Unsupervised Continual Learning based on Parameter Isolation

Ph.D. Thesis

By

ANKIT MALVIYA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JULY 2025

Unsupervised Continual Learning based on Parameter Isolation

A Thesis

Submitted in partial fulfillment of the requirements for the award of the degree
of
Doctor of Philosophy

by
ANKIT MALVIYA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE
JULY 2025

INDIAN INSTITUTE OF TECHNOLOGY INDORE

I hereby certify that the work which is being presented in the thesis entitled **Unsupervised Continual Learning based on Parameter Isolation** in the partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted in the **Department of Computer Science and Engineering, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from **July 2022** to **July 2025** under the supervision of **Dr. Chandresh Kumar Maurya, Associate Professor, Department of Computer Science and Engineering, Indian Institute of Technology Indore**.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

[illegible]

Signature of the student with date

03/11/2025

(Ankit Malviya)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

[illegible] 13/11/2025

Signature of Thesis Supervisor with date

(Dr. Chandresh Kumar Maurya)

Ankit Malviya has successfully given his Ph.D. Oral Examination held on **October 30, 2025**.

[illegible] 13/11/2025

Signature of Thesis Supervisor with date

(Dr. Chandresh Kumar Maurya)

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank a number of persons who in one or the other way contributed by making this time as learnable, enjoyable, and bearable as possible. First, I would like to express my heartfelt gratitude to my supervisor **Dr. Chandresh Kumar Maurya**, who has been a constant source of inspiration during my work. Without his constant guidance and research directions, this research work wouldn't have been possible. His continuous support and encouragement have always motivated me to remain streamlined in my research work.

I am thankful to **Prof. Anirban Sengupta** and **Dr. Ranveer Singh**, my research progress committee members, for taking time out of their busy schedules to evaluate my progress all these years. Their valuable comments and suggestions have helped me to improve my work at various stages.

My sincere acknowledgement and respect to **Prof. Suhas Joshi**, Director, Indian Institute of Technology Indore, for providing me the opportunity to explore my research capabilities at the Indian Institute of Technology Indore.

I would like to express my heartfelt respect to my parents for their love, care, and support they have provided to me throughout my life. Especially, I would like to thank my wife for her constant support, and encouragement.

Finally, I am thankful to *Almighty Lord Sri Sri Radha Madan Mohan*, the most gracious and merciful, and all his creatures who directly or indirectly contributed, helped, and supported me.

Ankit Malviya

Dedicated

to

My Parents, Wife, Children and Teachers

ABSTRACT

Continual learning (CL) is crucial for the development of intelligent systems that must evolve over time and learn from new experiences without forgetting previously acquired knowledge. Supervised Continual Learning (SCL) addresses this issue by adapting to changing data distributions with labeled data. In real-world applications, such as smart healthcare, robotics, autonomous systems, speech translation, etc., the ability to learn continuously is vital, especially when labeled data are sparse or unavailable. Unsupervised continual learning (UCL) addresses this challenge by enabling models to learn from unlabeled data, avoiding the need for manual annotation. The prominent obstacle in UCL is mitigating catastrophic forgetting (CF), where models forget previously learned knowledge when exposed to new information.

Many previous UCL methods rely on replay-based strategies to alleviate CF. Although effective, these techniques may not be feasible in scenarios where storing training data is impractical. Additionally, such methods often face challenges like representation drifts and overfitting, especially when buffer sizes are limited. Regularization-based methods determine the importance of parameters by leveraging knowledge from previous tasks. These methods add regularization constraint to the loss function to limit changes to important parameters. However, they still struggle to fully prevent CF. Architecture-based strategies take a different approach by allocating separate parameter sets to each individual task, gradually creating new networks as new tasks arise. However, this leads to a linear increase in memory and resource requirements.

To address aforementioned limitations, this thesis introduces novel strategies that leverage parameter isolation to reduce CF. Initially, we employ task-specific hard attention to prevent updates to parameters critical for previously learned tasks. However, hard attention suffers from the capacity saturation problem. To address this, we integrate it with gradient projection. Specifically, our approach leverages task-specific hard attention along with gradient projection to constrain updates to parameters crucial for previously learned tasks. Moreover, it offers notable advantages over architecture-based methods by eliminating the need for network expansion and enabling sequential learning within a fixed network struc-

ture.

Even so, relying solely on hard attention to construct sub-networks may result in retention of irrelevant weights and the formation of redundant sub-networks. This can lead to capacity saturation and information suppression for later tasks. To address this challenge, we propose task-specific sub-network masking inspired by the information bottleneck (IB) concept, which further minimizes interference with previously learned tasks. This method selectively accumulates valuable information into essential weights, creating redundancy-free sub-networks that effectively mitigate CF while supporting the learning of new tasks.

In this thesis, contrastive learning is adopted for unsupervised learning because of its ability to preserve consistent representations. We begin by incorporating instance-to-instance similarity through direct instance grouping. Although instance-level discrimination methods show good performance, they often face challenges due to an imbalance between positive and negative samples, which can reduce the model's robustness. To address this, we shift our focus from individual instances to group-level relationships by identifying local clusters within each batch. Therefore, we integrate instance-to-instance similarity into contrastive learning through both direct instance grouping and cross-level discrimination with local instance groups.

Previous research shows that out-of-distribution (OOD) detection capabilities can help in the learning of task boundaries. In closed-world or non-continual scenarios, OOD detection is straightforward because of the availability of labeled data in abundance but becomes more complex in CL setups where data arrive incrementally, especially in UCL setups where labeled information is absent. Therefore, we introduce the concept of pseudo-OOD detection by generating rotation-augmented views of the current batch of data samples, referred to as pseudo-OOD throughout this thesis. Our method employs these pseudo-labels for cross-level pseudo-group discrimination to improve OOD detection. This enables the model to prioritize in-distribution samples from the relevant task while assigning lower scores to samples from other tasks. Subsequently, we apply evidential deep learning (EDL) to rotation augmentation-based pseudo groups to improve OOD detection by identifying different sources of uncertainty and learning distinct class representations.

In the final approach, masking for the IB subnetwork faces challenges in balancing input compression with the preservation of meaningful patterns without labels. This method risks over-compression and the loss of important latent structures, which can degrade model performance. We address this issue by learning multiple semantic hierarchies within the data using unsupervised contrastive learning. However, traditional contrastive learning techniques typically contrast similar and dissimilar data points to learn meaningful representations, but lack the representational capacity required for data sets with multiple semantic hierarchies. The inherent hierarchical semantic structures in such datasets are vital for integrating semantically related clusters into larger, more general clusters. Existing contrastive learning methods often overlook these structures, limiting their semantic understanding. To overcome this, we propose constructing and updating hierarchical prototypes with cross-level group discrimination to more effectively capture and represent semantic structures in the latent space.

