLEDGE TRANSFER,
AND ADAPTIVE ARCHITECTURE OPTIMIZATION**

**(GIẢM THIỂU QUÊN NGHIÊM TRỌNG TRONG HỌC LIÊN TỤC SỬ DỤNG
CÁC PHƯƠNG PHÁP PHÁT LẠI BỘ NHỚ, CHUYỂN GIAO TRI THỨC
VÀ KIẾN TRÚC ĐỘNG)**

DOCTOR OF PHILOSOPHY IN INFORMATION SYSTEMS DISSERTATION

Major: Information Systems

Code: 9480104

PHD Candidate

SUPERVISORS

Hanoi - 2026

Abstract

Deep neural networks have achieved outstanding performance in conventional static learning settings where training data are fully available before model deployment. In real-world applications, however, data arrive continuously and in a non-stationary fashion, making it impractical to retrain models from scratch each time new information becomes available. Continual learning addresses this issue by training models incrementally, with the central objective of preserving prior task performance while assimilating new knowledge. The principal obstacle in this setting is *catastrophic forgetting*: the well-documented phenomenon in which parameter updates for a new task tend to interfere with and erode the representations that encode prior knowledge, resulting in significant drops in accuracy on earlier tasks.

Existing approaches to mitigating catastrophic forgetting fall into three broad families, each with characteristic limitations. Replay-based methods retain subsets of past data for periodic rehearsal but raise privacy concerns and face diminishing memory budgets as the task sequence grows. Knowledge-transfer methods avoid data retention by distilling previously learned representations into new models, yet remain constrained by the capacity of a fixed parameter space. Architecture-based methods expand or reconfigure the network to accommodate new tasks, but conventional expansion strategies incur parameter growth proportional to the number of tasks and fail to exploit shared structure across them.

This dissertation addresses these limitations through a progressive research programme in which each stage is motivated by the shortcomings of the preceding one. In the first stage, when data retention is feasible but storing raw samples is precluded by privacy constraints, we propose a prototype-based memory augmentation framework for continual relation extraction. Past class distributions are approximated by perturbing stored prototypes with derived Gaussian noise, eliminating the need to retain actual training examples. An Energy-based Latent feature Alignment (ELI) module is further

introduced to detect and correct representational drift in the latent space across sequential tasks, ensuring that previously learned class boundaries remain geometrically stable as new tasks are incorporated.

In the second stage, when even compact memory buffers are unavailable, we develop two complementary exemplar-free knowledge transfer methods. A Continual Learning Plugin (CL-plugin) is proposed, augmented with Contrastive Ensemble Distillation (CED) and Contrastive Supervised Classification (CSC) losses, which enables selective and discriminative knowledge transfer in the task-incremental aspect sentiment classification setting without storing any past data. Separately, ZeroRFF combines Random Fourier Feature mappings with a frozen pre-trained backbone and a closed-form analytic learning solution, reformulating the class-incremental learning problem as a regularised least-squares regression that can be solved recursively and exactly, thereby eliminating both exemplar storage and iterative gradient optimisation.

In the third stage, when task diversity and sequence length expose the capacity limitations inherent in fixed-parameter approaches, we introduce four adaptive architecture methods with dynamic parameter allocation. A multi-head framework incorporating Online Prototype Equilibrium (OPE) and Adaptive Prototypical Feedback (APF) explicitly disentangles within-task prediction from task-identity prediction, reducing inter-task class confusion through prototype-level balancing and targeted boundary reinforcement. An energy-based latent alignment extension of this framework further prevents latent space drift in the medical image classification setting. DAKTA integrates a Vision Transformer backbone adapted via Low-Rank Adaptation (LoRA) with a Kolmogorov-Arnold Classifier (KAC) head, whose learnable radial basis functions model local feature-class relationships more expressively than conventional linear classifiers; Fisher Information Matrix regularisation is applied jointly to the backbone and classifier to protect task-critical parameters throughout incremental training. Finally, SO-LoRA reconceptualises low-rank adaptation geometrically: a fixed universal orthogonal basis is shared across all tasks, and each task is restricted to learning sparse compositions over this stable coordinate system. Group sparsity explicitly bounds inter-task subspace overlap, and a gradient-aware exponential moving average mask protects historically important rank dimensions from interference, maintaining a constant adapter footprint regardless of sequence length.

Comprehensive experiments on standard continual learning for both text and image classification benchmarks demonstrate that the proposed methods achieve state-of-the-art

results across continual relation extraction, aspect sentiment classification and class-incremental image recognition. Systematic ablation studies confirm the contribution of each component, and a formal theoretical analysis establishes interference bounds for SO-LoRA grounded in the geometry of sparse subspace representations. Collectively, these contributions advance research in continual learning and deliver principled, practical solutions suited to real-world deployment, where privacy, memory, and computational resources are often tightly constrained.

Table of contents

ABBREVIATIONS	vii
LIST OF FIGURES	xi
LIST OF TABLES	xiii
PREAMBLE	1
1 THEORETICAL BACKGROUND AND PRELIMINARIES	12
1.1 Continual Learning Fundamentals	12
1.1.1 Definitions of Lifelong Learning	13
1.1.2 Continual Learning Definition	14
1.1.3 Mitigating Catastrophic Forgetting in Continual Learning: Objectives and Challenges	15
1.2 Typical Scenario and Approaches	19
1.2.1 Typical Scenario	19
1.2.2 Approaches to Continual Learning	23
1.3 Related Resources	30
1.3.1 Datasets for Text Analysis	30
1.3.2 Datasets for Image Analysis	35
1.4 Evaluation Metric	38
1.5 Summary	40
2 MEMORY-BASED REPLAY METHODS FOR CONTINUAL RELATION EXTRACTION	42
2.1 Introduction	42
2.2 Related Work	43
2.3 Proposed Method	46
2.3.1 Model Training Pipeline	46
2.3.2 Sample Encoder	46

2.3.3	Initial Training for New Task	47
2.3.4	Sample selection	48
2.3.5	Memory augmentation based on prototype using derived Gaussian noise	48
2.3.6	Memory Replay	49
2.3.7	Prediction	50
2.4	Extension With Energy-based Model Alignment (ELI)	51
2.5	Experimental Results	54
2.5.1	Experiment Setting	54
2.5.2	Metric and Baseline Comparisons	55
2.5.3	Result and discussion	55
2.6	Summary	59
3	KNOWLEDGE TRANSFER IN PRESERVING PREVIOUS INFORMATION IN CONTINUAL LEARNING	61
3.1	Introduction	61
3.2	Knowledge Distillation with Contrastive Learning Approach	62
3.2.1	Related Work	63
3.2.2	Continual Learning Plugin (CL-plugin)	65
3.2.3	Contrastive Ensemble Distillation (CED) Loss	68
3.2.4	Contrastive Supervised Learning of Current Task (CSC) Loss	69
3.2.5	Total Loss	70
3.2.6	Experimental Results	70
3.3	Exemplar-Free Knowledge Retention via Analytic Learning on Frozen Representations	75
3.3.1	Related Work	75
3.3.2	Preliminary	76
3.3.3	A Random Fourier Features and Analytic Learning Method for Class Incremental Learning	79
3.3.4	Algorithm Description	80
3.3.5	Experimental Results	83
3.4	Summary	85
4	ADAPTIVE ARCHITECTURES AND DYNAMIC PARAMETER ALLOCATIONS	87
4.1	Introduction	87

4.2	A Multi-Head Approach with Online Prototype Equilibrium and Adaptive Prototypical Feedback	89
4.2.1	Related Work	89
4.2.2	Preliminary	91
4.2.3	Feature representation optimization	94
4.2.4	Classification layer optimization	97
4.2.5	Experimental Results	99
4.2.6	Integrating Latent Alignment via Energy-Based Model	100
4.2.7	Experimental Results	102
4.3	DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic in Continual Learning	105
4.3.1	Related Work	106
4.3.2	Vision Transformer Backbone and Low-Rank Adaptation (LoRA)	108
4.3.3	Incremental Task Arithmetic (ITA) and Model Composition	108
4.3.4	Kolmogorov-Arnold Classifier Head	109
4.3.5	Directional Vector Fusion	110
4.3.6	Second-Order Regularization with the Fisher Information Matrix	110
4.3.7	Training Objective and Step-by-Step Procedure	110
4.3.8	Algorithm	112
4.3.9	Experimental Results	113
4.3.10	Ablation Study: Effect of Second-Order Regularization Placement	114
4.4	SO-LoRA: Sparse Orthogonal LoRA for Parameter-Efficient Continual Learning	115
4.4.1	Related Work	116
4.4.2	Preliminary	117
4.4.3	SO-LoRA: Sparse Orthogonal LoRA	120
4.4.4	Theoretical Analysis	121
4.4.5	Experimental results	122
4.5	Summary	127
	CONCLUSION	129
	LIST OF PUBLICATION	133
	BIBLIOGRAPHY	135

ABBREVIATIONS

AA	Average Accuracy
ACL	Analytic Continual Learning
AI	Artificial Intelligence
APF	Adaptive Prototypical Feedback
APT	A-la-carte Prompt Tuning
ASC	Aspect Sentiment Classification
AUC	Area Under the Curve
BBCL	Blurred Boundary Continual Learning
BERT	Bidirectional Encoder Representations from Transformers
BWT	Backward Transfer
CED	Contrastive Ensemble Distillation
CF	Catastrophic Forgetting
CIL	Class-Incremental Learning
CL	Continual Learning
CL-plugin	Continual Learning Plugin
CPT	Continual Pre-Training
CRE	Continual Relation Extraction
CRL	Contrastive Representation Learning
CSC	Contrastive Supervised learning of Current task Loss
CTR	Continual learning with Task-specific Representation

DAKTA	Directional Kolmogorov-Arnold Classifier for Task Arithmetic
DeiT	Data-efficient Image Transformers
DIL	Domain-Incremental Learning
EBM	Energy-Based Model
ELI	Energy-based Latent feature alignment
EMA	Exponential Moving Average
EWC	Elastic Weight Consolidation
FIM	Fisher Information Matrix
FM	Forgetting Measure
FWT	Forward Transfer
GACL	Generalized Analytic Continual Learning
GCIL	Generalized Class-Incremental Learning
GEM	Gradient Episodic Memory
GPU	Graphics Processing Unit
I.I.D.	Independent and Identically Distributed
iCaRL	Incremental Classifier and Representation Learning
ICS	Inter-task Class Separation
IE	Information Extraction
IIL	Instance-Incremental Learning
IM	Intransigence Measure
IND	In-Distribution
ITA	Incremental Task Arithmetic
KAC	Kolmogorov-Arnold Classifier
KB	Knowledge Base
KBP	Knowledge Base Population
KSM	Knowledge Sharing Sub-Module
LLL	LifeLong Learning

LML	Lifelong Machine Learning
LoRA	Low-Rank Adaptation
LwF	Learning without Forgetting
MAML	Model-Agnostic Meta-Learning
MAS	Memory Aware Synapses
MCMC	Markov Chain Monte Carlo
MLP	Multi-Layer Perceptron
MTL	Multi-Task Learning
NCM	Nearest Class Mean
NFH	No Forgetting Handling
NLP	Natural Language Processing
OCL	Online Continual Learning
OOD	Out-of-Distribution
OPE	Online Prototype Equilibrium
PEFT	Parameter-Efficient Fine-Tuning
PNN	Progressive Neural Networks
RBF	Radial Basis Function
RE	Relation Extraction
RFF	Random Fourier Features
SI	Synaptic Intelligence
SO-LoRA	Sparse Orthogonal Low-Rank Adaptation
SOTA	State of the Art
TACRED	TAC Relation Extraction Dataset
TFCL	Task-Free Continual Learning
TIL	Task-Incremental Learning
TP	Task-identity Prediction
TSM	Task-Specific Sub-Module

ViT	Vision Transformer
WP	Within-task Prediction
ZeroRFF	Zero-exemplar Random Fourier Features

List of Figures

1	(a) Static machine learning learns from all available data only once. (b) Adaptive machine learning learns continually from new data as it becomes available.	1
2	Annual publication trends for the keywords <i>continual learning</i> , <i>lifelong learning</i> , and <i>incremental learning</i> in DBLP	3
3	The Dissertation outline.	10
4	Overview of the dissertation’s three-stage approach to catastrophic forgetting: (1) memory replay without storing raw data, (2) exemplar-free knowledge transfer, and (3) adaptive architecture with dynamic capacity allocation. Each stage addresses a progressively stricter constraint, with residual limitations motivating the next.	11
1.1	(a) Adam experiences significant degradation in performance due to catastrophic forgetting. (b & c) As newer tasks are introduced, Adam exhibits decreased plasticity and performs notably worse compared to instances where Adam is restarted later.[106].	17
1.2	A comprehensive and advanced classification of notable continual learning methods [122].	23
1.3	A conceptual framework for continual learning includes [122].	29
2.1	CRL overall architecture [139]	44
2.2	Module of consistent representation using prototype augmentation(Ours).	45
2.3	A visualization of relation representation learned by our method at the different tasks on Fewrel and TACRED dataset.	50
2.4	Overall architecture of our proposed continual relation extraction model. Here, x_i denotes the input (including a sentence and an entity pair appearing in it) of our model.	52
3.1	Overview of CLASSIC [58]	65
3.2	Architecture of CTR [56]	66

3.3	The overall architecture of our proposed model.	66
3.4	General architecture of BERT (left) and the CTR system (middle) and the CL-plugin (right) that are integrated into BERT.	68
3.5	The architecture details of the CL-plugin within our model.	69
3.6	Overview of GACL [144]	76
3.7	Illustration of Class-Incremental Learning (CIL)	77
3.8	Overall architecture of ZeroRFF	80
4.1	Illustration of OnPro framework [126]	90
4.2	The overall architecture of our proposed method.	93
4.3	Task accuracy metrics during the training process of our method (green line) vs. the Onpro method (red line).	101
4.4	Energy-based latent aligner.	102
4.5	Example of image from 10 classes in CCH5000 dataset	103
4.6	Pipeline of the DAKTA framework, integrating a Vision Transformer (ViT) backbone with Low-Rank Adaptation (LoRA) and a Kolmogorov-Arnold Classifier (KAC) using Gaussian Radial Basis Functions (RBFs) for class-incremental learning. The process involves feature extraction from both old and new models, task-specific parameter composition via Incremental Task Arithmetic (ITA), and regularization using the Fisher Information Matrix to mitigate catastrophic forgetting.	106
4.7	Comparison of model accuracy across multiple datasets in a continual learning setting, highlighting the impact of catastrophic forgetting. . . .	114
4.8	SO-LoRA represents adaptation as sparse coefficients $B^{(t)}$ over a frozen orthonormal basis A . Group sparsity limits overlap across tasks, while a gradient-aware mask M protects historically important rank dimensions.	119
4.9	Average accuracy on ImageNet-R across sequence lengths.	124
4.10	Rank sensitivity. Performance peaks at $r = 40$, balancing capacity and interference.	126
4.11	Internal dynamics. Basis usage (Left) and gradient masks (Right) show dynamic allocation.	126

List of Tables

2	Summary of research gaps and the dissertation’s progressive research logic	7
1.1	Formal comparison of common continual learning scenarios, adapted and reorganized from the taxonomy proposed in [122].	22
1.2	Summary of dataset scale used throughout this dissertation	30
1.3	Summary of contributions in this dissertation.	41
2.1	Accuracy (%) comparison on the FewRel dataset across ten incremental tasks (T1–T10). Results for EA-EMR, EMAR, CML, RP-CRE, and CRL are as reported in their original papers.	56
2.2	Accuracy (%) comparison on the TACRED dataset across ten incremental tasks (T1–T10). Results for EA-EMR, EMAR, CML, RP-CRE, and CRL are as reported in their original papers.	57
2.3	Experimental accuracy of our proposed CRE model and existing state-of-the-art models across all tasks on the FewRel dataset. T is an abbreviation for Task.	57
2.4	Experimental accuracy of our proposed CRE model and existing state-of-the-art models across all tasks on the TACRED dataset. T is an abbreviation for Task.	58
3.1	Inherited components vs. new contributions in CL-plugin	65
3.2	Number of sentences per dataset, referred to as tasks in the context of continual learning [56]	71
3.3	Average Accuracy (Acc.) and Macro-F1 (MF1) over five random sequences of 19 tasks.	72
3.4	Ablation experimental results.	73
3.5	Experimental results on CIFAR-100 (%).	84
3.6	Comparison of principles between CL-plugin and ZeroRFF	85
4.1	Training time per task (minutes) for OnPro and ModifiedOnPro (ours) on CIFAR-10 and CIFAR-100 with memory size $\mathcal{M} = 200$	96

4.2	Average accuracy of our model and SOTA models with different buffer sizes $\mathcal{M}$	99
4.3	Average Forgetting of our model and SOTA models with different buffer sizes $\mathcal{M}$	100
4.4	Comparative results on CCH5000 dataset with one and two classes per task settings.	104
4.5	Systematic ablation analysis with computational profiling (GPU in GB and Time in minutes).	104
4.6	Comparative results on CCH5000 dataset with varying classes per task settings.	105
4.7	Comparison with SOTA (Final Accuracy $\uparrow$). Best results in bold. EWC, LwF-MC, DER++ (buffer size of 1000 examples), SEED, and TMC rely on full fine-tuning; L2P, CODA, and APT utilize prompt-based learning. Finally, InfLoRA adopts LoRA fine-tuning.	113
4.8	Ablation study on second-order regularization.	115
4.9	Comparison with the SOTA across several benchmarks(Final Forgetting $\downarrow$). Note that a lower forgetting score indicates better performance. . .	115
4.10	Comparison of LoRA-based continual learning methods.	116
4.11	Performance on ImageNet-R under different task granularities. Best is bold.	123
4.12	Performance on ImageNet-A, CIFAR-100, and CUB-200 (10 tasks). Best is bold.	124
4.13	Ablation on 10-task benchmarks. Impact of group sparsity and gradient-aware mask.	125
4.14	Ablation across sequence lengths on ImageNet-R.	125

PREAMBLE

Research Motivations

Static machine learning, a conventional approach characterized by training models on fixed datasets and deploying them without further adaptation, has been a cornerstone in various domains, spanning domains such as visual object recognition and text understanding [20]. However, despite its widespread adoption and success, this approach inherently carries several limitations that hinder its effectiveness in dynamic and evolving environments.

One of the primary limitations of static machine learning is its inability to adapt to changes in the underlying data distribution. Conventional machine learning models are built on the assumption that training data adequately represents deployment conditions. In practice, however, the statistical properties of input data frequently evolve due to factors including seasonal variation, shifting user behaviour, and the emergence of new trends. As a result, static models may become less accurate or even obsolete as they fail to capture the evolving patterns in the data.



Figure 1: (a) Static machine learning learns from all available data only once. (b) Adaptive machine learning learns continually from new data as it becomes available.

Furthermore, static models may exhibit limited generalization capabilities, particularly when faced with data that differs significantly from the training set. These models are trained to make predictions based on the patterns present in the training data, and their performance may degrade when confronted with new or unseen data during deployment. In dynamic environments where the data landscape is constantly evolving, static models may struggle to generalize effectively, leading to suboptimal performance and diminished reliability.

Another critical limitation of static machine learning is the lack of continual learning or updating mechanisms. Once deployed, static models remain static and cannot adapt to new information or incorporate feedback from real-world interactions. In dynamic environments with continuous data streams, static models may quickly become outdated, necessitating frequent retraining and redeployment to maintain performance. This incurs additional computational costs and resource overhead and introduces delays in responding to changing circumstances.

Moreover, static machine learning models are vulnerable to distribution shift, whereby the statistical relationship between input features and their associated labels evolves during deployment. Concept drift can occur due to various factors, such as changes in user behaviour, environmental conditions, or underlying processes. Static models, being static, are ill-equipped to recognize or adapt to these changes, resulting in deteriorating performance over time and potentially misleading predictions.

While static machine learning has been instrumental in many applications, its limitations become evident in dynamic and evolving environments. Alternative approaches, such as continual learning, have emerged to overcome these challenges and ensure robust performance. By incrementally incorporating new information, continual learning equips models with the capacity to evolve alongside changing data distributions, overcoming the rigidity of conventional static approaches and laying the groundwork for more robust and adaptable learning systems.

Conversely, continual learning solves these challenges by enabling models to adapt and improve over time. Continual learning algorithms are designed to incrementally learn from incoming data, allowing them to update their knowledge and adapt to environmental changes continuously. Through the selective integration of new task knowledge without disrupting previously learned representations, continual learning systems sustain high accuracy across dynamic and non-stationary environments. This capability is precious in applications where the data distribution is non-stationary or subject to concept drift.

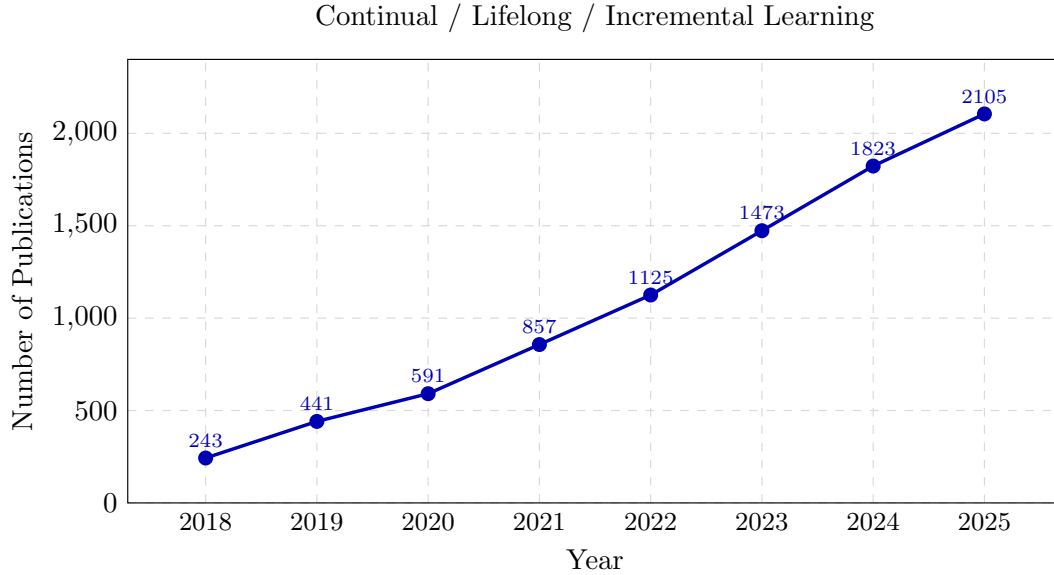


Figure 2: Annual publication trends for the keywords *continual learning*, *lifelong learning*, and *incremental learning* in DBLP

Moreover, continual learning simplifies model maintenance; models can keep learning without being fully retrained and redeployed each time. Instead of starting from scratch with each new dataset, continual learning models build upon existing knowledge, resulting in more efficient use of computational resources and reduced operational costs. Additionally, continual learning enables models to retain knowledge across different tasks and domains, improving transfer learning capabilities. Because these concepts emerge gradually across an entire lifetime, the literature often uses continual learning interchangeably with incremental learning or lifelong learning, without drawing a strict distinction between them [20, 67]. Figure 2 illustrates a consistent increase in the number of research publications on *continual*, *lifelong*, and *incremental learning* from 2018 to 2025 in DBLP¹. The growth from 243 papers in 2018 to 2,105 in 2025 demonstrates the rapidly rising academic and practical interest in adaptive learning systems, with the field nearly tripling in publication output between 2022 and 2025 alone. This steady, uninterrupted upward trend indicates that the research momentum has not slowed, reflecting sustained investment in solving the catastrophic forgetting problem across both academic and industrial settings. Overall, the trend suggests that continual learning remains one of the most active and rapidly evolving areas in machine learning research, with no sign of saturation as of 2025.

The research landscape of continual learning has evolved significantly, driven by

¹<https://dblp.org/>

several emerging trends. The proliferation of large-scale pre-trained models has sparked interest in parameter-efficient methods such as LoRA [48] and adapters [46], enabling efficient adaptation without catastrophic forgetting. Privacy-preserving approaches, including federated continual learning [136] and synthetic data generation [116], address data security concerns in sensitive domains. The field has expanded into multimodal learning scenarios [122] and neuroscience-inspired architectures [137] that mimic biological learning processes. The community increasingly emphasizes realistic evaluation paradigms with comprehensive metrics beyond accuracy [24, 76]. Despite these advances, critical challenges remain in selective knowledge transfer, computational efficiency, and the scalability-stability trade-off as models grow larger and tasks become more complex.

This dissertation addresses these contemporary challenges by proposing three complementary approaches: First, memory-efficient prototype methods that approximate data distributions without storing actual samples, addressing privacy concerns and memory constraints. Second, selective knowledge distillation approaches that identify and transfer only relevant knowledge from pre-trained and previously learned models, reducing computational overhead while improving task performance. Third, adaptive architectural strategies with dynamic parameter allocation efficiently manage model capacity across sequential tasks while preserving knowledge. These contributions directly respond to the identified gaps in current continual learning research and provide practical solutions for deploying adaptive AI systems in real-world scenarios.

In summary, continual learning confers upon machine learning systems the capacity for sustained adaptation, ensuring that deployed models remain accurate and dependable as the data landscape evolves. The research presented in this dissertation contributes to this evolving field by developing novel methods that balance knowledge retention, computational efficiency, and adaptation capabilities across diverse application domains.

Research Gaps and the Progressive Logic of This Dissertation

This dissertation focuses on three methodological categories [122]: (1) **regularization-based approaches**, which constrain updates to network parameters critical to prior tasks [5, 63, 137]; (2) **architecture-based methods**, which expand or reconfigure network structure for new knowledge [2, 79, 102]; and (3) **replay-based techniques**, which retain and revisit prior task examples during subsequent learning [17, 77, 98, 109].

Replay-based methods remain the de facto starting point for continual learning research: a small buffer of past-task exemplars is retained and replayed jointly with new task data at training time [12, 15]. This is intuitive and effective, but it is not always viable. In sensitive domains such as healthcare, finance, or personal communication, retaining raw data from previous tasks raises serious privacy concerns [6]. Moreover, memory requirements scale linearly with the number of tasks encountered, posing practical constraints in long-sequence settings [13]. These limitations motivate the first question: *can we achieve the benefits of replay without storing real data?*

This dissertation answers that question in Chapter 2 by proposing prototype-based memory augmentation: instead of retaining actual samples, the model approximates past data distributions through class prototypes perturbed with Gaussian noise. This approach preserves privacy and reduces memory overhead while maintaining competitive performance in continual relation extraction tasks. An energy-based latent alignment mechanism is further introduced to counteract representational drift across tasks. Together, these methods establish a privacy-preserving foundation for continual learning. However, prototype-based replay is not a complete solution. Generating informative pseudo-samples becomes increasingly difficult as the feature space grows more complex, and any approximation introduces some distribution mismatch. Furthermore, in scenarios where tasks are highly heterogeneous or the model is deployed with strict computational budgets, maintaining even a compact memory buffer may be impractical. These observations lead to the second gap: if memory is not available, can the model transfer and consolidate knowledge without accessing any stored examples?

Chapter 3 addresses this gap through knowledge transfer mechanisms: specifically, contrastive learning combined with knowledge distillation, and an analytic learning approach that reformulates continual learning as a closed-form least-squares problem. Rather than storing data, these methods exploit the knowledge already encoded in previously trained models and large-scale pre-trained representations. The CL-plugin framework with Contrastive Ensemble Distillation and Contrastive Supervised Losses enables selective knowledge transfer in task-incremental sentiment classification. The ZeroRFF method leverages random Fourier features over a frozen pre-trained backbone to achieve exemplar-free class-incremental learning with analytical guarantees. These contributions demonstrate that strategic knowledge reuse from pre-trained models can partially substitute for explicit data replay. Yet even exemplar-free knowledge transfer has boundaries. When tasks become sufficiently diverse, or when the number of sequential tasks grows large, a fixed model capacity becomes a bottleneck: the model either overfits new tasks

(losing stability) or fails to adapt fully (losing plasticity). The knowledge distillation objective is inherently conservative, it suppresses the very parameter changes needed for plasticity. This reveals the third and deepest gap: how can a model dynamically allocate its own capacity to accommodate new knowledge without sacrificing what it has learned?

Chapter 4 addresses this gap through adaptive architecture strategies with dynamic parameter allocation. Rather than constraining a fixed parameter set, these methods selectively expand or reconfigure the model’s effective capacity per task. The multi-head framework with Online Prototype Equilibrium and Adaptive Prototypical Feedback separates within-task and task-identity prediction, preventing inter-task class confusion. DAKTA integrates Low-Rank Adaptation with a Kolmogorov-Arnold classifier and Fisher-regularized task arithmetic, enabling flexible classifier expressiveness while protecting critical parameters. SO-LoRA reconceptualizes adaptation geometrically: by fixing a universal orthogonal basis and restricting each task to sparse compositions over this basis, it achieves constant parameter overhead while bounding task interference through group sparsity and gradient-aware masking. Taken together, these three chapters form a coherent and progressive research programme. Memory-based methods provide strong performance when data retention is feasible. Knowledge transfer methods extend continual learning to privacy-sensitive or memory-constrained settings. Adaptive architecture methods address the fundamental capacity limitations that constrain both of the preceding families. Each approach is not a replacement for the others; it is a response to the limitations that arise when the previous approach is pushed to its boundary.

Research Questions and Objectives

Research Questions

Drawing from the identified research challenges and the progressive logic described above, this dissertation seeks to address three research questions, illustrated in Table 2.

Objectives

Based on these research questions, this dissertation pursues three corresponding objectives:

Objective 1: To develop and evaluate novel memory-based continual learning frameworks that integrate prototype augmentation and energy-based latent alignment, achieving a significant reduction in catastrophic forgetting without storing actual training

Table 2: Summary of research gaps and the dissertation’s progressive research logic

Limitation of existing approach	Research gap	Research question	Ch.
Replay retains raw data → privacy risk, linear memory growth	Achieve replay benefits without storing real data	Q1. How can prototype augmentation and synthetic data generation mitigate catastrophic forgetting while preserving privacy and efficiency?	2
Exemplar-free transfer bounded by fixed backbone capacity	Transfer knowledge without stored examples	Q2. How can selective knowledge transfer via adaptive distillation and contrastive learning reduce redundancy while preserving task-relevant features?	3
Fixed parameter capacity limits long, diverse task sequences	Dynamically allocate capacity without sacrificing prior knowledge	Q3. How can dynamic capacity allocation preserve prior knowledge while enabling effective learning across long, heterogeneous task sequences?	4

samples.

Objective 2: To investigate and implement advanced knowledge transfer mechanisms, including contrastive distillation and analytic learning — to enable exemplar-free continual learning with measurable improvements in forward transfer and knowledge retention.

Objective 3: To develop and validate adaptive architecture strategies with dynamic parameter allocation that effectively balance stability, plasticity, and computational efficiency, demonstrating state-of-the-art performance across diverse class-incremental learning benchmarks.

Research Methodologies

This dissertation adopts a mixed-methods research approach combining qualitative and quantitative methodologies:

- Qualitative analysis involves: (i) a critical review of state-of-the-art continual learning methods and their underlying principles; (ii) the systematic identification

of existing limitations and research opportunities; (iii) the synthesis and development of novel approaches that build upon and extend current techniques.

- Quantitative evaluation includes: (i) Curation and preprocessing of benchmark datasets; (ii) systematic experimental design and implementation; (iii) performance assessment and comparative analysis of proposed frameworks; and (iv) peer-reviewed publication for community validation and reproducibility.

Main Contribution

This dissertation makes the following scientific contributions, organized according to the three-stage logic described above:

- **Privacy-preserving replay (Chapter 2).** We propose a prototype augmentation mechanism that generates Gaussian-perturbed pseudo-samples around class prototypes, eliminating the need to store real data while preserving distributional characteristics. An energy-based latent aligner (ELI) is introduced to counteract representational drift in the feature space, ensuring that previously learned class boundaries remain stable as new tasks are incorporated. These contributions are validated on continual relation extraction benchmarks (FewRel and TACRED) and published in [P1].
- **Exemplar-free knowledge transfer (Chapter 3).** We introduce two complementary methods. First, a Continual Learning Plugin augmented with Contrastive Ensemble Distillation (CED) and Contrastive Supervised Classification (CSC) losses achieves selective knowledge transfer in task-incremental aspect sentiment classification, outperforming state-of-the-art models without requiring exemplar storage [P2]. Second, ZeroRFF combines random Fourier features with a frozen pre-trained backbone and analytic learning to compute closed-form optimal solutions for class-incremental learning, achieving competitive performance under the challenging Si-Blurry scenario [P3].
- **Adaptive architecture with dynamic parameter allocation (Chapter 4).** We develop four methods that progressively address the capacity limitations of fixed-parameter approaches. (i) A multi-head framework with Online Prototype Equilibrium and Adaptive Prototypical Feedback disentangles within-task and task-identity prediction [P4]. (ii) Integration of ELI into this framework prevents latent space drift in medical image classification [P5]. (iii) DAKTA combines LoRA-adapted

Vision Transformers with a Kolmogorov-Arnold classifier and Fisher-regularized task arithmetic for fine-grained class-incremental learning [P6]. (iv) SO-LoRA enforces subspace separation through a fixed orthogonal basis and group sparsity, achieving constant parameter overhead with provable interference bounds across long task sequences [P7].

For each of the proposed models, the dissertation conducts a series of experiments and compares the results with those of the baseline continual learning models. Through experimental findings, the dissertation highlights the impact of the proposed model components in comparison to those of the original models. This PhD dissertation contributes to both the scientific and practical areas.

Dissertation Structure

This dissertation is organized into four main chapters following the progressive logic described above, as illustrated in 3. Preamble introduces the research context, establishing the motivations behind this work, defining the scope of the investigation, and articulating the research questions that guide this dissertation. The related publications are marked with their corresponding Chapter.

Chapter 1: THEORETICAL BACKGROUND AND PRELIMINARIES provides the theoretical background, defines key terminology, characterizes the eight principal continual learning scenarios, reviews the five major methodological families, and establishes the evaluation metrics used throughout.

Chapter 2: MEMORY-BASED REPLAY METHODS FOR CONTINUAL RELATION EXTRACTION addresses the privacy limitations of conventional replay by developing prototype-based memory augmentation and energy-based latent alignment for continual relation extraction.

Chapter 3: KNOWLEDGE TRANSFER IN PRESERVING PREVIOUS INFORMATION IN CONTINUAL LEARNING advances beyond memory-dependent methods by investigating exemplar-free knowledge transfer through contrastive distillation and analytic learning.

Chapter 4: ADAPTIVE ARCHITECTURES AND DYNAMIC PARAMETER ALLOCATIONS resolves the capacity constraints of fixed architectures by introducing four dynamic parameter allocation strategies that balance stability, plasticity, and computational efficiency. The dissertation concludes with a synthesis of findings, a unified

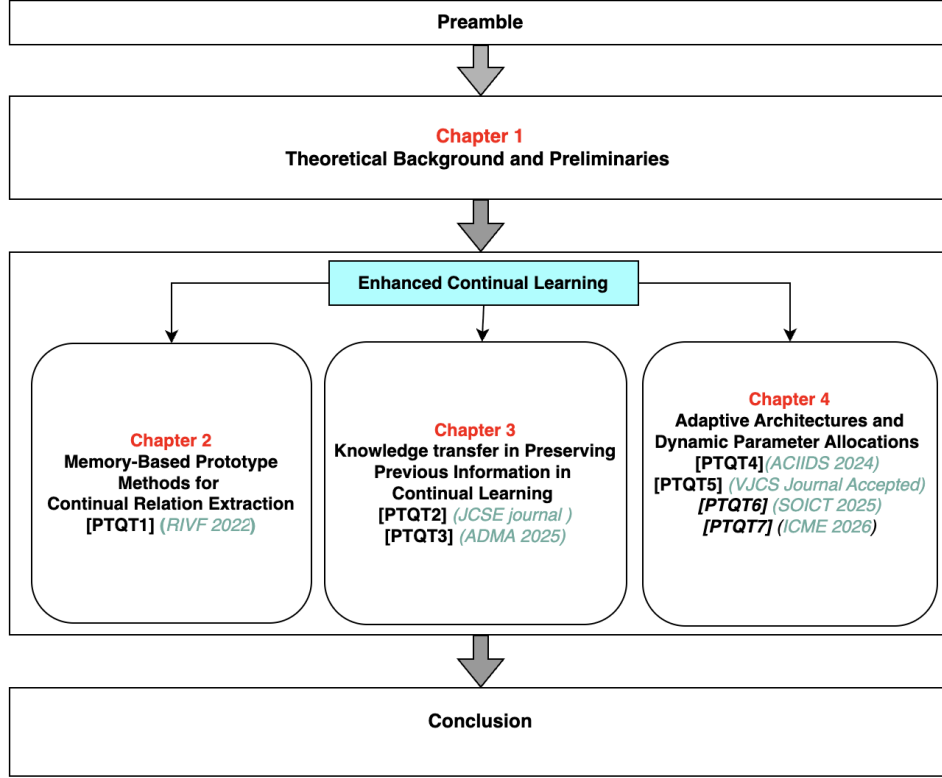


Figure 3: The Dissertation outline.

The related publications are listed in their corresponding Chapter.

comparison across contributions, and directions for future research.

Figure 4 summarizes the dissertation’s progression as a sequence of increasingly constrained solutions to the stability-plasticity dilemma. Catastrophic forgetting arises because gradient updates for new tasks overwrite parameters that encode previously acquired knowledge, and each stage of this work relaxes a different assumption about what resources are available to counteract that overwriting. Stage 1 assumes that some form of memory retention is permissible, provided raw samples are not stored, and addresses this via synthetic replay: Gaussian-perturbed prototypes substitute for exemplars, while ELI corrects the resulting drift in latent feature space through energy-based Langevin dynamics, validated on continual relation extraction benchmarks such as FewRel and TACRED. However, because pseudo-samples only approximate the true past distribution, and some deployment settings forbid any buffer at all, Stage 2 removes the memory assumption entirely and instead transfers knowledge through model-level mechanisms, contrastive distillation objectives (CED, CSC) and closed-form recursive least squares on frozen random features (ZeroRFF), generalizing the approach across both NLP and vision continual-learning settings. This shift introduces its own limitation: distillation regularizes away exactly the parameter changes that plasticity requires, so capacity eventually

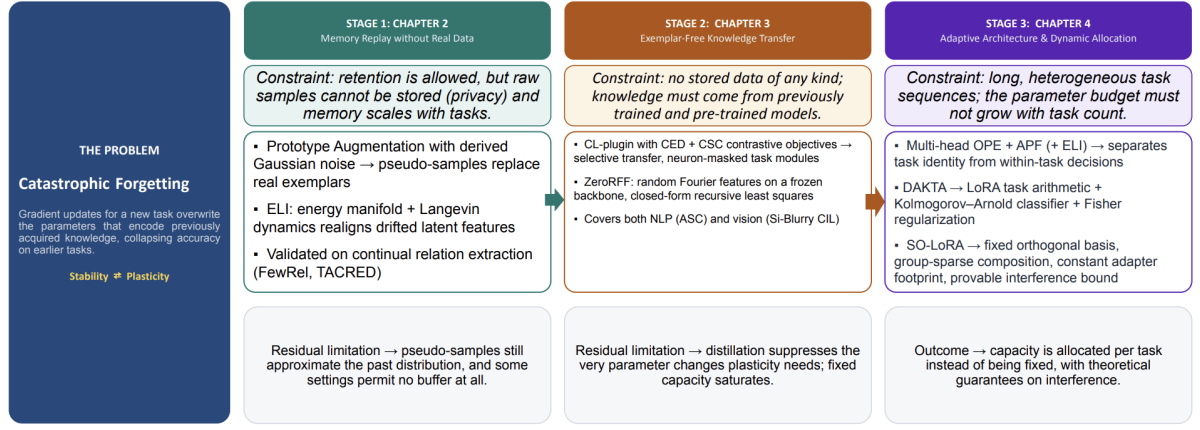


Figure 4: Overview of the dissertation’s three-stage approach to catastrophic forgetting: (1) memory replay without storing raw data, (2) exemplar-free knowledge transfer, and (3) adaptive architecture with dynamic capacity allocation. Each stage addresses a progressively stricter constraint, with residual limitations motivating the next.

saturates on long task sequences. Stage 3 therefore targets this residual problem directly by making architectural capacity adaptive rather than fixed: separating task identity from within-task decisions (OPE/APF), composing task-specific knowledge via LoRA-based arithmetic and Kolmogorov-Arnold classification (DAKTA), and constraining low-rank updates to an orthogonal, group-sparse basis with a provable interference bound (SO-LoRA). Read together, the three stages illustrate a coherent narrative arc: as constraints on memory and compute tighten, the burden of preventing forgetting shifts from data-level replay, to distillation-based transfer, to structural adaptation of the model itself.

The dissertation concludes with a summary of the key findings, theoretical and practical implications of the research, and suggestions for future work in this rapidly evolving field.

Chapter 1

Theoretical Background and Preliminaries

1.1 Continual Learning Fundamentals

Continual learning, variously termed lifelong learning or incremental learning in the literature, constitutes a machine learning paradigm in which a single unified model is trained sequentially across a series of distinct tasks. In the context of deep learning [20], it is a paradigm in which learning forms the foundation for intelligent systems to adapt to changing environments. Biological intelligence, shaped by evolutionary pressures, exhibits a remarkable capacity to progressively accumulate, revise, retain, and deploy knowledge as environmental conditions shift [37, 92]. Replicating this biological adaptability in artificial systems represents a central aspiration of modern AI research. The field of continual learning is motivated precisely by this goal: enabling neural networks to assimilate new knowledge sequentially while behaving as though all training data had been observed jointly. As a subfield of machine learning, CL specifically addresses the non-i.i.d. setting in which training distributions evolve over time. Its core aim is to develop algorithms that accumulate task-specific knowledge incrementally, allowing models to continuously adapt without performance degradation on earlier tasks [71]. Such content may include new skills, new instances of existing abilities, different environments, or various contexts, each presenting specific realistic challenges [92, 117]. As the content is introduced incrementally over time, the terms incremental learning and lifelong learning are frequently used interchangeably with continual learning across the research community, with no universally agreed distinction

[20, 92]. Catastrophic forgetting poses a fundamental challenge in sequential learning: when a model is updated on a new task, gradient-based parameter updates progressively overwrite the configurations that encode previously acquired knowledge, causing severe performance degradation on earlier tasks.

Continual learning has recently attracted many researchers. Timothée Lesort [71] focused on studying memory replay methods to address the problem of catastrophic forgetting, while Fei Mi et al. [85] address this problem without storing and replaying old data. The main approaches include using auxiliary data (such as unlabeled and synthetic data) and leveraging strongly pre-trained models with continuous editing techniques. The ultimate goal is to demonstrate that models can learn from changing data and adapt to new tasks without storing old data, thus unlocking the potential for lifelong machine learning solutions in real-world applications. Juan Manuel CORIA et al. [21] propose new methodologies for continually updating models in low-resource and dynamic production environments, where data is scarce and labels are unavailable, while Ngoc Quan Pham [94] focuses on advancing multilingual automatic speech recognition (ASR) systems, particularly in the context of continual learning. Specifically, it explores how to design end-to-end neural networks capable of learning to recognize multiple languages simultaneously and continually incorporate new languages without degrading the performance of previously learned languages. Richard Kurlle [68] combines the Bayesian probabilistic framework with deep neural network models to address challenges in continual learning, multi-source inference, and time series forecasting.

1.1.1 Definitions of Lifelong Learning

The earliest widely cited formulation of lifelong machine learning (LML) was put forward by Thrun [114], and can be stated as follows.

Definition 1 [114]. Given a system that has already completed N tasks, when it encounters task $(N + 1)^{\text{th}}$, it draws on the knowledge accumulated from the preceding N tasks to support learning of this new task.

An implicit but crucial assumption in this formulation is that, by the time the $(N + 1)^{\text{th}}$ task is being learned, the raw data from the earlier N tasks is typically no longer accessible. In other words, the system is not expected to retain and store every previously observed sample indefinitely. While this early definition conveys the core intuition behind continual learning, it leaves the notions of “task” and “knowledge” loosely specified.

A more refined formulation was later proposed by Chen and Liu.

Definition 2 [20]. Lifelong machine learning is conceived as an ongoing learning process in which, at any given moment, the learner has already worked through a sequence of N tasks, $T_1, T_2, \dots, T_N$, which may share the same type or domain or differ from one another. Upon encountering a new task T_{N+1} along with its associated data D_{N+1} , the learner draws on previously accumulated knowledge stored in a knowledge base (KB) to facilitate learning of T_{N+1} . While the primary goal of LML is typically to improve performance on the current task T_{N+1} , the framework is flexible enough that any task can become the optimization target, with the remaining tasks treated as prior experience. The KB serves as a repository that accumulates and retains knowledge derived from previously learned tasks; once T_{N+1} has been learned, the KB is refreshed with newly acquired knowledge, including intermediate outcomes and final results, obtained from this task. This refresh process may further involve checking for inconsistencies, performing inference, and mining higher-level knowledge from the accumulated information.

According to its proponents, this second definition can be distilled into three core properties: an ongoing, uninterrupted learning process; the accumulation and upkeep of knowledge within a dedicated knowledge base; and the capacity to apply previously acquired knowledge to facilitate learning of subsequent tasks. Relative to Thrun’s earlier formulation [115], the key addition here is the explicit role of a persistently maintained knowledge base. Under this view, LML performance can, in principle, be assessed and optimized for any task at any point in time, by treating all remaining tasks as prior knowledge regardless of the chronological order in which they were originally encountered. Furthermore, whereas Thrun’s original conception focused solely on improving performance for the newest task T_{N+1} , implicitly tolerating degradation in performance on earlier tasks, Chen and Liu [20] instead treat the preservation of all previously accumulated knowledge as a core objective of LML in its own right. A subsequent revision of this definition further extended it with two additional requirements: the capacity to autonomously identify new tasks, and the ability to learn continuously while the system is actively in operation.

1.1.2 Continual Learning Definition

In recent times, the concept of ongoing education throughout one’s life, known as lifelong learning, has garnered significant interest among deep learning researchers. In this field, it is frequently referred to as continual learning [20]. In the continual learning setting, a model parameterised by θ encounters training data from shifting distributions one task at a time, with restricted or no access to previously observed samples, yet must retain high

generalisation performance across all encountered task distributions.

Formally, we denote a training batch for task t as $D_{t,b} = \{X_{t,b}, Y_{t,b}\}$, where $X_{t,b}$ and $Y_{t,b}$ denote the input features and their corresponding labels, respectively. Here, $t \in T = \{1, \dots, k\}$ indexes the task and $b \in B_t$ indexes the batch within task t (T and B_t denote the corresponding index sets). We define a “task” as a set of training samples D_t drawn from a distribution $D_t := p(X_t, Y_t)$ (where D_t , X_t , and Y_t denote the full training set, omitting the batch index), and we assume no distribution shift between training and testing. Under realistic constraints, the labels Y_t and the task identity t may not always be available at inference time.

In continual learning, a **single model** f_θ , parameterized by θ , is trained sequentially across a sequence of tasks. The training data for each task may arrive either incrementally in batches, i.e., $\{\{D_{t,b}\}_{b \in B_t}\}_{t \in T}$, or all at once, i.e., $\{D_t\}_{t \in T}$. The core objective is to optimize θ across the task sequence T so that f_θ performs well on all tasks, without retraining from scratch.

1.1.3 Mitigating Catastrophic Forgetting in Continual Learning: Objectives and Challenges

Continual learning pursues several interconnected aims that address fundamental challenges in both artificial and biological learning systems. The foremost objective is to mitigate catastrophic forgetting, ensuring that knowledge acquired from previous tasks remains intact when learning new information. Closely related is the goal of transferring knowledge across tasks, enabling systems to leverage previously learned concepts to accelerate and improve performance on new challenges. However, continual learning must also manage task interference, where learning one task negatively impacts performance on other tasks, requiring a careful balance between knowledge sharing and task-specific specialization. These aims are further complicated by limited memory capacity, which necessitates efficient storage strategies that retain essential information while discarding redundant or less important details. Additionally, continual learning systems must handle distribution drift, adapting to gradual or sudden changes in data patterns over time without destabilizing existing knowledge. Finally, achieving sample efficiency is crucial, as continual learners should be able to acquire new skills from limited examples rather than requiring extensive retraining, mimicking the human ability to learn quickly from a few experiences. Together, these aims define the core challenges and objectives that drive research and development in continual learning, seeking to create more adaptive,

efficient, and robust learning systems.

Catastrophic forgetting problem [20]

Catastrophic forgetting is a central obstacle in continual learning: as a neural network adapts to a new task (e.g., task B), its performance on previously learned tasks (e.g., task A) often deteriorates sharply [20]. The root cause lies in gradient-based parameter updates: in optimising the network for the current task, they systematically overwrite weight configurations essential to prior tasks. This problem is particularly pronounced in sequential learning environments where multiple tasks are presented one after another. Researchers have extensively studied catastrophic forgetting [32, 81, 82, 97], have highlighted its significant impact on the performance and adaptability of neural networks in continual learning settings. This challenge is commonly characterised as the stability–plasticity trade-off, as described by [84]. A too-stable model resists incorporating new information from future training data. Conversely, a highly plastic model is prone to significant weight adjustments, leading to the loss of previously learned representations. It bears noting that catastrophic forgetting is not exclusive to modern deep architectures, it equally afflicts shallower networks such as traditional multi-layer perceptrons. Even simpler models like single-layered self-organizing feature maps have been shown to suffer from catastrophic interference, as demonstrated by [99]. This widespread issue underscores the critical need for effective solutions to enable true continual learning in neural network architectures.

Figure 1.1 demonstrates catastrophic forgetting observed with Adam in a streaming learning scenario. In this setup, the learner is exposed to a sequence of tasks derived from Label-Permuted EMNIST. These tasks are intentionally structured to exhibit high coherence, ensuring that features acquired in one task can be fully utilized in others. Ideally, if the learner can retain and utilize prior knowledge effectively, their performance should progressively improve with the introduction of more tasks. However, as depicted in Figure 1.1, Adam struggles to significantly enhance its performance, maintaining a consistently low level of accuracy. This stagnation suggests a cycle of forgetting and relearning. Notably, while catastrophic forgetting is conventionally examined in offline evaluations (solid lines), the issue persists even in online evaluations (dashed lines). This finding underscores the limitations of current representation learning methods, which fail to capitalize on previously acquired valuable features and instead exhibit a tendency to forget and relearn them when confronted with subsequent tasks.

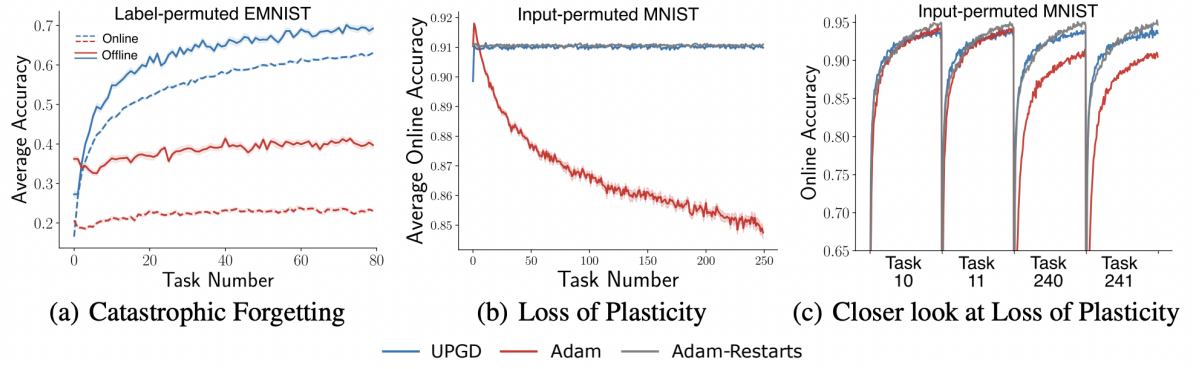


Figure 1.1: (a) Adam experiences significant degradation in performance due to catastrophic forgetting. (b & c) As newer tasks are introduced, Adam exhibits decreased plasticity and performs notably worse compared to instances where Adam is restarted later.[106].

To mitigate catastrophic forgetting, a natural approach is to minimize the reconfiguration of models’ parameters as much as possible. To this end, various CL models have been proposed, of which each task has its own specific module (or subset of the model’s parameters) to be optimized. When learning a new task, only the module specific to this task is tuned while others remain untouched. These modules can be task-specific adapters [46, 59] or continual learning plugins (CL-plugin) [56] or a relatively small 2-layer fully connected network (so-called a Capsule Networks) [44, 104], which are injected into the pre-trained BERT.

The isolation of task-specific modules helps CL models mitigate catastrophic forgetting. However, it hinders (or even prohibits) the shared knowledge utilization (including updates) among tasks, which can be useful for enhancing the model’s performance on related tasks. Knowledge utilization enables CL models to leverage insights acquired from past learned tasks in one domain and apply them effectively to new tasks in other domains. Without effective knowledge exploitation mechanisms, CL models may struggle to adapt to the nuances of different domains, resulting in reduced performance and reliability. This contradiction causes CL hard to achieve both objectives together, namely, preventing catastrophic forgetting and facilitating knowledge transfer. There are still two significant challenges for continual learning.

Other Challenges

Knowledge transfer [9] Knowledge transfer occupies a central position in continual learning research, serving as a key mechanism for overcoming the inherent difficulties of sequential task acquisition. Continual learning, while promising in its capacity to enable models to evolve and improve over time, encounters significant hurdles due to the ever-changing nature of data streams and the necessity to retain previously acquired knowledge while accommodating new information. Among these challenges, knowledge transfer emerges as a potent strategy to alleviate the difficulties associated with continual learning. In the field of continual learning, understanding the dynamics of knowledge transfer is critical to developing robust and adaptive models [91]. This involves addressing three fundamental questions: What knowledge should be transferred, when is the optimal time to transfer it, and how should the transfer be effectively implemented? By exploring these aspects, we can enhance the efficiency and effectiveness of continual learning systems.

What to Transfer: In continual learning, determining what knowledge to transfer is crucial, as it can vary significantly. Some knowledge is domain-specific, tailored to a particular task or environment, and may not be useful beyond its original context. Conversely, more general or transferable knowledge, such as fundamental skills or abstract concepts, can be beneficial for multiple tasks. Identifying and leveraging this transferable knowledge allows models to build on previous experiences, enhancing their ability to adapt to new challenges efficiently.

When to Transfer: The timing of knowledge transfer in continual learning is a delicate balance. Transfer can be beneficial when there is significant overlap or relevance between tasks, enabling the model to apply previously acquired insights to new situations. However, premature or inappropriate transfer can lead to negative transfer, where the model's performance on a new task is hindered by irrelevant or conflicting information from prior tasks. Thus, it is essential to develop strategies that assess the suitability of transfer based on the characteristics of the tasks involved.

How to Transfer: Implementing knowledge transfer in continual learning involves developing algorithms capable of integrating and combining information from previous tasks with new data. Methods such as transfer learning, where pre-trained models are fine-tuned on new tasks, and meta-learning, which aims to optimize learning processes across multiple tasks, are prominent approaches. In addition, techniques such as regular-

ization and modular networks can help manage and preserve valuable knowledge while minimizing interference between tasks. These algorithms ensure that the learning process remains robust and efficient as the model encounters new experiences.

Task interference [38]: Sequential task training inevitably introduces inter-task interference: the representations optimised for a given task can disrupt the acquisition of knowledge for subsequent tasks, thereby degrading the model’s cross-task generalisation capability.

Limited memory capacity [71]: Continual learning models often have limited memory capacity, making it challenging to retain a large number of tasks or experiences without experiencing catastrophic forgetting. Balancing the retention of old knowledge with acquiring new knowledge is a delicate trade-off, especially in resource-constrained environments.

Limited distribution drift [71]: Continual learning models must contend with concept drift and non-stationarity in the data distribution, where the underlying relationships between input features and target outcomes evolve. Adapting to these changes while preserving past knowledge poses a significant challenge, mainly when the model’s assumptions about the data distribution no longer hold.

Sample efficiency [107]: Continual learning models often require a large number of samples to learn new tasks effectively, especially when the tasks are diverse or complex. Balancing the exploration of new tasks with the exploitation of previously learned knowledge to maximize sample efficiency is a critical challenge in continual learning.

1.2 Typical Scenario and Approaches

1.2.1 Typical Scenario

Depending on how the incoming data streams are partitioned and whether task-identity information is provided, we identify the following representative continual learning settings (see Table 1.1 for a detailed comparison) [122]:

- **Instance-Incremental Learning (IIL) [122]:** The model is trained on a single task, but training examples are presented in a streaming fashion as successive mini-batches rather than as a complete offline dataset. The model must learn from streaming instances while maintaining consistent performance across the entire data distribution. This scenario simulates real-world online learning where data

accumulates gradually over time without task boundaries. *For example, a spam detection system that continuously receives new emails and must update its filtering model on the fly, without retraining from scratch on the full historical email corpus.*

- **Domain-Incremental Learning (DIL) [122]:** Tasks share identical label spaces (same set of classes) but exhibit different input distributions due to domain shifts such as changes in image style, lighting conditions, or sensor characteristics. Task identities are neither provided during training nor required during inference, as the model must learn domain-invariant representations that generalize across all encountered domains. *For example, a traffic sign classifier trained first on clear daytime images, then on rainy conditions, and finally on nighttime scenes, the set of sign categories remains fixed, but the visual appearance shifts substantially across domains.*
- **Task-Incremental Learning (TIL) [122]:** Tasks have completely disjoint label spaces, meaning each task introduces a distinct set of classes with no overlap. Task identities are explicitly provided during both training and testing phases, allowing the model to select the appropriate task-specific output head for prediction. This represents the simplest continual learning scenario, as task information eliminates ambiguity during inference. *For example, a multi-domain sentiment analysis system that first learns to classify reviews in the electronics domain (Task 1), then apparel (Task 2), and later restaurants (Task 3); at inference time, the domain identifier is provided so the correct classifier head is activated.*
- **Class-Incremental Learning (CIL) [122]:** Each task introduces a new set of non-overlapping classes, with task identity provided only at training time and withheld during inference. At test time, the model must discriminate among all classes seen so far without access to any task-indicator signal. This challenging scenario requires the model to maintain discriminative boundaries between all classes while preventing catastrophic forgetting of earlier tasks. *For example, an image recognition system that initially learns to distinguish cats and dogs, then incrementally acquires knowledge of birds and horses; at test time, any query image must be correctly classified among all four categories without any hint of when the model learned each class.*
- **Task-Free Continual Learning (TFCL) [122]:** While task label spaces remain disjoint, no explicit task boundary or identity information is provided at either training or inference time. The model must autonomously infer task transitions and

partition the data stream into coherent segments, acquiring task-relevant representations solely from the raw data stream. *For example, a medical imaging system deployed in a hospital where the data stream shifts silently from X-ray analysis to MRI interpretation to CT scan classification, with no explicit signal marking when or whether a domain shift has occurred.*

- **Online Continual Learning (OCL)** [122]: Tasks present disjoint label spaces with training samples arriving as a single-pass data stream, where each sample can only be observed once before being discarded. This strict constraint prevents any form of rehearsal or multiple passes over the data, demanding highly efficient learning algorithms that can extract maximum knowledge from limited exposure to each sample. *For example, an autonomous vehicle perceptron system that must learn to recognize new object categories (e.g., construction vehicles, unusual road signs) from a single encounter during deployment, as onboard storage constraints prohibit retaining raw sensor data for replay.*
- **Blurred Boundary Continual Learning (BBCL)** [122]: Task boundaries are not sharply defined but instead exhibit gradual transitions, characterized by overlapping label spaces where consecutive tasks may share some classes while introducing new ones. Additionally, samples from different tasks may arrive in an interleaved manner rather than in distinct sequential blocks, better reflecting real-world scenarios where task transitions are gradual rather than abrupt. *For example, a news topic classifier trained on a continuously evolving feed where articles about “politics” and “economy” appear throughout, while new topics such as “climate policy” or “cryptocurrency” gradually emerge and blend with existing categories, making it impossible to draw a clear boundary between old and new tasks.*
- **Continual Pre-training (CPT)** [122]: Large-scale pre-training data arrives in sequential stages rather than as a complete corpus. The objective is to continuously update the pre-trained model’s representations as new pre-training data becomes available, thereby improving knowledge transfer capabilities to downstream tasks. This scenario is particularly relevant for foundation models that must incorporate new knowledge domains without retraining from scratch. *For example, a large language model initially pre-trained on general web text is subsequently updated with domain-specific corpora, first biomedical literature, then legal documents, and later financial reports, without discarding previously acquired linguistic knowledge, so that the model progressively becomes capable of transferring richer representations*

to downstream tasks across all encountered domains.

Common Symbols:

- $\mathcal{D}_{t,b}$: Training dataset for time t and batch b
- t : Time index or training step
- $\mathcal{X}_i$: Input space for task i
- $\mathcal{Y}_i$: Output space (labels) for task i
- $p(\mathcal{X}_i)$: Probability distribution of input data for task i

Scenario	Training condition	Task label at inference
IIL	$\{\{\mathcal{D}_{t,b}, t\}_{b \in \mathcal{B}_t}\}_{t=j}$	$\{p(\mathcal{X}_t)\}_{t=j}$; no task label is needed at test time
DIL	$\{\mathcal{D}_t, t\}_{t \in \mathcal{T}}$, where $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ while $\mathcal{Y}_i = \mathcal{Y}_j$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; task identity is not needed for prediction
TIL	$\{\mathcal{D}_t, t\}_{t \in \mathcal{T}}$, where $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; the correct task label is given at inference
CIL	$\{\mathcal{D}_t, t\}_{t \in \mathcal{T}}$, where $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; the model must infer the task itself
TFCL	$\{\{\mathcal{D}_{t,b}\}_{b \in \mathcal{B}_t}\}_{t \in \mathcal{T}}$, where $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; task label may or may not be supplied
OCL	$\{\{\mathcal{D}_{t,b}\}_{b \in \mathcal{B}_t}\}_{t \in \mathcal{T}}$ with $ \mathcal{B} = 1$; $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; task label may or may not be supplied
BBCL	$\{\mathcal{D}_t, t\}_{t \in \mathcal{T}}$, where $p(\mathcal{X}_i) \neq p(\mathcal{X}_j)$, $\mathcal{Y}_i \neq \mathcal{Y}_j$, and $\mathcal{Y}_i \cap \mathcal{Y}_j \neq \emptyset$ for $i \neq j$	$\{p(\mathcal{X}_t)\}_{t \in \mathcal{T}}$; the model must infer the task itself
CPT	$\{\mathcal{D}_t^{pt}, t\}_{t \in \mathcal{T}}$ (pre-training phase), followed by fine-tuning on a downstream task j	$\{p(\mathcal{X}_t)\}_{t=j}$; no task label is needed at test time

Table 1.1: Formal comparison of common continual learning scenarios, adapted and reorganized from the taxonomy proposed in [122].

This dissertation evaluates both the Task-Incremental Learning (TIL) and Class-Incremental Learning (CIL) scenarios, as they represent two settings of increasing diffi-

culty, with and without task-identity information available at inference, and are the two most widely adopted scenarios in the continual learning literature, thereby enabling a comprehensive assessment of the proposed methods' resistance to catastrophic forgetting and their inter-class discriminative capability, while also facilitating fair comparison with prior work.

1.2.2 Approaches to Continual Learning

The continual learning literature has produced a diverse array of methods spanning multiple learning paradigms, which can be broadly organised into six methodological categories (see 1.2):

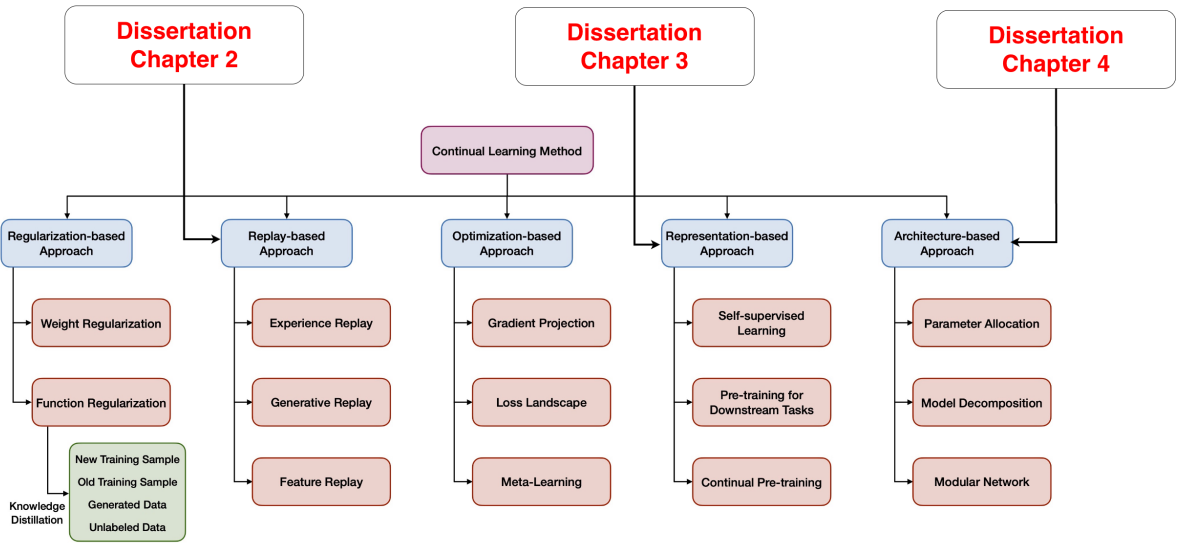


Figure 1.2: A comprehensive and advanced classification of notable continual learning methods [122].

1. **Regularization-based approach:** adding regularization terms with reference to the old model. These approaches employ a single model with a predetermined capacity, augmented by an extra component in the loss function. This additional term is designed to reinforce knowledge retention as the model encounters and learns from new tasks or data distributions.

A representative instantiation of this approach is Elastic Weight Consolidation (EWC) [63], which penalises changes to parameters deemed critical for previously learned tasks by incorporating a quadratic regularisation term into the training objective. This technique decelerates the learning process for parameters identified as crucial for previously learned tasks. The general formulation can be expressed as:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{new}}(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta_i^*)^2 \quad (1.1)$$

where $\mathcal{L}_{\text{new}}(\theta)$ denotes the loss on the new task, F_i is the Fisher information matrix that approximates the importance of each parameter, θ_i^* are the optimal parameters learned from previous tasks, and λ controls the strength of the regularization term. Through this mechanism, EWC seeks to preserve performance on prior tasks while still allowing the model to adapt to new information.

Other notable regularization-based methods include:

- **Synaptic Intelligence (SI)** [137]: Computes parameter importance online during training by tracking the contribution of each parameter to the change in loss.
- **Memory Aware Synapses (MAS)** [5]: Measures parameter importance based on sensitivity of learned function rather than loss.
- **Learning without Forgetting (LwF)**: Uses knowledge distillation from old model predictions as a form of functional regularization.

2. **Replay-based approach**: approximating and recovering the old data distributions. These approaches depend on preserving a subset of training examples from previously learned tasks, which can be revisited when addressing a current task. To address the increasing storage requirements, researchers recommend alternative strategies. These include latent replay techniques, as proposed by Pellegrini et al. (2019) [93], or pseudo-rehearsal methods, first introduced by Robins (1995) [100]. These alternative approaches aim to maintain the benefits of data revisitation while minimizing the storage overhead associated with keeping actual training examples. A notable example in this category is the iCaRL model [98], introduced by Rebuffi et al. (2017b), which has become a benchmark in incremental class learning. iCaRL maintains exemplar sets for each class using herding selection:

$$m_y = \frac{1}{|P_y|} \sum_{p \in P_y} \phi(p) \quad (1.2)$$

where m_y is the class mean, P_y is the exemplar set for class y , and $\phi(p)$ represents the feature representation.

However, this method comes with a trade-off: as the model retains samples for each task and periodically reviews them during the learning process, both computational

and memory demands escalate in proportion to the number of tasks encountered. To address these increasing storage requirements, researchers recommend alternative strategies:

- **Latent replay techniques** [93]: Store compressed representations in latent space rather than raw samples.
- **Pseudo-rehearsal methods** [100]: Generate synthetic samples using generative models to approximate previous task distributions.
- **Dark Experience Replay (DER)** [12]: Store both samples and their corresponding model outputs for more effective knowledge preservation.
- **Gradient Episodic Memory (GEM)** [77]: Uses stored examples as constraints to prevent negative backward transfer.

The memory buffer update strategy is crucial. Common approaches include:

- *Reservoir sampling*: Random selection with equal probability
- *Herding*: Select samples closest to class prototype
- *Gradient-based selection*: Retain samples with high gradient magnitude

3. **Optimization-based approach**: explicitly manipulate the optimization programs to enable continual learning. These methods focus on modifying the training procedure itself rather than the model architecture or loss function alone.

Key techniques include:

- **Gradient Projection Methods**: This family of approaches modifies the gradient update direction to prevent interference with knowledge learned on prior tasks. For instance, GEM [77] constrains the update so that performance on previous tasks does not degrade by solving the following constrained optimization problem:

$$\min_g \|g - g_t\|^2 \quad \text{s.t.} \quad \langle g, g_k \rangle \geq 0, \forall k < t \quad (1.3)$$

Here, g_t denotes the gradient computed for the current task t , and g_k denotes the gradients associated with each previous task k . Intuitively, rather than applying g_t directly, GEM searches for the gradient g closest to g_t (in the Euclidean sense) that still satisfies a non-negative inner-product constraint with every past-task gradient, ensuring the update does not push the loss on earlier tasks in an unfavorable direction.

- **Meta-Learning Approaches:** Learn initialization or learning rates that facilitate quick adaptation. Meta-learning methods, such as MAML, can be adapted for continual learning by identifying parameters that are easily fine-tuned for new tasks without forgetting.
 - **Online Continual Learning:** Methods like OSAKA [13] and OCL [64] that learn from streaming data with single-pass constraints.
 - **Uncertainty-based Methods [1]:** Use uncertainty estimation to determine which parameters to update and which to preserve.
4. **Representation-based approaches:** focus on learning robust and well-distributed representations that are more resistant to catastrophic forgetting. This approach emphasizes applying pre-trained models and learning transferable features.

Key strategies include:

- **Contrastive Learning:** Learn representations that maximize agreement between differently augmented views of the same data while minimizing agreement between different samples.
- **Self-supervised Pre-training:** Leverage large-scale pre-trained models (e.g., BERT, Vision Transformers) that have learned general representations, then adapt them for continual learning scenarios.
- **Feature Space Regularization:** Maintain consistency in the feature space rather than parameter space. This includes:
 - Distance-based regularization between old and new feature representations
 - Attention-based feature distillation
 - Prototype-based methods that maintain class prototypes in feature space
- **Representation Stability Analysis [23]:** Monitor and analyze representation drift to understand and mitigate forgetting at the feature level.

Knowledge-distillation techniques for continual learning have evolved past plain output matching: Dark Experience Replay [12] keeps soft logits next to the stored raw samples to give a richer distillation signal, and both FOSTER [119] and DER [135] grow the feature space while compressing it adaptively so that plasticity and stability stay balanced. For parameter-efficient continual learning, prompt-based methods such as L2P [125] and DualPrompt [124] leverage pre-trained Vision

Transformers by learning task-specific prompt tokens, while CLIP-based methods [140] align visual and textual modalities to improve zero-shot forward transfer. Despite these advances, two limitations remain: (i) methods relying on stored exemplars raise privacy concerns; (ii) methods using fixed-capacity parameters face saturation as task sequences grow. This dissertation addresses both limitations through the three research stages described in Chapter 3.

5. **Architecture-based approach** constructing task-adaptive parameters with a properly-designed architecture. These approaches combat forgetting by implementing modular network structure modifications and introducing task-specific parameters. This approach reconfigures the original network on the fly for a target task while keeping the original network’s parameters unchanged and shared across different tasks. This approach allows for task-specific adaptations without dramatically increasing the overall model size.

Notable methods include:

- **Progressive Neural Networks (PNN)** [102]: Add one new column of network weights per incoming task, with lateral links wiring it back to every column trained earlier:

$$h_i^{(k)} = f(W_i^{(k)} h_{i-1}^{(k)} + \sum_{j < k} U_i^{(k:j)} h_{i-1}^{(j)}) \quad (1.4)$$

where $h_i^{(k)}$ is the activation at layer i for task k , and $U_i^{(k:j)}$ represents lateral connections from task j to task k .

- **PackNet** [79]: Uses iterative pruning to free up capacity for new tasks while preserving performance on old tasks through binary masks.
- **Expert Gate** [2]: Employs a gating mechanism to dynamically select task-specific parameters or expert modules.
- **Adapter-based Methods** [46]: Insert small trainable adapter modules between frozen pre-trained layers, allowing task-specific adaptation with minimal parameter growth.
- **Multi-head Architectures**: Maintain separate classification heads for each task while sharing a common feature extractor. The challenge is preventing feature drift in the shared backbone.
- **Dynamic Network Expansion**: Selectively grow the network capacity when needed:

- Add new neurons or layers when the current capacity is insufficient
 - Use masking or sparsity to isolate task-specific parameters
 - Employ model compression techniques to manage parameter growth
- **Parameter Isolation Methods:** A closely related family of architecture-based approaches that explicitly dedicates a disjoint (or largely non-overlapping) subset of the network’s parameters to each task, rather than sharing all parameters across tasks. By construction, once a subset of parameters is assigned to task k and subsequently frozen, gradient updates for future tasks cannot alter it, which provides a strong, often near-guaranteed bound on forgetting – in contrast to regularization-based methods that only discourage large parameter shifts. Formally, given a shared backbone with parameters θ , parameter isolation methods learn a task-specific binary (or soft) mask $m^{(k)} \in \{0, 1\}^{|\theta|}$ such that the effective parameters used for task k are $\theta \odot m^{(k)}$, with masks for different tasks encouraged to have minimal overlap:

$$\theta_{\text{eff}}^{(k)} = \theta \odot m^{(k)}, \quad \text{s.t. } m^{(k)} \cap m^{(j)} \approx \emptyset \quad \forall j < k \quad (1.5)$$

Representative methods include **Hard Attention to the Task (HAT)** [105], which learns near-binary attention masks per layer via a sigmoid-with-temperature annealing scheme and uses the cumulative mask to gate gradients on already-important units; **PathNet** [31], which evolves discrete forward paths through a fixed-size network, freezing the modules on paths already used by prior tasks; **PiggyBack** [80], which learns task-specific binary masks over a single frozen pre-trained backbone, allowing many tasks to be encoded as lightweight masks rather than full weight copies; and **Supermasks in Superposition (SupSup)** [128], which finds a separate supermask per task within a fixed, randomly-initialized network and can further infer the task identity at test time from gradient information. Compared to expansion-based strategies, parameter isolation methods keep the total parameter count fixed (or nearly so), trading off long-term capacity, since the pool of freezable/available parameters is finite and mask overlap becomes unavoidable as the number of tasks grows, for stronger forgetting guarantees and, in several variants (e.g., PiggyBack, SupSup), the ability to store many tasks as compact masks on top of a single shared backbone.

This approach allows for task-specific adaptations without dramatically increasing the overall model size, though careful capacity management and knowledge sharing

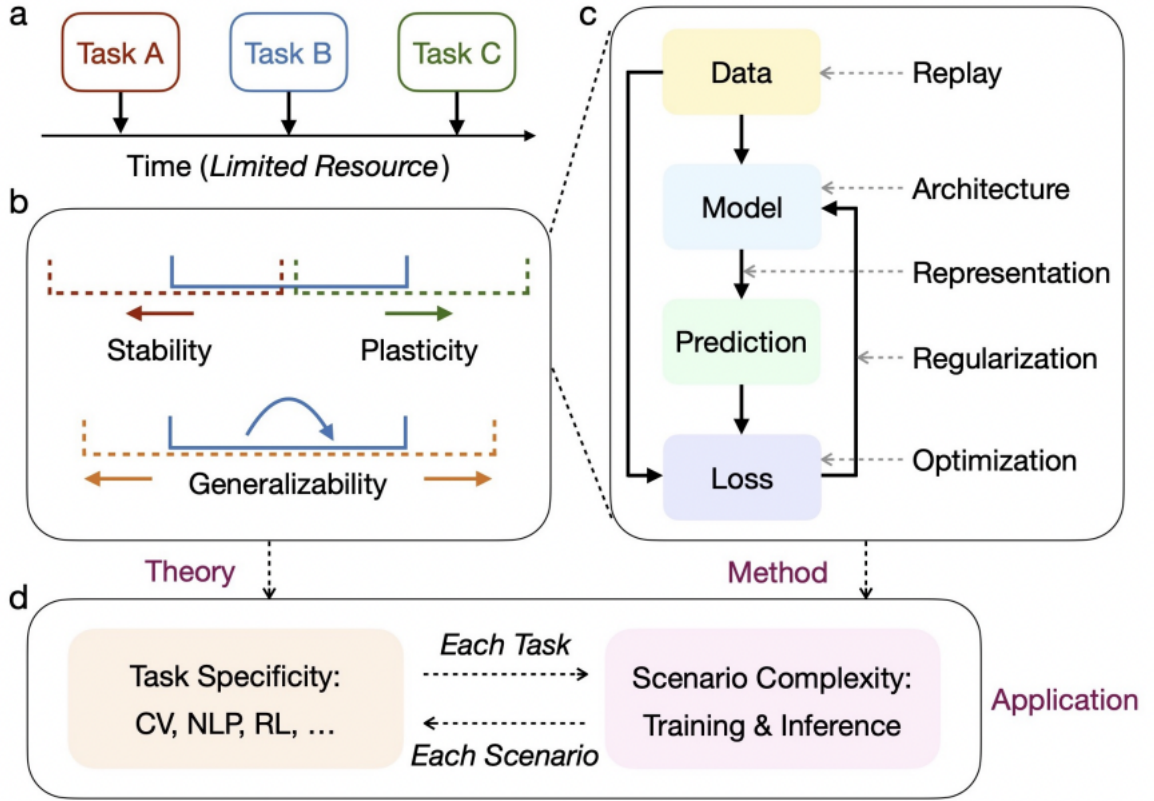


Figure 1.3: A conceptual framework for continual learning includes [122].

mechanisms are required to balance plasticity and stability.

Timothée Lesort’s dissertation [71] demonstrates that Regularization and Dynamics Architectures approaches have significant theoretical limitations in the context of continual learning. As a result, they have chosen to concentrate their efforts on exploring the applications of replay-based techniques. Specifically, they investigate Rehearsal and Generative Replay methods and their implementation in categorization tasks and reinforcement learning scenarios. This focus allows them to address the challenges of continual learning more effectively, bypassing the constraints associated with other methodologies. Fig. 1.3 shows the conceptual framework for continual learning, which includes: a) The need for continual learning to adapt to incremental tasks with changing data distributions. b) An ideal solution should balance stability (red arrow) and plasticity (green arrow) and ensure good generalization to differences within tasks (blue arrow) and between tasks (orange arrow). c) In pursuit of continual learning objectives, a range of well-established approaches has targeted distinct facets of machine learning. d) Moreover, continual learning is adapted to real-world use cases, addressing particular challenges such as the complexity of scenarios and the specificity of tasks.

Two active directions are related of this dissertation, and are noted here for completeness.

LLM-based continual learning. Recent work continually adapts large pretrained language models via instruction tuning, retrieval- augmented memory, or parameter-efficient adapters stacked over time [88]. These methods largely inherit the PEFT/LoRA and knowledge-distillation mechanisms studied in Chapters 3–4 of this dissertation, but operate at a scale (billions of parameters, open-ended generation objectives) that raises distinct engineering questions (e.g., catastrophic forgetting of instruction- following ability) not evaluated here.

Federated continual learning. This setting combines continual learning with federated optimization across multiple clients with non-IID, evolving local data streams [136]. It introduces communication-efficiency and client-drift concerns that are orthogonal to the single-model setting studied throughout this dissertation.

1.3 Related Resources

This section presents the benchmark datasets and evaluation protocols used throughout this dissertation to validate the proposed continual learning methods. We organize the datasets by application domain and describe their characteristics relevant to continual learning scenarios.

1.3.1 Datasets for Text Analysis

1.3.1.1 Relation Extraction Datasets

Table 1.2: Summary of dataset scale used throughout this dissertation

Dataset	#Samples	#Classes	#Tasks	Scenario	Domain	Used in
FewRel	56,000	80	10	TIL	NLP (RE)	Ch.2
TACRED	106,264	41	10	TIL	NLP (RE)	Ch.2
SemEval (19 tasks)	13,464	2–3	19	TIL	NLP (Sentiment)	Ch.3
CUB-200	11,788	200	10	CIL	Vision (fine-grained)	Ch.3, Ch.4
CIFAR-100	60,000	100	10	CIL	Vision	Ch.4
ImageNet-R	30,000	200	10	CIL	Vision	Ch.4

FewRel [39] is a large-scale few-shot relation classification dataset designed to evaluate models’ ability to learn new relation types with limited examples. The dataset

contains 70,000 sentences annotated with 100 relation types extracted from Wikipedia, with each relation type having 700 instances. The sentences are professionally annotated, cover diverse domains, and are suitable for continual learning scenarios where new relation types emerge over time. The dataset exhibits relatively balanced class distributions, with each relation type having an equal number of examples. This facilitates a fair evaluation across tasks in continual learning settings. This dataset is used for the task-incremental learning setting in this dissertation.

FewRel 2.0 [33] extends the original dataset by introducing two additional challenges crucial for realistic continual learning: (1) *domain adaptation*, where models must transfer knowledge across different domains (Wikipedia vs. PubMed), and (2) *none-of-the-above detection*, where models must identify instances that do not belong to any known relation type. This extension is particularly relevant for continual learning, as it simulates real-world scenarios where models encounter both domain shifts and unknown classes during sequential learning. The dataset contains 100 relation types from Wikipedia and 25 biomedical relation types from PubMed, with approximately 56,000 training instances and 15,000 test instances. This dataset is used for the task-incremental setting in this dissertation.

We settle on FewRel here on three grounds: every relation class carries exactly 700 instances, which keeps cross-task comparison free of imbalance effects; its 100 relation types are large enough to be partitioned into the 10-task protocol of [120]; and because the community treats it as the standard testbed for continual relation extraction, our numbers line up directly against CRL [139] and RP-CRE [22].

TACRED [138] is a large-scale supervised relation extraction dataset obtained through crowdsourcing, containing 106,264 examples covering 41 relation types defined in the TAC Knowledge Base Population (KBP) evaluations, plus a special `no_relation` label for negative instances. The dataset is extracted from newswire and web text from the TAC KBP corpus, with entity pairs annotated by crowdworkers. Unlike FewRel, TACRED exhibits significant class imbalance, with the `no_relation` class comprising approximately 79.5% of instances, while some relation types have fewer than 500 examples. This imbalance presents additional challenges for continual learning methods, as models must handle both catastrophic forgetting and class imbalance simultaneously. The dataset is split into 68,124 training instances, 22,631 development instances, and 15,509 test instances, following the standard TAC KBP evaluation protocol. This dataset is used for the task-incremental setting in this dissertation.

The selection of FewRel, FewRel 2.0, and TACRED is motivated by their complementary coverage of the core challenges that any continual relation extraction method must address.

We adopt FewRel as the principal benchmark in the TIL-CRE setting on the strength of three complementary properties. The class distribution is kept strictly balanced at 700 instances per relation type, so that any change in task-level performance can be attributed to the learning algorithm rather than to skewed sample sizes. Second, its 100 relation types drawn from Wikipedia provide sufficient task diversity to expose catastrophic forgetting over long task sequences, while the standard 10-task split protocol [120] enables direct and fair comparison with established CRE baselines such as CRL [139] and RP-CRE [22]. Third, because FewRel has become the de facto benchmark for few-shot and continual relation extraction, results obtained on it are immediately interpretable by the community.

FewRel 2.0 extends this evaluation in two directions that are essential for realistic continual learning. The cross-domain split (Wikipedia $\rightarrow$ PubMed) tests a model’s ability to transfer knowledge across heterogeneous distributions rather than within a single domain, directly probing the forward transfer capability of the proposed method. The none-of-the-above (NOTA) detection task additionally evaluates how a model handles novel, out-of-distribution classes during inference, simulating the open-world conditions that arise when new relation types emerge in deployed systems. Together, FewRel and FewRel 2.0 cover both the closed-world and open-world dimensions of continual relation extraction.

TACRED is included as a complementary benchmark specifically because it contrasts with FewRel in two important respects. Its severe class imbalance (the `no_relation` label comprises $\approx 79.5\%$ of instances) and its origin in real newswire text introduce distribution properties that are absent from FewRel’s controlled, Wikipedia-sourced corpus. This makes TACRED a stress test for methods that rely on prototype-based memory augmentation or contrastive replay, since the dominant negative class can overwhelm minority relation representations. Including TACRED therefore validates that the proposed approach is robust beyond balanced, curated benchmarks and generalises to more realistic extraction scenarios.

Collectively, these three datasets span the full spectrum of CRE difficulty: balanced few-shot evaluation (FewRel), cross-domain and open-world robustness (FewRel 2.0), and real-world class imbalance (TACRED). No single dataset covers all three dimensions simultaneously, which justifies the multi-benchmark evaluation strategy adopted in

1.3.1.2 Aspect Sentiment Classification Datasets

For continual learning of aspect sentiment classification tasks, this dissertation employs four benchmark datasets spanning multiple product domains for the task-incremental setting:

HL5Domains [49] comprises customer review sentences for 5 products: Digital Cameras, DVD Players, MP3 Players, Cell Phones, and Laptops. Each review sentence is annotated with aspect terms and their corresponding sentiment polarities (positive, negative, or neutral). The dataset contains approximately 7,000 annotated sentences with about 15,000 aspect-sentiment pairs, making it suitable for evaluating continual learning across related but distinct domains. The relatively small size of each domain (around 1,400 sentences per product) simulates low-resource continual learning scenarios.

Liu3Domains [75] focuses on 3 product categories: Books, Electronics, and Kitchen appliances. The dataset is extracted from Amazon product reviews and contains approximately 4,000 sentences per domain, with finer-grained aspect categories compared to HL5Domains. Each sentence may contain multiple aspects with different sentiment orientations, requiring models to perform multi-aspect sentiment analysis. This dataset is particularly challenging for continual learning due to subtle domain shifts and overlapping aspect categories across domains.

Ding9Domains [27] covers 9 diverse product categories: Bags, Boots, Keyboards, Televisions, Vacuums, Laptops, Phones, Routers, and Speakers. The dataset contains approximately 12,000 annotated sentences with over 20,000 aspect-sentiment pairs. The large number of domains and diverse product types make this dataset ideal for evaluating long-sequence continual learning, where models must retain knowledge across many sequential tasks. The domains exhibit varying levels of relatedness, allowing evaluation of both positive forward transfer (for related domains) and interference mitigation (for dissimilar domains).

SemEval14 [51] consists of customer reviews from two domains: Restaurants and Laptops, originally designed for the SemEval-2014 Task 4 on Aspect-Based Sentiment Analysis. The Restaurant domain contains 3,041 training sentences and 800 test sentences, while the Laptop domain has 3,045 training sentences and 800 test sentences. Unlike the previous datasets, SemEval14 provides explicit aspect category annotations (e.g.,

FOOD#QUALITY, SERVICE#GENERAL for restaurants) in addition to aspect terms and sentiments, enabling more structured continual learning experiments. The dataset uses a 3-class sentiment labeling scheme (positive, negative, neutral) and includes sentences with multiple aspects having different sentiments.

1.3.1.2.1 Rationale for dataset selection. HL5Domains, Liu3Domains, Ding9Domains and SemEval14 together make up a deliberately heterogeneous suite for the aspect-level sentiment task, chosen so that the principal sources of difficulty in continual sentiment learning are each represented at least once.

The primary challenge in continual ASC is domain shift: each product domain induces a distinct vocabulary, aspect distribution, and sentiment expression, so a model must retain previously acquired domain knowledge while adapting to new ones without interference. The datasets are selected to systematically vary the number and relatedness of sequential tasks. HL5Domains (5 tasks) and Liu3Domains (3 tasks) represent short-sequence continual learning, where the risk of forgetting accumulates over few transitions. Ding9Domains (9 tasks) represents the long-sequence regime, where a method must balance an increasingly larger set of prior domains; its product categories span very dissimilar domains (e.g., bags versus routers), enabling evaluation of both positive forward transfer for related domains and interference suppression for dissimilar ones. When the four datasets are concatenated into a single 19-task evaluation protocol following [57], the resulting benchmark assesses long-horizon continual learning performance across diverse domains in a unified setting.

SemEval14 is included for a qualitatively different reason: unlike the other three datasets, which consist of informal Amazon product reviews, SemEval14 is derived from a shared evaluation campaign with standardised annotation guidelines and structured aspect category labels (e.g., FOOD#QUALITY, SERVICE#GENERAL). Its inclusion ensures that the evaluation is not confined to a single annotation style and that the continual learning method generalises across both informal and structured review corpora.