LIST OF PUBLICATIONS

(A) From Ph.D. thesis work:

A1. Journal Articles:

J1. **A. Malviya**, S. Dhole, C. K. Maurya, *Unsupervised continual learning by cross-level, Instance-Group and Pseudo-Group Discrimination with Hard Attention*, **Journal of Computational Science**, 86, 102535–102553 (2025). DOI: 10.1016/j.jocs.2025.102535 (SCI, IF 3.7)

J2. **A. Malviya**, C. K. Maurya, *Unsupervised Continual Learning using Cross-Level Discrimination and Evidential Pseudo Out of Distribution Detection along with Gradient Projected Hard Attention*, **IEEE Access**, 12, 171143-171165 (2024). DOI: 10.1109/ACCESS.2024.3435555 (SCI, IF 3.6)

J3. **A. Malviya**, C. K. Maurya, *HiProIBM: Unsupervised Continual Learning through Hierarchical Prototypical Cross-Level Discrimination along with Information Bottleneck Subnetwork Masking*, **Applied Intelligence**, 55(6), 476-497 (2025). DOI: 10.1007/s10489-025-06362-z (SCI, IF 3.5)

(B) Other publications during Ph.D.:

B1. Journal Articles:

J1. B. Sarkar, P. Karande, **A. Malviya**, C. K. Maurya, *Continual End-to-End Speech-to-Text translation using augmented bi-sampler*, **Computer Speech and Language**, 96, 101885-101893 (2026). DOI: 10.1016/j.csl.2025.101885 (SCI, IF 3.4)

B2. Conference Articles:

C1. **A. Malviya**, D. D. Chakladar, C. K. Maurya, Foteini Simistira Liwicki, *Continual learning for EEG-based multiple neurological disorder classification*, **ACM, Proceedings of the 2nd International Workshop on Multimedia Computing for Health and Medicine**, (2025). DOI: 10.1145/3728424.3760765

B3. Patents:

Granted/Accepted

1. A. Malviya, A. Rao S, B. Sarkar, C. K. Maurya, *A system and method for end-to-end continual speech-to-speech translation*, **Indian Patent**, application No. 202421050380, 2024.

Under Review/Revision

2. C. K. Maurya, B. Sarkar, A. Malviya, P. Karande, *System and method for speech-to-text translation based on continual learning technique*, **Indian Patent**, Application No. 202321071567, 2023.

Contents

List of Figures

List of Tables

Table 1: **List of Abbreviations**

CF	Catastrophic Forgetting
CIL	Class Incremental Learning
CL	Continual Learning
CLD	Cross-Level Discrimination
Co2L	Contrastive Continual Learning
CGS	Core Gradient Space
DER	Dark Experience Replay
Dir	Dirichlet distribution
EDL	Evidential Deep Learning
GP-M-CLD-S-OOD	Model with Gradient Projection, hard attention, CLD, SimSiam and OOD detection capability
GP-M-CLD-S-EOOD (Framework-II)	Model with Gradient Projection, hard attention, CLD, SimSiam, and Evidential OOD detection capability
IB	Information Bottleneck
ID	In Distribution
KL	Kullback-Leibler divergence
$\mathcal{L}_{ECE}$	Evidential cross-entropy loss
$\mathcal{L}_{EDL-OOD}$	Pseudo-group level discrimination loss for evidential OOD detection
$\mathcal{L}_{GCD}$	Cross-level group discriminative contrastive loss
$\mathcal{L}_{ILD}$	Instance-level discrimination loss
$\mathcal{L}_{IB}$	Information bottleneck loss
$\mathcal{L}_{HCLD}$	Hierarchical cross-level cluster discrimination loss

Continued on next page

Table 1: **List of Abbreviations** (Continued)

M-CLD-N_OOD	Model with hard attention, CLD, NPID but without OOD detection capability
M-CLD-OOD (Framework-I)	Model with hard attention, CLD, NPID and OOD detection capability
M-CLD-S-OOD	Model with hard attention, CLD, SimSiam and OOD detection capability
N_M-CLD-N_OOD	Model with CLD and NPID but without hard attention and OOD detection capability
N_M-CLD-OOD	Model with CLD, NPID and OOD detection capability but without hard attention
LUMP	Lifelong Unsupervised Mixup
NPID	Non-Parametric Instance Discrimination
OOD	Out Of Distribution
PCA	Principal Component Analysis
PNN	Progressive Neural Networks
RGS	Residual Gradient Spaces
SCL	Supervised Continual Learning
SI	Synaptic Intelligence
SVD	Singular Value Decomposition
TIL	Task Incremental Learning
UCL	Unsupervised Continual Learning

Chapter 1

Introduction

Traditional machine learning algorithms perform admirably on tasks they are trained on, but are incapable of adapting to new concepts, classes, or data introduced after their initial training. In many applications, systems are required to continuously learn a sequence of tasks, which is known as either **continual learning (CL)** or lifelong learning [3]. It is the ability of a model to continually learn from different data distributions over time, allowing it to handle multiple tasks sequentially (see figure 1.1). Unlike humans, who can acquire new knowledge continuously with minimal forgetting of previous learning, neural networks often experience **catastrophic forgetting (CF)** [4]. This occurs when they forget previously acquired knowledge when learning new tasks. CL tackles the challenge of adapting algorithms to evolving data distributions.

Addressing CF is a crucial milestone in the progress toward developing increasingly

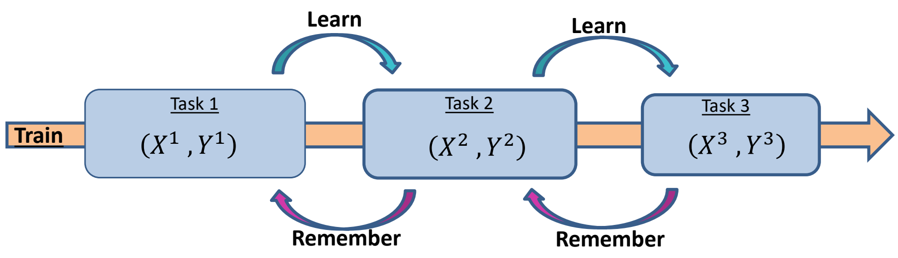


Figure 1.1: Continual learning process

versatile artificial intelligence systems. Such systems should possess the capability to effortlessly retain various tasks and acquire them in succession, adhering to the principles of lifelong learning. In addition to its alignment with biological principles, sequential learning systems are essential in numerous practical scenarios. Consider a situation in which a robotic system needs to adapt to new objects or tasks without the ability to completely retrain its underlying model each time. As it accumulates a vast database of objects and tasks, the cost and computational complexity of conducting concurrent or multitask learning at scale can become prohibitively high. Balancing the learning of new knowledge (plasticity) while retaining prior knowledge (stability) is crucial for CL [5].

1.1 Typical Scenarios of Continual Learning

In recent years, a wide range of deep learning approaches for CL have emerged. However, evaluating and comparing their performance remains challenging due to the absence of a unified framework. To address this issue, we outline three core scenarios in CL [1]: task-incremental, domain-incremental, and class-incremental learning, each presenting its own unique set of challenges. As shown in figure 1.2,

(a) In **task-incremental learning**, the algorithm is required to learn a sequence of clearly distinct tasks incrementally.

(b) In **domain-incremental learning**, the algorithm tackles the same type of problem across varying contexts or environments.

(c) In **class-incremental learning**, the algorithm must progressively learn to differentiate among an increasing number of objects or classes.

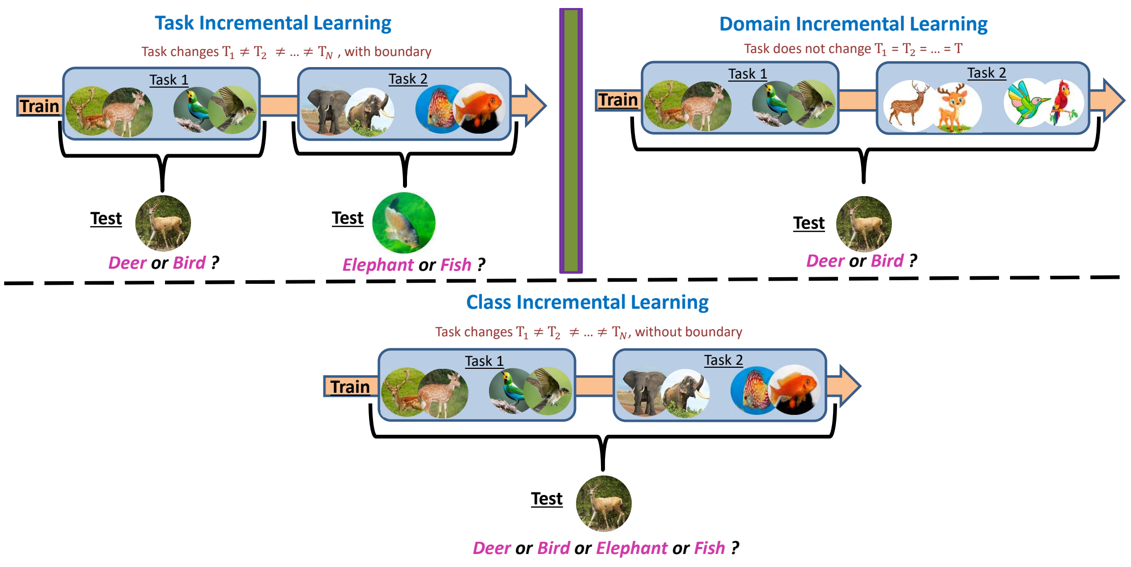

Task Incremental Learning
Task changes $T_1 \neq T_2 \neq ... \neq T_N$, with boundary
Train
Task 1
Task 2
Test
Deer or Bird ?
Test
Elephant or Fish ?
Domain Incremental Learning
Task does not change $T_1 = T_2 = ... = T$
Train
Task 1
Task 2
Test
Deer or Bird ?
Class Incremental Learning
Task changes $T_1 \neq T_2 \neq ... \neq T_N$, without boundary
Train
Task 1
Task 2
Test
Deer or Bird or Elephant or Fish ?

Figure 1.2: Continual learning scenarios [1]

1.2 Conceptual Framework of Continual Learning

To effectively operate in dynamic real-world environments, an intelligent system must be capable of continuously acquiring, updating, retaining, and utilizing knowledge throughout its lifespan. The increasing interest in this area highlights both its practical relevance and inherent complexity. Drawing from existing theoretical insights and empirical studies, Fig. 1.3 outlines the core principles of CL: achieving a balanced trade-off between stability and plasticity, while also promoting strong generalization both within and across tasks, all under constraints of resource efficiency.

In contrast to conventional machine learning methods that rely on a fixed data distribution, CL focuses on adapting to data that changes over time. A key challenge in this domain is CF, where learning new information can significantly impair the model's ability to retain previously acquired knowledge. This reflects a fundamental trade-off between plasticity (the ability to learn new data) and stability (the retention of prior knowledge) [3], excessive plasticity can cause forgetting, while too much stability can limit adaptability. A robust CL system should not only maintain a careful balance between these competing needs but also

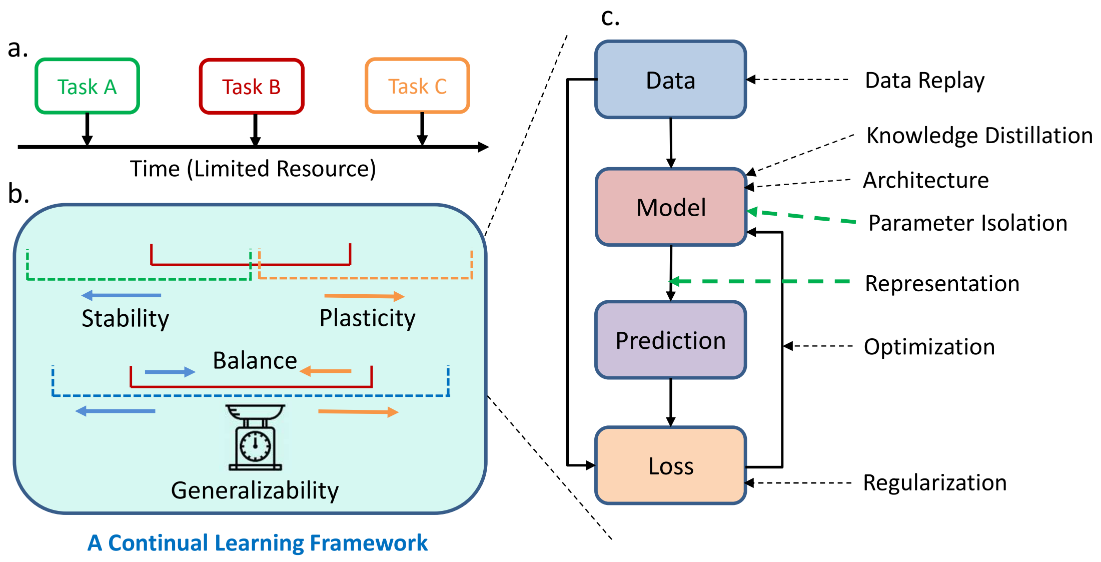


Figure 1.3: Conceptual framework of continual learning

exhibit strong generalization to handle distributional variations within and across tasks (see Fig. 1.3, b).

A straightforward baseline for addressing the challenges of CL is to reuse all previous training data, if permitted. However, this approach leads to significant computational and storage demands, along with potential privacy concerns. The core goal of CL, in contrast, is to enable efficient model updates with minimal reliance on old data, ideally focusing only on new training samples. Over the years, a variety of CL strategies have been developed, which can be broadly categorized into seven conceptual groups (see Fig. 1.3, c):

(a) **Replay-based methods:** Approximate and reconstruct previous data distributions to support CL. This is achieved either by storing a subset of past samples (episodic replay in GEM [6] and AGEM [7]) or generating pseudo-samples via generative models (generative replay [8]), enabling the model to interleave old and new data during training.

(b) **Architecture-based methods:** Design adaptive network structures that adjust parameters based on task-specific needs. These methods often expand the network dynamically (e.g., PNN [9], DEN [10]) to preserve prior knowledge while accommodating new information.

(c) **Representation-based methods:** Develop resilient and broadly applicable feature representations. By learning task-invariant embeddings, often via self-supervised or contrastive learning, these methods (e.g., LUMP [11], Co2L [12]) promote generalization and minimize interference across tasks.

(d) **Knowledge distillation-based methods:** Gradually transfer and preserve knowledge from earlier tasks using distillation techniques. A student model learns to mimic the output distributions of a frozen teacher model, thereby maintaining soft targets and capturing inter-class relationships from past tasks (e.g., LWF [13], DER [14]).

(e) **Regularization-based methods:** Introduce constraints referencing previous models to reduce forgetting. Techniques like EWC [15] and SI [16] penalize changes to important weights, leveraging Fisher Information or trajectory-based importance scores.

(f) **Parameter isolation-based methods:** Allocate distinct sets of parameters for different tasks to avoid interference. Strategies like task-specific masks (e.g., PackNet [17], HAT [18]) or modular subnetworks restrict cross-task parameter updates, ensuring knowledge compartmentalization.

(g) **Optimization-based methods:** Directly modify the optimization process to support CL dynamics. Meta-learning strategies, task-aware solvers, and gradient projection techniques (e.g., OGD [19]) reshape the learning trajectory to mitigate CF and promote forward/backward transfer.

1.3 Research Gaps

The initial strategies to mitigate CF involve preserving previously seen data and incorporating it into the retraining process of the model, commonly known as the replay or rehearsal-based approach [14, 11]. However, due to efficiency, capacity, and privacy concerns associated with the replay-based strategy, memory-free techniques are introduced [3]. *This motivates us to develop a novel framework that achieves near-zero forgetting without the need for a replay buffer.* Regularization-based methods determine the importance of parameters by leveraging knowledge from previous tasks. These methods add a regularization constraint to the loss function to limit changes to important parameters [15, 16, 20]. However, they still struggle to prevent CF fully [21]. Architecture-based strategies adopt a distinct approach by assigning separate parameter sets to each task, progressively expanding the network as new tasks emerge. However, this results in a linear growth in resource consumption [9], as these methods fail to effectively exploit the inherent sparsity of active neurons in deep networks. Some successful strategies, including parameter isolation, stand out as effective approaches with minimal or no CF [18, 22].