Furthermore, all four datasets use a common 3-class polarity scheme (positive/negative/neutral) and are evaluated under the same Average Accuracy and Macro-F1 metrics, enabling cross-dataset comparison on equal footing. This homogeneity of the evaluation protocol, combined with the heterogeneity of the domain content, makes this benchmark suite well-suited to isolating the effect of task order and domain relatedness on continual sentiment learning.

1.3.2 Datasets for Image Analysis

For evaluating continual learning in computer vision tasks, this dissertation employs several widely-adopted benchmark datasets that span different levels of complexity and visual characteristics. These datasets are used for the class-incremental learning setting.

CIFAR-10 [65] contains 60,000 32×32 color images across 10 mutually exclusive classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is split into 50,000 training images and 10,000 test images, with 6,000 images per class. Despite its relatively low resolution and simple class taxonomy, CIFAR-10 serves as a standard baseline for continual learning research due to its manageable computational requirements and well-studied characteristics. In continual learning scenarios, the 10 classes are typically divided into 2, 5, or 10 sequential tasks to simulate class-incremental learning.

CIFAR-100 [65] extends CIFAR-10 to 100 fine-grained classes organized into 20 superclasses, with the same image resolution (32×32) and dataset size (50,000 training and 10,000 test images). Each class contains 600 images (500 for training and 100 for testing). The fine-grained classification challenge and larger number of classes make CIFAR-100 particularly suitable for evaluating catastrophic forgetting in long-sequence continual learning scenarios. The hierarchical structure (superclasses and subclasses) also enables evaluation of different task construction strategies, such as grouping related classes within tasks or maximizing inter-task dissimilarity.

TinyImageNet [69] is a subset of ImageNet ILSVRC-2012 containing 200 classes with 500 training images per class (100,000 total), 50 validation images per class (10,000 total), and 50 test images per class (10,000 total). Images are downsampled to 64×64 resolution, providing a middle ground between CIFAR and full ImageNet in terms of both visual complexity and computational cost. The 200 classes cover diverse object categories including animals, vehicles, natural objects, and man-made objects, making it suitable for evaluating continual learning methods on more realistic and complex visual distributions. The dataset is commonly split into 10, 20, or 40 sequential tasks for continual learning experiments.

ImageNet-R [10] (ImageNet-Rendition) contains 30,000 images spanning 200 ImageNet classes, but with a focus on different artistic renditions including art, cartoons, deviantart, graffiti, embroidery, graphics, origami, paintings, patterns, plastic objects, plush objects, sculptures, sketches, tattoos, toys, and video game renditions. Each of

the 200 classes contains approximately 150 images. This dataset is particularly valuable for evaluating continual learning methods’ robustness to distribution shifts and domain changes, as models must learn to recognize objects across vastly different visual styles while retaining knowledge of previously learned styles. The significant domain gaps between different rendition types make this dataset challenging for continual learning methods that rely on feature similarity assumptions.

ImageNet-A [43] (ImageNet-Adversarial) consists of 7,500 naturally occurring adversarial examples that are correctly classified by humans but misclassified by standard ImageNet-trained models. The dataset covers 200 ImageNet classes and focuses on challenging real-world images that exploit common failure modes of deep networks. While primarily designed for robustness evaluation, ImageNet-A serves as a continual learning benchmark for evaluating whether models can adapt to naturally occurring distribution shifts while maintaining performance on standard ImageNet images. The dataset’s emphasis on hard examples makes it particularly suitable for evaluating the plasticity-stability trade-off in continual learning systems.

CCH5000 [53] is a custom continual learning benchmark dataset containing 5,000 histological images (625 images per class) of human colorectal cancer across eight tissue types: tumor epithelium, simple stroma, complex stroma, immune cell conglomerates, debris/mucus, mucosal glands, adipose tissue, and background. Images are 150×150 pixels (74×74 μm) extracted from 10 H&E-stained slides, capturing multiple texture scales from individual cells (10 μm) to larger structures (>50 μm). The dataset exhibits realistic variability in staining intensity and illumination, with carefully balanced class distributions. It is specifically designed for evaluating continual learning methods in scenarios with blurry task boundaries and overlapping class distributions.

No single image dataset stresses sequence length, distribution shift and domain specificity at once, so the CIL benchmark suite here is built deliberately from several datasets, each contributing one of those three axes of difficulty. CIFAR-10 and CIFAR-100 serve as the foundational benchmarks for CIL research. CIFAR-10 and CIFAR-100 act as the entry-level benchmarks. With only 10 low-resolution classes, CIFAR-10 keeps ablations and architecture checks cheap to run, since at this scale any bottleneck comes from the method itself rather than from sheer data volume. CIFAR-100, with its 100 fine-grained classes organised into 20 superclasses, introduces the long-sequence regime (up to 20 sequential tasks), which stresses catastrophic forgetting more severely and allows evaluation of different task construction strategies, grouping semantically related

classes within tasks versus maximising inter-task dissimilarity. Critically, both datasets are fully publicly available, have fixed train/test splits, and have been used as reference benchmarks in virtually every CIL study published in the past decade [77, 98, 133], ensuring that results are directly comparable with the literature.

Sitting between CIFAR and full ImageNet, TinyImageNet’s 200 classes at 64×64 resolution let us check that the method holds up beyond toy image sizes, without paying the full computational price of ImageNet-scale training. The 200-class scope also enables evaluation under a wider range of task granularities (10-, 20-, or 40-task splits), which is important for studying how the proposed method behaves as task boundaries become finer.

CIFAR and TinyImageNet say nothing about robustness, which is why ImageNet-R and ImageNet-A are added. ImageNet-R checks whether recognition survives large style shifts, artwork, cartoons, sketches, while past styles are still remembered, probing resilience to within-class distribution change directly. ImageNet-A, composed of naturally adversarial images that fool standard classifiers, tests the plasticity side of the stability-plasticity dilemma: can the model adapt to genuinely hard examples without catastrophic interference? Together, these two benchmarks evaluate generalization properties that go beyond standard accuracy on clean images.

Finally, CCH5000 is chosen as the sole domain-specific benchmark for medical continual learning. Its histopathology images introduce a distribution that is qualitatively distinct from natural photographs, with texture-based rather than shape-based class boundaries and realistic staining variability. Its deliberately blurry task boundaries, where tissue types share overlapping visual features, test whether the proposed method can operate under the non-stationary, boundary-ambiguous conditions that occur in clinical deployment. Evaluating on CCH5000 therefore validates the practical applicability of the method beyond standard computer vision benchmarks. In summary, the benchmark suite progresses from controlled low-resolution evaluation (CIFAR-10/100, TinyImageNet) through distribution-shift robustness testing (ImageNet-R, ImageNet-A) to real-world medical imaging (CCH5000), providing a rigorous and complete characterisation of the proposed method’s capabilities and limitations across diverse class-incremental learning conditions.

1.4 Evaluation Metric

Three aspects matter when judging a continual learner: how well it does on everything learned so far, how stable that knowledge stays over time, and how readily it picks up new tasks. This section provides a summary of typical evaluation metrics, using classification as an illustration.

To evaluate overall performance, we adopt two widely used metrics: Average Accuracy (AA) [16, 77] and Average Incremental Accuracy (AIA) [29, 45]. Let $a_{k,j} \in [0, 1]$ denote the accuracy evaluated on the test set of task j immediately after the model completes training on tasks 1 through k (where $j \leq k$). Depending on the evaluation protocol, the prediction space for computing $a_{k,j}$ is restricted either to $\mathcal{Y}_j$, in the case of multi-head (task-incremental) evaluation, or to the union $\cup_{i=1}^k \mathcal{Y}_i$, in the case of single-head (class-incremental) evaluation [16]. Based on this notation, AA and AIA at task k are formulated as

$$\begin{aligned} \text{AA}_k &= \frac{1}{k} \sum_{j=1}^k a_{k,j} \\ \text{AIA}_k &= \frac{1}{k} \sum_{i=1}^k \text{AA}_i \end{aligned} \tag{1.6}$$

Here, AA_k captures the model’s performance at the current stage, while AIA_k additionally accounts for performance accumulated over all previous stages.

To assess memory stability, two complementary metrics are commonly adopted: the forgetting measure (FM) [16] and backward transfer (BWT) [77]. Regarding FM, the degree of forgetting exhibited on a given task is quantified as the gap between the highest accuracy previously attained on that task and the accuracy observed at the current stage:

$$f_{j,k} = \max_{i \in \{1, \dots, k-1\}} (a_{i,j} - a_{k,j}), \forall j < k. \tag{1.7}$$

Following the completion of the k -th task, the overall FM is then obtained by averaging this forgetting score across all tasks encountered so far:

$$\text{FM}_k = \frac{1}{k-1} \sum_{j=1}^{k-1} f_{j,k}. \tag{1.8}$$

BWT, on the other hand, captures how acquiring knowledge from the k -th task influences, on average, the model’s performance on tasks learned earlier:

$$\text{BWT}_k = \frac{1}{k-1} \sum_{j=1}^{k-1} (a_{k,j} - a_{j,j}), \tag{1.9}$$

with a negative value of BWT generally signaling the presence of catastrophic forgetting.

Learning plasticity, in contrast to stability, can instead be quantified through the intransigence measure (IM) [16] and forward transfer (FWT) [77]. IM reflects how hard a model finds it to learn a new task, defined as the difference between the accuracy reached via joint training and the accuracy reached via continual learning on that same task:

$$\text{IM}_k = a_k^* - a_{k,k} \quad (1.10)$$

Here, a_k^* is the accuracy of a reference model, initialized randomly and trained jointly on $\cup_{j=1}^k \mathcal{D}_j$, when evaluated on task k . FWT, on the other hand, measures the average degree to which knowledge from earlier tasks helps performance on the current k -th task:

$$\text{FWT}_k = \frac{1}{k-1} \sum_{j=2}^k (a_{j,j} - \tilde{a}_j), \quad (1.11)$$

where $\tilde{a}_j$ denotes the accuracy of a randomly initialized reference model trained solely on $\mathcal{D}_j$ for task j . Note that the definition of $a_{k,j}$ is not fixed and depends on the nature of the task at hand. It may, for example, correspond to average precision (AP) in object detection [110], Intersection-over-Union (IoU) in semantic segmentation [14], Fréchet Inception Distance (FID) in image generation [129], or normalized reward in reinforcement learning [1], among other possibilities. The choice of metric should therefore be tailored to the task domain being studied.

As an illustration, Alaa El Khatib [30] relies on three complementary metrics to assess model performance: average task accuracy, forgetting, and intransigence.

Final Accuracy (Acc_{last}): the classification accuracy over all classes seen so far, measured once training on the last task is complete.

Average Accuracy captures the mean accuracy across every task encountered:

$$\text{ACC} = \frac{1}{T} \sum_{j=1}^T a_{T,j}, \quad (1.12)$$

where $a_{T,j}$ denotes the accuracy on task j after the model has finished training on all T tasks.

Average Forgetting quantifies how much performance on earlier tasks degrades once training on the final task is complete, and is given by:

$$\text{AF} = \frac{1}{T-1} \sum_{j=1}^{T-1} f_{T,j}, \quad (1.13)$$

where $f_{T,j} = \max_{k \in \{1, \dots, i-1\}} a_{k,j} - a_{i,j}$.

Beyond these, some additional metrics have been proposed in the literature, including linear probing to assess forgetting at the representation level [23], the largest eigenvalue of the Hessian to characterize the flatness of the loss landscape [86], the area under the accuracy curve to capture any-time inference performance [64], and measures of storage and computational overhead to assess resource efficiency, among others.

1.5 Summary

This chapter presents the theoretical foundation for research on continual learning and outlines the dissertation’s research directions. It begins by defining fundamental concepts, including life-long learning and continual learning, along with their primary objectives. The chapter then formalizes the problem statement by presenting typical continual learning scenarios, reviewing major methodological approaches, and establishing evaluation metrics for assessing system performance.

Looking closely at the three families of methods exposes limitations that together set the research agenda for this dissertation.

Replay-based methods (Chapter 2) achieve strong performance by revisiting past data, but storing raw training examples raises data privacy concerns that are unacceptable in sensitive domains such as medical imaging and personal communication. Reducing the memory budget alleviates privacy risk but causes exemplar under-representation, leading to overfitting and performance degradation on minority classes.

Knowledge-transfer methods (Section 3) avoid data retention by distilling knowledge from frozen model snapshots. However, their effectiveness is fundamentally bounded by the fixed parameter capacity of the shared backbone: as the number of tasks grows, the shared representation must compress increasingly diverse knowledge, eventually saturating.

Architecture-based methods (Section 4) address capacity constraints by expanding or reconfiguring the network, but conventional expansion strategies incur parameter growth proportional to the number of tasks, making unsustainable in long-horizon settings and failing to exploit cross-task shared structure.

Table 1.3: Summary of contributions in this dissertation.

Method	Chapter	Paper	Task	Benchmark	Key Metric / Improvement
ProtoAug+ELI	Ch.2	[P1]	TIL-CRE	FewRel, TACRED	+1.3% / +1.6% vs. CRL
CL-plugin (CED+CSC)	Ch.3	[P2]	TIL-ASC	HL5, Liu3, Ding9, SemEval14	Avg. Acc.
ZeroRFF	Ch.3	[P3]	CIL (Si-Blurry)	CIFAR-100	AAUC +2.5% vs. GACL
OPE/APF+ELI	Ch.4	[P4,P5]	CIL-Medical	CCH5000, CIFAR-10/100	Best Fgt
DAKTA	Ch.4	[P6]	CIL	C-100,CUB,Cal.,MIT,RES.,Crop.	Best Fgt vs. ITA-LoRA
SO-LoRA	Ch.4	[P7]	CIL	ImageNet-R(20T), A, CUB-200	Best Acc.

Chapter 2

Memory-Based Replay Methods for Continual Relation Extraction

2.1 Introduction

As a core subtask of Information Extraction (IE), Relation Extraction (RE) aims to identify and classify the semantic relationships between entities within a given sentence. This task underpins a wide range of downstream NLP applications, such as structured search, sentiment analysis, question answering, and summarization [50]. However, the conventional RE models always focus on working with predefined relations, which cannot handle the rapid emergence of new relations in real life [40].

In the process of finding solutions for this problem, Wang et al. proposed an embedding approach to enable continual learning for relation extraction tasks and opened up Continual Relation Extraction (CRE), which is a new direction for the study of the relation extraction problem [120]. However, continual learning has a notable drawback: catastrophic forgetting. Catastrophic forgetting is the phenomenon that occurs when a model forgets previous knowledge after learning new information. And CRE aims to help the model learn new relations without catastrophic forgetting [139].

Recent studies have proposed solutions for alleviating catastrophic forgetting in continual Learning, mainly concentrated on regularization methods [63, 137], dynamic architecture methods [19, 31], and memory-based methods [17, 77, 139]. Memory-based

approaches, which maintain memory to save some training examples from old tasks and retrain memorized examples simultaneously while training new tasks, have been shown to achieve the best performance in addressing the catastrophic forgetting problem, especially in NLP tasks [113, 120, 139]. However, memory-based methods struggle to avoid overfitting on memory samples and perform poorly on imbalanced datasets [139]. A contrastive representation learning method is proposed by Zhao et al. to solve these challenges by using supervised contrastive learning and knowledge distillation to restrict old tasks with no dramatic change [139].

Another notable memory-based method is the Prototype Augmentation (protoAug) mechanism, proposed by Zhu et al. [141]. Prototype Augmentation is intended to preserve the distinction and balance between old and new classes and to prevent significant changes in the decision boundary, thereby addressing the challenges of catastrophic forgetting [141]. In practice, retaining samples from earlier tasks is often limited by memory capacity or privacy concerns, and the ProtoAug mechanism offers a way to mitigate this constraint.

In this chapter, we employ the **task-incremental learning scenario**, expand on the work of Zhao et al. regarding CRL, take advantage of stability, and adopt prototype augmentation in this context to improve the ability to limit the catastrophic forgetting of CRL [139]. Specifically, our proposed method employs an alternative approach that limits the amount of real data stored from previously performed tasks. Specifically, our proposed method employs an alternative approach that limits the amount of real data stored from previously performed tasks. With this approach, the model can minimize the use of real data, thereby reducing data storage constraints. We then used an energy-based model to align the features of the previous and current tasks, thereby enhancing the discriminability of each task.

2.2 Related Work

Consistent Representation Learning (CRL) [139] is a memory-based method proposed to address catastrophic forgetting in continual relation extraction, the overall architecture is illustrated in Fig. 2.1. CRL maintains stable relation embeddings across sequential tasks by combining two complementary mechanisms: supervised contrastive learning and knowledge distillation. Specifically, during the training of each new task, CRL employs supervised contrastive loss to pull together the embeddings of samples belonging to the same relation while pushing apart those of different relations, thereby encouraging the

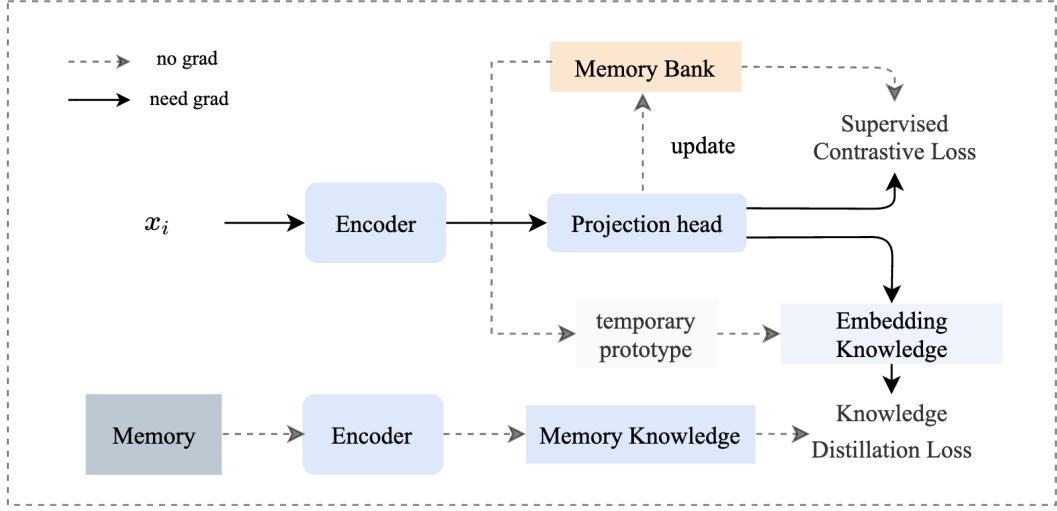


Figure 2.1: CRL overall architecture [139]

model to learn discriminative and compact representations. Subsequently, a memory replay stage is performed over stored real samples, where contrastive replay further consolidates previously learned relation embeddings. To prevent drastic shifts in the embedding space of old relations, knowledge distillation is applied on the memory bank, constraining the current model’s outputs to remain consistent with those of the previous model snapshot. Together, these components allow CRL to achieve strong performance in mitigating catastrophic forgetting, surpassing prior baselines such as RP-CRE on both FewRel and TACRED benchmarks.

However, CRL still exhibits several notable limitations. First, since CRL relies entirely on storing real data samples in the episodic memory bank, it is inherently constrained by memory capacity. It raises potential concerns regarding data privacy in sensitive domains. Second, as the number of observed tasks grows, the memory budget allocated per relation decreases, resulting in a sparse and potentially unrepresentative set of stored samples. This scarcity makes the model susceptible to *overfitting* on the memorized examples, degrading generalization to unseen instances of old relations. Third, CRL performs poorly on imbalanced datasets such as TACRED, where the unequal distribution of relation types causes the model to disproportionately favor dominant classes during contrastive replay, further amplifying forgetting for minority relations. These weaknesses motivate the integration of a prototype augmentation mechanism into the CRL framework. Instead of relying solely on real stored samples, the proposed method generates virtual data points by perturbing relation prototypes with Gaussian noise, thereby enriching the memory bank without increasing real data storage. This augmentation strategy mitigates

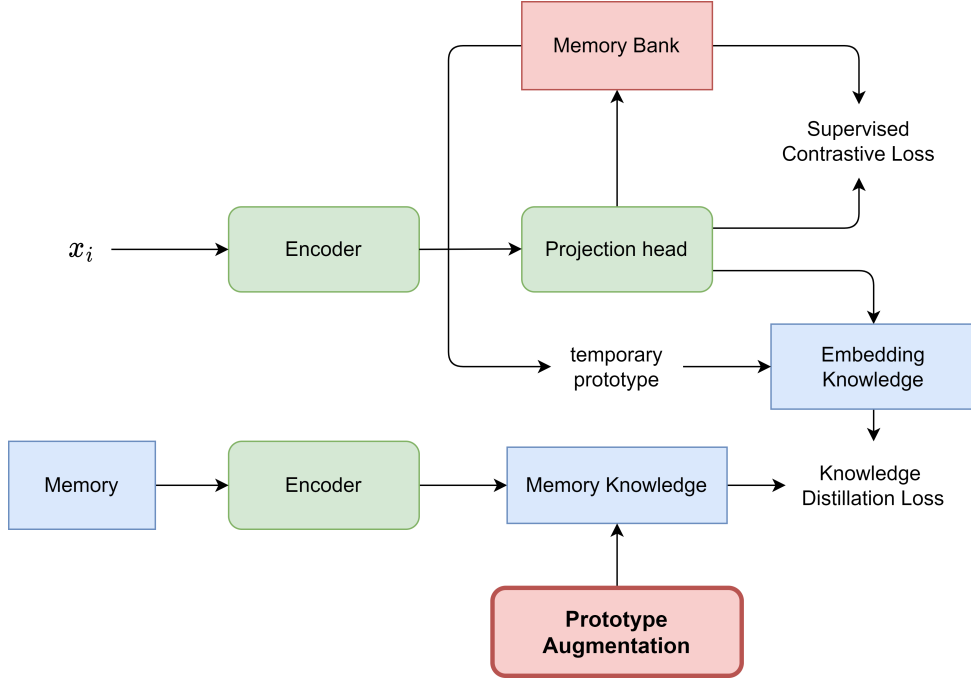


Fig. 2.2: Module of consistent representation using prototype augmentation(Ours).

overfitting, promotes class balance, and reduces dependency on privacy-sensitive data, ultimately improving the model’s ability to retain knowledge of previous relations across long task sequences.

We build on the prototype-augmentation idea of [141] but push it in two new directions. Whereas [141] targets class-incremental image classification with an encoder that never changes, our setting is continual relation extraction, where the encoder keeps updating and relation embeddings drift after every task. Second, [141] augments prototypes solely with isotropic Gaussian noise scaled by a fixed hyperparameter, whereas we derive the noise scale from the empirical covariance of stored representations (Equation 2.7), ensuring that the augmented distribution faithfully approximates the true class spread. Furthermore, we pair augmented replay with an Energy-based Latent feature alignment (ELI) module that corrects representational drift via Langevin dynamics, a component absent from [141], making our method the first to combine privacy-preserving prototype replay with energy-based latent correction in the CRE setting.

2.3 Proposed Method

2.3.1 Model Training Pipeline

Figure 2.2 shows the four interconnected components. The encoder and projection head are jointly trained with L_{CL} during the initial training phase (Section 2.3.3). Prototype augmentation (Section 2.3.5) then enriches the memory with synthetic samples before the replay phase, which combines contrastive replay and distillation to consolidate old-task knowledge without storing raw data.

Initial Training Encodes the input via the encoder $\mathbf{E}$. Then, the parameters from the encoder $\mathbf{E}$ and the projector $Proj$ are tuned with the current training samples in D_k using supervised contrastive learning.

Sample selection For each relation $r \in R_k$ that was unobserved in the previous tasks, we retrieve all samples with label r in D_k . Then, use the k-means algorithm to cluster these samples. For each cluster, the sample closest to the centroid is chosen to represent that cluster. Let's call it M_r . After that, generate a prototype p_r of r based on M_r to expand the prototype set $\mathbf{P}_r$.

Memory augmentation based on prototype Data storing from previous tasks might be limited because of memory or privacy issues. Therefore, in this part, we proposed a simple non-exemplar-based method, which uses prototype-based memory augmentation to solve the catastrophic forgetting problem in continual learning. On the one hand, proToAug memorizes a one-class representation prototype for each old class and augments memorized archetypes via Gaussian noise when learning new classes. Then, augmented prototypes and memory stored are joined to maintain the discrimination and balance between old and new classes.

Memory replay After the augmentation phase, combine the newly generated pseudo data and data in memory. And then train them with a bigger memory. In this process, memory replay is also applied iteratively to learn new relation prototypes while strengthening the discriminability of old relational prototypes.

2.3.2 Sample Encoder

Given an example x , we adopt an encoder to encode its semantic features for detection. To be specific, we first tokenized the given example into several tokens and then input the tokenized tokens into neural networks to compute their corresponding embedding. As

extracting relations from sentences is related to those entities mentioned in sentences, we thus add special tokens into the tokenized tokens to indicate the beginning and ending positions of those entities.

Specifically, we use $[E_{11}]$, $[E_{12}]$, $[E_{21}]$ and $[E_{22}]$ to denote the start and end position of the head and tail entity, respectively. Next, a sample's hidden representation is the concatenation of token embeddings of $[E_{11}]$ and $[E_{21}]$, which has been proven effective in previous works [7].

$$h = W[h_{[E_{11}]}; h_{[E_{21}]}] + b \quad (2.1)$$

where $h_{[E_{11}]}, h_{[E_{21}]} \in \mathcal{R}^{d_h}$ (d_h is the dimension of BERT hidden representation) are the hidden representations of $[E_{11}]$ and $[E_{21}]$, $W \in \mathcal{R}^{2d_h \times d_h}$ (d is sample embedding dimension) and $b \in \mathcal{R}^{d_h}$ are trainable parameters. The encoder is denoted as $\mathbf{E}$. Then, we use a projection head $Proj$ to obtain the low-dimensional embedding: For down-dimensional embedding, we feed the output Encoder $\mathbf{E}$ to the projector $Proj$:

$$\tilde{z} = \text{MLP}(h) \quad (2.2)$$

Where MLP is Multi-layer Perception. Finally, embedding $z = \tilde{z}/\|\tilde{z}\|$, here $\tilde{z}$ is the raw embedding vector from the MLP output and $\|\tilde{z}\|$ is the norm of vector $\tilde{z}$, typically the L2 norm (Euclidean length). Embedding z is used for contrastive learning, and the hidden representation is used for classification.

2.3.3 Initial Training for New Task

According to the general assumption of CRE, all relations in R_t are unobserved in former tasks $T_1 \sim T_{k-1}$. We first introduce the model's initial training on the current task for a new task. After feeding each batch B into encoder $\mathbf{E}$, projector $Proj$ and normalized output, we use supervised contrastive loss [60] to pull all samples in the same class closer and push others far away:

$$L_{CL} = \sum_{i=1}^{|B|} \frac{-1}{N_{y_i}} \sum_{j=1, |y_i=y_j}^{|B|} \log \frac{\exp(z_i \cdot z_j / \tau)}{\sum_{b=1}^{|B|} \exp(z_i \cdot z_b / \tau)} \quad (2.3)$$

Where N_{y_i} is the number of samples whose output is y_i . $\tau \in \mathcal{R}^+$ is a tunable parameter used to control the division of layers. The symbol $\cdot$ is used to represent the dot product.

2.3.4 Sample selection

Managing memory efficiently is crucial in continual learning, where the model must retain knowledge from previous tasks without exceeding a fixed storage budget. Before training begins, we set the total memory size M based on available resources and the number of tasks to be learned. As new tasks arrive, the number of stored samples per task is dynamically adjusted according to M and the current task count t . To make full use of the available storage, the memory quota assigned to task T_k is defined as:

$$M_k = \min\left(\max\left(M_{\text{left}}, \frac{M}{k}\right), |D_k|\right) \quad (2.4)$$

where M_{left} is the remaining free memory and $|D_k|$ is the total number of training samples in task T_k . To ensure that no single task dominates the memory, the budget M is divided equally among all tasks seen so far. When the memory is full, old samples must be removed to accommodate new ones. We first identify tasks whose current allocation M_i exceeds the fair share $\frac{M}{k}$, then randomly discard $M_i - \frac{M}{k}$ samples from those tasks until enough space M_k is freed for the new task. This rebalancing strategy keeps the per-task sample counts roughly equal, which helps prevent both overfitting to recent tasks and catastrophic forgetting of earlier ones.

For sample selection within each task, random sampling is insufficient because it may fail to capture the full diversity of the relation distribution. Instead, we apply k -means clustering to select more representative exemplars. Concretely, the text embeddings of all training instances in D_k are grouped into M_k clusters, and the sample closest to each cluster centroid is chosen for storage. By anchoring each stored sample at a cluster center, this approach maximizes the coverage of the feature space and ensures that the memory faithfully reflects the underlying data distribution of each task.

2.3.5 Memory augmentation based on prototype using derived Gaussian noise

Storing raw data from prior tasks is often limited by memory and privacy constraints, while generative replay approaches tend to be unstable and computationally inefficient to train. To address this, we propose a distribution-based augmentation memory designed to alleviate catastrophic forgetting.

We compute the expected feature of each class using the memory stored. For class c

in task k , we get:

$$\mu_{k,c} = \frac{1}{|M_c^k|} \cdot \sum_{i=1}^{|M_c^k|} \mathbf{E}(x_i) \quad (2.5)$$

Therefore the feature generated for class c in task k is augmented below:

$$f_{k,c} = \mu_{k,c} + \epsilon * r \quad (2.6)$$

where $\epsilon \sim \mathcal{N}(0, 1)$ is a noise vector drawn from a standard Gaussian distribution, with dimensionality identical to that of a prototype. r is a scale to control the uncertainty of the augmented prototypes. In particular, the scale r can be predefined or computed as the average variance of the class representations:

$$r_{k,c}^2 = \sum_{i=1}^{|M_{k,c}|} \frac{\text{Tr}(\Sigma_{k,c})}{D} \quad (2.7)$$

where D denotes the dimensionality of the deep feature space. $\Sigma_{k,c}$ is the covariance matrix for the features from class c at stage k , and the Tr operation computes the trace of a matrix. After the memory is augmented, we join this with our memory and train the next step, memory replay.

2.3.6 Memory Replay

After the initial training for the current task T_k , the old relations' embedding space is likely to be disrupted because the model is tuned towards fitting T_k 's learning objective. Inspired by [141], two replay strategies to learn consistent representation for alleviating this problem: contrastive replay and knowledge distillation on memory. One difference from the previous study is we augmented memory to alleviate catastrophic forgetting.

First, we use contrastive learning for replay memory. This step is a similar step to initial training for a new task; the difference here is that each batch uses all the samples in the entire memory bank for training:

$$\mathcal{L}_{CE} = \sum_{i=1}^{|M'_k|} \frac{-1}{N_{y_i}} \sum_{j=1|y_i=y_j}^{|M'_k|} \log \frac{\exp(z_i \cdot z_j / \tau)}{\sum_{m=1}^{|M'_k|} \exp(z_i \cdot z_m / \tau)} \quad (2.8)$$

Where M'_k is the memory size after augmentation, N_{y_i} is the number of samples of which label is equal to y_i .

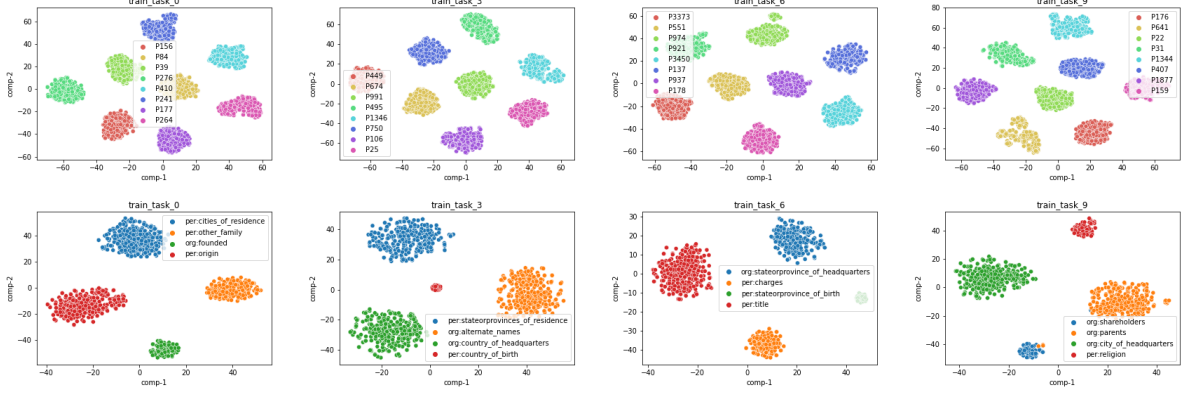


Fig. 2.3: A visualization of relation representation learned by our method at the different tasks on Fewrel and TACRED dataset.

2.3.7 Prediction

After finishing the k -th task, for each test sample x , we calculate the similarity between the embedding of x and each prototype embedding. For each label c we get:

$$p_c = \frac{1}{n_c} \sum_{i|y_i=c} \mathbf{E}(x_i) \quad (2.9)$$

If $g(\cdot)$ is Euclid distance:

$$y^* = \operatorname{argmin}_{c=1..k} g(f(x), p_c) \quad (2.10)$$

Else if $g(\cdot)$ is cosine similarity:

$$y^* = \operatorname{argmax}_{c=1..k} g(f(x), p_c) \quad (2.11)$$

The privacy-preserving design in Figure 2.2 (Section 2.4) still has two weak points. Gaussian augmentation only matches the true class distribution up to its second moment, so relations whose true distribution is multi-modal or heavy-tailed end up with augmented samples that misrepresent the original data and can bias the contrastive replay step. Second, the prototype set is updated only once per task using the mean of k-means cluster centres (Equation 2.5); if the encoder drifts significantly after learning subsequent tasks, the stored prototypes become geometrically misaligned with the current embedding space, reducing their utility. The ELI module introduced in Section 2.4 directly addresses this second limitation by realigning latent representations at test time.

2.4 Extension With Energy-based Model Alignment (ELI)

Distinction from prior energy-based and latent alignment methods. Energy-based models (EBMs) have been used for out-of-distribution detection and generative modelling [70], but their application to correcting representational drift in a continual learning context is distinct. Existing latent alignment methods in CL, such as feature distillation in iCaRL [98] or geometric regularization in PODNet [29], operate via loss penalties on intermediate features during training. Two things set ELI apart: it runs purely at inference time, applying Langevin dynamics to the current model’s encoding of past-task samples without any retraining, and it corrects the geometry of the latent manifold directly instead of penalising logits or feature norms. This test-time correction mechanism complements prototype augmentation and does not increase the training overhead.

ELI model training: When continuously learning a sequence of tasks, the optimized hidden feature space of previous tasks is changed over subsequent tasks, reducing the model’s performance on earlier tasks. To address this issue, we train a model based on the energy score and then feed it into the ELI algorithm to perform realignment of the skewed hidden spaces.

Energy-based Latent feature alignment (ELI) model

Once each task has been learned, the ELI module is optimized using three inputs, as illustrated in Figure 2.4: the raw samples $\mathbf{x}$ from the task currently being processed, together with two corresponding embeddings of $\mathbf{x}$ – one produced by the encoder as it stood after the previous task, $\mathbf{z}_{t-1} = \mathbf{E}_{t-1}(\mathbf{x})$, and one produced by the encoder in its present, post-update state, $\mathbf{z}_t = \mathbf{E}_t(\mathbf{x})$. Training the Energy-based Model (EBM) $\mathcal{E}_\psi$ consists of pushing the energy assigned to $\mathbf{z}_{t-1}$ downward while pushing that assigned to $\mathbf{z}_t$ upward; this procedure is summarized in Algorithm 1.

Once trained, $\mathcal{E}_\psi$ serves a corrective role: it is applied to offset the drift that old-task embeddings undergo once they are re-encoded by the updated model, i.e., $\mathbf{z}_t = \mathbf{E}_t(\mathbf{x})$. Because this drift moves $\mathbf{z}_t$ away from the region the encoder previously occupied, such embeddings end up sitting in higher-energy regions of the learned manifold. The alignment step therefore seeks new positions for $\mathbf{z}_t$ within the latent space that reduce this energy, effectively pulling the drifted representations back toward a low-energy configuration consistent with the earlier encoder.

EBM training To align feature space, we train the EBM model, which is constructed using a neural network that can map hidden representations to energy values

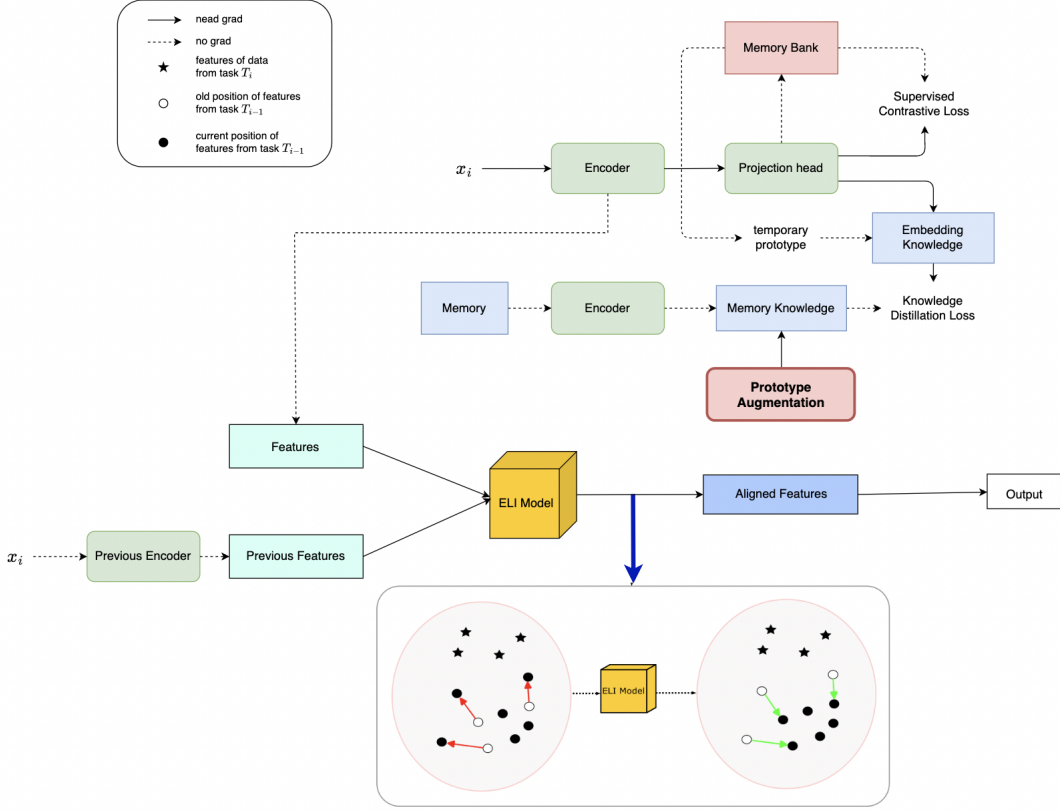


Figure 2.4: Overall architecture of our proposed continual relation extraction model. Here, x_i denotes the input (including a sentence and an entity pair appearing in it) of our model.

(constants)[52]. Specifically, for a hidden feature vector $\mathbf{z} \in \mathbb{R}^D$ in the latent space, To characterize the underlying structure of the data, a scalar-valued energy function $\mathcal{E}_\psi(\mathbf{z}) : \mathbb{R}^D \rightarrow \mathbb{R}$ is learned, which assigns each input an energy score. Based on this function, the corresponding energy-based model (EBM) is formulated as a Gibbs distribution $p_\psi(\mathbf{z})$ derived from $\mathcal{E}_\psi(\mathbf{z})$:

$$p_\psi(\mathbf{z}) = \frac{\exp(-\mathcal{E}_\psi(\mathbf{z}))}{\int_{\mathbf{z}} \exp(-\mathcal{E}_\psi(\mathbf{z})) d\mathbf{z}}, \quad (2.12)$$

where the denominator $\int_{\mathbf{z}} \exp(-\mathcal{E}_\psi(\mathbf{z})) d\mathbf{z}$ denotes the partition function, an integral over the entire space of $\mathbf{z}$ that is generally intractable to evaluate. The parameters of the EBM are optimized by maximizing the log-likelihood computed over samples drawn from the true underlying distribution $p_{true}(\mathbf{z})$:

$$L(\psi) = \mathbb{E}_{\mathbf{z} \sim p_{true}} [\log p_\psi(\mathbf{z})]. \quad (2.13)$$

The derivative of the function $L(\psi)$ is as follows [52]:

$$\partial_\psi L(\psi) = \mathbb{E}_{\mathbf{z} \sim p_{true}} [-\partial_\psi \mathcal{E}_\psi(\mathbf{z})] + \mathbb{E}_{\mathbf{z} \sim p_\psi} [\partial_\psi \mathcal{E}_\psi(\mathbf{z})]. \quad (2.14)$$

Algorithm 1 EBM training

Require: Feature extractor for the current task: $\mathbf{E}_t$; Feature extractor for the previous task: $\mathbf{E}_{t-1}$; Data distribution of the current task: $p_{data}^{T_t}$

Ensure: $\mathcal{E}_\psi$ is optimized

- 1: $\mathcal{E}_\psi \leftarrow$ Initialize
 - 2: **while** until required iterations **do**
 - 3: $\mathbf{x} \sim p_{data}^{T_t}$ $\triangleright$ Sample a mini-batch
 - 4: $\mathbf{z}^{T_{t-1}} \leftarrow \mathbf{E}_{t-1}(\mathbf{x})$
 - 5: $\mathbf{z}^{T_t} \leftarrow \mathbf{E}_t(\mathbf{x})$
 - 6: $\mathbf{z}_{sampled}^{T_t} \leftarrow$ Sample from EBM with $\mathbf{z}^{T_t}$ as starting point $\triangleright$ Refer Equation (2.15)
 - 7: in_dist_energy $\leftarrow E_\psi(\mathbf{z}^{T_{t-1}})$
 - 8: out_dist_energy $\leftarrow E_\psi(\mathbf{z}_{sampled}^{T_t})$
 - 9: Loss $\leftarrow$ (-in_dist_energy + out_dist_energy) $\triangleright$ Refer Equation (2.14)
 - 10: Optimize $\mathcal{E}_\psi$ with Loss.
 - 11: **end while**
-

The first component in (2.14) ensures the reduction of energy for samples $\mathbf{z}$ drawn from the genuine data distribution p_{true} , while the subsequent component guarantees that the samples generated from the model itself will possess elevated energy levels. In the ELI context, p_{true} corresponds to the distribution of latent representations from the model trained on the preceding task. Obtaining samples from $p_\psi(\mathbf{x})$ is challenging due to the normalization constant in Eq. (2.12). Approximate samples are iteratively generated using Langevin dynamics [127], a widely utilized Markov Chain Monte Carlo (MCMC) algorithm.

$$\mathbf{z}_{i+1} = \mathbf{z}_i - \frac{\lambda}{2} \partial_{\mathbf{z}} \mathcal{E}_\psi(\mathbf{z}) + \sqrt{\lambda} \omega_i, \omega_i \sim \mathcal{N}(0, \mathbf{I}) \quad (2.15)$$

where λ is the step size, and ω represents data uncertainty. In our experiments, we set λ to 0.1, as suggested by [52]. Equation (2.15) results in a Markov chain that converges to a stationary distribution after a few iterations, starting from an initial value $\mathbf{z}_i$.

We will feed the features through ELI (as depicted in Algorithm 2) for alignment during the testing phase.

At test time, Algorithm 2 takes the current model’s encoding z of a test sample and repeatedly steps it against the gradient of $E_\psi(z)$, nudging z toward whatever low-energy region matches the distribution learned from earlier tasks. Convergence is guaranteed by the contraction property of Langevin dynamics under mild smoothness assumptions [127].

Algorithm 2 Latent space alignment (ELI)

Require: Latent vector: $\mathbf{z}$; EBM: $\mathcal{E}_\psi$; Langevin iterations: L_{steps} ; Learning rate: λ

Ensure: $\mathbf{z}$

```
1: while until  $L_{steps}$  iterations do  
2:    $grad = \nabla_{\mathbf{z}} \mathcal{E}_\psi(\mathbf{z})$   
3:    $\mathbf{z} \leftarrow \mathbf{z} - \lambda \cdot grad$   
4: end while  
5: return  $\mathbf{z}$ 
```

The computational overhead is $L_{steps} \times O(D)$ per sample, where D is the latent dimension and $L_{steps} = 20$ in our experiments. This is negligible compared to the encoder forward pass. The ablation in Table 2.1 and Table 2.2 confirms that removing ELI decreases accuracy (e.g., by 1.3% on FewRel (2-class-per-task)), validating its contribution to representational stability.