The parameter isolation-based approach focuses on protecting specific parameters to retain knowledge from prior tasks. It enables the system to allocate a distinct sub-network for each task within a shared network structure through masking mechanisms [18, 17, 22, 23]. Inspired by compensatory mechanisms in neuroscience and leveraging the sparsity of neuron activations in deep networks, this approach preserves overall network stability while assigning distinct computational routes or subsets of neurons to different tasks within the shared architecture. Neuroscientific studies suggest that in healthy adult brains, the overall density of neural connections remains relatively stable, even as new tasks are learned. This indicates that the brain does not require significantly more neurons to accommodate

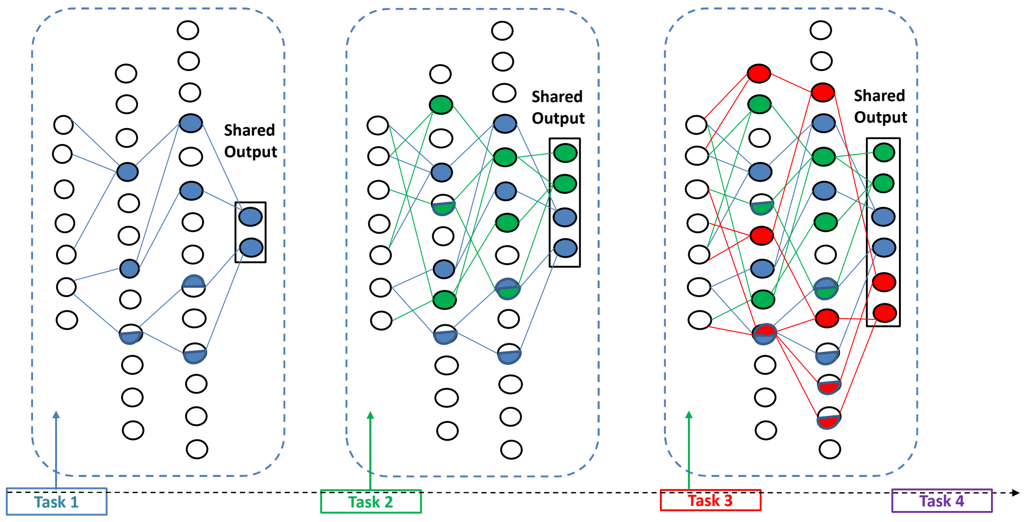


Figure 1.4: Parameter isolation based continual learning

additional knowledge. A similar pattern is observed in deep neural networks, where post-training, only a limited number of neurons are significantly active, leading to sparse activations [24, 25]. Due to the brain's hierarchical, sparse, and recurrent structure [26], neural activity often relies on sparse connections, with only a small subset of neurons responding to a given stimulus [27]. The brain acquires and preserves knowledge by reorganizing existing neurons to form more efficient routes. As a result, it becomes essential to safeguard these neural routes. We tested this hypothesis through a series of experiments for empirical validation. As shown in Fig. 1.4, a holistic view of the deep network is presented by assigning distinct activation neurons to each task using masking. This process effectively creates unique routes through the network, guiding information flow from the input layer, through the intermediate layers, and ultimately to the output layer.

Recent progress in CL has primarily focused on supervised continual learning (SCL), limiting its applicability in real-world scenarios where data is often unlabelled and exhibits distributional bias. Parameter isolation-based methods have also been predominantly explored in supervised settings, as reducing forgetting in unsupervised scenarios poses greater challenges. *Motivated by this gap, we investigate the use of parameter isolation for con-*

tinual learning of unsupervised representations. In this work, we focus on unsupervised continual learning (UCL), where feature representations are learned from a sequence of unlabelled tasks, demonstrating that effective CL can be achieved without relying on annotated data.

Some researchers have begun exploring CL in unsupervised settings, primarily driven by advancements in contrastive learning methods that aim to address the challenges of non-IID data distributions [11]. However, incorporating parameter isolation into UCL poses challenges, as it can lead to the learning of suboptimal representations and hinder performance. To support effective unsupervised learning, contrastive loss functions are commonly employed. Contrastive methods in the literature are traditionally divided into two main categories: instance-wise [28, 29, 30] and prototypical or group-level contrastive learning [31, 32]. Instance-wise contrastive learning aims to preserve and maintain local structure by bringing similar instances closer together and pushing dissimilar instances further apart in the latent space. On the other hand, prototypical contrastive learning is a method that identifies and leverages the natural clustering of data instances by detecting local groupings within a batch. It aims to form compact feature representations centered around cluster prototypes or centroids, effectively capturing the underlying semantic structure of the data. However, when both of these techniques are used separately, they face challenges due to imbalanced ratios of positive and negative samples within batches. Imbalanced datasets with a low positive-to-negative ratio favor instance discrimination, whereas those with a high positive-to-negative ratio support instance grouping. This imbalance can compromise the robustness of the model. As explained in [2], we address the aforementioned issue by incorporating cross-level instance and group discrimination, inspired by [33]. It achieves better invariant mapping by separately imposing grouping and discrimination objectives on features derived from a shared representation and demonstrates performance comparable to supervised classification tasks. Although the method proposed in [33] is not specifically tailored for a CL setup, where the additional challenge of class differentiation within sequential tasks is required. because it works in a simplified scenario by assuming all new data belong to new tasks, which is not realistic if the class labels are not provided. Therefore, to perform UCL in real-life applications, an out-of-distribution detector is required at the beginning to identify whether each new data point corresponds to a new task or already learned tasks. We extend our focus to explore potential group-level relationships. This involves identifying intrinsic clustering patterns of instances and pseudo-instances within batches of samples, thereby enhancing the adaptability of our model in the CL setup.

Prior research [34] has shown that out-of-distribution (OOD) detection capabilities can help in the learning of task boundaries. However, in closed-world or non-continual scenarios, OOD detection is straightforward because of the availability of labeled data in abundance, but becomes more complex in CL setups where data arrives incrementally, especially in UCL setups where labeled information is absent. *This motivates us to improve OOD detection within an unsupervised setting, enabling implicit learning of task boundaries without replaying previous task data.* Therefore, we introduce the concept of pseudo-OOD detection by generating rotation-augmented views of the current batch of data samples, referred to as pseudo-OOD throughout this thesis. Our method employs these pseudo-labels for cross-level pseudo-group discrimination to improve OOD detection. This enables the model to prioritize in-distribution samples from the relevant task while assigning lower scores to samples from other tasks.

Existing CL methods often focus on either task incremental learning (TIL) or class incremental learning (CIL) separately, rather than addressing both simultaneously. The primary distinction between CIL and TIL lies in the availability of task IDs during testing: in TIL, task IDs are assigned to each test sample, while in CIL, they are not provided. This distinction can impose constraints or limitations when applying CL methods designed for one problem to the other. Pseudo-OOD detection enables the model to assign a high score to in-distribution samples corresponding to the correct task while assigning a low score to samples from other tasks. It improves both TIL and CIL performance, with a higher impact observed in CIL.