The two components play complementary roles: prototype augmentation with Gaussian noise addresses the data-level problem of approximating past class distributions without storing real samples, while the ELI module addresses the model-level problem of latent representation drift across sequential tasks.

2.5 Experimental Results

2.5.1 Experiment Setting

The task sequence is built by drawing relations purely at random and splitting the full relation set into 10 disjoint groups, one per incremental task, the same partitioning protocol Cui et al. [22] used. We keep the random seed at 2021 throughout, matching Cui et al. [22], so the comparison stays fair, guaranteeing that the task sequence encountered by all models is exactly the same. It is worth noting that our reproduced baseline RP-CRE[†] and the proposed model CRL are trained and evaluated under strictly the same experimental environment.

We use `bert-base-uncased` [26] as the sentence encoder backbone, with a hidden size of 768. The input sequences are truncated or padded to a maximum length of 256 tokens, and entity markers are employed as the input pattern. A projection head maps the encoder output to a 64-dimensional feature space for contrastive learning. The model is optimized using the Adam optimizer with a learning rate of 5×10^{-6} , a batch size of

16, and gradient clipping at a maximum norm of 10. Training proceeds in two stages, each consisting of 10 epochs. The temperature hyperparameter τ for contrastive loss is set to 0.1, while the temperature for KL divergence is set to 10. For memory replay, we store 20 prototypes per relation class. All experiments are conducted over 5 independent rounds to report stable results.

2.5.2 Metric and Baseline Comparisons

For evaluating the efficiency of reducing catastrophic forgetting, we use average accuracy as the primary metric, consistent with prior work on continual relation extraction [22, 40]. Moreover, as in the previous paper, we use the average accuracy across 10 tasks at each step. We experiment on FewRel [33, 39] and TACRED [138]

We evaluate our combination of CRL and prototype augmentation and these two baselines, which have the best performance in CRE tasks at this time, with the above benchmarks for comparison:

- **RP-CRE** [22] achieves good performance in reducing catastrophic forgetting by refining sample embeddings. In this research, they save the prototype of each relation to extract relevant information from each relation in order to effectively leverage memorized samples. In particular, the prototype embedding for a given relation is calculated using examples of the relation that have been stored and gathered using the K-means algorithm.
- **CRL** [139] is proposed to maintain the relation embedding using contrastive learning and knowledge distillation. It is shown to perform better than other baselines at that time. This research trains each new task using supervised contrastive learning with a memory bank, enabling the model to learn relation representations efficiently. The samples are then contrastively replayed in memory. The model is designed to retain its understanding of past relationships through memory knowledge distillation to mitigate forgetting of previous tasks.

2.5.3 Result and discussion

Result without ELI model

Table 2.1 and table 2.2 show the result of the experiments. In this statistics table, $T_1, T_2, \dots, T_{10}$ represent ten independent tasks, with T_i as the i -th task; the division is based on the idea of Zhao et al. in the original paper about CRL [139]. Our method

Table 2.1: Accuracy (%) comparison on the FewRel dataset across ten incremental tasks (T1–T10). Results for EA-EMR, EMAR, CML, RP-CRE, and CRL are as reported in their original papers.

FewRel										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR [120]	89.0	69.0	59.1	54.2	47.8	46.1	43.1	40.7	38.6	35.2
EMAR [40]	88.5	73.2	66.6	63.8	55.8	54.3	52.9	50.9	48.8	46.3
CML [130]	91.2	74.8	68.2	58.2	53.7	50.4	47.8	44.4	43.1	39.7
EMAR+BERT*	98.8	89.1	89.5	85.7	83.6	84.8	79.3	80.0	77.1	73.8
RP-CRE [22]	97.8	95.1	91.8	90.5	89.9	87.7	86.6	85.6	84.4	82.6
CRL [139]	98.2	94.6	92.5	90.5	89.4	88.0	86.9	85.6	84.5	83.1
Ours	98.2	95.2	93.2	91.7	90.6	89.1	88.2	86.9	85.8	84.4

* BERT-based variant of EMAR, as reported in [22].

outperforms the two baseline methods, CRL and RP-CRE, on both datasets (FewRel and TACRED). This means that, without saving real data from previous tasks, we augment the data with a prototype for each relation to retrain on new tasks, and the model achieves higher performance than reusing real data.

With the FewRel dataset, in the first task, the CRL model and ours have similar results at 98.2%, but then the performance of CRL on the next task decreases significantly to 94.6%, and CRL only has 83.1% accuracy while our model has 84.4% on task 10, which is 1.3% higher than the CRL model. In addition, the RP-CRE model gets lower results on ten tasks. Likewise, with the TACRED dataset, both CRL and our models begin with 97.8 %. Afterwards, the performance of CRL on task 2 declined remarkably to 93.2% and results on the last task are only 78%, while our model still reached 81% on task 10. Compared with the RP-CRE model, all tasks get higher results from our model. The results show that our method produces the best representation for continual relation extraction.

Figure 2.3 illustrates that after the training process, the representations of data points are still good; specifically, the data points are close to each other in the same relation. Combined with the experimental results in Table I and Table II, our model performs stability on relations, which means this model alleviates the catastrophic forgetting problem.

Result when adding ELI model

Table 2.2: Accuracy (%) comparison on the TACRED dataset across ten incremental tasks (T1–T10). Results for EA-EMR, EMAR, CML, RP-CRE, and CRL are as reported in their original papers.

TACRED										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR [120]	47.5	40.1	38.3	29.9	24.0	27.3	26.9	25.8	22.9	19.8
EMAR [40]	73.6	57.0	48.3	42.3	37.7	34.0	32.6	30.0	27.6	25.1
CML [130]	57.2	51.4	41.3	39.3	35.9	28.9	27.3	26.9	24.8	23.4
EMAR+BERT*	96.6	85.7	81.0	78.6	73.9	72.3	71.7	72.2	72.6	71.0
RP-CRE [22]	97.6	93.2	90.6	85.1	82.7	81.1	78.3	76.0	76.1	75.7
CRL [139]	97.8	93.2	89.8	84.7	84.1	81.3	80.2	79.1	79.0	78.0
Ours	97.8	93.9	91.8	87.2	86.1	84.8	82.9	82.0	81.9	81.0

* BERT-based variant of EMAR, as reported in [22].

Table 2.3: Experimental accuracy of our proposed CRE model and existing state-of-the-art models across all tasks on the FewRel dataset. T is an abbreviation for Task.

FewRel										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR [120]	89	69	59.1	54.2	47.8	46.1	43.1	40.7	38.6	35.2
EMAR [40]	88.5	73.2	66.6	63.8	55.8	54.3	52.9	50.9	48.8	46.3
CML [130]	91.2	74.8	68.2	58.2	53.7	50.4	47.8	44.4	43.1	39.7
EMAR+BERT	98.8	89.1	89.5	85.7	83.6	84.8	79.3	80	77.1	73.8
RP-CRE [22]	97.9	92.7	91.6	89.2	88.4	86.8	85.1	84.1	82.2	81.5
CRL [139]	<u>98.2</u>	<u>94.6</u>	<u>92.5</u>	<u>90.5</u>	<u>89.4</u>	<u>87.9</u>	<u>86.9</u>	<u>85.6</u>	<u>84.5</u>	<u>83.1</u>
Ours	<u>98.2</u>	94.7	93.9	92.2	90.1	89.6	88.6	87.1	86.1	84.6

The experimental results of our CRE model reported on each dataset were averaged over five runs. We compare our model with six state-of-the-art existing methods for continual semantic relation extraction. Tables 2.3 and 2.4 present the accuracy scores for each task on the FewRel and TACRED datasets, respectively. Our model demonstrates superior performance on the FewRel dataset, achieving the highest accuracy on nine out of ten tasks on the FewRel dataset. Notably, it outperforms the recent CRL model [139] by 1-2% on 8 out of 10 tasks, as illustrated in Table 2.3. Our model is on par with CRL on two remaining tasks, i.e. Task 1 and 2. However, our method performs less effectively

Table 2.4: Experimental accuracy of our proposed CRE model and existing state-of-the-art models across all tasks on the TACRED dataset. T is an abbreviation for Task.

TACRED										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR [120]	47.5	40.1	38.3	29.9	24	27.3	26.9	25.8	22.9	19.8
EMAR [40]	73.6	57	48.3	42.3	37.7	34	32.6	30	27.6	25.1
CML [130]	57.2	51.4	41.3	39.3	35.9	28.9	27.3	26.9	24.8	23.4
EMAR+BERT	96.6	85.7	81	78.6	73.9	72.3	71.7	72.2	72.6	71
RP-CRE [22]	97.6	90.6	86.1	82.4	79.8	77.2	75.1	73.7	72.4	72.4
CRL [139]	<u>97.7</u>	<u>93.2</u>	<u>89.8</u>	<u>84.7</u>	<u>84.1</u>	<u>81.3</u>	<u>80.2</u>	<u>79.1</u>	<u>79</u>	<u>78</u>
Ours	97.8	95.2	90.6	88.2	84.6	81.7	82.8	80.5	80	79.6

than EMAR+BERT [40]. It’s worth noting that EMAR+BERT employs BERT, a powerful pre-trained model, which provides a highly effective initialization for Task 1. Without using BERT, EMAR [40] underperforms our model by significantly large margins on all 10 tasks, highlighting the effectiveness of our approach in leveraging memory and energy-based latent alignment for continual learning. We note that when learning only on Task 1, our proposed model does not suffer from catastrophic forgetting, so the effectiveness of our approach was not demonstrated in this specific scenario. However, as the number of tasks increases, our model demonstrates superior ability in mitigating catastrophic forgetting, consistently outperforming EMAR+BERT in subsequent tasks.

Similarly, the results on the TACRED dataset, presented in Table 2.4, further validate the effectiveness of our model. Our model achieves the highest accuracy scores across all tasks, demonstrating its robustness on this more challenging, imbalanced dataset. Especially on Task 4, our model can surpass CRL [40] by nearly 4%, the highest accuracy margin observed across all tasks. Unlike the results on the FewRel dataset, our model outperforms EMR+BERT on all tasks with significant large accuracy margins, ranging from 1.1% (on Task 1) to 11.1% on Task 7.

A few points stand out here. **(1) Task 1 parity.** At Task 1 on FewRel (Table 2.3) our model and CRL are tied at 98.2%, unsurprising, since with no earlier tasks yet ELI has no past distribution to correct toward. **(2) Compounding gains on later tasks.** The lead over CRL keeps growing as tasks pile up, just 0.1% at Task 2 but 1.5% by Task 10 (84.6% against 83.1%), which fits how ELI is meant to work: more tasks means more accumulated drift for it to correct, so its benefit becomes larger over time. **(3) TACRED imbalance**

robustness. On TACRED (Table 2.4), our model surpasses CRL by up to 3.5% (Task 4), which is the task with the most severe class imbalance. Prototype augmentation generates balanced synthetic samples, directly mitigating the representation collapse that affects CRL on minority relations. **(4) Comparison with EMAR+BERT.** EMAR+BERT’s advantage on early FewRel tasks is attributable to BERT’s stronger initialization; however, our method surpasses EMAR+BERT from Task 5 onwards on TACRED, demonstrating that task-specific BERT fine-tuning is not sufficient for long-horizon stability.

The consistent superior performance of our model across both datasets can be attributed to several factors:

- **Effective memory utilization:** Our memory-prototype approach allows for efficient use of stored information, enabling better retention of knowledge from previous tasks.
- **Energy-based Latent space Alignment (ELI):** The ELI module helps maintain consistency in the latent space across tasks, mitigating the representational shift problem common in continual learning scenarios.
- **Robustness to dataset characteristics:** The model’s ability to outperform existing methods on both the balanced FewRel and imbalanced TACRED datasets demonstrates its versatility and robustness to different data distributions.
- **Long-term stability:** Performance improvement becomes more pronounced in later tasks, indicating our model’s superior ability to mitigate catastrophic forgetting over extended sequences of tasks.

These results underscore the effectiveness of our proposed approach in addressing the challenges of continual relation extraction, particularly in maintaining performance across a sequence of tasks while mitigating catastrophic forgetting.

2.6 Summary

This chapter of the dissertation studies the effectiveness of memory-based methods. We propose a rehearsal method by generating random samples from the approximate normal distribution of each task. The experimental results demonstrate the effectiveness of the model. However, this approach is typically memory-intensive and involves data security issues. In the next chapter, the dissertation explores how to distill the learned knowledge of the model when learning on new tasks without requiring the storage and

use of memory. The contribution of this chapter was published in The 2022 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1-6. IEEE, 2022 [\[P1\]](#).

Chapter 3

Knowledge Transfer in Preserving Previous Information in Continual Learning

3.1 Introduction

Neural networks dramatically degrade performance on previously learned tasks when adapted to new ones. This phenomenon occurs because neural network parameters optimized for new tasks can overwrite representations critical for earlier tasks, leading to severe performance degradation. Addressing catastrophic forgetting while maintaining the capacity to learn new information efficiently constitutes the stability-plasticity dilemma, which has motivated extensive research into continual learning methodologies.

Among various continual learning approaches, knowledge transfer mechanisms have emerged as particularly promising for mitigating forgetting while enabling efficient learning. Rather than relying solely on storing past data or expanding model capacity, knowledge transfer methods leverage sophisticated techniques to distill and preserve essential information from previous tasks. These approaches can extract compact representations of past knowledge, align feature distributions across tasks, and selectively consolidate important parameters while maintaining computational efficiency and respecting privacy constraints.

This chapter investigates two complementary knowledge transfer strategies for **Task-Incremental Learning(TIL)** scenario in [\[P3\]](#) and **Class-Incremental Learning(CIL)**

[P4]. First, we explore knowledge distillation combined with contrastive learning principles to preserve discriminative representations across sequential tasks. This approach introduces a Continual Learning Plugin (CL-plugin) that facilitates both knowledge sharing and task-specific adaptation, augmented with Contrastive Ensemble Distillation (CED) and Contrastive Supervised Learning (CSC) losses. Second, we investigate how to effectively retain and transfer knowledge from pre-trained models using feature alignment techniques and Fisher-weighted parameter fusion, thereby enabling efficient adaptation while preventing representation drift.

The remainder of this chapter is organized as follows: Section 3.2 presents the knowledge distillation approach with contrastive learning, detailing the CL-plugin architecture and associated loss functions and experimental validation on aspect sentiment classification benchmarks. Section 3.3 introduces knowledge retention from pre-training models using analytic learning. Finally, Section 3.4 summarizes the key contributions and insights from both approaches.

3.2 Knowledge Distillation with Contrastive Learning Approach

Knowledge distillation (KD) trains a student model f_s to reproduce the output distribution of a teacher model f_t , typically a frozen snapshot of the model taken before the current task begins. Given an input x and temperature τ , the standard KD objective minimizes the Kullback–Leibler divergence between softened predictions:

$$\mathcal{L}_{KD} = \tau^2 \text{KL}\left(\sigma\left(\frac{z_t}{\tau}\right) \parallel \sigma\left(\frac{z_s}{\tau}\right)\right), \quad (3.1)$$

where z_t and z_s denote the teacher’s and student’s pre-softmax logits and $\sigma(\cdot)$ is the softmax function. In continual learning, KD is the mechanism behind regularization-based methods such as LwF), where the "teacher" is simply the model’s own state prior to the current task.

Plain output-level KD assumes teacher and student share an identical label space, which breaks down once classes accumulate across tasks or once the transfer target is a representation rather than a classifier. Contrastive transfer relaxes this assumption by aligning teacher and student embeddings instead of logits, using an InfoNCE-style objective:

$$\mathcal{L}_{NCE} = -\log \frac{\exp(\text{sim}(h_i, h_i^+)/\tau)}{\sum_j \exp(\text{sim}(h_i, h_j)/\tau)}, \quad (3.2)$$

where h_i^+ is a positive pair for anchor h_i (e.g., the teacher’s embedding of the same sample) and the sum in the denominator ranges over negatives in the batch. Because Eq. (3.2) operates purely in representation space, it transfers knowledge without requiring a shared, growing classification head, and tolerates ensembles of teachers. This is the principle underlying the Contrastive Ensemble Distillation (CED) and Contrastive Supervised learning of Current task (CSC) losses.

We address the core challenge of continual learning by balancing knowledge retention and transfer through a dual-component strategy that combines architectural protection with contrastive learning by implementing a CL-plugin with two modules: the Knowledge Sharing Sub-Module (KSM) for cross-task knowledge transfer and the Task-Specific Sub-Module (TSM) with neuron-level masks to prevent catastrophic forgetting.

Intuitively, once a neuron has been identified as important for a task, its parameters are locked and can no longer be modified, even when the model is subsequently trained on new tasks. This ensures that the knowledge learned for earlier tasks is preserved intact, while only the remaining, non-critical neurons stay trainable and are used to acquire new knowledge. As a result, learning new tasks does not overwrite or interfere with previously learned knowledge, effectively mitigating catastrophic forgetting.

We augment this with contrastive learning losses [58]: CED loss for ensemble knowledge distillation across all previous tasks, and CSC loss for enhanced current task performance, together improving both stability and plasticity in the TIL setting.

Figure 3.3 illustrates the overall architecture of our proposed method. The detailed components are presented as follows.

3.2.1 Related Work

Recent years have seen a surge of research efforts focusing on the integration of advanced pre-trained models, such as BERT (Bidirectional Encoder Representations from Transformers) [26], into continual learning approaches. BERT adapters, lightweight task-specific neural network components, have emerged as a promising solution [46], which chooses to freeze the parameters of the pre-trained BERT, only tuning a small number of parameters of injected task-specific adapters [46, 56, 59]. A BERT adapter can be a relatively small 2-layer fully connected network or a Capsule Networks (CapsNet) [44, 104] that uses vector capsules instead of scalar feature detectors. These task-specific adapters enable CL models to learn new tasks while preserving knowledge from previous tasks. Researchers have explored combining regularization and memory-based techniques

with BERT adapters to further enhance their performance in continual learning scenarios. Regularization methods are applied to adapter parameters to mitigate forgetting, while memory-based strategies such as Experience Replay and Generative Replay are integrated to retain knowledge from past tasks. By leveraging the strengths of both pre-trained models and continual learning techniques, these approaches aim to develop robust and adaptable models across diverse tasks and domains.

Although the CL capability of existing advanced models have been empirically demonstrated they still struggle with CF issues when tasks lack substantial shared knowledge [56, 58].

Isolating task-specific modules in CL models aids in mitigating catastrophic forgetting to some extent. However, it poses a challenge by limiting the utilization and updating of shared knowledge across tasks, which could enhance the model’s performance on related tasks. This contradiction makes it difficult for CL to simultaneously achieve both objectives: preventing catastrophic forgetting and facilitating knowledge transfer.

As far as we know, only a handful of continual learning (CL) models have effectively achieved both objectives simultaneously. Notably, CTR [56] and CLASSIC [58] are two prominent CL models that have recently demonstrated success in this regard. Both models employ task masks to isolate task-specific knowledge for dealing with CF. While CTR operates within the TIL setting, CLASSIC is designed for the DIL scenario. Remarkably, both models are specifically tailored for Aspect Sentiment Classification (ASC) tasks.

CLASSIC introduces an approach for knowledge transfer through contrastive learning, focusing on domain continual learning (i.e. DIL setting) [58], the overview is show in Fig 3.1. Its key innovation lies in a contrastive continual learning method facilitating both knowledge transfer across tasks and distillation from old tasks to the new task. CLASSIC utilizes BERT-Adapter, maintaining BERT parameters unchanged while achieving performance comparable to BERT fine-tuning. By proposing task masks to isolate task-specific knowledge and a contrastive continual learning method for knowledge transfer and distillation, CLASSIC significantly enhances the accuracy of all tasks.

CTR utilizes a key component known as the CL-plugin that is injected into BERT at two locations, illustrated in Fig 3.2. Compared to BERT-Adapter, CL-plugins differ significantly. While an adapter is a simple 2-layer fully-connected network inserted into BERT (to be fine-tuned) for each specific end task, the CL-plugin functions as a capsule network, which deviates from traditional neural networks by utilizing vector-output capsules instead of scalar activations, thereby preserving additional information in

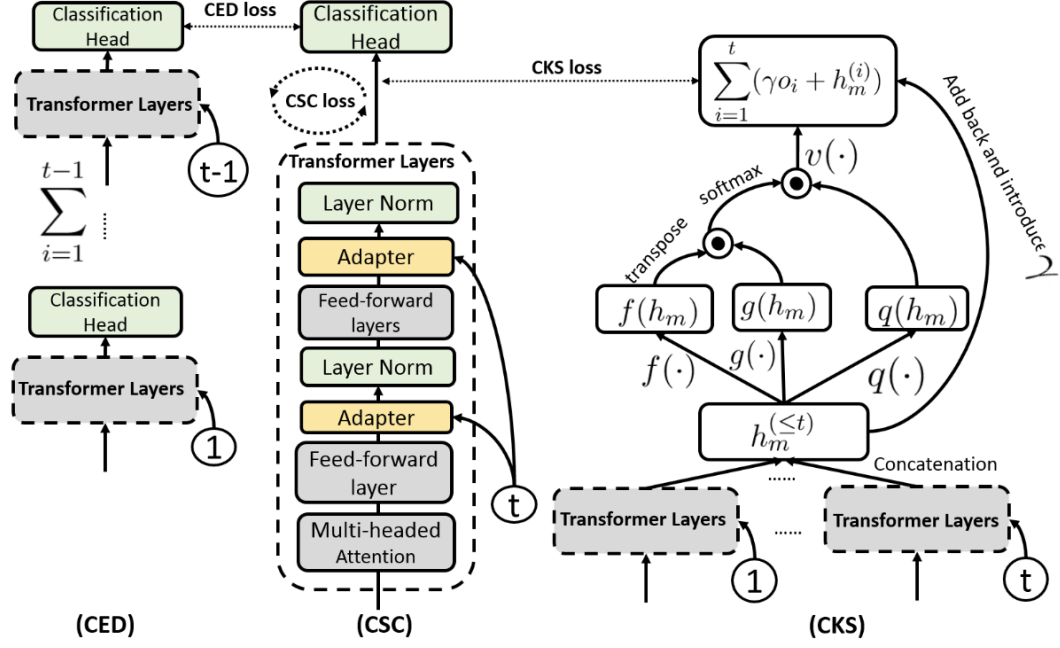


Figure 3.1: Overview of CLASSIC [58]

a more nuanced manner. CL-plugin integrates a novel transfer routing mechanism that facilitates knowledge exchange across tasks while safeguarding task-specific information to prevent interference. CTR can learn all tasks using only one pair of CL-plugin modules inserted into BERT. By this strategic integration, CTR eliminates the need for individual fine-tuning of BERT for each task, which often leads to catastrophic forgetting [56].

3.2.2 Continual Learning Plugin (CL-plugin)

Table 3.1: Inherited components vs. new contributions in CL-plugin

Component	Origin	Role in our model
CL-plugin base architecture	Inherited from [56]	Plug-in module backbone
Neuron mask m_l (p.64)	Inherited, adapted	Prevents forgetting at layer l
Contrastive Ensemble Distillation (p.64)	New	Selective knowledge distillation
Contrastive Supervised Classification (p.65)	New	Improves class discrimination

Given hidden states $h(t)$ extracted from the feed-forward layer within a transformer

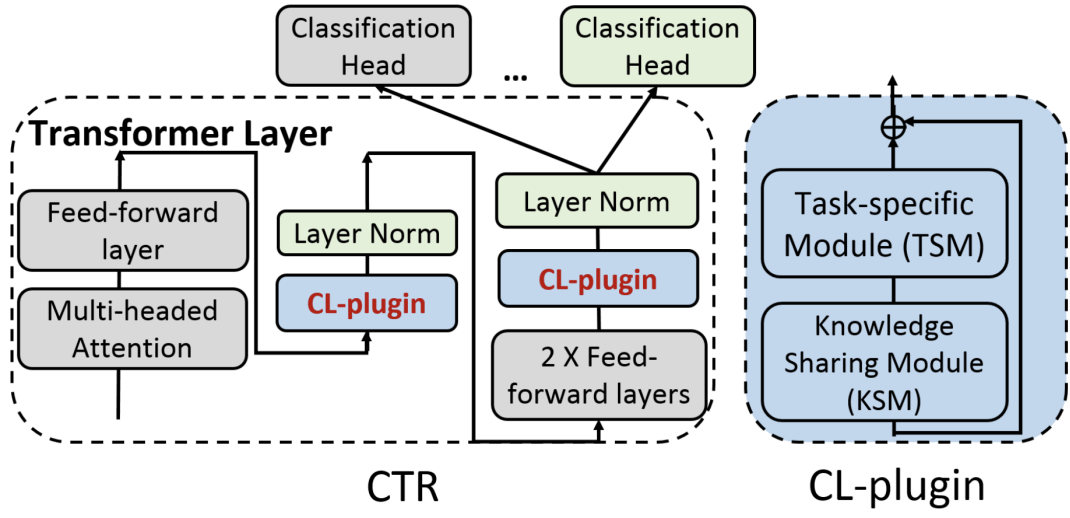


Figure 3.2: Architecture of CTR [56]

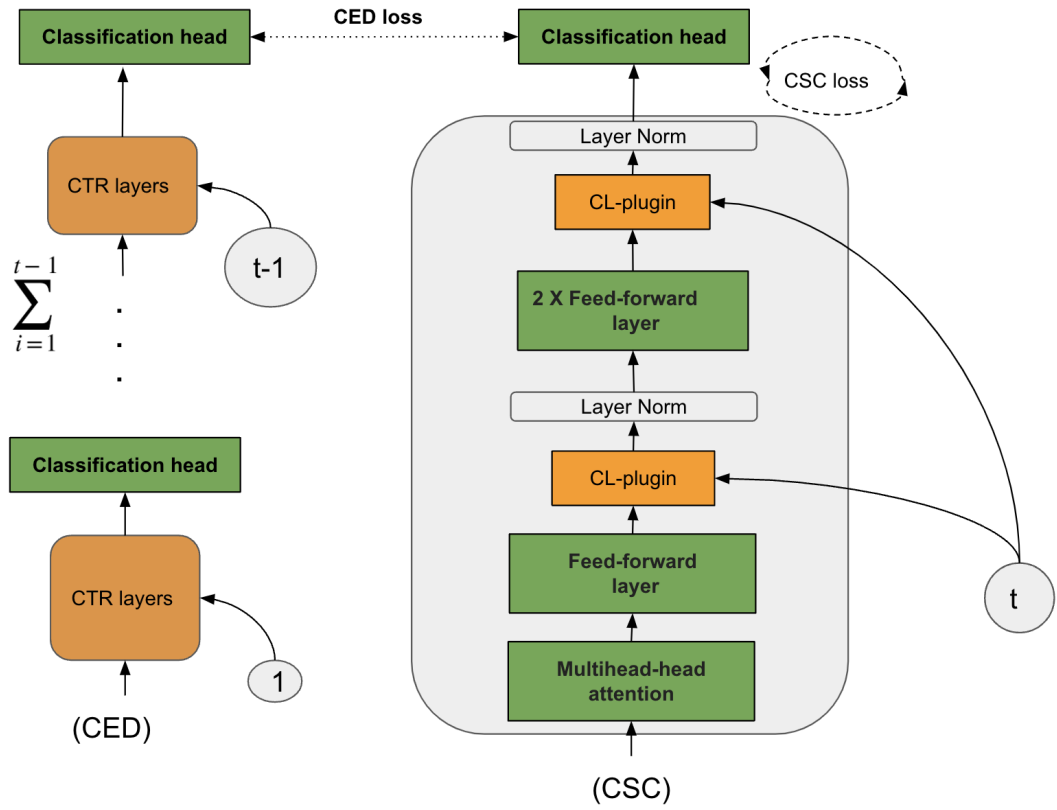


Figure 3.3: The overall architecture of our proposed model.

layer, and the task ID t , the CL-plugin yields outputs that comprise hidden states with informative features specific to the t^{th} task. Finally, CL-plugin employs the cross-entropy loss to optimize the model's parameters, expressed as follows:

$$\mathcal{L}_{CE} = \sum_{i=1}^N -y_i \log \hat{y}_i \quad (3.3)$$

where $\hat{y}_i$ is the probability score of the i^{th} sample in the batch data after passing through the model.

Design rationale for CL-plugin placement. The two CL-plugin modules are inserted after the Layer Normalization (LayerNorm) sublayers rather than after the attention or feed-forward sublayers, for two reasons: First, LayerNorm outputs are scale-normalized representations that are more amenable to capsule-based routing, as the dynamic routing algorithm in the KSM assumes bounded input magnitudes. Second, putting a plugin after each of the two LayerNorms, one following self-attention, one following the feed-forward block, lets the KSM pick up syntactic cues from the attention side and semantic cues from the feed-forward side, enriching what the shared representation captures[56].

Frozen vs. trainable components. During training on task t , the following components are **frozen**: all original BERT parameters Θ_{BERT} , and the task masks $m_l^{(t')}$ for all completed tasks $t' < t$. The following components are **trainable**: the KSM capsule layers, the TSM fully-connected layers, the task embedding $e_l^{(t)}$ for the current task, and the classification head for task t . This frozen/trainable split ensures that knowledge from past tasks encoded in the masks is not overwritten during new task adaptation.

Inside the CL-plugin, the KSM consists of two capsule layers, namely the task capsule layer and the knowledge-sharing capsule layer, equipped with a dynamic routing algorithm. This configuration effectively clusters similar tasks and shared knowledge, enabling knowledge transfer among tasks with commonalities. In contrast, the TSM comprises differentiable fully-connected layers, with each layer's output undergoing additional processing using a task-specific mask. This mask indicates which neurons need protection for the specific task to address CF, preventing gradient updates on these neurons masked for the specific task during back-propagation when adapting to a new task (see Figure 3.4).

Specially, for a specific task ID t , $e_l^{(t)}$ denotes its embedding trained for the l^{th} layer within the TSM. This embedding encompasses differentiable parameters that can be learned concurrently with other components of the network. The task mask $m_l^{(t)}$, akin to a "soft" binary mask, is generated by employing a pseudo-gate function denoted as σ

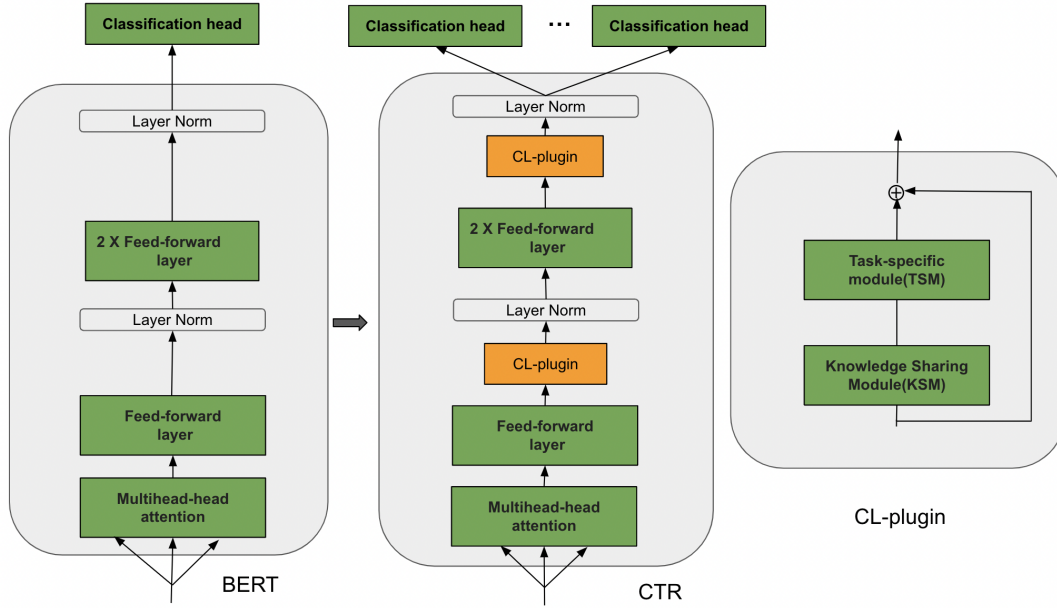


Figure 3.4: General architecture of BERT (left) and the CTR system (middle) and the CL-plugin (right) that are integrated into BERT.

and incorporating a positive scaling hyper-parameter denoted as s throughout the training procedure. The calculation of $m_l^{(t)}$ is outlined as follows:

$$m_l^{(t)} = \sigma(se_l^{(t)}) \quad (3.4)$$

For tasks sharing some common knowledge, their masks may have some neurons that coincide with each other. Each layer output $k_l^{(t)}$ in TSM when learning the task t is element-wise multiplied with $m_l^{(t)}$. To this end, the masked output $k_l^{(t)}$ from the last layer is then passed to the subsequent BERT model layer through a skip connection (see TSM in Figure 3.5). The final $m_l^{(t)}$ for all layers after learning the task t is saved for further use.

3.2.3 Contrastive Ensemble Distillation (CED) Loss

This loss employs contrastive learning to distill knowledge from all previous tasks into the current task t . For each previous task i , the Contrastive Ensemble Distillation (CED) is calculated as follows:

$$\mathcal{L}_{\text{CED}}^{(i)} = \sum_{n=1}^N -\log \frac{\exp((z_n^{(i)} \cdot z_n^{(t)})/\tau)}{\sum_{j=1, j \neq n}^N \exp((z_n^{(i)} \cdot z_j^{(t)})/\tau)} \quad (3.5)$$

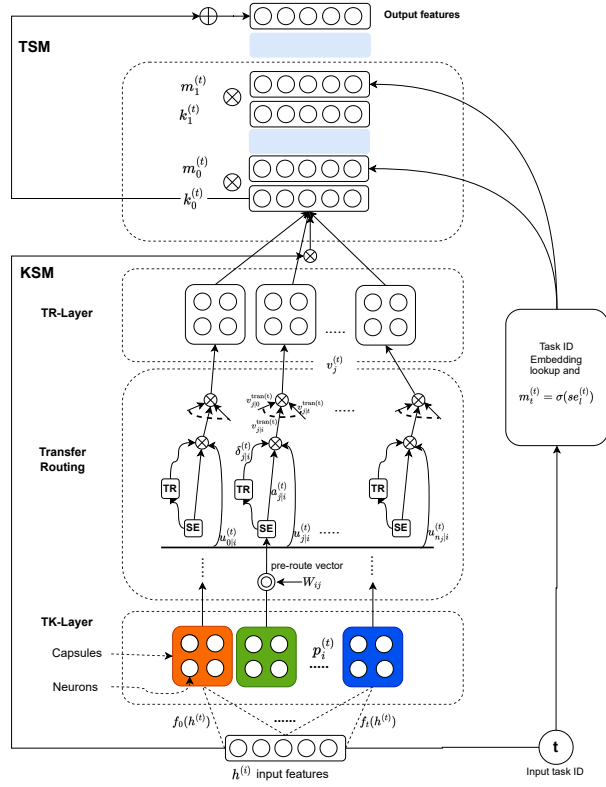


Figure 3.5: The architecture details of the CL-plugin within our model.

where N is the batch size; τ is an adjustable temperature parameter controlling the separation of classes; n is the index of the data sample in the batch; $z_n^{(i)}$ and $z_n^{(t)}$ are the logits for the same input data at the index n , generated from our model for the previous task i and current task t , respectively. This logit pair is treated as the positive pair and all of the other pairs are considered as negative pairs.

Since the model for each previous task remains fixed during the current task’s training process, it effectively serves as a teacher for the current task (acting as the student). For $t - 1$ models corresponding to all previous tasks, the final CED is formulated as per formula 3.6.

$$\mathcal{L}_{\text{CED}} = \sum_{i=1}^{t-1} \mathcal{L}_{\text{CED}}^{(i)} \quad (3.6)$$

3.2.4 Contrastive Supervised Learning of Current Task (CSC) Loss

To improve current task performance, we employ a contrastive objective that enhances class discrimination. Inspired by CLASSIC [58], we introduce the Contrastive Supervised Learning of Current Task (CSC) Loss, which leverages contrastive learning to maximize intra-class similarity while minimizing inter-class similarity for the current task’s samples.

$$\mathcal{L}_{\text{CSC}} = \sum_{n=1}^N \frac{1}{N_{y_n} - 1} \sum_{j=n, y_j=y_n}^N \log \frac{\exp\left(\frac{h_n^{(t)} \cdot h_j^{(t)}}{\tau}\right)}{\sum_{k=1, y_k \neq y_n}^N \exp\left(\frac{h_n^{(t)} \cdot h_k^{(t)}}{\tau}\right)} \quad (3.7)$$

Where N_{y_n} is the number of samples in the data batch that have the same label as y_n , while $h_j^{(t)}$ is the masked output of the TSM for the j^{th} sample in the batch, derived from the layer before the classification head of the task t model.

3.2.5 Total Loss

We combine the three losses described above to optimize our CL model in the TIL setting, as follows:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{CED}} + \mathcal{L}_{\text{CSC}} \quad (3.8)$$

where L_{CE} denotes the cross-entropy loss, L_{CED} represents the contrastive ensemble distillation loss calculated using formula 3.6, and L_{CSC} indicates the contrastive supervised learning of the current task as expressed in formula 3.7.

Equal weights across the three loss terms are justified on three grounds. **(1) Scale compatibility.** Empirically, L_{CE} , L_{CED} and L_{CSC} sit on comparable numerical scales throughout training, their mean values differ by less than a factor of 2, so treating them equally is a reasonable default. **(2) Avoiding over-regularization.** Preliminary experiments with $\alpha, \beta \in \{0.1, 0.5, 1, 2\}$ showed that increasing $\beta > 1$ causes the model to over-distil from past tasks, reducing plasticity on new tasks (accuracy drop of $\sim 1.5\%$ on the first domain). A sensitivity analysis is provided in Table 3.4. **(3) Simplicity.** Equal weighting reduces hyperparameter search cost, an important practical consideration in continual learning where each hyperparameter sweep requires re-training over the entire task sequence.

3.2.6 Experimental Results

We conducted experimental evaluations comparing existing non-continual learning and task-incremental continual learning models for a sequence of ASC tasks on benchmark datasets. The reported results are averaged across five random task orders.

3.2.6.1 Experimental Datasets

Following the evaluation protocol of [56], we report both Average Accuracy and Macro-F1 to account for class imbalance across the 19 ASC tasks. We employ four benchmark

Table 3.2: Number of sentences per dataset, referred to as tasks in the context of continual learning [56]

Data source	Task/domain	Train	Valid.	Test
Liu3domain	Speaker	352	44	44
	Router	245	31	31
	Computer	283	35	36
HL5domain	Nokia6610	271	34	34
	Nikon4300	162	20	21
	Creative	677	85	85
	CanonG3	228	29	29
	ApexAD	343	43	43
Ding9domain	CanonD500	118	15	15
	Canon100	175	22	22
	Diaper	191	24	24
	Hitachi	212	26	27
	Ipod	153	19	20
	Linksys	176	22	23
	MicroMP3	484	61	61
	Nokia6600	362	45	46
	Norton	194	24	25
SemEval14	Rest.	3452	150	1120
	Laptop	2163	150	638

Table 3.3: Average Accuracy (Acc.) and Macro-F1 (MF1) over five random sequences of 19 tasks.

Scenarios	Category	Model	Acc.(%)	MF1(%)
Non-continual Learning (SDL)	BERT	MTL	91.91	88.11
	BERT	SDL	85.84	76.35
	BERT (Frozen)	SDL	78.14	58.13
	Adapter-Bert	SDL	85.96	78.07
	W2V	SDL	77.01	51.89
Continual Learning	BERT	NFH	49.60	43.08
	BERT (Frozen)	NFH	85.51	76.64
	Adapter-BERT	NFH	85.51	76.64
	W2V	NFH	82.69	73.56
		L2	56.04	38.40
		A-GEM	86.06	78.44
		DER++	84.27	75.08
		KAN	85.49	77.38
	BERT (frozen)	SRK	84.76	78.52
		EWC	86.37	74.52
		UCL	83.89	74.82
		OWM	87.02	79.31
		HAT	86.74	78.16
		CAT	83.68	68.64
		L2	63.97	52.43
		A-GEM	45.88	28.21
		DER++	47.63	35.54
		EWC	56.30	49.58
		UCL	64.46	36.64
		OWM	72.99	66.51
	Adapter-BERT	HAT	86.14	78.52
		BCL	88.29	81.40
		LAMOL	88.91	80.59
		CTR	89.47	83.62
		CTR*	88.21	81.19
		Ours	88.50	82.35

Table 3.4: Ablation experimental results.

Model	Acc.(%)	MF1(%)
Ours	88.50	82.35
Without CL-plugin	85.83	79.75
Without CSC+CED	86.48	79.32
Without CSC	87.52	79.93
Without CED	87.98	80.10

datasets, namely: (1) HL5Domains [49], comprising review sentences for 5 products; (2) Liu3Domains [75], focusing on 3 products; (3) Ding9Domains [27], covering 9 products; and (4) SemEval14 [51], relating to 2 products. These datasets collectively encompass sentiments about 19 aspects, each treated as an individual aspect-based sentiment classification task. Detailed statistics for each dataset are provided in Table 3.2.

In alignment with previous studies, we partitioned 10% of the original data for validation and another 10% for testing, for each dataset (1), (2), and (3). However, for dataset (4), we opted for a validation set consisting of only 150 examples from the training set. To maintain methodological consistency with prior research, we present a comprehensive breakdown of the sample numbers for each task in Table 3.2.

3.2.6.2 Experiment Setting

Our approach utilises a two-layer fully connected network with 768 dimensions in the capsule layers and three transfer capsules. Concerning the task-specific modules, we employ 2000 dimensions for the final states. To optimize our model, we use the Adam optimizer and set the learning rate to $3e - 5$. We conduct training for ten epochs on the SemEval datasets and extend it to 30 epochs for other datasets. For all the baseline models, we utilize the code provided by their respective authors and adapt it for classification purposes.

3.2.6.3 Results and Analysis

Firstly, we employ multi-task learning, an approach that incorporates comprehensive knowledge of all preceding tasks and is trained on the entire dataset, to evaluate the upper bound of this problem. Then, we employ non-continual learning baselines independently

for each task. We have three baselines of such, namely BERT, BERT(frozen), and Adapter-BERT. Continuously, we set the foundation based on "no forgetting handling" (NFH). To this end, we compare our model against 12 state-of-the-art continual learning models in the TIL setting, including KAN [55], HAT [105], CAT [54], UCL [1], EWC, L2 [63], OWM [134], A-GEM [17], DER++ [12], BCL [59], LAMOL [113].

As shown in Table 3.3, our model achieves an Average Accuracy of 88.50% and a Macro-F1 of 82.35%, attaining the highest Macro-F1 score among all evaluated continual learning methods. While LAMOL [113] reports a higher accuracy (88.91%), this advantage stems from its generative language modelling objective (GPT-2 backbone), which enables pseudo-rehearsal by generating synthetic task data, a fundamentally different strategy from our distillation-based approach and one that incurs substantially higher computational cost at generation time. Similarly, CTR [56] achieves 89.47% accuracy when using the original task ordering, but drops to 88.21% under the same randomised ordering used in our experiments (CTR*), falling below our model on both metrics. These results confirm that our method provides a stronger and more reliable trade-off between accuracy and class-level fairness (as measured by Macro-F1) within the Adapter-BERT family of methods.

3.2.6.4 Ablation Study

We conducted experiments by removing specific components from our method, and the results in Table 3.4 highlight the effectiveness of the CL-plugin, CED, and CSC. The outcomes, averaged across two evaluation metrics (accuracy and mF1) from 5 random task orders, demonstrate notable improvements. Specifically, when the CL-plugin is omitted, a two-layer fully connected adapter is employed. The results reveal that the inclusion of the CL-plugin enhances the model’s performance by approximately 2.6% in both accuracy and mF1 score. Additionally, Table 3.4 illustrates the positive impact of incorporating two contrastive learning-based losses (CED and CSC) on the continual ASC performance of our model. We also conducted experiments with our model, incorporating another contrastive learning-based loss known as the Knowledge Sharing Loss (CKS), inspired by [58]. However, experimental results revealed that CKS did not yield any improvement. This could be attributed to the fact that our model already includes the KSM module, which facilitates knowledge sharing, as in CKS.

3.3 Exemplar-Free Knowledge Retention via Analytic Learning on Frozen Representations

While Section 3.2 transfers knowledge through ensemble distillation from stored model snapshots, this section investigates a complementary strategy: retaining all knowledge in a frozen pre-trained backbone and computing closed-form optimal classifiers analytically, thereby eliminating both exemplar storage and iterative gradient updates entirely. This analytic approach guarantees exact weight-space solutions at each incremental step, achieving a qualitatively different form of stability compared to gradient-based distillation.

3.3.1 Related Work

Generalized Analytic Continual Learning (GACL) [144] is an exemplar-free method specifically designed for the Generalized Class-Incremental Learning (GCIL) scenario, where training data across tasks may contain overlapping classes and unevenly distributed samples, the architecture of GACL is shown in Fig. 3.6. GACL builds upon the foundation of Analytic Continual Learning (ACL), which reformulates the continual learning objective as a regularized least squares problem and computes the closed-form optimal solutions by directly solving the equation where the derivative equals zero, rather than relying on iterative gradient descent optimization. This analytical approach eliminates the need to store exemplars from previous tasks, thereby addressing data privacy concerns and reducing memory overhead. By maintaining and recursively updating a correlation matrix across tasks, GACL is able to incorporate knowledge from all previously seen tasks without explicitly accessing their raw data. This makes GACL a competitive exemplar-free baseline in the GCIL setting, outperforming other memory-free methods such as LwF, L2P, and DualPrompt by a considerable margin on standard benchmarks. Although GACL [144] demonstrates promising results as an exemplar-free method for GCIL, it applies the analytic learning classifier directly on the original feature space without any non-linear transformation, limiting its ability to handle linearly non-separable class distributions. This restriction becomes particularly pronounced in complex real-world scenarios such as CIFAR-100 under the Si-Blurry setting, where class boundaries are intricate and overlapping. Motivated by this observation, we propose ZeroRFF, which integrates Random Fourier Features (RFF) into the analytic continual learning framework to map extracted features into a higher-dimensional space where data becomes approximately linearly separable. This is simple yet effective augmentation preserves

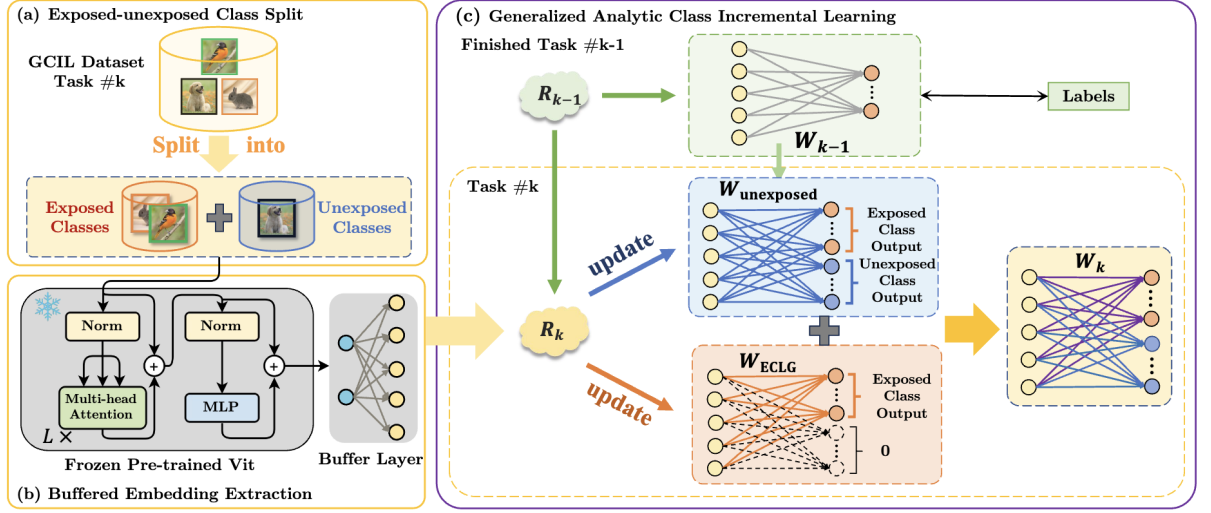


Figure 3.6: Overview of GACL [144]

the exemplar-free and closed-form properties of analytic learning while substantially enhancing the model’s classification capability, bridging the performance gap between memory-free and replay-based methods.

3.3.2 Preliminary

Generalized Class Incremental Learning

Class-Incremental Learning (CIL) is a common scenario in continual learning, in which the model learns new classes in stages, assuming that classes in each task are completely disjoint and that the number of samples per task is fixed, as illustrated in Fig. 3.7. However, this assumption often does not accurately reflect real-world scenarios, in which task data may include both new and previously encountered classes, and the number of samples per class varies across tasks.

Generalized Class-Incremental Learning (GCIL) is proposed as an extended scenario of CIL, addressing the limitations of CIL by allowing the model to handle data with characteristics more similar to reality. In GCIL, training data in a task may contain previously learned classes mixed with new classes, and the number of samples in each task is not fixed but changes flexibly. GCIL better reflects real-world scenarios, where data is not neatly organized into separate tasks but often appears in a mixed and unpredictable manner.

The main difference between GCIL and CIL lies in flexibility and the ability to handle complex data. While CIL requires clearly divided tasks, GCIL allows the model to learn under conditions closer to reality, where data continuously changes and does not

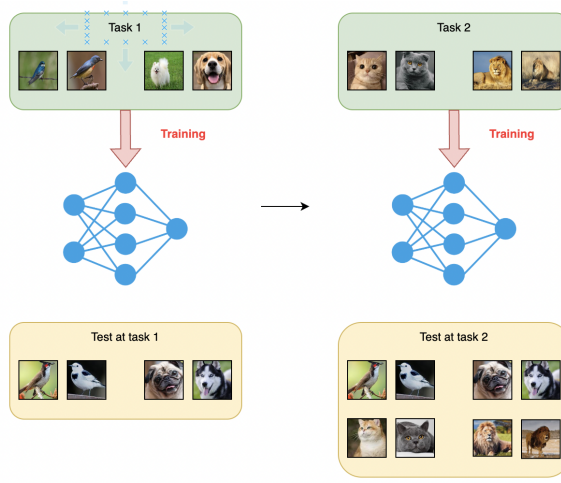


Figure 3.7: Illustration of Class-Incremental Learning (CIL)

follow fixed boundaries. However, the complexity of GCIL also increases the challenge of *catastrophic forgetting*, especially when minority classes may be overlooked during training due to uneven data distribution.

Specifically, the continual learning problem includes a sequence of continuous tasks $\mathcal{D} = \{\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_T^{\text{train}}\}$, where $\mathcal{D}_t^{\text{train}} = \{x_i, y_i\}_{i=1}^{N_t}$ is the training dataset of the t -th task, and T is the total number of tasks.

The training dataset $\mathcal{D}_t^{\text{train}}$ consists of N_t data pairs, y_i is the label of data x_i and y_i is a one-hot vector of size $|\mathcal{C}|$, where $\mathcal{C}$ is the label set. x_i is an input image with dimensions $c \times w \times h$. The goal of GCIL in the k -th task is to train a model using the dataset $\mathcal{D}_k^{\text{train}}$ and evaluate the performance of that model on the test dataset $\mathcal{D}_{1:k}^{\text{test}}$. Here, $\mathcal{D}_{1:k}$ denotes the merged dataset including tasks from 1 to k .

The Generalized Class-Incremental Learning (GCIL) scenario simulates real-world continual learning, where data distribution in terms of quantity and type can vary between tasks. In this scenario, the model is trained on a set of tasks, where each task may include classes belonging to the set $\mathcal{S}$, with a total number of classes being N . The number of samples of different classes in each task k is represented by a random vector $\mathbf{c}_k \in \mathbb{R}^N$. With $c_{k,i}$ being the i -th component of vector $\mathbf{c}_k$, representing the number of samples of class i in task k .

Analytic Continual Learning

Analytic Continual Learning (ACL) is an emerging branch in the field of Continual Learning, developed based on the foundation of Analytic Learning research [35]. Unlike traditional deep learning methods, which primarily rely on gradient descent algorithms

to find optimal solutions for loss functions, ACL has a completely different approach. Specifically, instead of using iterative steps to adjust model parameters through gradients, ACL seeks to directly solve the equation where the derivative equals zero to determine the optimal solution. The ACIL method [142] transforms the continual learning problem into a least squares regression problem, eliminating the need for sample storage by maintaining correlation matrices. RanPAC [83] applies this technique to pre-trained models. GKEAL [143] focuses on continual learning scenarios with few samples (few-shot CIL) by using a Gaussian kernel projection. ACL is an emerging competitive branch in the field of continual learning.

Random Fourier Features Random Fourier Features (RFF) approximate a shift-invariant kernel, most commonly the Gaussian (RBF) kernel, by an explicit, low-dimensional random feature map, avoiding the $O(n^2)$ cost of an exact kernel matrix. For an input $x \in \mathbb{R}^d$, RFF draws random projections $\Omega \sim \mathcal{N}(0, \sigma^{-2}I) \in \mathbb{R}^{D \times d}$ and phase offsets $b \sim \text{Uniform}(0, 2\pi)^D$, and defines

$$\phi(x) = \sqrt{\frac{2}{D}} \cos(\Omega x + b) \in \mathbb{R}^D, \quad k(x, y) \approx \phi(x)^\top \phi(y). \quad (3.9)$$

Two properties make RFF attractive for exemplar-free continual learning: (i) the mapping $\phi(\cdot)$ is fixed and task-agnostic, so it introduces no additional forgetting; and (ii) it lifts a frozen backbone’s embeddings into a higher-dimensional, approximately-linear-separable space, which improves the quality of the closed-form classifier obtained via the RLS update.

Kernel function

Kernel functions play an important role in solving nonlinear classification and regression problems. To handle data that is not linearly separable, we need a transformation to map the original data to a new space where the data becomes linearly separable. This transformation often creates data in a space with higher dimensions, even infinite dimensions, compared to the original data. Direct computation of these functions can cause memory and computational performance issues. Instead of directly computing each data point in the new space, an efficient approach is to use kernel functions to describe the relationship between any two data points in the new space. Some popular kernel functions widely used in machine learning applications include:

- **Linear Kernel:** $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$. This is the basic case, suitable when data can be linearly separated.
- **Polynomial Kernel:** $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top \mathbf{y} + c)^d$, where $c \geq 0$ and d is the degree of

the polynomial. This function allows modeling nonlinear relationships between features.

- **Radial Basis Function (RBF) Kernel:** $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$, where $\gamma > 0$. RBF is one of the most popular kernel functions, particularly effective in handling nonlinear data.
- **Sigmoid Kernel:** $K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x}^\top \mathbf{y} + c)$, with parameters κ and c chosen appropriately. This function originates from neural networks and can model nonlinear relationships.

3.3.3 A Random Fourier Features and Analytic Learning Method for Class Incremental Learning

3.3.3.1 Overall Architecture

Fig. 3.8 illustrates the architecture of our proposed method ZeroRFF, for Generalized Class-Incremental Learning (GCIL), which combines transformer-based feature extraction with analytic learning mechanisms to enable fast learning and uses a Fourier Features Module to enhance the model’s classification capability for sequential task learning.

The process begins with the input of task k , represented as $\mathcal{D}_k^{\text{train}} = \{X_k^{\text{train}}, Y_k^{\text{train}}\}$, containing training samples and their corresponding labels. The methodology is organized into four distinct phases, each serving a specific purpose in the continual learning pipeline.

Feature Extraction Phase: The input data first passes through a pre-trained DeiT-S/16 backbone network with fixed parameters Θ_{backbone} . This feature extractor, which has been pre-trained on ImageNet, transforms the raw input images into high-level feature representations $X_k^{(E)} = f_{\text{backbone}}(X_k^{\text{train}})$. The backbone remains frozen throughout the training process to prevent catastrophic forgetting at the feature level and ensure consistent representations across different tasks.

Kernel Mapping Phase: The extracted features $X_k^{(E)}$ are then transformed using Random Fourier Features (RFF) to obtain $X_k^{(B)} = \text{RFF}(X_k^{(E)})$. This transformation maps the input features into a higher-dimensional space where the data becomes approximately linearly separable. The RFF transformation is defined as $z(x) = \sqrt{\frac{2}{D}} \cos(\omega^T x + \beta)$, where $\omega \sim \mathcal{N}(0, \sigma^{-2}I)$ and $\beta \sim \text{Uniform}(0, 2\pi)$. With dimension $D = 5000$, this approximation effectively captures the behavior of the RBF kernel while avoiding the computational overhead of explicit kernel matrix calculations.

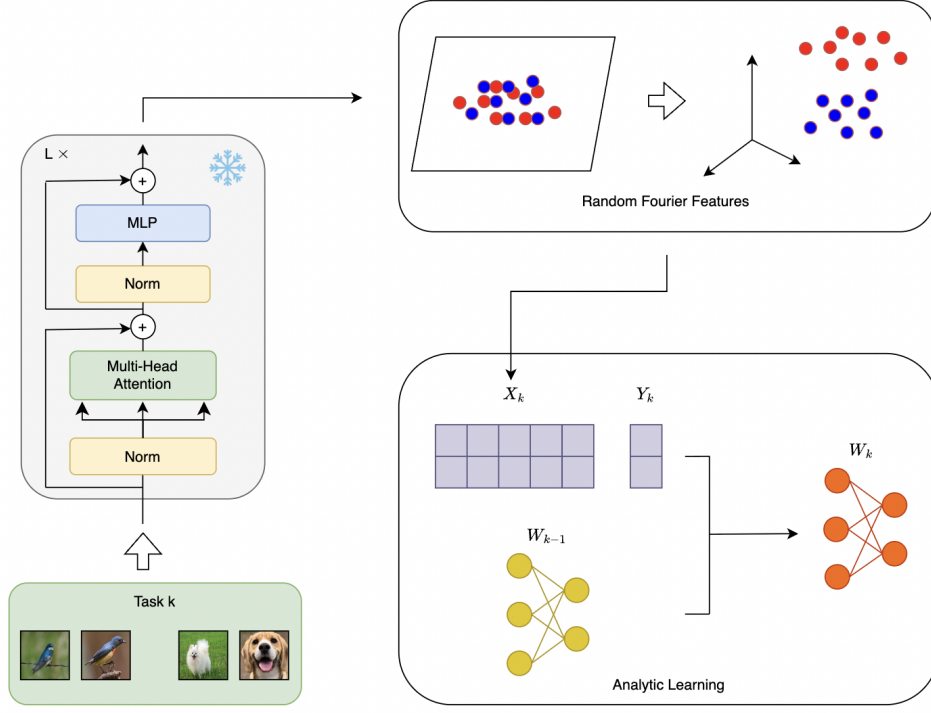


Figure 3.8: Overall architecture of ZeroRFF

Exemplar-Free Learning Phase: The transformed features enter the analytic learning component, which represents the core innovation of our approach. Unlike traditional gradient-based optimization methods, this phase employs direct matrix solutions to find optimal parameters. The process maintains two key matrices: the correlation matrix R_k and the weight matrix $\hat{W}^{(k)}$. The correlation matrix is updated using the Sherman-Morrison formula for efficient recursive computation, while the weight matrix is updated through a direct analytical solution. Crucially, this phase requires no storage of previous task exemplars, addressing privacy concerns and reducing memory requirements.

Task Management Phase: After processing each task, the system checks whether all tasks have been completed. If additional tasks remain, the process increments the task counter ($k = k + 1$) and loads the next task dataset $\mathcal{D}_{k+1}^{\text{train}}$, creating a seamless loop for sequential task processing. Upon completion of all tasks, the methodology outputs the final model $\hat{W}_{\text{FCN}}^{(K)}$, which is ready for inference on the complete set of learned classes.

3.3.4 Algorithm Description

Algorithm 3 presents the complete ZeroRFF training procedure, which implements an exemplar-free approach to Generalized Class-Incremental Learning. The algorithm is

designed to learn from a sequence of tasks without storing previous task data, addressing both privacy concerns and memory constraints inherent in traditional continual learning approaches.

Input and Initialization: The algorithm takes as input a sequence of training tasks $\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_K^{\text{train}}$, where each task $\mathcal{D}_k^{\text{train}}$ consists of input samples X_k^{train} and their corresponding labels Y_k^{train} . Additionally, the algorithm requires a pre-trained backbone model with fixed weights Θ_{backbone} , typically a DeiT-S/16 transformer pre-trained on ImageNet. The initialization phase establishes two critical matrices: the correlation matrix $R_0 = \gamma I$, where γ is the regularization parameter, and the weight matrix $\hat{W}_{\text{FCN}}^{(0)} = 0$.

Sequential Task Processing: The core of the algorithm operates through a sequential loop over all tasks from $k = 1$ to K . For each task, the algorithm performs four essential operations that collectively enable exemplar-free continual learning.

The first operation (Line 3) involves feature extraction using the pre-trained backbone: $X_k^{(E)} \leftarrow f_{\text{backbone}}(X_k^{\text{train}}, \Theta_{\text{backbone}})$. This step transforms raw input images into high-level feature representations while maintaining consistency across tasks through the use of frozen pre-trained weights. The feature extraction process ensures that the input data is represented in a meaningful semantic space suitable for subsequent kernel transformations.

The second operation (Line 4) applies the Random Fourier Features transformation: $X_k^{(B)} \leftarrow \text{rff}(X_k^{(E)})$. This transformation maps the extracted features into a higher-dimensional space where the data becomes approximately linearly separable. The RFF transformation is crucial for enabling the analytic learning approach, as it provides a finite-dimensional approximation to the infinite-dimensional feature space induced by the RBF kernel.

Analytic Learning Updates: The third and fourth operations (Lines 5-6) implement the core analytic learning mechanism through recursive matrix updates. The correlation matrix update (Line 5) employs the Sherman-Morrison formula:

$$R_k \leftarrow R_{k-1} - R_{k-1} X_k^{(B)T} (I + X_k^{(B)} R_{k-1} X_k^{(B)T})^{-1} X_k^{(B)} R_{k-1} \quad (3.10)$$

This update maintains the inverse of the accumulated covariance matrix without explicitly storing or accessing previous task data. The Sherman-Morrison formula enables efficient rank-one updates to matrix inverses, making the computation tractable even for high-dimensional feature spaces.

The weight matrix update (Line 6) computes the optimal parameters using a direct analytical solution:

$$\hat{W}_{\text{FCN}}^{(k)} \leftarrow \hat{W}_{\text{FCN}}^{(k-1)} - R_{k-1} X_k^{(B)T} X_k^{(B)} \hat{W}_{\text{FCN}}^{(k-1)} + R_k X_k^{(B)T} Y_k^{\text{train}} \quad (3.11)$$

This recursive formulation ensures that the weight matrix incorporates information from all previous tasks while being updated solely based on the current task data. The direct analytical nature of this update eliminates the need for iterative optimization procedures, resulting in faster convergence and more stable learning dynamics.

Computational Efficiency: The algorithm’s computational complexity is dominated by matrix operations involving $D \times D$ matrices, where $D = 5000$ is the RFF dimension. The Sherman-Morrison update requires $O(D^2)$ operations per task, while the weight matrix update scales as $O(D \cdot C)$, where C is the number of classes. This complexity is significantly lower than approaches that require storing and processing exemplars from previous tasks.

Memory Efficiency: A key advantage of Algorithm 3 is its constant memory footprint concerning the number of tasks. The algorithm maintains only the correlation matrix R_k and the weight matrix $\hat{W}_{\text{FCN}}^{(k)}$, both of which have fixed sizes independent of the number of previously encountered tasks. This stands in stark contrast to replay-based methods, which require memory that grows linearly with the number of tasks.

Theoretical Foundation: The algorithm is grounded in the principle of analytic learning, which seeks to find closed-form solutions to optimization problems. By formulating the continual learning objective as a regularized least squares problem and employing the RFF approximation, the algorithm can compute exact solutions without iterative optimization. This theoretical foundation provides convergence guaranties and eliminates the hyperparameter tuning typically required for gradient-based methods.

The algorithm concludes by returning the final weight matrix $\hat{W}_{\text{FCN}}^{(K)}$, which encapsulates all learned knowledge from the sequence of tasks. This final model can then be used for inference on any class from any of the encountered tasks, effectively achieving the goal of continual learning without catastrophic forgetting for recursive updates throughout the learning process. The recursive nature of the updates, combined with the direct analytical solutions, enables the methodology to maintain performance across tasks without requiring access to historical data, making it particularly suitable for scenarios where data privacy and storage efficiency are paramount concerns.

Algorithm 3 ZeroRFF Training Algorithm

Require: Tasks $\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_K^{\text{train}}$ with $\mathcal{D}_k^{\text{train}} \sim \{X_k^{\text{train}}, Y_k^{\text{train}}\}$; Pre-trained model with fixed weights Θ_{backbone}

Ensure: Weight matrix $\hat{W}_{\text{FCN}}^{(K)}$

- 1: **Initialize:** $R_0 \leftarrow \gamma I, \hat{W}_{\text{FCN}}^{(0)} \leftarrow 0$
 - 2: **for** $k = 1$ to K **do**
 - 3: $X_k^{(E)} \leftarrow f_{\text{backbone}}(X_k^{\text{train}}, \Theta_{\text{backbone}})$ ▷ Feature extraction
 - 4: $X_k^{(B)} \leftarrow \text{rff}(X_k^{(E)})$ ▷ Apply Random Fourier Features
 - 5: $R_k \leftarrow R_{k-1} - R_{k-1} X_k^{(B)T} (I + X_k^{(B)} R_{k-1} X_k^{(B)T})^{-1} X_k^{(B)} R_{k-1}$ ▷ Update correlation matrix
 - 6: $\hat{W}_{\text{FCN}}^{(k)} \leftarrow [\hat{W}_{\text{FCN}}^{(k-1)} - R_{k-1} X_k^{(B)T} X_k^{(B)} \hat{W}_{\text{FCN}}^{(k-1)} + R_k X_k^{(B)T} Y_k^{\text{train}}]$ ▷ Update weight matrix
 - 7: **end for**
 - 8: **return** $\hat{W}_{\text{FCN}}^{(K)}$
-

Three properties of Algorithm 3 are worth flagging. **(1) Exact recursive equivalence.** The Sherman-Morrison update in Line 5 is algebraically equivalent to recomputing the closed-form solution on all data seen so far, but requires only $O(D^2)$ operations per task rather than $O(N_{\text{total}} D^2)$ for full recomputation, where N_{total} grows with the number of tasks. **(2) No gradient optimization.** Unlike CL-plugin (Section 3.2), ZeroRFF requires no backward pass, eliminating the risk of task interference through gradient updates entirely. **(3) Constant memory footprint.** The algorithm maintains only two fixed-size matrices ($R_k \in \mathbb{R}^{D \times D}$ and $\hat{W}^{(k)} \in \mathbb{R}^{D \times C}$) regardless of the number of tasks K , in contrast to replay-based methods whose memory grows as $O(K)$. The primary computational cost is the matrix inversion in Line 5, which costs $O(D^2 N_k)$ per task using the efficient block-wise formula.

3.3.5 Experimental Results

3.3.5.1 Experimental Settings

We conducted experiments on a popular dataset in the fields of machine learning and computer vision: CIFAR-100 [66]. We evaluated under the Si-Blurry scenario [87], with 5 different seed values to ensure stability and reliability of the results. In the Si-Blurry setting, data is organized to reflect complex real-world scenarios where classes are not completely separated between tasks, and there is label overlap. Specifically, the disjoint

class ratio (*disjoint class ratio* (r_d)) was set to 50%, and the blurry sample ratio (*blurry sample ratio* (r_b)) was set at 10%.

3.3.5.2 Results and discussion

On the CIFAR-100 dataset, the proposed ZeroRFF method excels in the memory-free scenario, achieving $A_{\text{AUC}} = 60.48 \pm 4.33$, $A_{\text{AVG}} = 58.55 \pm 6.05$, and $A_{\text{Last}} = 72.44 \pm 0.13$. Compared to the second-best memory-free method, GACL ($A_{\text{AUC}} = 57.99 \pm 2.46$, $A_{\text{AVG}} = 56.24 \pm 3.12$, $A_{\text{Last}} = 70.31 \pm 0.06$), ZeroRFF improves by approximately 2% across all three criteria. Furthermore, the ZeroRFF method eliminates the dependence on memory storage while still outperforming most memory-based methods. Compared to MVP-R ($A_{\text{AUC}} = 60.62 \pm 1.03$ with memory size 2000), ZeroRFF is only slightly inferior in A_{AUC} , but superior in A_{AVG} and A_{Last} .

Table 3.5: Experimental results on CIFAR-100 (%).

Memory-size	Method	A_{AUC} (%)	A_{AVG} (%)	A_{Last} (%)
2000	EWC++ [63]	53.31 ± 1.70	50.95 ± 1.50	52.55 ± 0.71
	ER [101]	56.17 ± 1.84	53.80 ± 1.46	55.60 ± 0.69
	RM [8]	53.22 ± 1.82	52.99 ± 1.65	55.25 ± 0.61
	MVP-R [87]	60.62 ± 1.03	57.58 ± 0.56	64.30 ± 0.29
500	EWC++ [63]	48.31 ± 1.81	44.56 ± 0.40	40.52 ± 0.83
	ER [101]	51.59 ± 1.91	48.03 ± 0.80	44.09 ± 0.80
	RM [8]	41.07 ± 1.30	38.10 ± 0.59	32.66 ± 0.34
	MVP-R [87]	56.20 ± 1.44	53.61 ± 0.04	55.35 ± 0.43
0	LwF [72]	40.71 ± 1.21	38.49 ± 0.56	27.03 ± 2.92
	L2P [125]	42.68 ± 2.70	39.89 ± 0.45	28.59 ± 3.34
	DualPrompt [124]	41.34 ± 2.59	38.59 ± 0.82	22.74 ± 3.40
	MVP [87]	45.07 ± 1.24	44.93 ± 0.54	39.94 ± 0.47
	SLDA [41]	53.00 ± 3.85	50.09 ± 2.77	61.79 ± 3.81
	GACL [144]	57.99 ± 2.46	56.24 ± 3.12	70.31 ± 0.06
	ZeroRFF(Ours)	60.48 ± 4.33	58.55 ± 6.05	72.44 ± 0.13

Table 3.5 reveals several important patterns. **(1) Memory-free vs. replay comparison.** ZeroRFF (memory = 0) achieves $A_{\text{AUC}} = 60.48$, which is within 0.14 points

of MVP-R with memory size 2000, demonstrating that analytic closed-form learning can match or approach replay-based methods without storing any past data. This is a qualitatively important result: it shows that the performance gap between memory-free and memory-based methods in the Si-Blurry setting is bridgeable through expressive feature transformation. **(2) GACL comparison.** The improvement over GACL (+2.49% AAUC, +2.31% AAVG) is attributable to the RFF mapping: by projecting features into a 5000-dimensional space approximating the RBF kernel, ZeroRFF enables the analytic classifier to separate class boundaries that are non-linearly interleaved in the original DeiT-S/16 feature space. **(3) ALast metric.** The high ALast = 72.44 (vs. 70.31 for GACL) indicates that ZeroRFF retains its advantage even after the full task sequence, a critical property for deployment scenarios where the model is evaluated only after all tasks have been learned.

Table 3.6: Comparison of principles between CL-plugin and ZeroRFF

Criterion	CL-plugin	ZeroRFF
Application domain	NLP (aspect sentiment)	Vision (image CIL)
Scenario	TIL	CIL
Anti-forgetting mechanism	Neuron mask + distillation	Closed-form analytic learning
Requires gradient descent?	Yes	No (analytic/closed-form)

Both methods share the common principle of avoiding exemplar storage by separating the frozen feature-extraction stage from the knowledge-update stage.

3.4 Summary

The two methods developed in this chapter tackle exemplar-free continual learning from different angles: CL-plugin relies on distillation-based knowledge transfer to preserve stability (Section 3.2), whereas ZeroRFF obtains the same stability through closed-form analytic learning over frozen representations (Section 3.3). The former is better suited to tasks with rich inter-task semantic overlap, while the latter is preferable when a strong frozen backbone is available and training efficiency is paramount. This chapter presents two novel approaches for knowledge transfer in continual learning to address the challenge of preserving previously learned information while acquiring new knowledge. Contrastive Learning Approach for Knowledge Distillation introduces a framework with three key components: the Continual Learning Plugin (CL-plugin) for modular integration, Con-

trastive Ensemble Distillation (CED) Loss for maintaining discriminative representations from previous tasks, and Contrastive Supervised Learning of Current Task (CSC) Loss for effective new task learning. These are combined into a unified Total Loss function that balances knowledge retention and acquisition. Knowledge Retention from Pre-trained Models using Feature Alignment addresses knowledge utilization from pre-trained models through Feature Alignment techniques that prevent representation drift. Comprehensive experiments on standard benchmarks demonstrate significant improvements in knowledge retention while maintaining competitive performance on new tasks. Ablation studies and comparison results confirm the effectiveness of individual components and establish superiority over existing state-of-the-art methods. The proposed approaches provide practical, modular solutions for representation-based continual learning that effectively balance knowledge preservation with new learning capacity across various scenarios and architectures.

The contribution of this chapter was published in [\[P2\]](#) and [\[P3\]](#).

Chapter 4

Adaptive Architectures and Dynamic Parameter Allocations

4.1 Introduction

Architecture-based approaches represent one of the three main families of continual learning methods, alongside regularization-based and replay-based techniques. These methods address catastrophic forgetting by dynamically allocating dedicated model capacity for each new task, thereby creating isolated parameter spaces that minimize interference between sequential learning phases. Unlike regularization methods that constrain parameter updates or replay methods that store historical data, architecture-based approaches modify the model structure itself to accommodate new knowledge while preserving old parameters.

Despite their theoretical appeal, traditional architecture-based methods face critical challenges. First, conventional dynamic architecture expansion strategies often result in excessive parameter growth, making them impractical for resource-constrained deployment scenarios. Second, strict parameter isolation while effective at preventing forgetting limits beneficial knowledge transfer between related tasks, potentially hindering the model's ability to leverage shared representations. Third, determining optimal capacity allocation remains challenging, as overallocation wastes computational resources while underallocation restricts the learning capacity for new tasks.

This chapter addresses these limitations through four complementary contributions in adaptive architectures and dynamic parameter allocation in **class-incremental learning**

scenario. First (Section 4.2), we propose a multi-head approach with online prototype equilibrium and adaptive prototypical feedback, which optimizes both feature representations and classification layers through dual mechanisms. Specifically, Online Prototype Equilibrium (OPE) maintains stable and discriminative class representations by balancing prototypes from both current and previously learned classes, while Adaptive Prototypical Feedback (APF) selectively reinforces decision boundaries for class pairs that are prone to misclassification. We further incorporate an energy-based latent alignment mechanism (ELI) to counteract representational drift in the feature space across sequential tasks, ensuring that latent representations of old classes are realigned to their optimal regions after each new task is introduced. Second (Section 4.3), we introduce DAKTA, a continual learning framework that integrates a Vision Transformer (ViT) backbone with Low-Rank Adaptation (LoRA) and a Kolmogorov-Arnold Classifier (KAC) head. DAKTA leverages Incremental Task Arithmetic (ITA) to compose task-specific parameter updates without storing past exemplars, while second-order Fisher Information Matrix regularization is applied to both the backbone and classifier head to protect parameters critical to previous tasks from being overwritten during sequential training. Third (Section 4.4), SO-LoRA is a rehearsal-free continual learning scheme that reconceptualizes low-rank adaptation geometrically by maintaining a fixed orthonormal basis and restricting each task to learn sparse compositions over this stable coordinate system. Group sparsity explicitly reduces subspace overlap between tasks, while a gradient-aware soft mask protects historically important rank dimensions from interference, yielding a constant adapter footprint regardless of the number of tasks encountered.

These methods collectively advance the state-of-the-art in architecture-based continual learning by balancing three critical objectives: (1) maintaining parameter efficiency through controlled capacity expansion, (2) preserving previously acquired knowledge through intelligent parameter protection, and (3) enabling effective knowledge transfer through strategic parameter sharing. Comprehensive experiments on standard benchmarks, including CIFAR-10, CIFAR-100, TinyImageNet, ImageNet-R, ImageNet-A, and CCH5000, demonstrate that our approaches achieve superior performance compared to existing methods while significantly reducing computational overhead.

Note on backbone selection. The four methods presented in this chapter are organised into two methodological lineages that differ in backbone architecture, reflecting two distinct deployment contexts in continual learning research. Sections 4.2 employ a ResNet-18 backbone trained from random initialization, consistent with the standard evaluation protocol for online and exemplar-based CIL benchmarks [17, 98]. Sections 4.3

and 4.4 (DAKTA and SO-LoRA) employ a ViT-B/16 backbone pre-trained on ImageNet-21k, consistent with the emerging paradigm of parameter-efficient fine-tuning (PEFT) for continual learning [111, 124]. These lineages address different practical constraints: the ResNet-18 methods target settings where large pre-trained models are unavailable or domain-inappropriate (e.g., specialised medical imaging), whereas the ViT-B/16 methods leverage the increasingly dominant foundation model paradigm. Accordingly, each method is evaluated against baselines sharing the same backbone (Tables 4.2, 4.3 versus Tables 4.7, 4.11, 4.12), and direct cross-method comparisons within this chapter are not intended.

4.2 A Multi-Head Approach with Online Prototype Equilibrium and Adaptive Prototypical Feedback

In this section, we aim to optimize the quality of hidden feature representations through Online Prototype Equilibrium (OPE) and Adaptive Prototypical Feedback (APF). To enhance the model’s classification capability, during the training of each task, we incorporate two heads: the WP head (to improve task recognition) and the OOD head (to enhance class recognition).

4.2.1 Related Work

Online Prototype Learning (OnPro) [126] is a rehearsal-based The method is designed for online Class-Incremental Learning (CIL), where the model learns from a single-pass data stream without access to task identifiers at inference. The overview of the OnPro framework is illustrated in Fig. 4.1. OnPro introduces a prototype-based representation learning framework that maintains online prototypes for each class and optimizes the embedding space through a contrastive objective. Specifically, OnPro computes online prototypes as the averaged feature representations of samples within each mini-batch and applies a contrastive loss to pull prototypes of the same class closer while pushing those of different classes apart, thereby encouraging the model to learn discriminative and well-separated class representations. By operating at the prototype level rather than the individual sample level, OnPro achieves a more stable and compact representation of class distributions, which effectively alleviates catastrophic forgetting under the constrained memory buffer setting. OnPro demonstrates strong performance on standard benchmarks such as CIFAR-10 and CIFAR-100 across various memory buffer sizes, consistently outperforming prior rehearsal-based methods including OCM, SCR, and

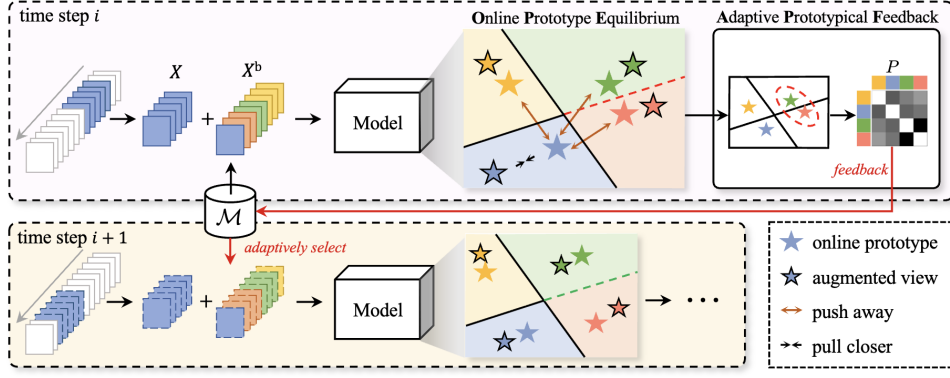


Figure 4.1: Illustration of OnPro framework [126]

DVC, establishing itself as a competitive state-of-the-art baseline in online CIL.

However, OnPro exhibits two notable limitations that motivate the proposed method. First, OnPro relies on a single shared classification head for all classes across all tasks, which conflates the two distinct sub-problems inherent in CIL: within-task prediction (WP), which determines which class within a given task an input belongs to, and task-ID prediction (TP), which identifies which task the input originates from. Without explicitly disentangling these two objectives, the shared head must simultaneously handle inter-task and intra-task discrimination, leading to inter-task class separation (ICS) errors that reduce overall classification accuracy. Second, although OnPro’s contrastive loss globally balances class representations, it treats all class pairs uniformly without accounting for the varying degrees of confusion between different classes. In practice, some class pairs are inherently more prone to misclassification than others, and a uniform treatment of all class pairs may cause the model to underemphasize decision boundaries that are most critical for reducing classification errors.

Motivated by these observations, the proposed method augments the OnPro framework with two complementary components. On one hand, the Adaptive Prototypical Feedback (APF) mechanism is introduced to identify class pairs that are most susceptible to confusion based on the distances between their stored prototypes, and to adaptively oversample these confusing pairs during memory replay, thereby sharpening the decision boundaries where they are needed most. On the other hand, inspired by the theoretical framework of Kim et al. [61], the proposed method replaces the single shared classification head with a multi-head architecture consisting of a Within-task Prediction (WP) head and an Out-of-Distribution (OOD) head for each task. The WP head focuses on fine-grained intra-task classification, while the OOD head enables task-ID prediction by distinguishing in-distribution samples of the current task from out-of-distribution

samples of other tasks. This explicit decomposition of the CIL prediction into WP and TP components allows the model to handle inter-task class separation more effectively, leading to substantial gains in average accuracy over OnPro across all evaluated memory buffer sizes on both CIFAR-10 and CIFAR-100.

4.2.2 Preliminary

Apart from CF, CIL presents another intricate challenge, i.e. the inter-task class separation (ICS). ICS occurs when the learner lacks access to the training dataset of former tasks when learning a new task. Recent studies underpin three conditions necessary and sufficient for a robust and effective CIL model. They are (i) proficient within-task prediction (WP), (ii) proficient task-id prediction (TP) and (iii) excellent out-of-distribution (OOD) detection for each task.

For a training set $D = \{(x^i, y^i)_{i=1}^n\}$ of each task, where n is the number of data samples, $x^i \in \chi$ is an input sample and $y^i \in \gamma$ is its corresponding label. γ is the set of all labels in D and is referred to as the in-distribution (IND). The goal of OOD detection is to construct a function $f : \chi \rightarrow \gamma \cup \{O\}$ that can identify Out-of-Distribution (OOD) instances not belonging to any class in γ . The class of such instances is denoted as O . When the model can effectively recognize OOD instances for each task, it hence addresses the inter-task class separation (ICS) problem [61]. ICS refers to the situation in which the learner encounters challenges in establishing clear boundaries between classes of different tasks. Regarding WP and TP, Kim et al. [61, 62], present a formula for calculating the probability of a test sample belong to a task, which is the product of two probabilities, i.e. WP and TP as in formula 4.1 and 4.2 as below, respectively.

$$\mathbf{P}_{WP} = \mathbf{P}(x \in X_{k,j} | x \in \mathbf{X}_k, D), \quad (4.1)$$

$$\mathbf{P}_{TP} = \mathbf{P}(x \in \mathbf{X}_k | D), \quad (4.2)$$

where $\mathbf{X}_k$ is the dataset of task k , $\mathbf{Y}_k$ is the set of classes for task k and $\mathbf{X}_k = \{\mathbf{X}_{k,j}\}$, j represents the j^{th} class of the k^{th} task. The classes within each task are distinct, i.e., $\mathbf{X}_{k,j} \cap \mathbf{X}_{k,j'} = \emptyset, \forall j \neq j'$. The tasks are also distinct, i.e., $\mathbf{X}_k \neq \mathbf{X}_{k'} = \emptyset, \forall k \neq k'$. In this line, the prediction probability of a CIL model is given by formula 4.3 as follows:

$$\begin{aligned} \mathbf{P}(x \in \mathbf{X}_{k_0,j_0} | D) &= \sum_{k=1, \dots, n} \mathbf{P}(x \in X_{k,j_0} | x \in \mathbf{X}_k, D) \mathbf{P}(x \in \mathbf{X}_k | D) \\ &= \mathbf{P}(x \in X_{k_0,j_0} | x \in \mathbf{X}_{k_0}, D) \mathbf{P}(x \in \mathbf{X}_{k_0} | D), \end{aligned} \quad (4.3)$$

Kim et al. (2023) proposed a replay-based continual learning method for Class Incremental Learning known as ROW [62] (short for Replay, OOD, WP). At the k^{th} task, the system takes in the dataset D_k and makes use of replayed samples retained from earlier tasks in the memory buffer M , treating them as out-of-distribution data for task k in order to train an OOD detection head, while a WP head is tuned in parallel. More specifically, the model associated with each task is composed of two heads: an OOD head used to compute TP, and a WP head. Training updates the parameters ψ_k of the feature extractor f_k , the parameters θ_k of the OOD head h_k , and the parameters ϕ_k of the WP head, which functions analogously to a classification head g_k . Both h_k and g_k operate on the same set of features produced by f_k to derive their respective probabilities.

The training procedure is carried out in three stages: (1) the feature extractor f_k and the OOD head h_k are trained jointly using the IND data from D_k together with the OOD data drawn from the memory buffer; (2) the WP head g_k is fine-tuned for the current task using D_k alone, with the feature extractor f_k kept fixed; and (3) the OOD heads corresponding to all previously learned tasks are fine-tuned. Stages (2) and (3) amount to nothing more than fine-tuning a single classifier layer. The outputs of these two heads are subsequently combined to yield the final CIL prediction probability, as given in formula 4.3.

The architecture uses two complementary losses. **OPE loss** (Online Prototype Equilibrium): Within each mini-batch, OPE balances the contribution of each class by computing a prototype from the current batch and penalising predictions that deviate from the prototype ordering. This addresses the class imbalance problem inherent in online continual learning, where mini-batches may contain unequal numbers of samples from current and past classes. OPE is applied only to the within-task (WP) head because WP prediction requires fine-grained within-class discrimination. **CE loss** (Cross-Entropy): Standard CE is applied to both WP and OOD heads to provide a direct classification signal. The OOD head uses CE with a uniform prior over past-task classes to avoid over-confidence on in-distribution samples from the current task. The combination of OPE and CE is motivated by OnPro [126]: our modification adds the OOD head and APF feedback (Section 4.2.3), which together disentangle within-task and cross-task decision boundaries.

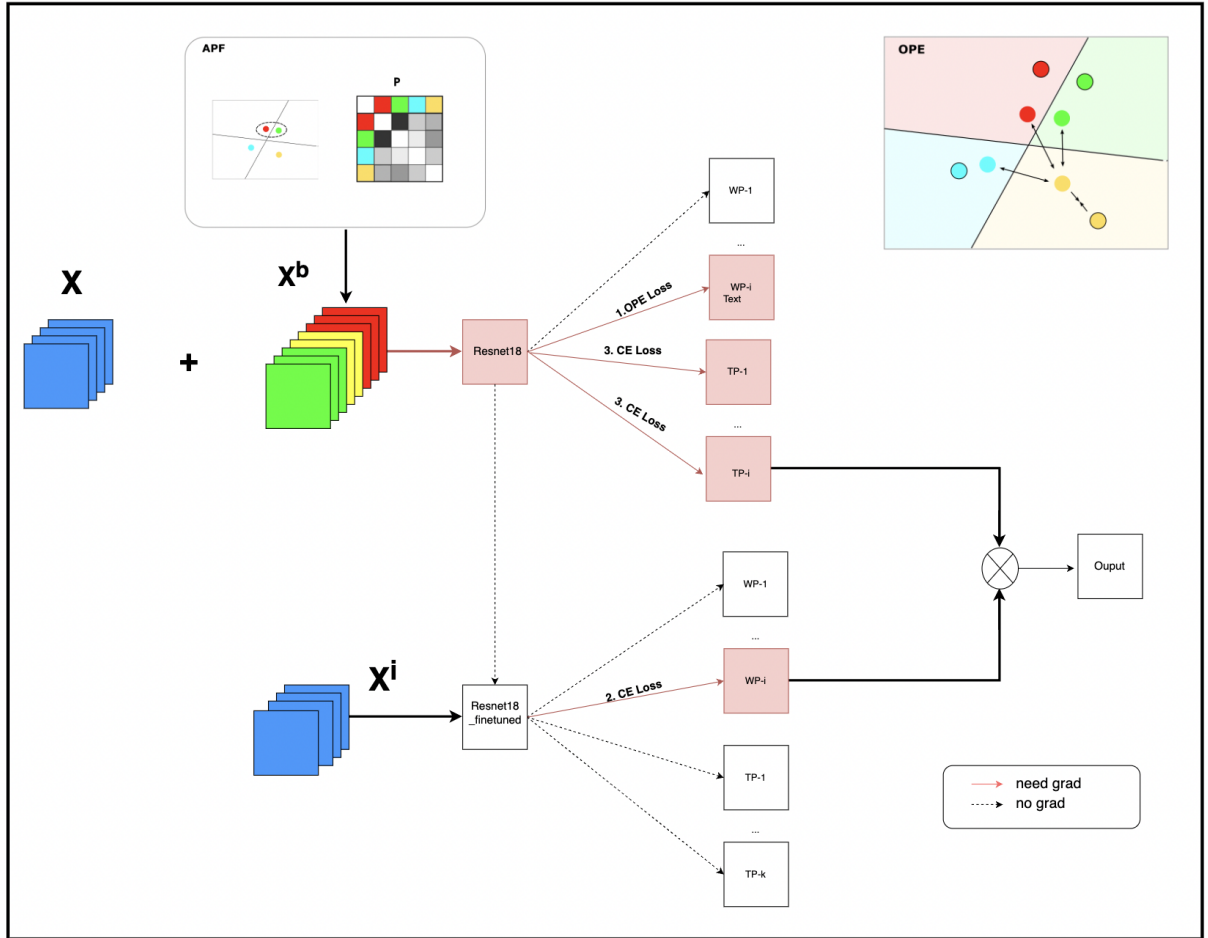


Figure 4.2: The overall architecture of our proposed method.