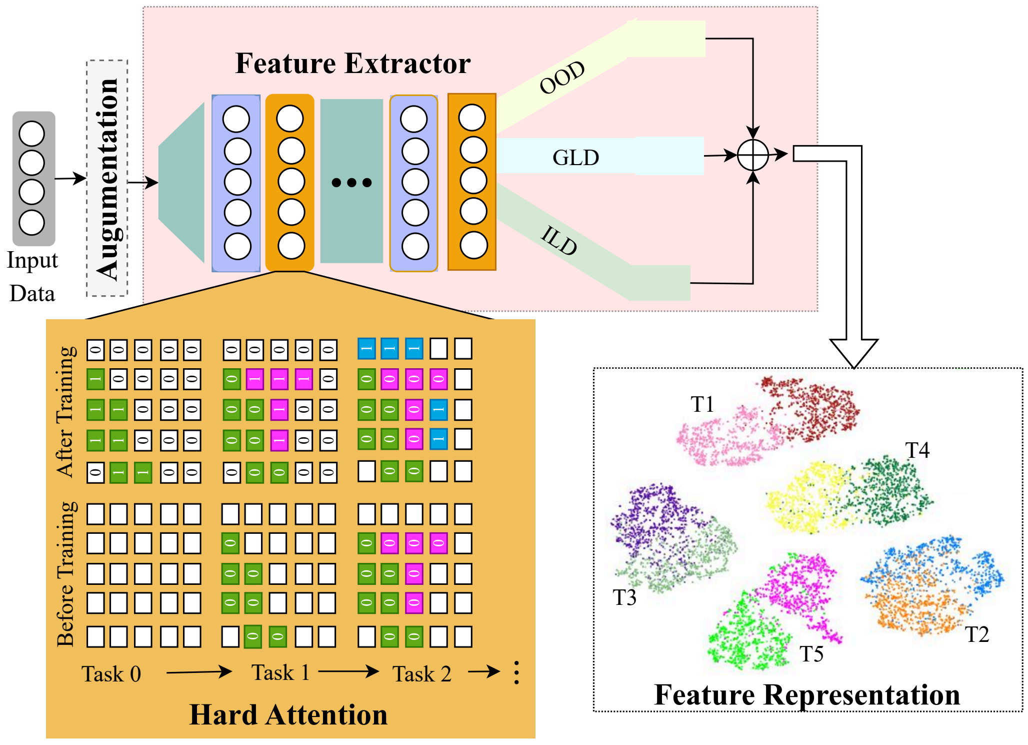


Figure 1.5: Overview of the proposed Framework-I: Augmentation Strategies, Hard Attention-Based Feature Extraction, Multi-Level Projection Heads for Instance, Group, and Pseudo-OOD Discrimination, and Task-Specific Feature Representations

To address the aforementioned advantages and limitations, our proposed Framework-I integrates a task-specific hard attention mechanism, inspired by the foundational work of [18]. However, our incorporation of hard attention differs, as it has not been previously applied in a UCL setting. UCL emerges as a forward-looking learning paradigm capable of incrementally acquiring new knowledge from unlabeled data, while hard attention ensures the retention of experience from previous tasks with minimal forgetting. During the learning of a new task, attention vectors are concurrently learned. Utilizing these vectors from previous tasks, a mask is defined, and the network weights for current tasks are then updated accordingly. The resulting masks demonstrate near-binary behavior, where some weights remain unchanged while others adapt to accommodate the new task. Figure 1.5 presents a conceptual overview of the proposed Framework-I.

Recent research in Mean Magnitude of Channels (MMC) [35] highlighted that hard attention (weight magnitudes) do not always reflect importance, thus leading to redundant sub-networks. It results in network capacity overload and performance decline. Therefore, in Framework-II, we propose integrating parameter isolation with gradient projection, inspired by seminal works [18, 36] and adapting to the context of UCL. Parameter isolation using hard attention maintains prior task knowledge through binary adjustments of attention vectors, derived from past tasks, which serve as masks for updating network parameters during the current task. This selective updating prevents the forgetting of previously learned information. On the other hand, gradient projection partitions weight gradient space and constrains gradient directions to ensure scalability and preserve past knowledge while facilitating learnability in new tasks, minimizing interference, and mitigating CF. However, when applied individually, hard attention and gradient projection can be overly restrictive for gradient updates, compromising performance in addressing new tasks. Therefore, a synergistic approach is proposed, wherein the constraint of gradient update is halved, enabling improved performance across both initial and subsequent tasks while preserving the benefits of both techniques.

The efficiency of detecting OOD instances can be further improved when the Framework-I takes into account the uncertainty of neural network predictions. Among the techniques for assessing the uncertainty of neural networks, evidential deep learning (EDL) [37] stands out for its ability to identify various sources of uncertainty. This allows

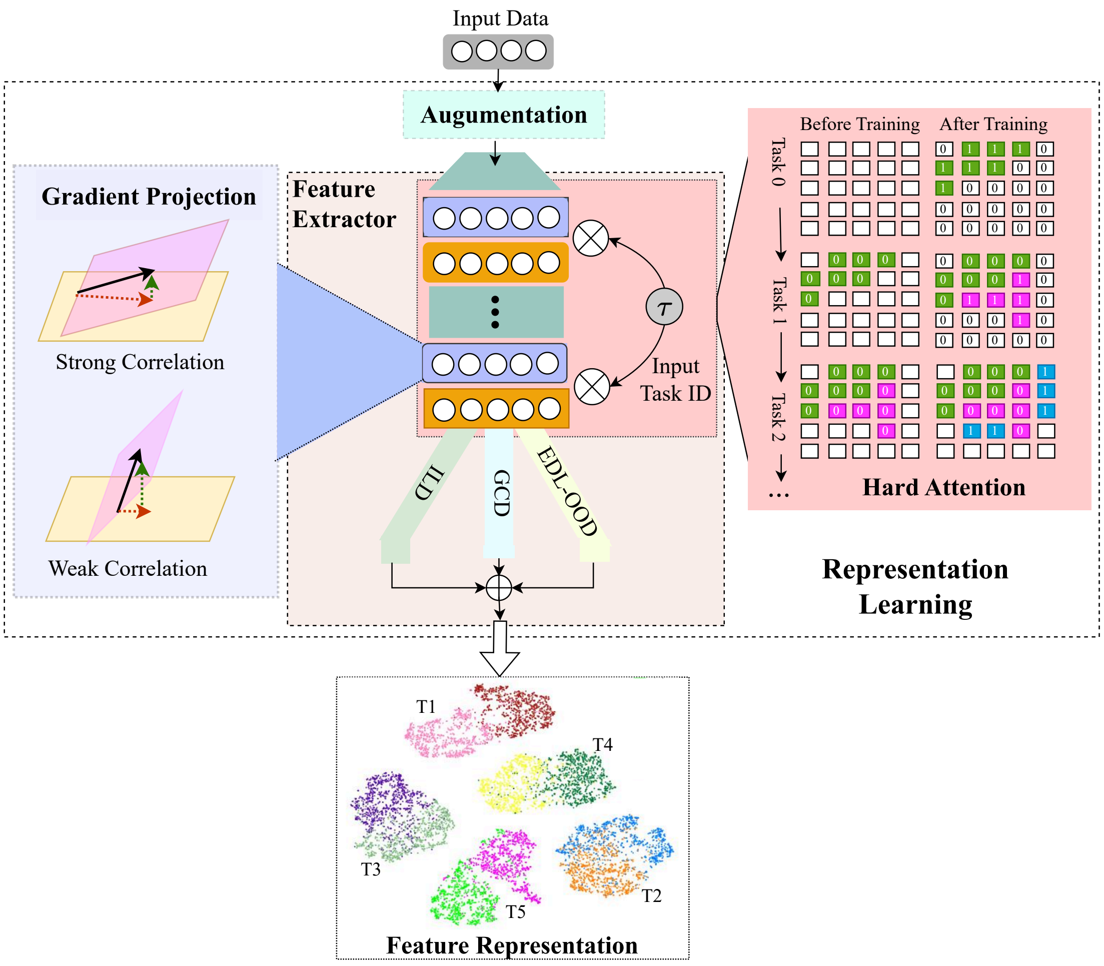


Figure 1.6: Overview of the proposed Framework-II: Augmentation Strategies, Hard Attention-Based Feature Extraction, Gradient Projection, Multi-Level Projection Heads for Instance, Group, and Pseudo-EDL-OOD Discrimination, and Task-Specific Feature Representations.

Framework-II to score the in-distribution sample high for the correct task while scoring it low for other tasks due to data distribution mismatch, because it can differentiate between a lack of confidence and conflicting evidence. It should be noted that, despite achieving the capability to discern task boundaries for individual tasks and class boundaries within the task distribution via cross-level instance grouping and evidential pseudo-OOD detection, the necessity to retain previous task experiences persists. An abstract representation of the proposed Framework-II is shown in Figure 1.6.

The integration of hard attention and gradient projection in Framework-II still imposes constraints on the network, preventing it from fully utilizing its capacity. Therefore, an alternative method is required to optimize parameter isolation and harness the network's full potential. *This drives us to explore alternative approaches to circumvent these challenges and refine the construction of subnetworks.* Inspired by Information Bottleneck (IB) theory [38, 39, 40], we use subnetwork masking through IB to achieve CL in Framework-III. This approach integrates IB to minimize redundancy while maintaining or freezing essential parameters, thereby mitigating CF. Simultaneously, irrelevant information is channeled into expendable parameters. It optimizes inter-layer mutual information to construct redundancy-free sub-networks, addressing challenges in CL. Additionally, a feature decomposing module aids in regulating the compression process, offering automatic and flexible ratio settings for deeper networks.

Although the approaches discussed above have shown effectiveness, they often lack the representational capability needed to handle data sets with multiple semantic hierarchies, such as the hierarchies present in the ImageNet dataset [41]. *This motivates us to use hierarchical prototypes to capture the underlying semantic structures within the data.* Incorporating hierarchical semantic structures into representations has the potential to significantly enhance semantic understanding across diverse downstream tasks. These prototypes, organized in a tree structure in the latent space, are dynamically updated during training to align with current data representations. By leveraging these hierarchical prototypes in Framework-III, we enhance both instance-wise and prototypical contrastive learning.

As discussed above, Framework-III proposes using subnetwork masking via IB for parameter isolation. While subnetwork masking with IB has been explored in SCL [40], applying IB to minimize forgetting in UCL presents inherent challenges [42]. The absence of labels makes it difficult to identify relevant information, increasing the risk of losing es-

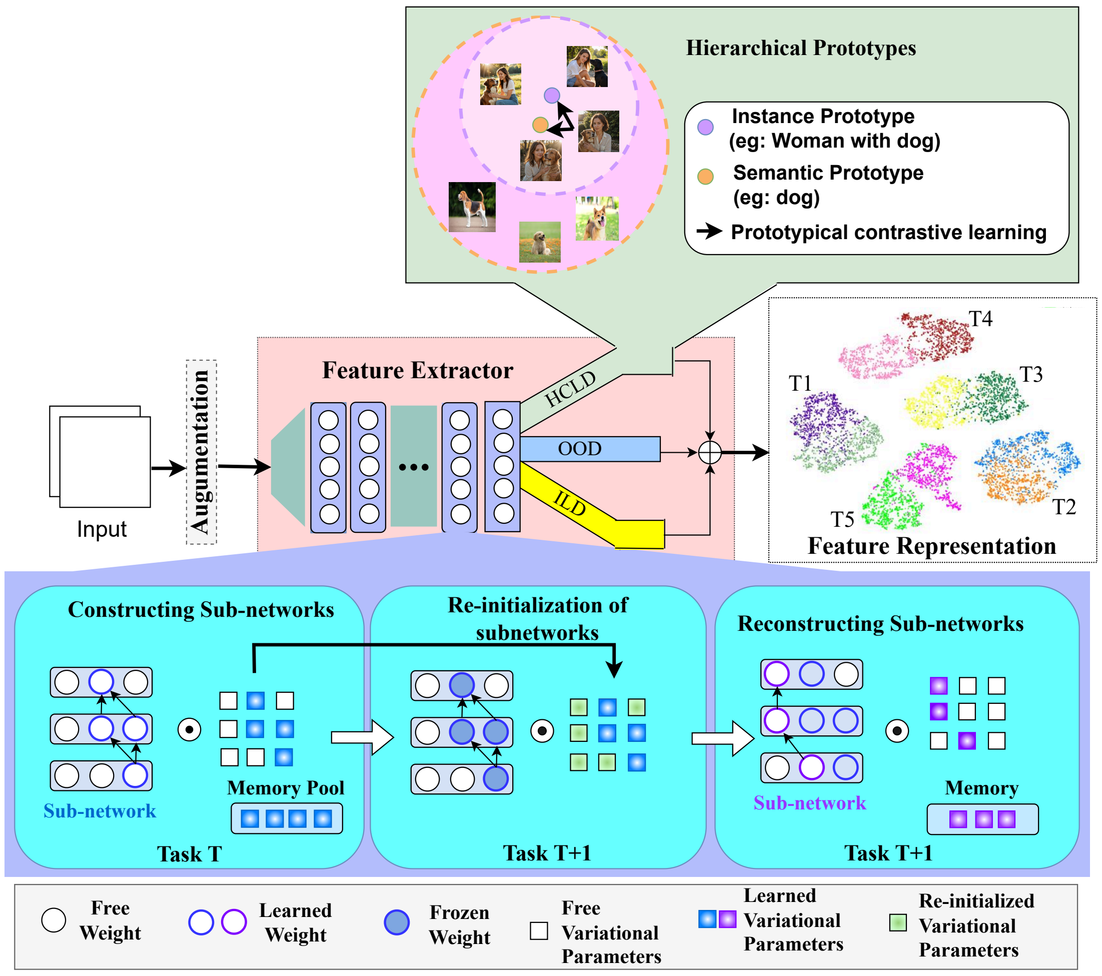

Hierarchical Prototypes
Instance Prototype (eg: Woman with dog)
Semantic Prototype (eg: dog)
Prototypical contrastive learning
Input
Augumentation
Feature Extractor
HCLD
OOD
ILD
T1
T2
T3
T4
T5
Feature Representation
Constructing Sub-networks
Sub-network
Memory Pool
Task T
Re-initialization of subnetworks
Task T+1
Reconstructing Sub-networks
Sub-network
Memory
Task T+1
Free Weight
Learned Weight
Frozen Weight
Free Variational Parameters
Learned Variational Parameters
Re-initialized Variational Parameters

Figure 1.7: Overview of proposed Framework-III: Augmentation Strategies, Information Bottleneck-Based Subnetwork Masking, and Multi-Level Projection along with Hierarchical Prototypical Clustering.

sential features and leading to potential overcompression that can obscure important latent structures [43]. Hierarchical prototypical cross-level discrimination complements IB subnetwork masking by identifying relevant information and preventing the loss of essential features due to overcompression. Together, these components address each other's limitations, enabling effective management of the trade-off between stability and plasticity in the UCL setup. Figure 1.7 shows an abstract view of the proposed Framework-III.

1.4 Motivation and Objectives

UCL represents a critical challenge in the development of intelligent systems that must adapt to new tasks or environments without the availability of labeled data. The ability to learn from an unending stream of data while retaining previously acquired knowledge without CF is essential for real-world applications. However, existing methods in CL often rely on supervised signals, making them less applicable to unsupervised settings, where labels are not provided. The problem is compounded by the need for models to adapt continuously to new and diverse data distributions while preserving previously learned knowledge.

The current approaches to CL, including regularization-based methods, replay-based methods, and architecture-based methods, each have limitations. Regularization techniques, while effective in mitigating forgetting, can lead to inefficient use of model capacity. Replay-based methods face difficulties in managing memory and efficiently storing relevant data samples. Architecture-based approaches, though effective, can be computationally expensive and may not fully exploit the underlying parameter space.