4.2.3 Feature representation optimization

OPE learns the representative features for each class by **pulling** online prototypes both from the data batch $\mathcal{P}$ and the augmented data batch $\hat{\mathcal{P}}$ that are close to each other in the embedding space (depicted by two arrows pointing towards each other in Figure 4.2). **OPE** enhances the discriminative boundaries between classes by **pushing** online prototypes from other classes away from $\mathcal{P}$ and $\hat{\mathcal{P}}$ in the embedding space (depicted by the bidirectional arrows in Figure 4.2).

In each epoch of task t , the data received in small batches is denoted as X . Our proposed method operates on the entire data batch instead of processing each data sample. Accordingly, we define the **online prototype** of each class within an epoch as the mean representation of all online embeddings belonging to that class in a mini-batch:

$$\mathbf{p}_i = \frac{1}{n_i} \sum_j \mathbf{z}_j \cdot \mathbb{1}\{y_j = i\}, \quad (4.4)$$

where n_i denotes the number of instances belonging to class i within the mini-batch, and $\mathbb{1}$ represents an indicator function. We denote by $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^K$ the collection of K online prototypes computed over X ($K = |\mathcal{P}| \leq |\mathcal{C}_t|$). Analogously, $\mathcal{P}^b = \{\mathbf{p}_i^b\}_{i=1}^{K^b}$ denotes the corresponding set of online prototypes computed over X^b ($K = |\mathcal{P}| \leq |\mathcal{C}_t|$ and $K^b = |\mathcal{P}^b| \leq \sum_{i=1}^t |\mathcal{C}_i|$), where $\mathcal{C}$ refers to the set of classes present in the data batch.

OPE adjusts the network parameters to optimize the two aforementioned training phases by minimizing the contrastive loss function given by the formula:

$$l(\mathcal{P}, \hat{\mathcal{P}}) = \frac{-1}{|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} \log \frac{\exp(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_i}{\tau})}{\sum_j \exp(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_j}{\tau}) + \sum_{j \neq i} \exp(\frac{\mathbf{p}_i^T \mathbf{p}_j}{\tau})}, \quad (4.5)$$

where τ denotes the temperature hyperparameter, and both $\mathcal{P}$ and $\hat{\mathcal{P}}$ have been l_2 normalized. The contrastive loss, summed over all positive pairs from both $(\mathcal{P}, \hat{\mathcal{P}})$ and $(\hat{\mathcal{P}}, \mathcal{P})$, is expressed as follows:

$$\mathcal{L}_{\text{pro}}(\mathcal{P}, \hat{\mathcal{P}}) = \frac{1}{2} [l(\mathcal{P}, \hat{\mathcal{P}}), l(\hat{\mathcal{P}}, \mathcal{P})], \quad (4.6)$$

To this end, **OPE** adjusts the model parameters to effectively learn both discriminative and distinctive features for both new and previously learned classes using $\mathcal{L}_{\text{pro}}$, given by the formula:

$$\mathcal{L}_{\text{OPE}} = \mathcal{L}_{\text{pro}}^{\text{new}}(\mathcal{P}, \hat{\mathcal{P}}) + \mathcal{L}_{\text{pro}}^{\text{seen}}(\mathcal{P}^b, \hat{\mathcal{P}}^b), \quad (4.7)$$

where $\mathcal{L}_{\text{pro}}^{\text{new}}$ is responsible for acquiring knowledge from the new classes, while $\mathcal{L}_{\text{pro}}^{\text{seen}}$ serves to retain the knowledge accumulated from all previously encountered classes. *This*

dynamic resembles a zero-sum game, where OPE seeks to strike a balance that turns it into a mutually beneficial one. Specifically, as the model learns, the knowledge from a new class is gathered and added to prototypes in the memory bank $\mathcal{M}$, causing $\mathcal{L}_{\text{pro}}^{\text{new}}$ to gradually shift towards a balanced state that separates all known classes from each other better, including the new one. This adaptation is crucial for minimizing CF and aligns with the goals of Class Incremental Learning.

Although OPE achieves balance at a global level, it treats all classes equally, whereas in practice classes exhibit varying degrees of inter-class confusion. Consequently, the model should focus more on classes that are more likely to be confused with one another. To address this, we introduce APF, which leverages feedback from online prototypes to identify classes prone to misclassification and strengthen their decision boundaries accordingly. Specifically, for every pair of classes in the memory bank $\mathcal{M}$, APF calculates the distance between their online prototypes across all previously seen classes, using this distance as a proxy for confusion — the smaller the distance between two prototypes, the greater their likelihood of being confused. APF then converts this prototype distance matrix into a probability distribution P over classes by applying a symmetric Gaussian kernel, formulated as:

$$P_{i,j} \propto \exp(-\|\mathbf{p}_i^b - \mathbf{p}_j^b\|_2^2) \quad (4.8)$$

where $i, j \in 1, \dots, |\mathcal{P}^b|$ and $i \neq j$. These probabilities are then normalized so that they form a valid probability mass function summing to 1. The resulting probabilities over $\mathcal{M}$ are used by APF to inform the subsequent sampling process, thereby sharpening the decision boundaries for classes that are more susceptible to misclassification.

APF is realized through a sampling-based ensemble strategy. In particular, it preferentially draws more instances from classes within $\mathcal{M}$ that are prone to misclassification, mixing them according to the probability distribution P . However, since over-emphasizing these classes risks disrupting the current balanced state of the online prototypes, we adopt a two-stage sampling scheme for constructing the replay data X^b of size m . In the first stage, n_{APF} samples are drawn according to P , such that a larger $P_{a,b}$ leads to more samples being drawn from classes a and b , where $n_{\text{APF}} = \alpha \cdot m$ and α is a ratio that controls the strength of APF. In the second stage, the remaining $m - n_{\text{APF}}$ samples are drawn uniformly at random from the entire memory bank, which prevents the model from over-focusing on confusion-prone classes and thereby preserves the balanced state already established. Algorithm 4 summarizes the overall procedure for computing APF.

Algorithm 4 Adaptive Prototype-guided Fusion (APF) sampling, adapted from [126]

Require: Memory buffer $\mathcal{M}$ and online class prototypes $\{\mathbf{p}_i\}_{i=1}^{K^b}$ accumulated up to the current step.

Ensure: Fused sample set X^b

- 1: **Init:** $\mathcal{S} \leftarrow \emptyset$; set the fusion budget $n_{\text{APF}} = \alpha \cdot m$; for every class pair, derive a pairing probability $P_{i,j}$ from the prototype similarity between $\mathbf{p}_i^b$ and $\mathbf{p}_j^b$
 - 2: **for** each pair probability $P_{i,j} \in P$ **do**
 - 3: Allocate $X_i, X_j \leftarrow \lfloor P_{i,j} \cdot n_{\text{APF}} + 0.5 \rfloor$ instances to classes i and j , respectively
 - 4: Fuse the allocated instances and add them to the pool: $\mathcal{S} \leftarrow \mathcal{S} \cup \text{Mixup}(X_i, X_j)$
 - 5: **end for**
 - 6: Draw X_{base} as the leftover $m - n_{\text{APF}}$ instances, sampled uniformly at random from $\mathcal{M}$
 - 7: Combine the pools: $X^b \leftarrow \mathcal{S} \cup \text{Mixup}(X_{\text{base}}, X_{\text{base}})$
 - 8: **return** X^b
-

Analysis of the APF mechanism. APF (Adaptive Prototypical Feedback) tackles a subtle but important problem in prototype-based classifiers: as the feature extractor continues to be updated, prototypes saved from earlier tasks can gradually fall out of alignment with the model’s current decision boundaries, leading to misclassifications in regions where tasks overlap. To detect this drift, APF compares the predictions made by the WP prototype head against the entropy of the OOD head; a discrepancy between the two signals that a boundary has shifted. Once such a misalignment is identified, APF generates targeted hard negatives close to the affected boundary, forcing the model to re-establish the correct class partition. In terms of computational cost, APF requires one extra forward pass per class per mini-batch to produce these boundary samples, which introduces an $O(|C_t|)$ overhead per iteration. In practice, however, this cost is small enough to be negligible relative to the overall training budget (see Table 4.1).

Table 4.1: Training time per task (minutes) for OnPro and ModifiedOnPro (ours) on CIFAR-10 and CIFAR-100 with memory size $\mathcal{M} = 200$.

Method	Extra passes	CIFAR-10 (min)	CIFAR-100 (min)
OnPro [126]	0	23.8	89.4
ModifiedOnPro (Ours)	$O(C_t)$	25.4	95.1

4.2.4 Classification layer optimization

Building on the probabilistic framework for handling distribution shifts, we decompose the prediction probability as:

$$\mathbf{P}(X_{k,j}|x) = \mathbf{P}(X_{k,j}|x, k)\mathbf{P}(X_k|x), \quad (4.9)$$

This decomposition naturally suggests a two-stage architecture. We therefore introduce two specialized heads at the classifier layer: an OOD Head that estimates task probability (TP), and a WP Head that computes within-task prediction (WP). Our model learns the parameter set $(\Psi_k, \theta_k, \phi_k)$, where Ψ_k parameterizes the OOD head h_k , and ϕ_k parameterizes the WP head. Following this idea, the training pipeline is organized into three consecutive steps:

1. Jointly optimize the feature extractor f_k and the OOD head h_k , using IND samples from D_k together with OOD samples drawn from $\mathcal{M}$.
2. With f_k frozen, adapt the WP head g_k to the current task by fine-tuning on D_k .
3. Update the OOD head so that it remains accurate for all previously seen tasks. Notably, steps 2 and 3 only update the final layer of each head, leaving the feature extractor unchanged.

Training Feature Extractor and OOD Head: The goal of this stage is to jointly learn the feature extractor f_k and the OOD head h_k for task k , since the WP head relies on the same shared extractor (detailed further below); Fig. 1(a) depicts this procedure. Because any sample not belonging to task k is by definition treated as OOD, the current task data D_k is used as the IND set, whereas replayed samples from earlier tasks $k' \neq k$, drawn from the memory buffer $\mathcal{M}$, are assigned to a dedicated OOD category (highlighted in red) within h_k . Optimization of $h_k \circ f_k$ proceeds along two objectives simultaneously: it maximizes $p(y|x, k) = \text{softmax } h_k(f(x, k; \Psi_k); \theta_k)_y$ for the IND samples $x \in D_k$, and it maximizes $p(ood|x, k)$ for the OOD samples $x \in \mathcal{M}$. In practice, this dual objective is implemented as the minimization of a combined cross-entropy loss, defined in formula 4.10.

$$\mathcal{L}_{ood}(\Psi_t, \theta_t) = -\frac{1}{2N} \left(\sum_{(x,y) \in D_k} \log p(y|x, k) + \sum_{(x,y) \in \mathcal{M}} \log p(ood|x, k) \right), \quad (4.10)$$

where N is the number of instances in D_k . We utilize upsampling with the replay instances to achieve an equal number of samples as the current task data D_k . The first loss discriminates IND instances, while the second is introduced to distinguish between IND and OOD instances.

Fine-Tuning the WP Head: Starting from the feature extractor produced in the first stage, we freeze it entirely and update only the WP head g_k (the WP classifier), i.e., only the parameters ϕ_k are trained on D_k . Optimization proceeds by minimizing the cross-entropy objective below.

$$\mathcal{L}_{WP}(\phi_k) = -\frac{1}{N} \sum_{(x,y) \in D_k} \log p(y|x, k). \quad (4.11)$$

The **WP probability** of each class in task k is simply given by the corresponding softmax output.

Fine-Tuning the OOD Heads of All Tasks: A limitation of the OOD head h_k produced in step 1) is that it is imbalanced across tasks: because the samples retained in $\mathcal{M}$ for earlier tasks are less varied, the OOD heads learned for those tasks end up weaker than the ones for later tasks, which benefit from a more diverse set of stored instances in $\mathcal{M}$. To address this imbalance, after each task finishes training we re-tune the OOD heads for all tasks seen so far, using exclusively the replay samples held in $\mathcal{M}$. Once task k has been trained, $\mathcal{M}$ holds replay instances covering every class from task 1 through task k . For every task $k' \leq k$, we build a new IND set $\tilde{D}_{k'}$ by pulling out of $\mathcal{M}$ the instances that belong to task k' , and define a pseudo memory buffer $\tilde{\mathcal{M}}$ as the remainder of $\mathcal{M}$ once those instances have been removed. All OOD heads are then jointly updated by minimizing the loss defined in formula 4.12.

$$\mathcal{L}_{TP}(\theta_{k'}) = -\frac{1}{M} \left(\sum_{(x,y) \in \tilde{\mathcal{M}}} \log p(ood|x, k') + \sum_{(x,y) \in \tilde{D}_{k'}} \log p(y|x, k') \right), \quad (4.12)$$

where M is $|\tilde{D}_k| + |\tilde{\mathcal{M}}|$.

Both components are expressed as log-likelihoods over the same probability space of model $\theta_{k'}$, ensuring their gradient contributions are naturally comparable in magnitude. Equal weighting thus reflects a balanced optimization between retaining knowledge of previous tasks (via OOD regularization on $\hat{\mathcal{M}}$) and learning the current task k' (via task-specific training on $\tilde{D}_{k'}$), directly addressing the stability-plasticity trade-off in continual learning.

4.2.5 Experimental Results

We employ two benchmark datasets for image classification, namely CIFAR-10 and CIFAR-100, to assess the effectiveness of our proposed model, which is based on the online Class Incremental Learning (CIL) approach. We utilize two metrics to measure the performance of our method: average accuracy and average forgetting. Our model achieves the highest effectiveness compared to several other baseline methods in terms of the average accuracy metric, as demonstrated in table 4.2. When using our proposed multi-head method, the model outperforms the Onpro method that uses a single head, with experiments on the Cifar-10 dataset showing an increase of nearly 9% from 52.4% to 63.62% when $\mathcal{M}=100$, and a 6% increase when $\mathcal{M}=200$, and 5% when $\mathcal{M}=500$. The experimental results on the Cifar-100 dataset also show significant improvements, increasing by approximately 7%, 3%, and 1.3% respectively for $\mathcal{M}=500$, $\mathcal{M}=1000$, and $\mathcal{M}=2000$. This indicates that our method improves significantly more when using a smaller memory buffer size.

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL [98]	31	33.9	42	12.8	16.5	17.6
DER++ [12]	31.5	39.7	50.9	16	21.4	23.9
PASS [141]	33.7	33.7	33.7	7.5	7.5	7.5
AGEM [17]	17.7	17.5	17.5	5.8	5.9	5.8
GSS [4]	18.4	19.4	25.2	8.1	9.4	10.1
ER [15]	19.4	20.9	26	8.7	9.9	10.7
MIR [3]	20.7	23.5	29.9	9.7	11.2	13
GDumb [96]	23.3	27.1	34	8.2	11	15.3
ASER [108]	20	22.8	31.6	11	13.5	17.6
SCR [78]	40.2	48.5	59.1	19.3	26.5	32.7
CoPE [24]	33.5	37.3	42.9	11.6	14.6	16.8
DVC [34]	35.2	41.6	53.8	15.4	20.3	25.2
OCM [36]	47.5	59.6	70.1	19.7	27.5	34.4
OnPro [126]	52.4	63	67.7	20.7	27.2	34.9
ModifiedOnPro(Ours)	63.6	69.3	72.8	27.1	30.3	36.3

Table 4.2: Average accuracy of our model and SOTA models with different buffer sizes $\mathcal{M}$

Table 4.3: Average Forgetting of our model and SOTA models with different buffer sizes $\mathcal{M}$.

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL [98]	52.7	49.3	38.3	16.5	11.2	10.4
DER++ [12]	57.8	46.7	33.6	41.0	34.8	33.2
PASS [141]	21.2	21.2	21.2	10.6	10.6	10.6
AGEM [17]	64.8	64.8	64.5	41.7	41.8	41.7
GSS [4]	67.1	65.8	61.2	48.7	46.7	44.7
ER [15]	64.7	62.9	57.5	47.0	46.4	44.7
MIR [3]	62.6	58.5	51.1	45.7	44.2	42.3
GDumb [96]	28.5	28.4	28.1	25.0	23.2	20.7
ASER [108]	64.8	62.6	53.2	52.8	50.4	46.8
SCR [78]	43.2	35.5	24.1	29.3	20.4	11.5
CoPE [24]	49.7	45.7	39.4	25.6	17.8	14.4
DVC [34]	40.2	31.4	21.2	32.0	32.7	28.0
OCM [36]	35.5	23.9	13.5	18.3	15.2	10.8
OnPro [126]	24.2	16.5	13.7	17.8	14.9	10.0
ModifiedOnPro(Ours)	23.2	18.25	13.5	19.24	14.47	11.19

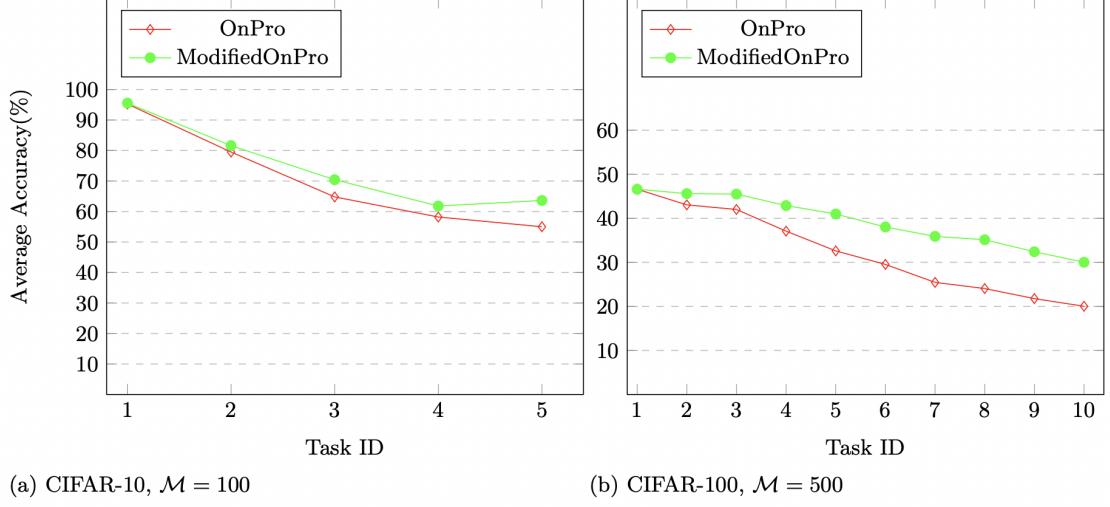
The results in Table 4.3 indicate that, despite achieving higher average accuracy, our proposed model still faces the issue of catastrophic forgetting. For example, in experiments on CIFAR-10, with $\mathcal{M}=100$ and $\mathcal{M}=500$, our model has the smallest forgetting scores, but with $\mathcal{M}=200$, it is larger compared to Onpro (Note that a smaller forgetting score is better). For the CIFAR-100 dataset, our model only forgets the least with $\mathcal{M}=1000$, while in the other two cases, it is higher than Onpro. This is explained by the fact that our method is currently focusing on optimizing for the classification ability of the model and has not addressed the issue of catastrophic forgetting as much.

We track task accuracy metrics during the training process, illustrating that our method consistently outperforms the Onpro method with higher results (see Figure 4.3).

4.2.6 Integrating Latent Alignment via Energy-Based Model

Building on the ELI mechanism introduced in Section 2.4, we now apply it to the multi-head architecture to address latent space drift in medical image classification. Energy-

Figure 4.3: Task accuracy metrics during the training process of our method (green line) vs. the Onpro method (red line).



based latent aligner (ELI) is a novel methodology that helps counter the representational shift in the latent space of incremental learning models. ELI learns an energy manifold for the latent representations such that previous task latents have low energy and the current task latents have high energy values (see Figure 4.4). The feature representations are optimized for the circle recognition task when trained on the circular dataset. However, when learning an additional task of square recognition, the hidden feature space of circles shifts closer to stars (as indicated by the red arrow, shifting in the wrong direction). Then, the ELI model is used to align and pull them back towards each other along the direction of the green arrow.

Once all WP-heads and OOD-heads are fine-tuned, we train the ELI models. We use three components: samples of current task $\mathcal{D}_t$, features of x encoded by the encoder network trained until previous task $\mathbf{e}_{t-1} = f_{t-1}(x)$, and features of x encoded by the encoder network trained until current task $\mathbf{e}_t = f_t(x)$. ELI learns to assign a small energy value for $\mathbf{e}_{t-1}$ and a high energy value for $\mathbf{e}_t$.

We train a model $E_\psi(\mathbf{e}) : \mathbb{R}^D \rightarrow \mathbb{R}$ to map the feature $\mathbf{e} \in \mathbb{R}^D$ to energy value. An energy-based model (EBM) is defined as a Gibbs distribution $p_\psi(\mathbf{e})$ trên $E_\psi(\mathbf{e})$:

$$p_\psi(\mathbf{e}) = \frac{\exp(-E_\psi(\mathbf{e}))}{\int_{\mathbf{e}} \exp(-E_\psi(\mathbf{e})) d\mathbf{e}}, \quad (4.13)$$

EBM learns to maximize the log-likelihood of a sample taken from the distribution

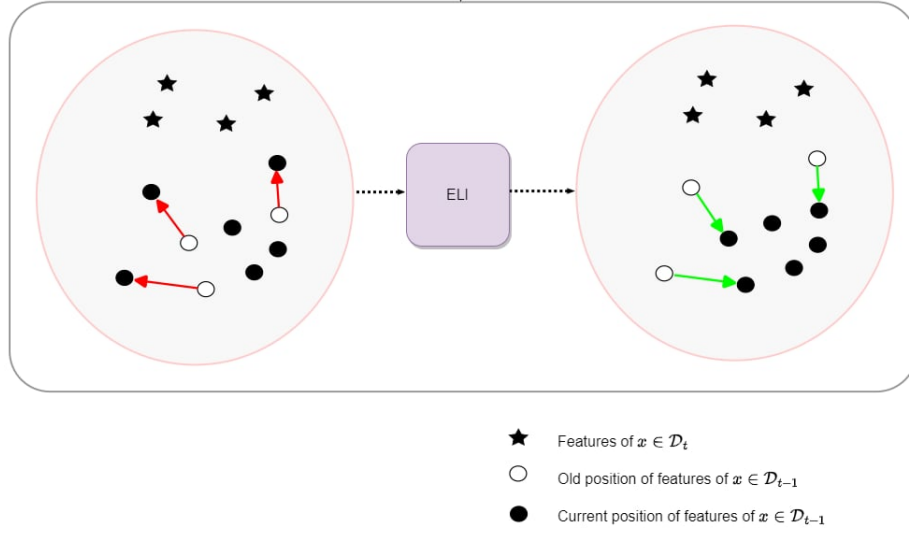


Figure 4.4: Energy-based latent aligner.

$p_{true}(\mathbf{e})$:

$$L(\psi) = \mathbb{E}_{\mathbf{e} \sim p_{true}} [\log p_{\psi}(\mathbf{e})]. \quad (4.14)$$

The derivative of $L(\psi)$ is:

$$\partial_{\psi} L(\psi) = \mathbb{E}_{\mathbf{e} \sim p_{true}} [-\partial_{\psi} E_{\psi}(\mathbf{e})] + \mathbb{E}_{\mathbf{e} \sim p_{\psi}} [\partial_{\psi} E_{\psi}(\mathbf{e})]. \quad (4.15)$$

The first term in 4.15 assures that the energy for $\mathbf{e}$ taken from the distribution p_{true} is minimized. The second one maximizes the samples from the current model. In ELI, p_{true} is the distribution of features extracted from the model trained until the previous task. Taking samples from $p_{\psi}(\mathbf{x})$ is difficult because of the normalization constant in equation 4.13. Approximate samples are recursively drawn using Langevin dynamics [89, 127], which is a popular MCMC algorithm,

$$\mathbf{e}_{i+1} = \mathbf{e}_i - \frac{\lambda}{2} \partial_{\mathbf{e}} E_{\psi}(\mathbf{e}) + \sqrt{\lambda} \omega_i, \omega_i \sim \mathcal{N}(0, \mathbf{I}) \quad (4.16)$$

This equation ensures the Markov chain stabilizes to a stationary distribution within a few iterations, starting from an initial value $\mathbf{e}_i$.

4.2.7 Experimental Results

4.2.7.1 Experimental Settings

We evaluate our approach across multiple datasets: CCH5000 for medical image classification [53], and CIFAR-10/100 for general computer vision tasks. The CCH5000 dataset

comprises 5000 colorectal cancer histology images spanning eight tissue types (see Figure 4.5 for some examples). We employ an 80-20 train-test split with random partitioning, training initially on four classes before incremental learning on the remaining classes across two or four tasks.

Our implementation uses ResNet-18 backbone pre-trained on ImageNet, optimized via Adam with learning rate 5×10^{-4} . Batch configurations include current task samples ($N = 10$) and replay buffer samples ($m = 32$). All comparisons maintain identical backbone initialization and memory constraints, with results averaged across three independent runs.

Performance assessment employs two complementary metrics, namely average accuracy and average forgetting.

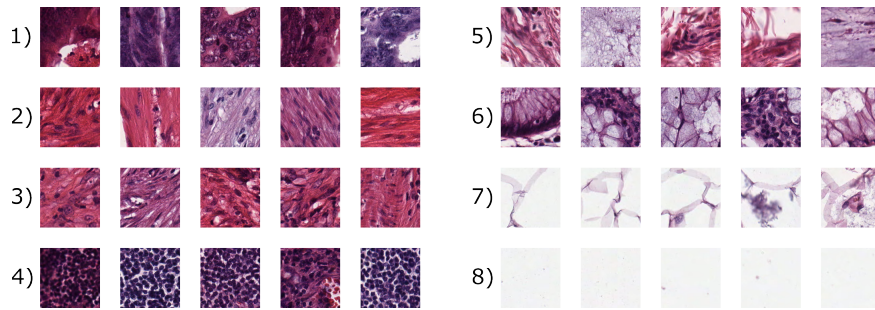


Figure 4.5: Example of image from 10 classes in CCH5000 dataset

Fig. 4.5 show five random images from each class of the CCH5000 dataset, including: 1) tumour epithelium, 2) simple stroma, 3) complex stroma (stroma that contains single tumour cells and/or single immune cells), 4) immune cell conglomerates, 5) debris and mucus, 6) mucosal glands, 7) adipose tissue, 8) background To evaluate models, we use Average Accuracy and Average Forgetting.

4.2.7.2 Results

We conduct an ablation experiment to evaluate the effect of WP-heads, OOD-heads and ELI on the model’s performance. Specifically, we evaluate and compare the performance of our model, the model without ELI, and the model without ELI, WP-heads and OOD-heads. With the scenario of 2 classes per task and without ELI, accuracy decreases by 1.3%, while the forgetting score increases from 2.9 to 4.6. Without WP, OOD, and ELI, accuracy decreases by 1.8%, and the forgetting score also increases by 0.9.

In the scenario of one class per task, removing the ELI component results in a 0.2%

Table 4.4: Comparative results on CCH5000 dataset with one and two classes per task settings.

Classes per task	1		2	
Metrics	Acc	Fgt	Acc	Fgt
iCaRL [98]	91.1	9.0	93.0	6.8
UCIR [45]	92.0	5.5	93.9	4.4
PODNet [29]	89.2	6.0	92.0	5.2
DER [135]	91.0	5.6	93.0	6.4
LO2LN [18]	94.5	3.9	94.6	4.0
Onpro[126]	89.5	3.6	92.8	3.8
Ours	91.6	4.6	94.6	2.9

increase in the model’s performance, indicating that ELI is not effective in this scenario. However, when excluding all three components, WP, OOD, and ELI, the model’s accuracy decreases by 2.1

Table 4.5: Systematic ablation analysis with computational profiling (GPU in GB and Time in minutes).

Classes per task	1				2			
Model	Acc	Fgt	GPU	Time	Acc	Fgt	GPU	Time
Ours	91.6	4.6	13.4	34.82	94.6	2.9	13.4	25.3
– ELI	91.8	4.8	13.4	33.18	93.3	8.5	13.4	24.1
– <i>WP, OOD</i>	89.7	2.3	13.4	20.32	92.1	5.7	13.4	18.8
– ELI (Onpro[126])	89.5	3.6	13.4	19.13	92.8	3.8	13.4	18.1

4.2.7.3 Medical Image Classification Results

We compare our model’s results to several state-of-the-art methods for solving incremental learning problems. The experiments are implemented in two settings: one and two classes per task. We conducted each experiment three times and averaged the results.

Performance Analysis: Table 4.4 demonstrates that our model achieves significant improvements in both accuracy and forgetting metrics. With two classes per task, our method achieves the best forgetting score of 2.9, representing a 1.1 point (27.5%)

improvement over LO2LN [18] and a substantial 3.9 (57.4%) point improvement over iCaRL [98]. The average accuracy matches the best performer (LO2LN) at 94.6% and exceeds PODNet [29] by 2.6%.

However, our method shows reduced effectiveness in the single class per task scenario, achieving 91.6% accuracy compared to LO2LN’s 94.5%. This performance degradation is primarily attributed to the limited discriminative power of ELI when each task contains insufficient class diversity for effective energy manifold learning.

Computational Overhead Analysis: Our approach introduces additional computational costs through the multi-head architecture and ELI training. Specifically, the training time increases by approximately 82% compared to the baseline OnPro method (34.82 minutes vs. 19.13 minutes for one class per task), while GPU memory consumption remains comparable at 13.4 GB, indicating efficient parameter sharing in our architecture.. The ELI component alone contributes approximately 8% of the total training time through its Langevin dynamics sampling process.

Table 4.6: Comparative results on CCH5000 dataset with varying classes per task settings.

Method	1 Class/Task		2 Classes/Task	
	Acc	Fgt	Acc	Fgt
iCaRL [98]	91.1	9.0	93.0	6.8
UCIR [45]	92.0	5.5	93.9	4.4
PODNet [29]	89.2	6.0	92.0	5.2
DER [135]	91.0	5.6	93.0	6.4
LO2LN [18]	94.5	3.9	94.6	4.0
OnPro [126]	89.5	3.6	92.8	3.8
Ours	91.6	4.6	94.6	2.9

4.3 DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic in Continual Learning

DAKTA is designed to train each task on each composed model, and is inspired by the Kolmogorov-Arnold representation theorem that combines many fundamental functions for learnable activation functions on the edges of the classifier. Additionally, DAKTA uses the second-order regularization loss, which helps the model protect important knowledge from pre-trained models built upon a Vision Transformer (ViT) backbone adapted for

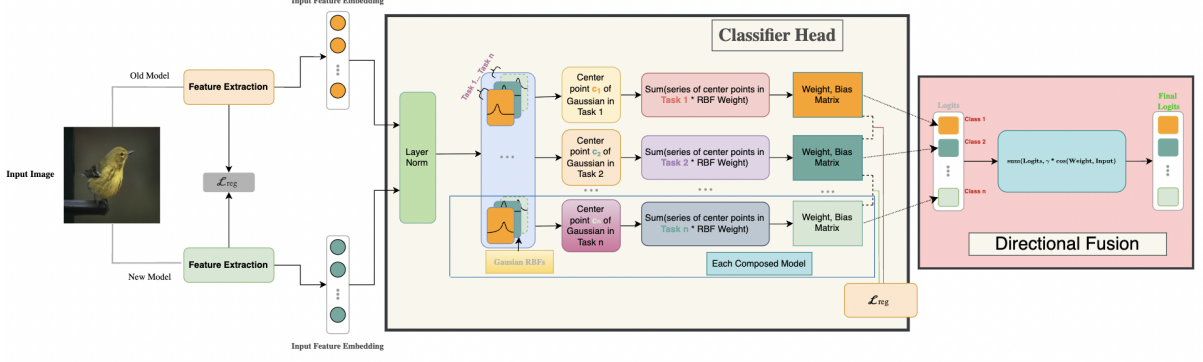


Figure 4.6: Pipeline of the DAKTA framework, integrating a Vision Transformer (ViT) backbone with Low-Rank Adaptation (LoRA) and a Kolmogorov-Arnold Classifier (KAC) using Gaussian Radial Basis Functions (RBFs) for class-incremental learning. The process involves feature extraction from both old and new models, task-specific parameter composition via Incremental Task Arithmetic (ITA), and regularization using the Fisher Information Matrix to mitigate catastrophic forgetting.

each new task using Low-Rank Adaptation (LoRA). The following subsections detail each component and the overall learning process.

Figure 4.6 illustrates the overall architecture of DAKTA, comprising three main components. First, the pre-trained ViT-B/16 backbone is adapted using low-rank adapters (LoRA, rank $r = 8$) inserted into each attention layer. The backbone weights remain frozen during training; only the LoRA parameters are updated, reducing trainable parameters while preserving pre-trained representations. Second, in place of an ordinary linear classifier, DAKTA uses a Kolmogorov-Arnold Classifier (KAC) head whose learnable radial-basis-function activations let it model non-linear feature-to-class relationships, sharpening discrimination as tasks accumulate. Third, Fisher Information Matrix (FIM) regularization is applied to both the LoRA parameters and KAC weights, penalising updates to parameters critical for previously learned tasks and thereby mitigating catastrophic forgetting. In the figure, orange arrows indicate active gradient flow, while blue arrows denote frozen pathways.

4.3.1 Related Work

Incremental Task Arithmetic with LoRA (ITA-LoRA) [95] is a parameter-efficient continual learning method that builds upon the task arithmetic framework to enable class-

incremental learning without storing past exemplars. ITA-LoRA adapts a pre-trained Vision Transformer (ViT) backbone to each new task using Low-Rank Adaptation (LoRA), which inserts small trainable matrices of low rank into the attention and feed-forward layers of the ViT while keeping the majority of backbone parameters frozen. After training on each task, ITA-LoRA records the displacement of the adapted parameters relative to the pre-trained initialization as a task vector, and at inference time composes the final model by averaging all stored task vectors. This task arithmetic formulation is further grounded in a second-order Taylor approximation perspective, which ensures that the composed task vectors remain within the pre-training loss basin, thereby providing theoretical justification for the stability of the composed model. ITA-LoRA achieves competitive performance across diverse benchmarks, outperforming prompt-based methods (CODA, APT), full fine-tuning variants (SEED, TMC), and LoRA-based baselines (InfLoRA) in terms of both final accuracy and forgetting score, establishing itself as a strong state-of-the-art baseline in the task arithmetic paradigm for continual learning.

However, ITA-LoRA exhibits a fundamental limitation that motivates the proposed DAKTA method. While ITA-LoRA effectively handles parameter-level task compositionality through LoRA-based adaptation and task vector averaging, it retains the conventional *linear classifier head* for all tasks. A linear classifier computes class logits purely through a dot product between the feature vector and the weight matrix, which provides a global and uniform decision boundary that does not adapt to the local structure of the feature space. As new tasks are introduced and the feature space evolves, the decision boundaries of old classes are increasingly disturbed by the introduction of new class weights, leading to inter-task interference in the classification space. This is particularly problematic in fine-grained recognition scenarios such as CUB-200, where classes share highly similar visual features and the subtle distinctions between classes demand a more locally sensitive classifier. Furthermore, a standard linear head treats all dimensions of the feature vector equally, without selectively activating the channels that are most discriminative for each class, limiting the model’s ability to suppress interference from irrelevant feature dimensions. Second, ITA-LoRA applies second-order regularization only to the backbone parameters and does not extend this principled regularization to the classifier head, leaving the classifier vulnerable to catastrophic forgetting as new classes are appended.

Motivated by these observations, the proposed DAKTA framework replaces the linear classifier with a Kolmogorov-Arnold Classifier (KAC) built upon learnable Gaussian Radial Basis Functions (RBFs). Unlike a linear classifier, the KAC computes class activations as weighted sums of Gaussian kernels centered at learnable locations in the

feature space, providing selective channel activation and enhanced locality that significantly reduces interference between tasks. Additionally, DAKTA introduces a directional logit fusion mechanism that combines the RBF-based magnitude logits with a cosine similarity term, enabling the classifier to capture both local similarity patterns and global directional relationships for more robust class discrimination. Finally, DAKTA extends the second-order Fisher Information Matrix regularization to both the backbone and the KAC classifier head, ensuring that parameters critical to previous tasks are protected throughout incremental training. Together, these innovations lead to consistent improvements over ITA-LoRA in both final accuracy and forgetting score across all evaluated benchmarks.

4.3.2 Vision Transformer Backbone and Low-Rank Adaptation (LoRA)

Our framework uses a Vision Transformer (ViT) as the feature extractor. The ViT is a neural network architecture that processes images by dividing them into patches and applying self-attention, enabling it to capture complex patterns in visual data. To efficiently adapt the ViT to new tasks without retraining all its parameters, we use Low-Rank Adaptation(LoRA). LoRA works by inserting small, trainable matrices (of low rank) into the attention and feedforward layers of the ViT. During training on each new task, only these LoRA parameters are updated, while the majority of the ViT’s parameters remain fixed. This approach enables fast adaptation to new tasks and helps prevent the model from overwriting previously learned knowledge.

4.3.3 Incremental Task Arithmetic (ITA) and Model Composition

To enable the model to learn new tasks without forgetting the previous ones, we use Incremental Task Arithmetic(ITA). For each task t , we keep track of the changes made to both the LoRA parameters of the ViT and the parameters of the KAC classifier. Let θ_0 and ψ_0 denote the initial parameters of the ViT and KAC, respectively, before any task-specific adaptation. After training on task t , the updated parameters are θ_t and ψ_t . We define the task vectors as:

$$\tau_t^{\text{backbone}} = \theta_t - \theta_0, \quad \tau_t^{\text{cls}} = \psi_t - \psi_0. \quad (4.17)$$

At inference time (i.e., when making predictions after all tasks have been learned), we combine the knowledge from all T tasks by averaging these task vectors:

$$\theta_P = \theta_0 + \frac{1}{T} \sum_{t=1}^T \tau_t^{\text{backbone}}, \quad \psi_P = \psi_0 + \frac{1}{T} \sum_{t=1}^T \tau_t^{\text{cls}}. \quad (4.18)$$

This means that the final model is a balanced combination of all adaptations made for each task, thereby retaining knowledge from earlier tasks while incorporating new information.

4.3.4 Kolmogorov-Arnold Classifier Head

We introduce the Kolmogorov-Arnold Classifier (KAC), which is more flexible than a standard linear classifier. The KAC is inspired by a mathematical result called the Kolmogorov-Arnold representation theorem, which states that any complex function can be constructed from sums and compositions of simple, one-dimensional functions. In our implementation, the KAC uses Gaussian Radial Basis Functions (RBFs) to model the relationship between features and classes.

Given a feature vector $x \in \mathbb{R}^d$ (where d is the number of features), the KAC computes, for each class c , an activation as follows:

$$\phi_c(x) = \sum_{j=1}^N w_{c,j} \exp \left(-\frac{(x - \mu_{c,j})^2}{2\sigma_{c,j}^2} \right), \quad (4.19)$$

where:

- N is the number of RBFs per feature channel (a small integer, e.g., 3 or 4),
- $w_{c,j}$ are learnable weights for class c and RBF j ,
- $\mu_{c,j}$ are learnable centers (the location where each RBF is focused),
- $\sigma_{c,j}$ are learnable widths (which control how wide each RBF is),
- $\exp(-\frac{(x-\mu_{c,j})^2}{2\sigma_{c,j}^2})$ is the Gaussian RBF, which outputs a value close to 1 when x is near $\mu_{c,j}$ and quickly drops to 0 as x moves away.

Given a feature vector $x \in \mathbb{R}^d$ (where d is the number of features), the KAC constructs an effective weight matrix for each class c by combining the RBF centers and weights: $W_c = \sum_{j=1}^N w_{c,j} \mu_{c,j}$, where $w_{c,j}$ are learnable weights, $\mu_{c,j}$ are learnable centers, and N is the number of RBFs per feature channel. The initial logits are computed via a linear transformation:

$$l_c^{\text{init}}(x) = W_c \cdot x + b_c, \quad (4.20)$$

where b_c is a learnable bias term for class c . To enhance the classifier's expressiveness, a directional alignment term is computed as the cosine similarity between the normalized feature vector $\frac{x}{\|x\|_2}$ and the normalized effective weight vector $\frac{W_c}{\|W_c\|_2}$, defined as $\cos(x, W_c) = \frac{x \cdot W_c}{\|x\|_2 \|W_c\|_2}$.

4.3.5 Directional Vector Fusion

In addition to utilizing vector magnitude information, we also integrate directional information into our approach. The final logits are computed by adding this term, scaled by a learnable parameter γ_c initialized to 0.1, to the initial logits:

$$l_c(x) = l_c^{\text{init}}(x) + \gamma_c \cdot \cos(x, W_c). \quad (4.21)$$

This design combines the linear transformation with a directional component, where the learnable scaling parameter γ_c optimizes the contribution of the cosine similarity term, improving robustness and expressiveness compared to a standard linear classifier.

4.3.6 Second-Order Regularization with the Fisher Information Matrix

To further reduce forgetting, we use a regularization technique based on the Fisher Information Matrix (FIM). The FIM is a mathematical tool that measures how important each parameter is for the model’s predictions. After training on each task, we estimate the diagonal of the FIM for both the ViT and the KAC parameters, denoted as $\hat{F}_{\theta_0}$ and $\hat{F}_{\psi_0}$. During training on a new task, we penalize large changes to parameters that the FIM indicates are important for previous tasks. The regularization terms are as follows:

$$L_{\text{reg}}^{\text{backbone}} = \frac{\alpha}{2} \sum_i \hat{F}_{\theta_0}^{(i)} (\theta_t^{(i)} - \theta_0^{(i)})^2, \quad (4.22)$$

$$L_{\text{reg}}^{\text{cls}} = \frac{\beta}{2} \sum_j \hat{F}_{\psi_0}^{(j)} (\psi_t^{(j)} - \psi_0^{(j)})^2, \quad (4.23)$$

where α and β are hyperparameters that control how strongly we penalize changes to important parameters. This approach encourages the model to keep parameters that are crucial for previous tasks close to their original values, thus preserving old knowledge.

4.3.7 Training Objective and Step-by-Step Procedure

For each batch of data (x, y) in task t , the total loss is:

$$L = L_{\text{CE}}(h_{\psi_t}(f_{\theta_t}(x)), y) + L_{\text{reg}}^{\text{backbone}} + L_{\text{reg}}^{\text{cls}}, \quad (4.24)$$

where L_{CE} is the cross-entropy loss, which measures how well the model’s predictions match the true labels, f_{θ_t} is the ViT with LoRA parameters for task t , h_{ψ_t} is the KAC classifier for task t , $L_{\text{reg}}^{\text{backbone}}$ and $L_{\text{reg}}^{\text{cls}}$ are the regularization terms described above.

The total loss L in Equation 4.24 consists of three terms. The first term, L_{CE} , is the cross-entropy loss that trains the model to correctly classify the current task t : the input x is processed by the LoRA-augmented ViT backbone f_{θ_t} to extract features, which are then passed to the KAC classifier h_{ψ_t} to produce predictions compared against the ground-truth labels y . The remaining two terms, $L_{\text{reg}}^{\text{backbone}}$ and $L_{\text{reg}}^{\text{cls}}$, are regularization losses applied to the backbone and classifier, respectively, to prevent catastrophic forgetting by constraining the model parameters from deviating too far from previously learned knowledge. Together, these three terms balance the trade-off between learning new tasks (plasticity) and retaining old ones (stability).