Parameter isolation presents a promising avenue to address these challenges. By isolating specific subsets of parameters for different tasks or data distributions, models can retain their flexibility to learn new information without interfering with previously learned knowledge. This method allows for a more efficient and scalable approach to CL, as different parts of the model can specialize in different aspects of the data, enabling more nuanced learning without overloading the shared parameters.

The motivation for exploring UCL based on Parameter Isolation is driven by the need to develop more effective and efficient learning algorithms that can handle the dynamic nature of real-world data. By focusing on isolating relevant parameters for each new task or data distribution, we aim to reduce the risk of CF while ensuring the model maintains its capacity to adapt to new, unseen data. This approach could open the door to more robust and scalable systems in applications such as autonomous robotics, speech translation, and smart healthcare, where labeled data is scarce and systems need to operate continuously in changing environments.

Ultimately, this research seeks to advance the field of CL by leveraging parameter isolation to address the limitations of existing methods, offering a pathway to more efficient, scalable, and sustainable learning systems in unsupervised settings.

Research Objective I: To develop replay-free, architecture-preserving UCL framework with hard attention and cross-level discrimination for near-zero forgetting and improve TIL & CIL through effective OOD detection.

Research Objective II: To address capacity saturation in hard attention by integrating gradient projection and further improve TIL & CIL accuracy through evidential OOD detection using uncertainty-aware learning.

Research Objective III: To improve network capacity in UCL through effective subnetwork masking beyond hard attention, and enhance representation learning by hierarchical prototypical clustering.

1.5 Thesis Contributions

This thesis proposes three parameter isolation-based approaches for UCL aimed at mitigating CF. Each approach builds progressively upon the previous one, offering successive improvements. Collectively, these methods present replay-free, architecture-preserving, and nearly zero forgetting solutions for CL in an unsupervised setting. The three contributions of the thesis are briefly discussed below.

1.5.1 Replay-free UCL through Hard Attention and Cross-level Discrimination

This work proposes a UCL Framework-I that combines hard attention with cross-level instance-to-group discrimination, as shown in Figure 1.5. Our methodology strategically leverages the benefits offered by both of these approaches within the UCL paradigm. In UCL, it is crucial to preserve the representation learning of previous tasks. This requires the model to distinguish between tasks during training. This can be done using either a task identifier or a guided mechanism. Task information can be encoded by implementing hard attention, wherein each layer of the feature extractor is conditioned accordingly. Hard attention helps in leveraging the learned information to prevent the model from forgetting previous tasks.

To facilitate unsupervised learning, we utilize a contrastive loss function. However, a challenge arises concerning instance-level contrastive loss, primarily due to imbalanced ratios between positive and negative samples within batches. We call similar data points positive samples and dissimilar data points negative samples. Imbalance data with a low positive/negative ratio helps with instance discrimination. On the other hand, a high positive/negative ratio favors instance grouping. Therefore, to mitigate this limitation, we integrate cross-level instance discrimination with group discrimination mechanisms, as shown in Figure 1.5 with green and blue box.

The combination of instance-level discrimination and instance-group-level discrimination effectively generates representations that distinguish correlated instances and maintain semantic groupings within a distribution. This approach is well-suited for TIL scenarios, where the task-ID indicates the distribution of test sample. However, in the context of OOD data, this approach tends to misclassify OOD instances as the most similar in-distribution class. This vulnerability to adversarial attacks makes it less suitable for CIL settings, where accurately distinguishing data from different distributions is crucial for predictions. Successful CL necessitates the ability of the model to discern task boundaries. This differentiation can be enabled through OOD detection capability, where, samples of one task can be considered as OOD for the other tasks and vice versa. While OOD detection is straightforward in closed-world scenarios [44], it becomes challenging in CL setups, where data arrives incrementally. In the case of UCL setup, it is particularly non-trivial due to unlabeled data. Since in the unsupervised domain labeled information is absent, it is difficult to consider a sample of one task to be OOD for the sample of other tasks. Therefore, we explicitly created pseudo OOD samples through rotation-augmentations. Our proposed approach involves detecting pseudo-OOD samples by learning cross-level pseudo-group discrimination from the augmented views. It enables the model to assign a high score to in-distribution samples corresponding to the correct task while assigning a low score to samples from other tasks.

Contributions to the proposed Framework-1 are as follows:

- We propose a novel framework for UCL achieving negligible or zero forgetting without relying on replay buffer.
- Our novel approach aims to uncover and incorporate instance-to-instance discrimination into contrastive learning. Instead of relying solely on instance grouping, we achieve this through discrimination at the cross-level between instances and pseudo instances (pseudo-OOD) with their respective local instance groups. It improves task and class incremental learning (TIL & CIL).
- Our extensive experiments across four standard datasets demonstrate significant performance improvements with minimal forgetting over SOTA baselines for task sequences ranging from 5 to 100. The average TIL accuracy of 76.79% and CIL ac-

curacy of 62.96% across all the standard datasets, with negligible forgetting. This is particularly notable as the SOTA baselines achieve only 74.28% and 60.68%, respectively, in the UCL setting. These baselines experience significant forgetting of almost over 4%. Furthermore, it attains a level of performance that is highly competitive with the SCL baseline and even surpasses it on certain standard datasets, reducing forgetting drastically from over 14.5% to nearly zero.

1.5.2 Addressing Capacity Saturation in Hard Attention and Improving Accuracy with Evidential OOD Detection

The previously presented Framework-I based on hard attention and cross-level discrimination has performed pretty well compared to existing SOTA approaches. However, it suffers from the "capacity saturation problem". When hard attention is used independently, it imposes excessive constraints on parameter updates, leading to suboptimal learning performance for new tasks. This exacerbates the capacity problem by restricting the availability of free neurons and potentially increases forgetting. Therefore, instead of solely relying on the importance of individual parameters, we also focus on constraining the direction of gradients. This ensures that the neural network learns new tasks by updating its parameters in directions orthogonal to the gradient subspaces considered crucial for previous tasks. Through the empirical study, we found that integrating hard attention with gradient projection can complement the advantages and drawbacks of each other when given equal importance.

Unsupervised contrastive learning is effective in maintaining representation continuity but faces challenges with imbalanced positive-to-negative ratios, leading to biased instance discrimination. As discussed in section §1.5.1, to address this issue in Framework-1, we proposed an approach that combines direct instance grouping with cross-level discrimination for both real and pseudo groups, enhancing unsupervised representation learning.

In Framework-II, we additionally utilize the capabilities of EDL to improve pseudo-OOD detection by identifying different sources of uncertainty. Cross-entropy is commonly used for image classification but is known to be biased towards the training data, often assigning high probabilities even to incorrect predictions [45]. Instead of adopting a traditional loss function, we employ a novel approach that combines cross-entropy with evidential KL-divergence for rotation-augmented pseudo labels. This strategy enables the quantification of uncertainty, a vital aspect for ensuring correct predictions [45].

Including evidential cross-entropy loss for OOD detection in the final loss function improves both TIL and CIL accuracy. Intuitively, a model capable of distinguishing between OOD and ID data learns more descriptive features, leading to improved TIL accuracy. For CIL, it is evident that any task inference mechanism necessitates discriminative features between ID and OOD.

Contributions of the proposed Framework-II are as follows:

- We propose a novel framework for UCL using cross-level discrimination and evidential pseudo-OOD detection along with gradient-projected hard attention. In this framework also we are able to achieve nearly zero forgetting in UCL without a replay buffer.

- Our novel approach integrates instance-to-instance discrimination into contrastive learning, focusing on cross-level discrimination between instances and pseudo-instances (OOD) with their respective local instance groups. Additionally, we incorporate Evidential Deep Learning (EDL) into UCL for detecting OOD samples using rotation-augmented pseudo labels, which has not been explored previously.