The training procedure for each task is as follows Algorithm 5.

Algorithm 5 DAKTA Algorithm

Require: Pre-trained Vision Transformer (ViT) backbone parameters θ_0 ; Initial KAC classifier parameters ψ_0 ; Tasks $\mathcal{D}_1, \dots, \mathcal{D}_T$

Ensure: Final composed model parameters (θ_P, ψ_P)

```

1: for  $t = 1$  to  $T$  do
2:   Extend KAC classifier to accommodate new classes for task  $t$            ▷ Classifier
   expansion
3:    $\theta_t \leftarrow \theta_0, \psi_t \leftarrow \psi_0$                                ▷ Parameter initialization
4:   for each training iteration on task  $t$  do
5:     Sample mini-batch  $(x, y)$  from  $\mathcal{D}_t$ 
6:      $z \leftarrow f_{\theta_t}(x)$                                              ▷ Feature extraction with ViT+LoRA
7:      $l \leftarrow h_{\psi_t}(z)$                                            ▷ Compute logits with KAC
8:      $\mathcal{L}_{\text{task}} \leftarrow \text{CrossEntropy}(l, y)$                        ▷ Classification loss
9:     Compute  $\mathcal{L}_{\text{reg}}^{\text{backbone}}$  and  $\mathcal{L}_{\text{reg}}^{\text{cls}}$  via Eq. 4.22 and Eq. 4.23
10:     $\mathcal{L} \leftarrow \mathcal{L}_{\text{task}} + \mathcal{L}_{\text{reg}}^{\text{backbone}} + \mathcal{L}_{\text{reg}}^{\text{cls}}$            ▷ Total loss (Eq. 4.24)
11:    Update  $\theta_t, \psi_t$  via gradient descent on  $\mathcal{L}$ 
12:  end for
13:  Estimate Fisher information matrices  $\hat{F}_{\theta_0}, \hat{F}_{\psi_0}$  using data from  $\mathcal{D}_t$ 
14:   $\tau_t^{\text{backbone}} \leftarrow \theta_t - \theta_0, \tau_t^{\text{cls}} \leftarrow \psi_t - \psi_0$            ▷ Store task vectors
15: end for
16: Compose final parameters  $(\theta_P, \psi_P)$  via directional vector fusion (Eq. 4.21)
17: return  $(\theta_P, \psi_P)$ 

```

After all tasks have been learned, the final model is composed of averaging the task vectors as described above. This ensures that the model maintains a balance between

retaining knowledge from earlier tasks and adapting to new ones, achieving robustness in continual learning without catastrophic forgetting.

4.3.8 Algorithm

We introduce **DAKTA**, a continual learning algorithm that unifies parameter-efficient adaptation and a flexible classifier head for class-incremental scenarios. As detailed in Algorithm 5, DAKTA operates in two main phases for each new task: classifier expansion and regularized fine-tuning.

Classifier Expansion. When a new task arrives, the KAC classifier is expanded to accommodate the additional classes. To ensure stable integration of new knowledge, we initialize the task-specific parameters for both the backbone and classifier from their pre-trained values. In this stage, we optionally perform a brief linear probing step, where only the new classifier parameters are updated while the backbone remains frozen. This step helps align the new head with the pre-trained feature space, promoting optimal initialization for subsequent learning.

Regularized Fine-Tuning. After pre-consolidation, DAKTA proceeds to jointly update the backbone (via LoRA modules) and the KAC classifier using the data from the current task. The training objective combines the standard cross-entropy loss with second-order regularization terms for both the backbone and classifier parameters. These regularizers, based on the Fisher information estimated after each task, penalize deviations from the pre-trained weights, thereby mitigating catastrophic forgetting. Throughout training, only a small subset of parameters (the LoRA adapters and the KAC head) are updated, ensuring parameter efficiency and scalability.

Task Vector Storage and Model Composition. Upon completion of each task, DAKTA records the parameter changes (task vectors) for both the backbone and the classifier. The Fisher information matrices are also updated to reflect the importance of each parameter. At inference, the final model is composed by aggregating the pre-trained parameters with a weighted sum of the task vectors, balancing knowledge across all tasks.

DAKTA is compatible with various parameter-efficient adaptation strategies, such as LoRA and (IA)³, and is designed to be robust to the continual introduction of new classes. By combining a compositional classifier with principled regularization and efficient adaptation, DAKTA achieves strong performance and knowledge retention in class-incremental learning.

Table 4.7: Comparison with SOTA (Final Accuracy $\uparrow$). Best results in bold. EWC, LwF-MC, DER++ (buffer size of 1000 examples), SEED, and TMC rely on full fine-tuning; L2P, CODA, and APT utilize prompt-based learning. Finally, InfLoRA adopts LoRA fine-tuning.

Model	C-100	CUB	Caltech	MIT	RESISC	CropDis.
CODA [111]	86.48	78.54	90.57	77.73	69.50	74.65
SEED [103]	83.39	85.35	90.04	86.34	74.81	92.77
TMC [74]	78.42	71.72	82.30	68.66	60.66	66.56
DER++ [12]	83.02	76.10	86.11	68.88	67.23	98.77
InfLoRA [73]	87.17	79.14	90.53	79.14	79.92	89.05
ITA-FFT [95]	89.38	84.80	92.32	85.35	80.50	91.81
ITA-LoRA [95]	89.96	85.55	92.65	86.60	82.00	95.85
ITA-(IA)3 [95]	90.66	85.67	92.67	84.74	83.73	95.41
DAKTA (Ours)	91.21	87.59	93.61	87.97	85.82	97.75

4.3.9 Experimental Results

We evaluate our proposed method on a diverse set of continual learning benchmarks, including ImageNet-R (IN-R), CIFAR-100 (C-100), CUB-200 (CUB), Caltech-256 (Caltech), MIT-67 (MIT), RESISC-45 (RESISC), and CropDiseases (CropDis.). Our experiments compare the final classification accuracy of our approach with that of a range of state-of-the-art (SOTA) continual learning methods, including both fine-tuning and prompt-based strategies. The compared baselines include CODA, SEED, TMC, DER++, and InfLoRA, as well as several ITA-based variants (ITA-FFT, ITA-LoRA, ITA-(IA)3). Each method is evaluated under the same experimental protocol, with results reported as the final accuracy (%) after learning all tasks. For methods that utilize experience replay, such as DER++, a buffer size of 1,000 examples is used. Prompt-based methods (L2P, CODA, APT) and LoRA-based fine-tuning (InfLoRA) are also included for comprehensive comparison. Table 4.7 summarizes the results across all datasets. Our method achieves the highest accuracy on the majority of benchmarks, demonstrating the effectiveness of combining ITA-LoRA with the Kolmogorov-Arnold Classifier (KAC) for continual learning. For each benchmark, the highest-performing result is indicated in bold typeface.

To assess the performance of our model in a continual learning scenario, we also

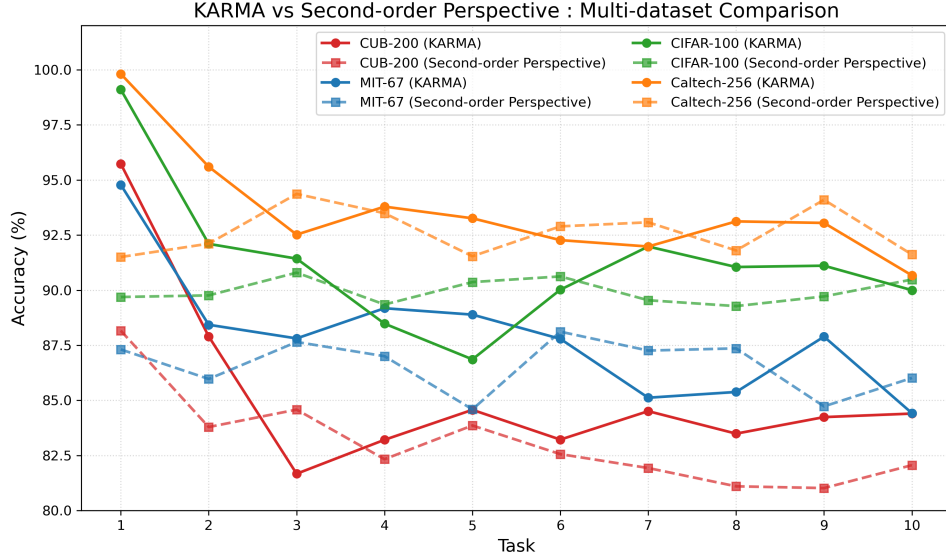


Figure 4.7: Comparison of model accuracy across multiple datasets in a continual learning setting, highlighting the impact of catastrophic forgetting.

evaluate its accuracy across multiple datasets and visualize the results to highlight the impact of catastrophic forgetting. Figure 4.7 presents a comprehensive comparison of the model’s accuracy as it learns new tasks sequentially, revealing trends in performance retention and degradation over time.

4.3.10 Ablation Study: Effect of Second-Order Regularization Placement

To investigate the impact of second-order (Fisher-based) regularization in our framework, we conduct an ablation study focusing on where the regularization loss is applied. Specifically, we compare two settings:

1. **Backbone-only Regularization:** The second-order regularization loss is applied only to the ViT backbone parameters, while the KAC classifier head is trained without any regularization.
2. **Backbone + Classifier Regularization:** The second-order regularization loss is applied to both the ViT backbone and the KAC classifier head parameters.

We evaluate both settings on the class-incremental learning benchmark and report the average accuracy (%) over all tasks and the accuracy on the last task, which is a common measure of resistance to catastrophic forgetting.

Table 4.8: Ablation study on second-order regularization.

Method	C-100	CUB	Cal.	MIT	RES.	Crop.
w/o Reg.	85.24	80.57	89.1	82.25	79.38	90.84
w/o KAC	89.91	84.12	92.35	85.24	82.25	94.82
Full Model	91.21	87.59	93.61	87.97	85.82	97.75

Table 4.9: Comparison with the SOTA across several benchmarks(Final Forgetting [↓]).

Note that a lower forgetting score indicates better performance.

Model	C-100	CUB	Caltech	MIT	RESISC	CropDis.
CODA [111]	5.28	8.82	2.64	7.49	22	15.37
SEED [103]	6.39	5.12	2.67	4.53	8.66	1.19
InfLoRA [73]	4.05	4.58	1.89	6.85	10.77	4.79
APT [11]	6.71	8.82	3.64	6.33	27.74	14.01
TMC [74]	9.2	7.37	6.39	12.43	16.43	16.94
ITA-LoRA [95]	3.22	2.08	2.38	4.68	8.30	0.68
ITA-FFT [95]	3.89	1.39	2.44	4.80	6.19	0.95
ITA-(IA)3 [95]	3.57	2.23	2.39	4.48	7.33	0.76
DAKTA(Ours)	2.98	1.14	1.93	3.08	4.38	0.75

As shown in Table 4.8, applying regularization (shown in Equations (4.22) and (4.23)) to both the backbone and the KAC classifier head leads to a clear improvement in the accuracy of each task. This demonstrates that regularizing the classifier head, in addition to the backbone, further mitigates catastrophic forgetting and enhances overall continual learning performance.

4.4 SO-LoRA: Sparse Orthogonal LoRA for Parameter-Efficient Continual Learning

We propose **SO-LoRA**, a rehearsal-free continual learning scheme that encodes each task as a sparse composition over a fixed orthogonal LoRA basis. The key idea is to keep the low-rank coordinate system stable over time, and let tasks compete only through a sparse set of shared coefficients.

4.4.1 Related Work

Existing LoRA-based continual learning methods expose a fundamental structural tension. Architecture-based approaches such as per-task LoRA isolation [73] eliminate gradient interference by allocating separate adapters per task, but incur linear parameter growth that becomes unsustainable over long task sequences. At the other extreme, methods that share a single adapter across tasks maintain a constant memory footprint but suffer from catastrophic interference, as gradients from different tasks compete within the same low-rank subspace and progressively overwrite previously acquired knowledge [112]. Recent methods such as O-LoRA [123] and SD-LoRA [131] attempt to mitigate interference through orthogonality regularization or decoupled optimization heuristics; however, O-LoRA relies on regularizers that can drift or become costly over long sequences, while SD-LoRA treats interference as an optimization artifact rather than addressing its underlying geometric cause, namely the overlap between task representations in the shared subspace. This distinction becomes critical in long-horizon settings, as evidenced by O-LoRA’s sharp accuracy drops from 72.76% to 60.98% as the number of tasks grows from 5 to 20 on ImageNet-R. Motivated by these observations, we propose SO-LoRA, which reconceptualizes the problem geometrically by fixing a universal orthonormal basis and restricting each task to learn sparse compositions over this stable coordinate system. Group sparsity explicitly reduces subspace overlap between tasks, while a gradient-aware soft mask protects historically important basis directions from being overwritten, yielding an interference bound proportional to the product of task sparsity norms and maintaining a constant parameter footprint regardless of sequence length.

Table 4.10: Comparison of LoRA-based continual learning methods.

Method	Basis fixed?	Sparsity	Grad mask	Param footprint
O-LoRA [123]	No	No	No	$O(T)$
SD-LoRA [132]	Partial	No	No	$O(1)$
InfLoRA [73]	Yes (SVD)	No	No	$O(1)$
SO-LoRA (Ours)	Yes (QR)	Yes ($\ell_{2,1}$)	Yes (EMA)	$O(1)$

Table 4.10 sets these methods side by side. InfLoRA gets its fixed basis from an SVD of the pre-trained weight matrix and projects new gradients into the resulting null space, but without any sparsity constraint it has no analytical bound on inter-task

interference. SD-LoRA decouples stable and dynamic components heuristically without a geometric guarantee. SO-LoRA is the only method that simultaneously (i) fixes an orthonormal basis constructed by QR decomposition (Eq. 4.33), (ii) enforces group sparsity via the $\ell_{2,1}$ norm to minimise subspace overlap, and (iii) applies a gradient-aware EMA mask to protect historically important rank dimensions, yielding the interference bound in Proposition 4.29.

4.4.2 Preliminary

Problem setting. We study rehearsal-free Class-Incremental Learning (CIL). A model f_Φ receives a stream of T tasks $\{\mathcal{T}_1, \dots, \mathcal{T}_T\}$, where each task $\mathcal{T}_t$ provides data $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t} \sim P_t(x, y)$ and has a disjoint label set C_t ($C_t \cap C_k = \emptyset$ for $t \neq k$). We adopt parameter-efficient fine-tuning by splitting parameters as $\Phi = \{\Theta, \Psi\}$, with a frozen backbone Θ and a small trainable adapter Ψ ($|\Psi| \ll |\Theta|$). The ideal objective minimizes risk over all tasks:

$$\Phi^* = \operatorname{argmin}_{\Phi} \sum_{t=1}^T \mathbb{E}_{(x,y) \sim \mathcal{D}_t} [\mathcal{L}(f_\Phi(x), y)]. \quad (4.25)$$

In rehearsal-free CIL, however, task $\mathcal{T}_t$ is optimized using only $\mathcal{D}_t$; gradients $\nabla_\Psi \mathcal{L}_t$ can therefore move parameters to regions that increase loss on earlier tasks (catastrophic forgetting).

Parameter-Efficient Fine-Tuning and Low-Rank Adaptation

Parameter-efficient fine-tuning (PEFT) adapts a large pre-trained model to a downstream task by updating only a small subset of parameters, leaving the backbone frozen; common instantiations include adapter modules, prompt tuning, and low-rank updates. Low-Rank Adaptation (LoRA) [48] is the PEFT variant used throughout Chapter 4. Given a frozen pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA parameterizes the update as a low-rank product:

$$W = W_0 + \Delta W = W_0 + BA, \quad B \in \mathbb{R}^{d \times r}, \quad A \in \mathbb{R}^{r \times k}, \quad r \ll \min(d, k), \quad (4.26)$$

with only A and B trained. LoRA has three properties that continual learning methods exploit directly: (i) parameter efficiency, since r is typically two to three orders of magnitude smaller than d, k ; (ii) mergeability, since $W_0 + BA$ collapses back into a dense matrix of the original shape at inference time, adding no latency; and (iii) composability across tasks, since multiple task-specific low-rank updates $\{B^{(t)} A^{(t)}\}_{t=1}^T$ can be combined post-hoc, the property exploited by task arithmetic and by the frozen, orthogonal-basis variant used in SO-LoRA (Section 4.4). **LoRA in continual learning.** LoRA [47] adapts

a frozen weight $W_0 \in \mathbb{R}^{d \times k}$ using a low-rank update $\Delta W = AB$, with $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times k}$:

$$h = xW_0 + x\Delta W = xW_0 + xAB. \quad (4.27)$$

In a task stream, one can either isolate adapters per task (stable but memory grows with T):

$$W^{(t)} = W_0 + A_t B_t, \quad \Omega_{\text{iso}} = \bigcup_{t=1}^T \{A_t, B_t\} \Rightarrow |\Omega_{\text{iso}}| \propto \mathcal{O}(T), \quad (4.28)$$

or share a single adapter (constant memory but prone to interference):

$$W^{(t)} = W_0 + A^{(t)} B^{(t)}, \quad \Omega_{\text{shared}} = \{A, B\} \Rightarrow |\Omega_{\text{shared}}| \propto \mathcal{O}(1), \quad (4.29)$$

often yielding

$$\mathcal{L}_{<t}(W_0 + A^{(t)} B^{(t)}) \gg \mathcal{L}_{<t}(W_0 + A^{(t-1)} B^{(t-1)}). \quad (4.30)$$

SO-LoRA targets this trade-off: it keeps $\mathcal{O}(1)$ parameters while enforcing subspace separation within the shared adapter.

Task Arithmetic

Task arithmetic treats the effect of fine-tuning as a vector in parameter space that can be composed algebraically [90]. For a pre-trained model with parameters θ_0 fine-tuned on task t to obtain θ_t , the task vector is defined as $\tau_t = \theta_t - \theta_0$. Multiple task vectors can then be added, scaled, or subtracted to compose or suppress task-specific behavior in a single model:

$$\theta_{\text{new}} = \theta_0 + \sum_{t=1}^T \lambda_t \tau_t, \quad (4.31)$$

where the coefficients λ_t control how strongly each task’s knowledge is incorporated. When the fine-tuned parameters are constrained to a LoRA update (Eq. (4.26)), the task vector reduces to $\tau_t = B^{(t)} A^{(t)}$, and Eq. (4.31) becomes a sum of low-rank matrices rather than dense ones – making the composition both cheap to store and cheap to apply. This is the mechanism behind Incremental Task Arithmetic (ITA), used by DAKTA in Section 4.3 to fuse task-specific LoRA updates into a single backbone without retraining.

Representation

Throughout this dissertation, the representation of an input x refers to the feature vector $h = f_\theta(x)$ produced by the backbone prior to the task-specific classification head, as distinct from the raw input x or the model’s output logits z . Representation-based continual learning methods, pursue representations that are simultaneously discriminative (useful for the current task) and stable (resistant to drift as new tasks arrive). Representation

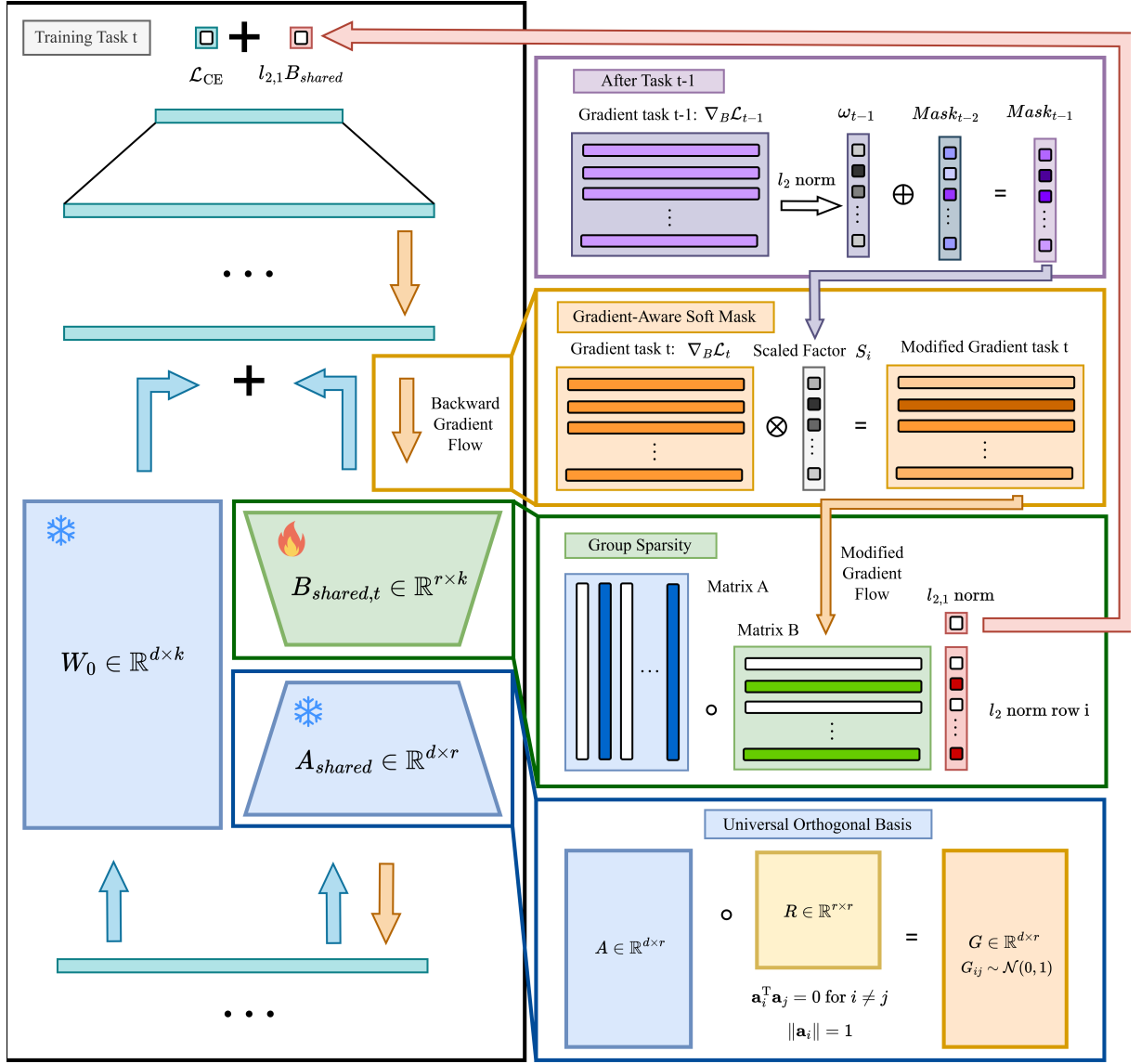


Figure 4.8: SO-LoRA represents adaptation as sparse coefficients $B^{(t)}$ over a frozen orthonormal basis A . Group sparsity limits overlap across tasks, while a gradient-aware mask M protects historically important rank dimensions.

drift refers to uncontrolled change in f_θ over the course of continual training, which is the failure mode that Eqs. (3.1), (3.2) regularize against, that analytic continual learning avoids by freezing f_θ outright, and that SO-LoRA (Section 4.4) bounds explicitly by constraining updates to an orthogonal basis. Representation stability is most commonly measured via linear probing, training a fresh linear classifier on frozen representations extracted at different points in the task sequence, as referenced in Section 1.4.

4.4.3 SO-LoRA: Sparse Orthogonal LoRA

Given frozen pre-trained weights W_0 , SO-LoRA parameterizes the task-adapted weights as

$$W^{(t)} = W_0 + AB^{(t)}, \quad (4.32)$$

where $A \in \mathbb{R}^{d \times r}$ is a fixed orthogonal basis and $B^{(t)} \in \mathbb{R}^{r \times k}$ is the shared, evolving coefficient matrix. Relative to standard LoRA (where both factors drift), Eq. 4.32 freezes the low-rank coordinate system and restricts learning to a controlled coefficient space.

Representation Stability via Fixed Orthogonal Basis A major source of forgetting in low-rank adaptation is subspace rotation: when both factors are updated, the meaning of rank dimensions changes over time. We prevent this by constructing A once and freezing it. Concretely, we sample $G \sim \mathcal{N}(0, 1)^{d \times r}$ and perform a QR decomposition $G = QR$, setting $A \leftarrow Q$. This yields

$$A^\top A = I_r \Rightarrow \langle A_{:,i}, A_{:,j} \rangle = \delta_{ij}. \quad (4.33)$$

With a fixed orthonormal dictionary, tasks can only differ through $B^{(t)}$, which reduces representational drift and makes interference analyzable in the coefficient space.

Interference Minimization via Group Sparsity With A fixed, interference arises when multiple tasks activate the same basis dimensions. Since $AB^{(t)} = \sum_{j=1}^r A_{:,j} B_{j,:}^{(t)}$, the row $B_{j,:}^{(t)}$ controls whether basis vector $A_{:,j}$ is used. We encourage row-wise sparsity with the $\ell_{2,1}$ norm (group lasso):

$$\mathcal{R}(B^{(t)}) = \|B^{(t)}\|_{2,1} = \sum_{j=1}^r \|B_{j,:}^{(t)}\|_2 = \sum_{j=1}^r \sqrt{\sum_{i=1}^k (B_{ji}^{(t)})^2}. \quad (4.34)$$

The per-task objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}}(f_\Phi(x), y) + \lambda \sum_{j=1}^r \|B_{j,:}^{(t)}\|_2. \quad (4.35)$$

This pushes each task to reuse as few rank directions as necessary, reducing subspace overlap and thus expected interference.

Protecting important directions with a gradient-aware mask Sparsity lowers collision probability, but cannot guarantee that critical directions will not be overwritten. We therefore maintain an importance vector $M \in \mathbb{R}^r$, updated only at task boundaries. After finishing task $\mathcal{T}_t$, we measure the sensitivity of each rank dimension by the gradient norm of the corresponding row:

$$\omega_{t,j} = \|(\nabla_{B^{(t)}} \mathcal{L}_t)_{j,:}\|_2. \quad (4.36)$$

We aggregate importance over time using an EMA:

$$M_{t,j} = \alpha \cdot \frac{\omega_{t,j}}{\max_k(\omega_{t,k}) + \epsilon} + (1 - \alpha) \cdot M_{t-1,j}. \quad (4.37)$$

During training of $\mathcal{T}_{t+1}$, we compute batch-wise relevance $\phi_j = \frac{\|g_{j,:}\|_2}{\max_k \|g_{k,:}\|_2 + \epsilon}$ for $g = \nabla_{B^{(t+1)}} \mathcal{L}_{t+1}$, and scale gradients by

$$S_j = 1 - (M_{t,j}(1 - \phi_j)), \quad g'_{j,:} = S_j \cdot g_{j,:}. \quad (4.38)$$

Thus, historically important but currently irrelevant rows ($M_{t,j} \approx 1, \phi_j \approx 0$) are suppressed, while unused capacity ($M_{t,j} \approx 0$) remains free to adapt.

Intuitively, group sparsity bounds the number of shared orthogonal-basis directions any two tasks can simultaneously activate, which directly limits gradient interference between tasks during joint optimization. Formally, as shown in Proposition 1, since $A^\top A = I_r$, the cross-task interference $|\langle \Delta W_i, \Delta W_j \rangle_F|$ is upper-bounded by the product of the row-wise ($\ell_{2,1}$) sparsity norms of the coefficient updates, $s_i s_j = \|\Delta B_i\|_{2,1} \|\Delta B_j\|_{2,1}$. Because this bound is non-vacuous only along rank dimensions l where both $\|(\Delta B_i)_{l,:}\|_2$ and $\|(\Delta B_j)_{l,:}\|_2$ are simultaneously non-zero, minimizing the $\ell_{2,1}$ penalty in Eq. (8) drives each task toward a compact subset of active basis directions. This reduces the overlap between $\text{supp}(B_i^{(t)})$ and $\text{supp}(B_j^{(t)})$ across tasks, thereby tightening the interference bound in Eq. (13) and providing a theoretical justification for why row-wise sparsity over a fixed orthogonal dictionary suppresses catastrophic forgetting without requiring per-task parameter isolation.

4.4.4 Theoretical Analysis

We formalize interference through subspace overlap. For task $\mathcal{T}$, define the effective update $\Delta W = A\Delta B$. Interference between tasks i and j is captured by the Frobenius inner product $\langle \Delta W_i, \Delta W_j \rangle_F$.

Proposition 1. Let $\Delta W_i = A\Delta B_i$ and $\Delta W_j = A\Delta B_j$ for two tasks, with $A^\top A = I_r$. Let $s_i = \|\Delta B_i\|_{2,1}$ and $s_j = \|\Delta B_j\|_{2,1}$. Then

$$|\langle \Delta W_i, \Delta W_j \rangle_F| \leq s_i s_j. \quad (4.39)$$

Proof. Starting from the definition,

$$\langle \Delta W_i, \Delta W_j \rangle_F = \text{Tr}((\Delta W_i)^\top \Delta W_j) = \text{Tr}((A\Delta B_i)^\top (A\Delta B_j)). \quad (4.40)$$

Using $A^\top A = I_r$, this becomes $\text{Tr}(\Delta B_i^\top \Delta B_j)$, which decomposes row-wise:

$$\text{Tr}(\Delta B_i^\top \Delta B_j) = \sum_{l=1}^r \langle (\Delta B_i)_{l,:}, (\Delta B_j)_{l,:} \rangle. \quad (4.41)$$

Applying the triangle inequality and Cauchy, Schwarz,

$$\left| \sum_{l=1}^r \langle (\Delta B_i)_{l,:}, (\Delta B_j)_{l,:} \rangle \right| \leq \sum_{l=1}^r \|(\Delta B_i)_{l,:}\|_2 \|(\Delta B_j)_{l,:}\|_2. \quad (4.42)$$

Let $u_l = \|(\Delta B_i)_{l,:}\|_2$ and $v_l = \|(\Delta B_j)_{l,:}\|_2$ (non-negative). Then $\sum_l u_l v_l \leq (\sum_l u_l)(\sum_l v_l) = \|\Delta B_i\|_{2,1} \|\Delta B_j\|_{2,1} = s_i s_j$, proving Eq. 4.39. $\square$

Interpretation. Eq. 4.42 shows that interference is driven by shared active rows: the product $\|(\Delta B_i)_{l,:}\|_2 \|(\Delta B_j)_{l,:}\|_2$ is non-zero only when both tasks use rank dimension l . Compared to standard LoRA (where $A^\top A \neq I$ and cross-dimension correlations appear), SO-LoRA isolates geometry in the coefficient space. Minimizing the $\ell_{2,1}$ norm in Eq. 4.34 therefore directly suppresses overlap and tightens the bound in Eq. 4.39.

4.4.5 Experimental results

We evaluate SO-LoRA on standard class-incremental learning (CIL) benchmarks and compare against strong rehearsal-free prompt- and LoRA-based baselines. Across longer task sequences and severe distribution shifts, SO-LoRA achieves higher retention and a steadier learning trajectory, while keeping a constant adapter footprint.

4.4.5.1 Experimental Setup

Datasets. We use four CIL benchmarks with disjoint class splits per task. ImageNet-R [42] is our primary scalability benchmark, evaluated under 5/10/20-task partitions. To test robustness and granularity, we additionally report results on ImageNet-A [43], CIFAR-100 [65], and CUB-200 [118], each configured as a 10-task sequence.

Metrics. Let $a_{t,i}$ be the accuracy on task i after training through task t . We report (i) Average Accuracy $\text{Acc} = \frac{1}{T} \sum_{i=1}^T a_{T,i}$, which measures final performance after the full stream, and (ii) Average Anytime Accuracy $\text{AAA} = \frac{1}{T} \sum_{t=1}^T \frac{1}{t} \sum_{i=1}^t a_{t,i}$, which summarizes stability at every task boundary.

Baselines. We compare with representative rehearsal-free prompt methods (L2P [125], DualPrompt [124], CODA-Prompt [111], HiDe-Prompt [121]) and PEFT baselines (O-LoRA [123], InfLoRA [73], SD-LoRA [131]).

4.4.5.2 Implementation Details

Training. We follow prior PEFT-CIL settings: a ViT-B/16 [28] backbone pre-trained on ImageNet-21k [25] is frozen, and SO-LoRA is inserted into the query/value projections with rank $r=40$. We minimize cross-entropy plus an $\ell_{2,1}$ penalty ($\lambda=10^{-4}$) using Adam for 15 epochs per task (learning rate 5×10^{-3} , batch size 16). The mask update uses $\alpha=0.8$. We report mean $\pm$ std over five random seeds.

Inference. To reduce classifier bias in incremental evaluation, we use Nearest Class Mean (NCM). After each task, class prototypes are computed as $\mu_c = \frac{1}{|\mathcal{D}_c|} \sum_{x \in \mathcal{D}_c} f_\theta(x)$, and prediction uses cosine similarity $\hat{y} = \arg \max_c \frac{f_\theta(x)^\top \mu_c}{\|f_\theta(x)\| \|\mu_c\|}$.

4.4.5.3 Main Results

Tables 4.11 4.12 summarize the main comparison between comprehensive analysis of SO-LoRA against current state-of-the-art rehearsal-free methods.

Table 4.11: Performance on ImageNet-R under different task granularities. Best is bold.

Method	IN-R (5)		IN-R (10)		IN-R (20)	
	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$
L2P	71.54 (0.64)	66.30 (0.35)	70.62 (0.23)	64.59 (0.48)	67.35 (0.53)	59.97 (0.31)
DualPrompt	73.28 (0.41)	71.43 (0.46)	71.23 (0.74)	67.72 (0.54)	68.61 (0.74)	62.88 (0.80)
CODA	78.68 (0.23)	75.29 (0.52)	77.11 (0.71)	73.03 (0.63)	73.71 (0.55)	68.53 (0.61)
HiDe	77.42 (0.21)	72.45 (0.31)	76.84 (0.35)	72.63 (0.61)	75.88 (0.25)	71.56 (0.66)
O-LoRA	76.71 (1.81)	72.76 (1.52)	71.30 (1.02)	64.13 (1.86)	68.04 (0.36)	60.98 (0.98)
InfLoRA	81.63 (0.27)	77.83 (0.33)	80.24 (0.36)	77.04 (0.68)	76.78 (0.86)	70.94 (0.90)
SD-LoRA	82.85 (0.40)	79.02 (0.28)	81.05 (0.42)	77.92 (0.22)	80.62 (0.50)	75.43 (0.57)
SO-LoRA	83.94 (0.24)	79.78 (0.37)	83.71 (0.25)	78.83 (0.21)	83.18 (0.29)	78.15 (0.12)

Scalability and Long-horizon retention. SO-LoRA ranks first on ImageNet-R for all task granularities. Its degradation from 5 to 20 tasks is small (Acc: from 79.78% to 78.15%), whereas O-LoRA drops sharply (Acc: from 72.76% to 60.98%), consistent with subspace drift over long horizons. In the hardest 20-task setting, SO-LoRA improves over SD-LoRA by +2.72 Acc, suggesting better scalability when a single shared adapter must serve all tasks.

Table 4.12: Performance on ImageNet-A, CIFAR-100, and CUB-200 (10 tasks). Best is bold.

Method	IN-A		C100		CUB	
	AAA↑	Acc↑	AAA↑	Acc↑	AAA↑	Acc↑
L2P	50.61 (0.21)	41.17 (0.39)	87.58 (0.52)	82.76 (0.98)	69.30 (0.39)	56.48 (0.29)
DualPrompt	54.86 (0.53)	43.07 (0.29)	90.57 (0.23)	83.88 (0.48)	78.57 (0.54)	67.26 (0.49)
CODA	56.30 (0.84)	45.94 (0.49)	91.37 (0.27)	86.82 (0.69)	82.36 (0.40)	72.41 (0.63)
HiDe	55.72 (0.44)	44.67 (0.82)	91.17 (0.34)	86.53 (0.31)	81.71 (0.51)	74.15 (0.72)
O-LoRA	54.17 (0.81)	44.46 (1.48)	89.96 (1.43)	82.51 (1.52)	79.66 (1.57)	70.12 (0.40)
InfLoRA	61.63 (0.19)	51.02 (0.26)	91.44 (0.46)	86.09 (0.28)	81.43 (0.46)	71.13 (0.67)
SD-LoRA	63.24 (0.20)	55.35 (0.47)	91.71 (0.33)	87.28 (0.23)	85.15 (0.55)	77.38 (0.35)
SO-LoRA	67.89 (0.63)	60.27 (0.28)	90.88 (0.17)	86.61 (0.06)	87.71 (0.24)	81.58 (0.43)

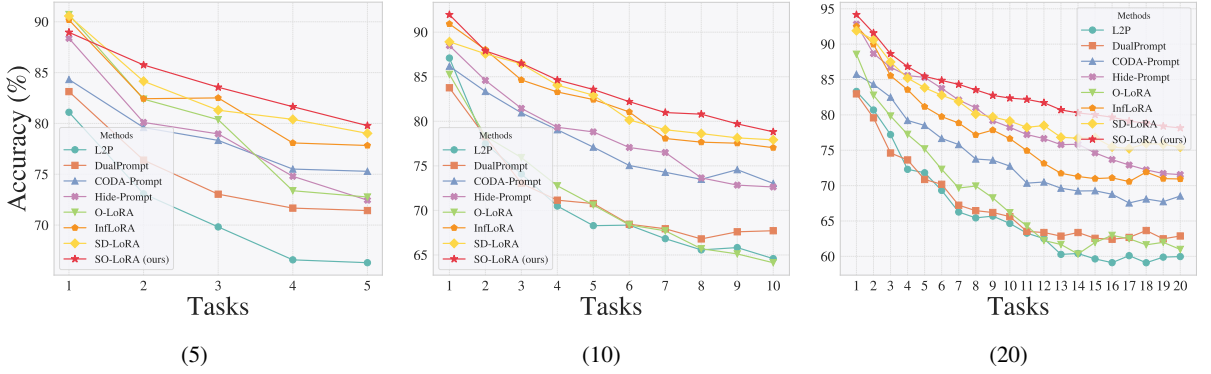


Figure 4.9: Average accuracy on ImageNet-R across sequence lengths.

Robustness under domain shift. ImageNet-A induces a substantial distribution shift. SO-LoRA achieves the best performance (Acc 60.27, AAA 67.89), exceeding SD-LoRA by nearly five Acc points. This gap supports the role of the gradient-aware mask: when current data depart from the pre-training manifold, explicitly protecting historically important directions becomes critical and prompt-only strategies become less reliable.

Fine-grained transfer. On CUB-200, SO-LoRA attains the highest accuracy (81.58), indicating that sparse recombinations over an orthogonal basis can capture subtle cues without amplifying interference. On CIFAR-100, SO-LoRA remains competitive and close to the best baseline, suggesting that the proposed constraints preserve plasticity.

Table 4.13: Ablation on 10-task benchmarks. Impact of group sparsity and gradient-aware mask.

Method	ImageNet-A		ImageNet-R	
	AAA	Acc	AAA	Acc
SO-LoRA	67.89 _(0.63)	60.27 _(0.28)	83.71 _(0.25)	78.83 _(0.21)
w/o sparsity	66.86 _(0.96)	59.30 _(0.54)	83.52 _(0.28)	78.77 _(0.31)
w/o mask	66.31 _(1.25)	58.15 _(0.83)	83.40 _(0.24)	78.37 _(0.14)

Table 4.14: Ablation across sequence lengths on ImageNet-R.

Method	5 tasks		20 tasks	
	AAA	Acc	AAA	Acc
SO-LoRA	83.94 _(0.24)	79.78 _(0.37)	83.18 _(0.29)	78.15 _(0.12)
w/o sparsity	83.11 _(0.59)	79.71 _(0.38)	83.20 _(0.33)	78.07 _(0.14)
w/o mask	83.41 _(0.32)	79.03 _(0.34)	82.66 _(0.31)	77.30 _(0.29)

4.4.5.4 Ablation Study

We ablate group sparsity and the gradient-aware mask to isolate their effects on retention (Acc) and stability (AAA).

Effect of group sparsity. On ImageNet-R, removing $\ell_{2,1}$ only marginally changes performance, suggesting that the fixed orthogonal basis already provides strong geometric separation for in-domain tasks. Under the larger shift of ImageNet-A, sparsity becomes more beneficial (Acc: from 60.27% to 59.30%), consistent with row-sparse task codes reducing overlap when tasks are less aligned.

Effect of gradient-aware masking. Masking yields the most consistent gains. Removing it causes the largest drop on ImageNet-A (Acc: from 60.27% to 58.15%) and a clearer penalty on ImageNet-R with 20 tasks (Acc: from 78.15% to 77.30%), indicating that long horizons still benefit from explicitly protecting historically important directions even within an orthogonal parameterization.

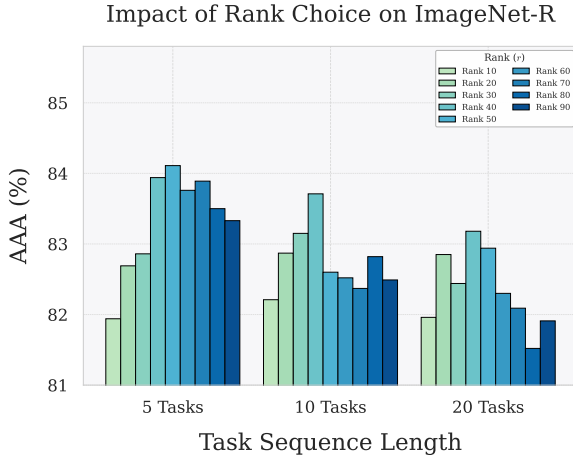


Figure 4.10: **Rank sensitivity.** Performance peaks at $r = 40$, balancing capacity and interference.

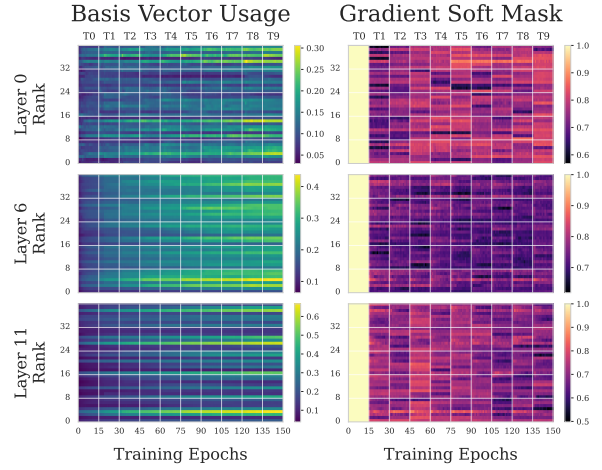


Figure 4.11: **Internal dynamics.** Basis usage (Left) and gradient masks (Right) show dynamic allocation.

4.4.5.5 Discussion

Rank Sensitivity and Basis Capacity. We vary the subspace rank $r \in [10, 90]$ (Fig. 4.10) to quantify the capacity–interference trade-off. Accuracy rises quickly for small ranks, suggesting that a minimal orthogonal “vocabulary” is needed to cover ImageNet-R variation. Gains saturate around $r = 40$, and performance declines once $r \gtrsim 50$. Larger bases increase degrees of freedom and weaken the practical effect of sparsity, allowing task solutions to spread across many directions and increasing overlap. We therefore set $r = 40$ as a stable operating point: enough capacity to model new tasks without inviting unnecessary interference.

Basis Vector Usage and Subspace Evolution. Figure 4.11 (Left) visualizes the evolution of $B^{(t)}$ via row-norm heatmaps. The clear banded structure indicates that each task activates only a small subset of basis directions, consistent with the intended group sparsity behavior. Notably, inactive directions early in training become selectively recruited later, implying a “reserve” of capacity that is gradually allocated as new concepts arrive. This staggered recruitment helps avoid early saturation of the fixed basis and supports longer task horizons.

Basis Vector Usage and Subspace Evolution. Figure 4.11 (Right) shows how the gradient-aware mask distributes stability and plasticity across layers. Early layers exhibit sharper separation between frozen and adaptable dimensions, consistent with protecting general low-level primitives (edges/textures) once they stabilize ($S_j \approx 0$ for high-importance directions). Deeper layers display smoother, less binary masks, reflecting

the need for controlled adaptation of higher-level semantics that may partially overlap across tasks. Overall, the mask behaves as intended: it strongly protects historically sensitive directions while leaving sufficient flexibility where transfer and re-organization are beneficial.

4.5 Summary

This chapter presents three innovative dynamic parameter allocation strategies for architecture-based continual learning that address catastrophic forgetting while maintaining computational efficiency. The multi-head approach with the online prototype equilibrium introduces a dual optimization framework that separately handles feature representation and classification layer optimization. The method employs online prototype equilibrium to maintain stable representations and adaptive prototypical feedback to prevent drift in learned features, ensuring effective knowledge retention across sequential tasks. Multi-Head Approach with Energy-Based Latent Alignment proposes an energy-based model for maintaining latent space alignment across tasks. This approach uses energy functions to preserve the geometric relationships of previously learned representations while accommodating new task knowledge, preventing the typical latent space drift that occurs in continual learning scenarios. DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic successfully addresses the stability-plasticity tradeoff in continual learning. By integrating Incremental Task Arithmetic (ITA) with Kolmogorov-Arnold Classifiers (KAC), DAKTA achieves superior performance in class-incremental learning scenarios. SO-LoRA is a parameter-efficient continual learning framework that avoids the usual choice between shared-adapter interference and per-task memory growth. By representing updates as sparse compositions over a fixed orthogonal basis and modulating learning with a gradient-aware soft mask, SO-LoRA limits destructive overlap while keeping constant adapter size. Comprehensive experiments demonstrate that these dynamic allocation strategies significantly outperform traditional architecture-based methods by reducing computational overhead, preventing catastrophic forgetting, and enabling effective cross-task knowledge transfer. The proposed approaches provide practical solutions for scalable continual learning in resource-constrained environments while maintaining competitive performance across diverse sequential learning scenarios.

The contribution of this chapter was published in [P4], [P5], [P6] and [P7].

Algorithm 6 SO-LoRA Training for a New Task $\mathcal{T}_t$

Input: Pre-trained weights W_0 , frozen basis A , shared matrix $B_{\text{shared},t-1}$, mask M_{t-1} , task data $\mathcal{D}_t$.

Output: Updated $B_{\text{shared},t}$, updated mask M_t , class prototypes P_t .

```
1:  $B_{\text{shared}} \leftarrow B_{\text{shared},t-1}$ 
2: Extend classifier head  $h_\phi$  for new classes in  $C_t$ 
3: Freeze  $W_0$ ,  $A$ , and past classifier weights
4: for each epoch do
5:   for each batch  $(x, y) \in \mathcal{D}_t$  do
6:                                      $\triangleright$  Forward
7:      $W_{\text{adapted}} \leftarrow W_0 + AB_{\text{shared}}$ 
8:      $f(x) \leftarrow \text{Backbone}(x; W_{\text{adapted}})$ 
9:      $\hat{y} \leftarrow h_\phi(f(x))$ 
10:                                      $\triangleright$  Gradients
11:      $\mathcal{L}_{\text{task}} \leftarrow \mathcal{L}_{\text{CE}}(\hat{y}, y)$ 
12:      $g_{\text{task}} \leftarrow \nabla_{B_{\text{shared}}} \mathcal{L}_{\text{task}}$ 
13:      $g_{\text{reg}} \leftarrow \nabla_{B_{\text{shared}}} (\lambda \|B_{\text{shared}}\|_{2,1})$ 
14:                                      $\triangleright$  Gradient-aware modulation
15:     for  $i \in \{1, \dots, r\}$  do
16:        $\phi_i \leftarrow \frac{\|(g_{\text{task}})_{i,:}\|_2}{\max_j \|(g_{\text{task}})_{j,:}\|_2 + \epsilon}$ 
17:        $S_i \leftarrow 1 - (M_{t-1,i}(1 - \phi_i))$ 
18:        $(g'_{\text{task}})_{i,:} \leftarrow S_i \cdot (g_{\text{task}})_{i,:}$ 
19:     end for
20:      $g' \leftarrow g'_{\text{task}} + g_{\text{reg}}$ 
21:     Update  $B_{\text{shared}}$  and new classifier weights using  $g'$ 
22:   end for
23: end for
24:  $B_{\text{shared},t} \leftarrow B_{\text{shared}}$ 
25:                                      $\triangleright$  Prototypes (NCM)
26: For each class  $c \in C_t$ :  $\mu_c \leftarrow \frac{1}{|\mathcal{D}_c|} \sum_{(x_i, y_i) \in \mathcal{D}_c} \text{Backbone}(x_i; W_0 + AB_{\text{shared},t})$ 
27:  $P_t \leftarrow \{\mu_c\}_{c \in C_t}$ 
28:                                      $\triangleright$  Mask update (EMA of row-gradient norms)
29: Compute  $g_{\text{final}} \leftarrow \nabla_{B_{\text{shared}}} \mathcal{L}_{\text{CE}}$  over  $\mathcal{D}_t$  using  $B_{\text{shared},t}$ 
30: For  $i \in \{1, \dots, r\}$ :  $\omega_{t,i} \leftarrow \|(g_{\text{final}})_{i,:}\|_2$ 
31:  $M_t \leftarrow \alpha \cdot \frac{\omega_t}{\max_j (\omega_{t,j}) + \epsilon} + (1 - \alpha) \cdot M_{t-1}$ 
```

Conclusion and Future Work

This dissertation has addressed catastrophic forgetting in continual learning through a three-stage progression motivated by the practical and theoretical limitations that arise at each level of the problem. The contributions are summarized below, together with the specific gap each stage resolves.

Stage 1: Privacy-preserving replay (Chapter 2). The first contribution eliminates the need to store real training samples, a key privacy concern in rehearsal-based methods. Instead, past class distributions are approximated via prototype augmentation with Gaussian noise, preserving enough distributional structure for effective contrastive replay. An energy-based alignment module (ELI) further mitigates representational drift by learning an energy manifold after each task and using Langevin dynamics to realign shifted features. Experiments on FewRel and TACRED confirm that this approach matches or surpasses methods that retain real data.

Stage 2: Exemplar-free knowledge transfer (Chapter 3). The second contribution removes the remaining dependency on stored data entirely. Two methods are proposed. CL-plugin uses a dual-module architecture one for cross-task knowledge sharing, one for task-specific forgetting prevention via neuron masks, combined with contrastive distillation losses, achieving state-of-the-art Macro-F1 on a 19-task sentiment benchmark without any replay data. ZeroRFF extends this to class-incremental vision learning by pairing random Fourier features with a frozen backbone and closed-form updates, remaining competitive even under the challenging Si-Blurry scenario. Together, these methods show that exemplar-free continual learning is viable across both NLP and vision settings.

Stage 3: Adaptive architectures with dynamic parameter allocation (Chapter 4). The third contribution tackles a fundamental bottleneck: a fixed parameter budget struggles to accommodate many diverse tasks without forgetting or underfitting. Four methods address this through complementary strategies. **OPE/APF** disentangles class-

incremental learning into two separate heads, one for OOD detection (task identity), one for within-task classification. OPE maintains discriminative boundaries across all seen classes; APF reinforces boundaries most prone to confusion. Combined with ELI, this yields the best forgetting score on CCH5000. **DAKTA** replaces the standard linear classifier with a Kolmogorov-Arnold Classifier (KAC) using learnable radial basis functions, enabling more expressive local feature-class modeling. Paired with LoRA-based task arithmetic and Fisher Information regularization on a ViT backbone, DAKTA consistently outperforms ITA-LoRA across six benchmarks. **SO-LoRA** offers the most parameter-efficient solution: a shared orthogonal basis is fixed, and each task learns only a sparse composition over it, keeping adapter overhead constant regardless of task count. Group sparsity limits inter-task interference, and a gradient-aware EMA mask protects important rank dimensions. A formal bound on task interference provides theoretical support. SO-LoRA achieves top accuracy on ImageNet-A, CUB-200, and a 20-task ImageNet-R setting among all rehearsal-free baselines.

Unified View: A Progressive Answer to Catastrophic Forgetting

Viewed as a whole, the three stages of this dissertation constitute a progressive answer to catastrophic forgetting that is sensitive to the constraints of different deployment contexts. When data retention is feasible, and privacy constraints permit limited storage, prototype-based replay (Chapter 2) provides the strongest performance guarantee. When even approximate data retention is infeasible, selective knowledge transfer from previously learned models (Chapter 3) offers an effective alternative that is both privacy-preserving and computationally efficient. When tasks are highly diverse, sequences are long, and a fixed parameter budget becomes a binding constraint, adaptive architecture strategies (Chapter 4) provide the only principled path to sustained performance.

The three approaches are not mutually exclusive. The energy-based latent alignment mechanism developed in Chapter 3 is shown in Chapter 4 to improve performance when applied to the multi-head architecture, demonstrating that components transfer across stages. The analytic learning foundation of ZeroRFF (Chapter 3) shares conceptual ground with SO-LoRA’s closed-form stability (Chapter 4). Future work that integrates these stages into a unified, context-adaptive framework, selecting among replay, distillation, and architecture expansion based on available resources and task statistics, represents a natural extension of this research programme.

Limitations

Task boundary assumptions. The majority of methods proposed in this dissertation

operate under the assumption that task boundaries are known during training and, in the task-incremental setting, also during inference. This assumption is violated in many real-world scenarios, where data distributions shift gradually, and no explicit signal demarcates when a new task begins. Extending these methods to the task-free and blurred-boundary settings identified in Chapter 1 remains an open challenge.

Scalability to very long sequences The proposed methods are evaluated on sequences of 5 to 20 tasks. Scalability to hundreds or thousands of tasks has not been systematically investigated. In prototype-based methods, the number of stored prototypes grows linearly with the number of classes. In knowledge distillation methods, the computational cost of distillation from all previous tasks grows with sequence length. In LoRA-based methods, the rank sensitivity analysis suggests that interference increases with the number of tasks even when sparsity is enforced, though the theoretical bound in SO-LoRA provides a path toward characterizing this degradation.

Domain coverage The empirical evaluation covers text classification (relation extraction, aspect sentiment classification) and image classification (natural images, fine-grained recognition, medical imaging). Other modalities, such as audio, time series, multimodal data and other task types of object detection, semantic segmentation, and generative modelling, are not addressed. The generalizability of the proposed frameworks to these settings is an important open question.

Inconsistent backbone selection in Chapter 4. The multi-head and ELI methods use a ResNet-18 backbone, while DAKTA and SO-LoRA use a ViT-B/16 backbone pre-trained on ImageNet-21k. This difference reflects the distinct methodological lineages of the methods but limits direct cross-method comparison within Chapter 4. A unified evaluation on a single backbone would clarify the relative contributions of the architectural innovations versus the backbone capacity.

Future Research Directions

Unified context-adaptive framework. A natural next step is a meta-framework that automatically selects among replay, knowledge transfer, and architecture expansion based on task statistics such as distribution shift, memory budget, and task similarity. Reinforcement learning or Bayesian optimization could drive this selection, replacing fixed design-time commitments with deployment-aware adaptation.

Task-agnostic and online continual learning. Removing the assumption of known task boundaries requires automatic boundary detection and online prototype updating.

The OPE mechanism from Chapter 4 offers a natural starting point, as its prototype distance signals can detect distribution shifts; pairing OPE with a clustering-based discovery module could enable task-free operation.

Continual learning for large language models. The LoRA-based methods in Chapter 4 (DAKTA and SO-LoRA) transfer directly to LLMs, where full fine-tuning is infeasible. Key questions include how sparse orthogonal adaptation scales to billion-parameter models, how Fisher Information regularization can be efficiently approximated at that scale, and how SO-LoRA’s subspace geometry relates to emerging work on model merging and task vectors.

Multi-modal and cross-domain continual learning. Real-world systems process heterogeneous streams spanning vision, language, and structured data. Extensions to multi-modal settings will require modality-specific parameter allocation and shared representations for cross-modal transfer. SO-LoRA’s orthogonal basis structure is a promising fit, as different modalities naturally occupy distinct parameter subspaces.

Closing Remark. Catastrophic forgetting remains a core obstacle to truly adaptive AI. This dissertation has shown that privacy-preserving replay, exemplar-free knowledge transfer, and adaptive parameter allocation offer viable, complementary solutions across diverse deployment settings. Together, they represent principled steps toward a learning system that accumulates knowledge over its lifetime without forgetting, scales to many tasks, and respects real-world resource constraints.

List of Publications

- [P1] **Quynh-Trang Pham Thi**, Anh-Cuong Pham, Ngoc-Huyen Ngo and Duc-Trong Le. "Memory-Based Method using Prototype Augmentation for Continual Relation Extraction." In 2022 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1-6. IEEE, 2022. Scopus.
- [P2] Thanh-Hai Dang, **Quynh-Trang Pham Thi**, Duc-Trong Le, and Tri-Thanh Nguyen. "Contrastive Learning for Boosting Knowledge Transfer in Task-Incremental Continual Learning of Aspect Sentiment Classification Tasks". VNU Journal of Science: Computer Science and Communication Engineering 40, no. 1 (2024).
- [P3] Duc-Hung Nguyen, Tri-Thanh Nguyen, Thanh-Hai Dang and **Quynh-Trang Pham Thi**. "ZeroRFF: A Random Fourier Features and Analytic Learning Method for Generalized Class Incremental Learning". The 21st International Conference on Advanced Data Mining and Applications 2025. Scopus Rank B.
- [P4] **Quynh-Trang Pham Thi**, Duc-Hung Nguyen, Thanh Hai Dang, Duc-Trong Le, Tri-Thanh Nguyen, and Quang-Thuy Ha. "Towards Robust Continual Learning: A Multi-Head Approach with Online Prototype Equilibrium and Adaptive Prototypical Feedback." In Asian Conference on Intelligent Information and Database Systems, pp. 275-286. Singapore: Springer Nature Singapore, 2024. Scopus Rank B.
- [P5] **Quynh-Trang Pham Thi**, Duc-Hung Nguyen, Anh-Duc Nguyen-Tran, Hung-Dung Nguyen, Tri-Thanh Nguyen, Quang-Thuy Ha and Thanh Hai Dang. "Addressing Catastrophic Forgetting in Continual Learning using Multihead Approach and Latent Alignment via Energy-Based Model." Vietnam journal of computer science. (Accepted). WoS Q3.
- [P6] **Quynh-Trang Pham Thi**, Van-Truong Le, Thanh-Huyen Pham, Thanh-Nga Hoang Thi and Duc-Trong Le. "DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic in Continual Learning". SOICT 2025. Scopus.

- [P7] **Quynh-Trang Pham Thi**, Quang-Dat Mac, Thanh-Ha Nguyen and Duc-Trong Le
."SO-LoRA: Sparse Orthogonal LoRA for Parameter-Efficient Continual Learning".
In 2026 IEEE International Conference on Multimedia and Expo (ICME 2026).
Scopus Rank A.

The list contains seven publications.

References

- [1] H. Ahn, S. Cha, D. Lee, and T. Moon, “Uncertainty-based continual learning with adaptive regularization,” Advances in neural information processing systems, vol. 32, 2019.
- [2] R. Aljundi, P. Chakravarty, and T. Tuytelaars, “Expert gate: Lifelong learning with a network of experts,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7120–7129. [Online]. Available: <https://ieeexplore.ieee.org/document/8100236>
- [3] R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin, and L. PagoCaccia, “Online continual learning with maximal interfered retrieval,” Advances in neural information processing systems, vol. 32, 2019.
- [4] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio, “Gradient based sample selection for online continual learning,” Advances in neural information processing systems, vol. 32, pp. 11 817 – 11 826, 2019.
- [5] R. Aljundi et al., “A statistical theory of regularization-based continual learning,” in International Conference on Machine Learning. PMLR, 2024. [Online]. Available: <https://icml.cc/virtual/2024/poster/34787>
- [6] N. Asadi, M. Davari, S. Mudur, R. Aljundi, and E. Belilovsky, “Prototype-sample relation distillation: towards replay-free continual learning,” in International conference on machine learning. PMLR, 2023, pp. 1093–1106. [Online]. Available: <https://dl.acm.org/doi/10.5555/3618408.3618453>
- [7] L. Baldini Soares, N. FitzGerald, J. Ling, and T. Kwiatkowski, “Matching the blanks: Distributional similarity for relation learning,” in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, A. Korhonen, D. Traum, and L. Màrquez, Eds. Florence, Italy: Association

for Computational Linguistics, Jul. 2019, pp. 2895–2905. [Online]. Available: <https://aclanthology.org/P19-1279/>

- [8] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, “Rainbow memory: Continual learning with a memory of diverse samples,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 8218–8227.
- [9] D. Benavides-Prado and P. Riddle, “A theory for knowledge transfer in continual learning,” in Conference on Lifelong Learning Agents. PMLR, 2022, pp. 647–660.
- [10] M. Boschini, L. Bonicelli, A. Porrello, G. Bellitto, M. Pennisi, S. Palazzo, C. Spampinato, and S. Calderara, “Transfer without forgetting,” in European conference on computer vision. Springer, 2022, pp. 692–709. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-20050-2_40
- [11] B. Bowman, A. Achille, L. Zancato, M. Trager, P. Perera, G. Paolini, and S. Soatto, “a-la-carte prompt tuning (apt): Combining distinct data via composable prompting,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 14 984–14 993.
- [12] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, “Dark experience for general continual learning: a strong, simple baseline,” in Proceedings of the 34th International Conference on Neural Information Processing Systems, ser. NIPS ’20. Red Hook, NY, USA: Curran Associates Inc., 2020.
- [13] M. Caccia, P. Rodríguez, O. Ostapenko, F. Normandin, M. Lin, L. Caccia, I. Laradji, I. Rish, A. Lacoste, D. Vazquez, and L. Charlin, “Online fast adaptation and knowledge accumulation (osaka): a new approach to continual learning,” in Proceedings of the 34th International Conference on Neural Information Processing Systems, ser. NIPS ’20. Red Hook, NY, USA: Curran Associates Inc., 2020.
- [14] F. Cermelli, M. Mancini, S. R. Buló, E. Ricci, and B. Caputo, “Modeling the background for incremental learning in semantic segmentation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9233–9242.
- [15] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. Dokania, P. Torr, and M. Ranzato, “Continual learning with tiny episodic memories,” in Workshop

on Multi-Task and Lifelong Reinforcement Learning, 2019. [Online]. Available: <https://ora.ox.ac.uk/objects/uuid:6e7580c4-85c9-4874-a52d-e4184046935c/files/ma2bb7107b453e6b1081bba3b1c35f894>

- [16] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, “Riemannian walk for incremental learning: Understanding forgetting and intransigence,” in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 532–547.
- [17] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, “Efficient lifelong learning with a-gem,” in International Conference on Learning Representations, 2019. [Online]. Available: <https://ai.meta.com/research/publications/efficient-lifelong-learning-with-a-gem/>
- [18] E. Chee, M. L. Lee, and W. Hsu, “Leveraging old knowledge to continually learn new classes in medical images,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 12, June 2023, pp. 14 178–14 186.
- [19] T. Chen, I. Goodfellow, and J. Shlens, “Net2net: Accelerating learning via knowledge transfer,” in International Conference on Learning Representations, 2016. [Online]. Available: <http://arxiv.org/abs/1511.05641>
- [20] Z. Chen and B. Liu, Lifelong machine learning. Springer, 2018, vol. 1.
- [21] J. M. Coria, “Continual representation learning in written and spoken language,” Ph.D. dissertation, Université Paris-Saclay, 2023.
- [22] L. Cui, D. Yang, J. Yu, C. Hu, J. Cheng, J. Yi, and Y. Xiao, “Refining sample embeddings with relation prototypes to enhance continual relation extraction,” in Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2021, pp. 232–243.
- [23] M. Davari, N. Asadi, S. Mudur, R. Aljundi, and E. Belilovsky, “Probing representation forgetting in supervised and unsupervised continual learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 16 712–16 721.
- [24] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, “A continual learning survey: Defying forgetting in classification tasks,” IEEE Transactions on Pattern Analysis and

Machine Intelligence, vol. 44, no. 7, pp. 3366–3385, 2021. [Online]. Available: <https://doi.org/10.1109/TPAMI.2021.3057446>

- [25] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2009, pp. 248–255.
- [26] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” CoRR, vol. abs/1810.04805, 2018. [Online]. Available: <http://arxiv.org/abs/1810.04805>
- [27] X. Ding, B. Liu, and P. S. Yu, “A holistic lexicon-based approach to opinion mining,” in Proceedings of the 2008 ACM Conference on Web Search and Data Mining, ser. WSDM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 231–240. [Online]. Available: <https://doi.org/10.1145/1341531.1341561>
- [28] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Int. Conf. Learn. Represent. (ICLR), 2021.
- [29] A. Douillard, M. Cord, C. Ollion, T. Robert, and E. Valle, “Podnet: Pooled outputs distillation for small-tasks incremental learning,” in Computer vision—ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part XX 16. Springer International Publishing, 2020, pp. 86–102.
- [30] A. El Khatib, “Continual learning and forgetting in deep learning models,” Ph.D. dissertation, University of Waterloo, 2020.
- [31] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra, “Pathnet: Evolution channels gradient descent in super neural networks,” 2017.
- [32] R. M. French, “Catastrophic forgetting in connectionist networks,” Trends in Cognitive Sciences, vol. 3, no. 4, pp. 128–135, 1999. [Online]. Available: [https://doi.org/10.1016/S1364-6613\(99\)01294-2](https://doi.org/10.1016/S1364-6613(99)01294-2)
- [33] T. Gao, X. Han, H. Zhu, Z. Liu, P. Li, M. Sun, and J. Zhou, “FewRel 2.0: Towards more challenging few-shot relation classification,” in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing

- (EMNLP-IJCNLP), K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6250–6255. [Online]. Available: <https://aclanthology.org/D19-1649/>
- [34] Y. Gu, X. Yang, K. Wei, and C. Deng, “Not just selection, but exploration: Online class-incremental continual learning via dual view consistency,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 7442–7451.
- [35] P. Guo and M. R. Lyu, “A pseudoinverse learning algorithm for feedforward neural networks with stacked generalization applications to software reliability growth data,” Neurocomputing, vol. 56, pp. 101–121, 2004.
- [36] Y. Guo, B. Liu, and D. Zhao, “Online continual learning through mutual information maximization,” in International Conference on Machine Learning. PMLR, 2022, pp. 8109–8126.
- [37] M. B. Gurbuz and C. Dovrolis, “Nispa: Neuro-inspired stability-plasticity adaptation for continual learning in sparse networks,” in International Conference on Machine Learning. PMLR, 2022, pp. 8157–8174. [Online]. Available: <https://proceedings.mlr.press/v162/gurbuz22a/gurbuz22a.pdf>
- [38] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, “Embracing change: Continual learning in deep neural networks,” Trends in cognitive sciences, vol. 24, no. 12, pp. 1028–1040, 2020.
- [39] X. Han, H. Zhu, P. Yu, Z. Wang, Y. Yao, Z. Liu, and M. Sun, “FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4803–4809. [Online]. Available: <https://aclanthology.org/D18-1514/>
- [40] X. Han, Y. Dai, T. Gao, Y. Lin, Z. Liu, P. Li, M. Sun, and J. Zhou, “Continual relation learning via episodic memory activation and reconsolidation,” in Proceedings of the 58th annual meeting of the association for computational linguistics, 2020, pp. 6429–6440.

- [41] T. L. Hayes and C. Kanan, “Lifelong machine learning with deep streaming linear discriminant analysis,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, 2020, pp. 220–221.
- [42] D. Hendrycks et al., “The many faces of robustness: A critical analysis of out-of-distribution generalization,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), 2021, pp. 8340–8349.
- [43] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, “Natural adversarial examples,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 15 262–15 271. [Online]. Available: <https://ieeexplore.ieee.org/document/9578772>
- [44] G. E. Hinton, A. Krizhevsky, and S. D. Wang, “Transforming auto-encoders,” in International conference on artificial neural networks. Springer, 2011, pp. 44–51.
- [45] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, “Learning a unified classifier incrementally via rebalancing,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 831–839.
- [46] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, “Parameter-efficient transfer learning for NLP,” in Proceedings of the 36th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 2790–2799. [Online]. Available: <https://proceedings.mlr.press/v97/houlsby19a.html>
- [47] E. J. Hu et al., “LoRA: Low-rank adaptation of large language models,” in Int. Conf. Learn. Represent. (ICLR), 2022.
- [48] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen et al., “Lora: Low-rank adaptation of large language models.” ICLR, vol. 1, no. 2, p. 3, 2022.
- [49] M. Hu and B. Liu, “Mining and summarizing customer reviews,” in Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 168–177. [Online]. Available: <https://doi.org/10.1145/1014052.1014073>

- [50] Y. Y. Huang and W. Y. Wang, “Deep residual learning for weakly-supervised relation extraction,” in Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, M. Palmer, R. Hwa, and S. Riedel, Eds. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 1803–1807. [Online]. Available: <https://aclanthology.org/D17-1191/>
- [51] Q. Jiang, L. Chen, R. Xu, X. Ao, and M. Yang, “A challenge dataset and effective models for aspect-based sentiment analysis,” in Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), 2019, pp. 6280–6285.
- [52] K. Joseph, S. Khan, F. S. Khan, R. M. Anwer, and V. N. Balasubramanian, “Energy-based latent aligner for incremental learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 7452–7461. [Online]. Available: <https://www.computer.org/csdl/proceedings-article/cvpr/2022/694600h442/1H0KP8DVARO>
- [53] J. N. Kather, C.-A. Weis, F. Bianconi, S. M. Melchers, L. R. Schad, T. Gaiser, A. Marx, and F. G. Zöllner, “Multi-class texture analysis in colorectal cancer histology,” Scientific reports, vol. 6, no. 1, pp. 1–11, 2016.
- [54] Z. Ke, B. Liu, and X. Huang, “Continual learning of a mixed sequence of similar and dissimilar tasks,” in Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 18 493–18 504. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/d7488039246a405baf6a7cbc3613a56f-Paper.pdf
- [55] Z. Ke, B. Liu, H. Wang, and L. Shu, “Continual learning with knowledge transfer for sentiment classification,” in Joint European conference on machine learning and knowledge discovery in databases. Springer, 2020, pp. 683–698. [Online]. Available: <https://www.cs.uic.edu/~liub/publications/ECML-PKDD-2020.pdf>
- [56] Z. Ke, B. Liu, N. Ma, H. Xu, and L. Shu, “Achieving forgetting prevention and knowledge transfer in continual learning,” CoRR, vol. abs/2112.02706, 2021. [Online]. Available: https://papers.nips.cc/paper_files/paper/2021/file/bcd0049c35799cdf57d06eaf2eb3cff6-Supplemental.pdf

- [57] —, “Achieving forgetting prevention and knowledge transfer in continual learning,” Advances in Neural Information Processing Systems, vol. 34, pp. 22 443–22 456, 2021. [Online]. Available: https://papers.nips.cc/paper_files/paper/2021/file/bcd0049c35799cdf57d06eaf2eb3cff6-Supplemental.pdf
- [58] Z. Ke, B. Liu, H. Xu, and L. Shu, “CLASSIC: Continual and contrastive learning of aspect sentiment classification tasks,” in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6871–6883. [Online]. Available: <https://aclanthology.org/2021.emnlp-main.550>
- [59] Z. Ke, H. Xu, and B. Liu, “Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks,” CoRR, vol. abs/2112.03271, 2021. [Online]. Available: <https://arxiv.org/abs/2112.03271>
- [60] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, “Supervised contrastive learning,” Advances in neural information processing systems, vol. 33, pp. 18 661–18 673, 2020.
- [61] G. Kim, C. Xiao, T. Konishi, Z. Ke, and B. Liu, “A theoretical study on solving continual learning,” Advances in Neural Information Processing Systems, vol. 35, pp. 5065–5079, 2022.
- [62] G. Kim, C. Xiao, T. Konishi, and B. Liu, “Learnability and algorithm for continual learning,” in International conference on machine learning. PMLR, 2023, pp. 16 877–16 896. [Online]. Available: <https://proceedings.mlr.press/v202/kim23x/kim23x.pdf>
- [63] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska et al., “Overcoming catastrophic forgetting in neural networks,” Proceedings of the national academy of sciences, vol. 114, no. 13, pp. 3521–3526, 2017.
- [64] H. Koh, D. Kim, J. W. Ha, and J. Choi, “Online continual learning on class incremental blurry task configuration with anytime inference,” in 10th International Conference on Learning Representations, ICLR 2022, 2022. [Online]. Available: https://openreview.net/forum?id=nrGGfMbY_qK
- [65] A. Krizhevsky, G. Hinton et al., “Learning multiple layers of features from tiny images,” Technical Report, University of Toronto, 2009.

- [66] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105, 2012.
- [67] D. Kudithipudi, M. Aguilar-Simon, J. Babb, M. Bazhenov, D. Blackiston, J. Bongard, A. P. Brna, S. Chakravarthi Raja, N. Cheney, J. Clune et al., “Biological underpinnings for lifelong learning machines,” Nature Machine Intelligence, vol. 4, no. 3, pp. 196–210, 2022.
- [68] R. Kurle, “Variational bayes for continual learning and time-series forecasting,” Ph.D. dissertation, Technische Universität München, 2023.
- [69] Y. Le and X. Yang, “Tiny imagenet visual recognition challenge,” CS 231N, vol. 7, no. 7, p. 3, 2015.
- [70] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. J. Huang, “A tutorial on energy-based learning,” Predicting Structured Data, vol. 1, no. 0, 2006. [Online]. Available: <http://yann.lecun.com/exdb/publis/pdf/lecun-06.pdf>
- [71] T. Lesort, “Continual learning: Tackling catastrophic forgetting in deep neural networks with replay processes,” Ph.D. dissertation, Institut polytechnique de Paris, 2020.
- [72] Z. Li and D. Hoiem, “Learning without forgetting,” IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 12, pp. 2935–2947, 2017.
- [73] Y.-S. Liang and W.-J. Li, “Inflora: Interference-free low-rank adaptation for continual learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 23 638–23 647.
- [74] T. Y. Liu and S. Soatto, “Tangent model composition for ensembling and continual fine-tuning,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 18 676–18 686.
- [75] T. Liu, L. Ungar, and J. Sedoc, “Continual learning for sentence representations using conceptors,” in 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), 2019, pp. 3274–3279. [Online]. Available: <https://aclanthology.org/N19-1331/>

- [76] V. Lomonaco, L. Pellegrini, A. Cossu, A. Carta, G. Graffieti, T. L. Hayes, M. De Lange, M. Masana, J. Pomponi, G. M. Van de Ven et al., “Avalanche: an end-to-end library for continual learning,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 3600–3610.
- [77] D. Lopez-Paz and M. Ranzato, “Gradient episodic memory for continual learning,” Advances in neural information processing systems, vol. 30, 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf
- [78] Z. Mai, R. Li, H. Kim, and S. Sanner, “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, 2021, pp. 3589–3599. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021W/CLVision/papers/Mai_Supervised_Contrastive_Replay_Revisiting_the_Nearest_Class_Mean_Classifier_in_CVPRW_2021_paper.pdf
- [79] A. Mallya and S. Lazebnik, “Packnet: Adding multiple tasks to a single network by iterative pruning,” in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 7765–7773. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/papers/Mallya_PackNet_Adding_Multiple_CVPR_2018_paper.pdf
- [80] A. Mallya, D. Davis, and S. Lazebnik, “Piggyback: Adapting a single network to multiple tasks by learning to mask weights,” in Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV. Berlin, Heidelberg: Springer-Verlag, 2018, p. 72–88. [Online]. Available: https://doi.org/10.1007/978-3-030-01225-0_5
- [81] J. L. McClelland, B. L. McNaughton, and R. C. O’Reilly, “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, vol. 102, no. 3, p. 419, 1995.
- [82] M. McCloskey and N. J. Cohen, “Catastrophic interference in connectionist networks: The sequential learning problem,” Psychology of Learning and Motivation, vol. 24, pp. 109–165, 1989. [Online]. Available: [https://doi.org/10.1016/S0079-7421\(08\)60536-8](https://doi.org/10.1016/S0079-7421(08)60536-8)

- [83] M. D. McDonnell, D. Gong, A. Parvaneh, E. Abbasnejad, and A. Van den Hengel, “Ranpac: Random projections and pre-trained models for continual learning,” Advances in Neural Information Processing Systems, vol. 36, pp. 12 022–12 053, 2023.
- [84] M. Mermillod, A. Bugaiska, and P. Bonin, “The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects,” p. 504, 2013.
- [85] F. Mi, “Lifelong machine learning with data efficiency and knowledge retention,” EPFL, Tech. Rep., 2021.
- [86] S. I. Mirzadeh, M. Farajtabar, R. Pascanu, and H. Ghasemzadeh, “Understanding the role of training regimes in continual learning,” Advances in Neural Information Processing Systems, vol. 33, pp. 7308–7320, 2020.
- [87] J.-Y. Moon, K.-H. Park, J. U. Kim, and G.-M. Park, “Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning,” in Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 11 731–11 741.
- [88] N. S. Nayak, K. Killamsetty, L. Han, A. Bhandwaldar, P. Chanda, K. Xu, O. Silkin, M. Eyceoz, H. Wang, A. Pareja et al., “Sculpting subspaces: Constrained full fine-tuning in llms for continual learning,” in International Conference on Learning Representations, vol. 2026, 2026, pp. 20 212–20 243.
- [89] R. M. Neal, “Mcmc using hamiltonian dynamics,” Handbook of Markov Chain Monte Carlo, pp. 113–162, 2011.
- [90] G. Ortiz-Jimenez, A. Favero, and P. Frossard, “Task arithmetic in the tangent space: Improved editing of pre-trained models,” Advances in Neural Information Processing Systems, vol. 36, pp. 66 727–66 754, 2023.
- [91] S. J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Transactions on knowledge and data engineering, vol. 22, no. 10, pp. 1345–1359, 2009.
- [92] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, “Continual lifelong learning with neural networks: A review,” Neural Networks, vol. 113, pp. 54–71, 2019. [Online]. Available: <https://doi.org/10.1016/j.neunet.2019.01.012>

- [93] L. Pellegrini, G. Graffieti, V. Lomonaco, and D. Maltoni, “Latent replay for real-time continual learning,” in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 10 203–10 209.
- [94] N. Q. Pham, “Continually learning new languages,” Ph.D. dissertation, KIT - Karlsruher Institut für Technologie, 2023.
- [95] A. Porrello, L. Bonicelli, P. Buzzega, M. Millunzi, S. Calderara, and R. Cucchiara, “A second-order perspective on model compositionality and incremental learning,” 2024.
- [96] A. Prabhu, P. H. Torr, and P. K. Dokania, “Gdumb: A simple approach that questions our progress in continual learning,” in Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. Springer, 2020, pp. 524–540.
- [97] R. Ratcliff, “Connectionist models of recognition memory: constraints imposed by learning and forgetting functions.” Psychological review, vol. 97, no. 2, p. 285, 1990.
- [98] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, “icarl: Incremental classifier and representation learning,” in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 2001–2010. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2017/papers/Rebuffi_iCaRL_Incremental_Classifier_CVPR_2017_paper.pdf
- [99] F. M. Richardson and M. S. Thomas, “Critical periods and catastrophic interference effects in the development of self-organizing feature maps,” Developmental science, vol. 11, no. 3, pp. 371–389, 2008.
- [100] A. Robins, “Catastrophic forgetting, rehearsal and pseudorehearsal,” Connection Science, vol. 7, no. 2, pp. 123–146, 1995.
- [101] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, “Experience replay for continual learning,” Advances in neural information processing systems, vol. 32, 2019.
- [102] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, “Progressive neural networks,” in Proceedings of the 30th International

- Conference on Neural Information Processing Systems, 2016, pp. 523–532. [Online]. Available: <https://proceedings.neurips.cc/paper/2016/hash/0e0b8a3e6b8c0d0c7c8e0e0e0e0e0e0-Abstract.html>
- [103] G. Rypeść, S. Cygert, V. Khan, T. Trzcinski, B. M. Zieliński, and B. Twardowski, “Divide and not forget: Ensemble of selectively trained experts in continual learning,” in The Twelfth International Conference on Learning Representations, 2024.
- [104] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in Proceedings of the 31st International Conference on Neural Information Processing Systems, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 3859–3869. [Online]. Available: <https://dl.acm.org/doi/pdf/10.5555/3294996.3295142>
- [105] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, “Overcoming catastrophic forgetting with hard attention to the task,” in Proceedings of the 35th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 4548–4557. [Online]. Available: <https://proceedings.mlr.press/v80/serra18a.html>
- [106] G. Shi, J. Chen, W. Zhang, L.-M. Zhan, and X.-M. Wu, “Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima,” Advances in neural information processing systems, vol. 34, pp. 6747–6761, 2021.
- [107] H. Shi, Z. Xu, H. Wang, W. Qin, W. Wang, Y. Wang, Z. Wang, S. Ebrahimi, and H. Wang, “Continual learning of large language models: A comprehensive survey,” ACM Comput. Surv., May 2025. [Online]. Available: <https://doi.org/10.1145/3735633>
- [108] D. Shim, Z. Mai, J. Jeong, S. Sanner, H. Kim, and J. Jang, “Online class-incremental continual learning with adversarial shapley value,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 11, 2021, pp. 9630–9638.
- [109] H. Shin, J. K. Lee, J. Kim, and J. Kim, “Continual learning with deep generative replay,” in Advances in Neural Information Processing Systems, vol. 30, 2017. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/hash/0efbe98067c6c73dba1250d2beaa81f9-Abstract.html>

- [110] K. Shmelkov, C. Schmid, and K. Alahari, “Incremental learning of object detectors without catastrophic forgetting,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 3400–3409.
- [111] J. S. Smith, L. Karlinsky, V. Gutta, P. Cascante-Bonilla, D. Kim, A. Arbelle, R. Panda, R. Feris, and Z. Kira, “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, 2023, pp. 11 909–11 919. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2023/papers/Smith_CODA-Prompt_COntinual_Decomposed_Attention-Based_Prompting_for_Rehearsal-Free_Continual_Learning_CVPR_2023_paper.pdf
- [112] J. S. Smith, L. Valkov, S. Halbe, V. Gutta, R. Feris, Z. Kira, and L. Karlinsky, “Adaptive memory replay for continual learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 3605–3615.
- [113] F.-K. Sun, C.-H. Ho, and H. yi Lee, “Lamol: Language modeling for lifelong language learning,” in International Conference on Learning Representations, 2019.
- [114] S. Thrun, “Explanation-based neural network learning,” in Explanation-Based Neural Network Learning: A Lifelong Learning Approach. Springer, 1996, pp. 19–48.
- [115] S. Thrun and A. Bücken, “Integrating grid-based and topological maps for mobile robot navigation,” in Proceedings of the national conference on artificial intelligence, 1996, pp. 944–951.
- [116] G. M. Van de Ven, H. T. Siegelmann, and A. S. Tolias, “Brain-inspired replay for continual learning with artificial neural networks,” Nature communications, vol. 11, no. 1, p. 4069, 2020.
- [117] G. M. van de Ven, T. Tuytelaars, and A. S. Tolias, “Three types of incremental learning,” Nature Machine Intelligence, vol. 4, no. 12, pp. 1185–1197, 2022.
- [118] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011.
- [119] F.-Y. Wang, D.-W. Zhou, H.-J. Ye, and D.-C. Zhan, “Foster: Feature boosting and compression for class-incremental learning,” in European conference on computer vision. Springer, 2022, pp. 398–414.

- [120] H. Wang, W. Xiong, M. Yu, X. Guo, S. Chang, and W. Y. Wang, “Sentence embedding alignment for lifelong relation extraction,” in 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), J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 796–806. [Online]. Available: <https://aclanthology.org/N19-1086/>
- [121] L. Wang, J. Xie, X. Zhang, M. Huang, H. Su, and J. Zhu, “Hierarchical decomposition of prompt-based continual learning,” in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 36, 2023, pp. 60 527–60 544.
- [122] L. Wang, X. Zhang, H. Su, and J. Zhu, “A comprehensive survey of continual learning: Theory, method and application,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024. [Online]. Available: <https://doi.org/10.1109/TPAMI.2024.3367329>
- [123] X. Wang et al., “Orthogonal subspace learning for language model continual learning,” in Findings of the Assoc. Comput. Linguist. (EMNLP), 2023, pp. 10 658–10 671.
- [124] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy et al., “Dualprompt: Complementary prompting for rehearsal-free continual learning,” in European conference on computer vision. Springer, 2022, pp. 631–648.
- [125] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, and T. Pfister, “Learning to prompt for continual learning,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 139–149.
- [126] Y. Wei, J. Ye, Z. Huang, J. Zhang, and H. Shan, “Online prototype learning for online continual learning,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 18 764–18 774. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2023/papers/Wei_Online_Prototype_Learning_for_Online_Continual_Learning_ICCV_2023_paper.pdf
- [127] M. Welling and Y. W. Teh, “Bayesian learning via stochastic gradient langevin dynamics,” in Proceedings of the 28th International Conference

on Machine Learning, 2011, pp. 681–688. [Online]. Available: <https://www.stats.ox.ac.uk/~teh/research/compstats/WelTeh2011a.pdf>

- [128] M. Wortsman, V. Ramanujan, R. Liu, A. Kembhavi, M. Rastegari, J. Yosinski, and A. Farhadi, “Supermasks in superposition,” in Proceedings of the 34th International Conference on Neural Information Processing Systems, ser. NIPS ’20. Red Hook, NY, USA: Curran Associates Inc., 2020.
- [129] C. Wu, L. Herranz, X. Liu, J. Van De Weijer, B. Raducanu et al., “Memory replay gans: Learning to generate new categories without forgetting,” Advances in neural information processing systems, vol. 31, 2018.
- [130] T. Wu, X. Li, Y.-F. Li, G. Haffari, G. Qi, Y. Zhu, and G. Xu, “Curriculum-meta learning for order-robust continual relation extraction,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 12, 2021, pp. 10 363–10 369.
- [131] Y. Wu et al., “Sd-loRA: Scalable decoupled low-rank adaptation for class-incremental learning,” in Int. Conf. Learn. Represent. (ICLR), 2025.
- [132] Y. Wu, H. Piao, L.-K. Huang, R. Wang, W. Li, H. Pfister, D. Meng, K. Ma, and Y. Wei, “Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning,” in The Thirteenth International Conference on Learning Representations, 2025. [Online]. Available: https://proceedings.iclr.cc/paper_files/paper/2025/file/92f43b1d33fae4aa1958f75317f0cec1-Paper-Conference.pdf
- [133] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, “Large scale incremental learning,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 374–382.
- [134] W. Xue and T. Li, “Aspect based sentiment analysis with gated convolutional networks,” in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 2514–2523. [Online]. Available: <https://aclanthology.org/P18-1234>
- [135] S. Yan, J. Xie, and X. He, “Der: Dynamically expandable representation for class incremental learning,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 3014–3023.

- [136] J. Yoon, W. Jeong, G. Lee, E. Yang, and S. J. Hwang, “Federated continual learning with weighted inter-client transfer,” in International conference on machine learning. PMLR, 2021, pp. 12 073–12 086.
- [137] F. Zenke, B. Poole, and S. Ganguli, “Continual learning through synaptic intelligence,” in International Conference on Machine Learning. PMLR, 2017, pp. 3987–3995. [Online]. Available: <https://proceedings.mlr.press/v70/zenke17a>
- [138] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, “Position-aware attention and supervised data improve slot filling,” in Proceedings of the 2017 conference on empirical methods in natural language processing, 2017, pp. 35–45.
- [139] K. Zhao, H. Xu, J. Yang, and K. Gao, “Consistent representation learning for continual relation extraction,” in Findings of the association for computational linguistics: ACL 2022, 2022, pp. 3402–3411.
- [140] Z. Zheng, M. Ma, K. Wang, Z. Qin, X. Yue, and Y. You, “Preventing zero-shot transfer degradation in continual learning of vision-language models,” in Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 19 125–19 136.
- [141] F. Zhu, X.-Y. Zhang, C. Wang, F. Yin, and C.-L. Liu, “Prototype augmentation and self-supervision for incremental learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 5871–5880.
- [142] H. Zhuang, Z. Weng, H. Wei, R. Xie, K.-A. Toh, and Z. Lin, “Acil: Analytic class-incremental learning with absolute memorization and privacy protection,” Advances in Neural Information Processing Systems, vol. 35, pp. 11 602–11 614, 2022.
- [143] H. Zhuang, Z. Weng, R. He, Z. Lin, and Z. Zeng, “Gkeal: Gaussian kernel embedded analytic learning for few-shot class incremental task,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 7746–7755.
- [144] H. Zhuang, Y. Chen, D. Fang, R. He, K. Tong, H. Wei, Z. Zeng, and C. Chen, “Gacl: Exemplar-free generalized analytic continual learning,” Advances in Neural Information Processing Systems, vol. 37, pp. 83 024–83 047, 2024.