- Our comprehensive experiments demonstrate substantial performance improvements compared to SOTA baselines on benchmark datasets. We achieve an average TIL accuracy of 77.82% and CIL accuracy of 67.39% across all standard datasets, with minimal forgetting. In contrast, the SOTA baselines achieve up to only 73.33% and 60.08%, respectively in the UCL setting, exhibiting significant forgetting. Moreover, Framework-II shows a significant improvement over the SOTA SCL baseline, with an increase of 2.82% and 3.68% in average accuracy, while significantly reducing forgetting from approximately 12.66% and 19.56% to nearly zero in TIL and CIL respectively.

1.5.3 Improving Network Capacity with Sub-network Masking and Accuracy through Hierarchical Representation Learning

Relying solely on weight magnitude (hard attention) for subnetwork construction can result in retaining unimportant weights and redundant subnetworks, leading to capacity saturation. This drives us to explore alternative approaches to circumvent these challenges and refine the construction of subnetworks. Inspired by Information Bottleneck (IB) theory [38, 39, 40], we use subnetwork masking through IB to achieve CL. This approach integrates IB to minimize redundancy while maintaining or freezing essential parameters, thereby mitigating CF. Simultaneously, irrelevant information is channeled into expendable parameters. It optimizes inter-layer mutual information to construct redundancy-free subnetworks, addressing challenges in CL. Additionally, a feature decomposing module aids in regulating the compression process, offering automatic and flexible ratio settings for deeper networks. Although subnetwork masking using IB is explored in SCL [40], however minimizing forgetting through IB in UCL poses inherent challenges [42]. The absence of labels makes it difficult to identify relevant information, increasing the risk of losing essential features and leading to potential overcompression that can obscure important latent structures

[43].

Organizing instances into groups is crucial for determining their similarity, rather than just focusing on individual discrimination. This is necessary because discrimination on an individual level may compromise both stability and effectiveness. However, previous research suggests that clustering can result from attraction as well as shared repulsion, relying solely on instance clustering may also lead to problems like grouping dissimilar instances [33]. Therefore, our endeavors are twofold: first, to direct each instance towards its associated cluster, and second, to simultaneously repel distant clusters associated with other instances.

Prototypical contrastive learning endeavors to create condensed representations of images within a latent space, ensuring close proximity to their respective cluster centers. While traditional approaches [31, 32] typically compare an image with a singular set of prototypes, this overlooks potential semantic relationships with multiple clusters. Instead of solely associating an image with the most similar prototypes, it is beneficial to consider other clusters as well, which are then regarded as negative samples. Consequently, we employ hierarchical prototypes to represent semantic structures in data and dynamically update them during training. By leveraging these hierarchical semantic representations, we aim to enhance traditional instance-wise and prototypical contrastive learning. This involves optimizing positive and negative pair selections to better align with the semantic structures.

Hierarchical prototypical cross-level discrimination complements IB subnetwork masking by identifying relevant information and preventing the loss of essential features due to overcompression. Together, these components address each other's limitations, enabling effective management of the trade-off between stability and plasticity in UCL setup. Figure 1.7 shows an abstract view of the proposed Framework-III.

The proposed Framework-III contributes as follows:

- We propose a novel framework for UCL that leverages cross-level discrimination of instances and hierarchical prototypical clusters to enhance representational capability for datasets with multiple semantic hierarchies.

- Hard Attention often results in redundant sub-networks and capacity saturation, which degrades performance. Therefore, we employ information bottleneck-based subnetwork masking in UCL to mitigate redundancy and improve CL.

- Our comprehensive experiments demonstrate substantial performance gains over SOTA baselines by achieving nearly zero forgetting in UCL without a replay buffer on both long and small task-sequence data.

1.6 Thesis Organization

The rest of the thesis is organized as follows. In **Chapter 2**, essential background and preliminaries are outlined. It begins with a comparison of SCL and UCL paradigms. It then outlines the benchmark datasets, data augmentation strategies, and evaluation protocols used to assess model performance under TIL and CIL settings.

In **Chapter 3**, a comprehensive review of the literature is presented by discussing various existing techniques related to CL. The review critically examines these approaches, with a particular emphasis on UCL based on parameter isolation. Each method is analyzed in the context of its strengths, limitations, and relevance to the proposed research, providing a foundation for the exploration and advancement of UCL using parameter isolation.

In **Chapter 4**, a memory-free UCL approach, focusing on innovative methods and strategies that enable CL without relying on an external memory buffer, is proposed. It presents a novel Framework-I for UCL through Cross-Level, Instance-Group, and Pseudo-Group Discrimination with Hard Attention, enabling efficient adaptation to new data while effectively preserving previously learned knowledge in a memory-efficient manner.

In **Chapter 5**, the issue of capacity saturation associated with hard attention is addressed by utilizing gradient projection memory. It presents a novel Framework-II for UCL through Cross-Level Discrimination and Evidential Pseudo Out-of-Distribution Detection, combined with Gradient Projected Hard Attention. This approach enhances the model's ability to learn continuously while mitigating the limitations of capacity saturation.

In **Chapter 6**, the problem of capacity saturation associated with gradient projected hard attention is further addressed. It presents a novel Framework-III for UCL by introducing information bottleneck subnetwork masking. It also develops a strategy for learning the hierarchical structure inherent in the data through hierarchical prototypical cross-level discrimination.

In **Chapter 7**, the key findings of the research are highlighted and their practical implications are presented. Additionally, it summarizes the contributions of the thesis and outlines potential future directions for further research in the field

Chapter 2

Background and Preliminaries

This chapter begins with a comparative analysis of SCL and UCL paradigms. Next, we describe the datasets employed, including MNIST, CIFAR-10, CIFAR-100, and Tiny-ImageNet, detailing their task configurations and preprocessing steps. We then elaborate on the data augmentation strategies critical to representation learning. Finally, we present the evaluation metrics and protocols used to assess model performance under TIL and CIL scenarios, including the formulation of accuracy and forgetting metrics. This setup ensures a comprehensive and fair evaluation of the proposed methods and baselines.

2.1 Comparison between Supervised and Unsupervised Continual Learning

Table 2.1 presents a comparative overview of SCL and UCL across key dimensions such as label dependence, learning objectives, and methods, etc. It highlights the fundamental distinctions in supervision requirements, task definitions, and cost-efficiency between the two paradigms.

Table 2.1: Comparison between Supervised Continual Learning (SCL) and Unsupervised Continual Learning (UCL)

Aspect	**Supervised Continual Learning (SCL)**	**Unsupervised Continual Learning (UCL)**
Label Availability	Requires labeled data for each task or instance.	No labels are available; relies on the structure of the data itself.
Learning Objective	Minimize classification or regression loss using labeled data.	Learn representations or clusters, often via self-supervised signals.
Task Definition	Explicitly defined based on label differences.	Tasks may be defined based on distribution shifts or augmentations.
Forgetting Problem	Catastrophic forgetting of previously learned labeled knowledge.	Forgetting occurs in learned representations or clustering structures.
Knowledge Transfer	Transfer learned label information across tasks.	Transfer unsupervised representations or embeddings across tasks.
Supervision Cost	High, requires annotated labels.	Low, no manual labeling needed.
Common Methods	EWC [15], LwF [13], GEM [6], PNN [9], HAT [18], etc.	LUMP [11], CURL [46], CCSL [47], PFR [48], etc.

2.2 Dataset Used

We use four image classification benchmark datasets in our experiments: MNIST, CIFAR-10, CIFAR-100, and Tiny-ImageNet, all resized to a uniform input dimension of $32 \times 32 \times 3$ pixels.

1. **MNIST** (M-5T)[0] [49]: This dataset contains 60,000 training and 10,000 testing grayscale images of handwritten digits, each of size 16×16 pixels, across 10 classes. For our experiments, we split it into 5 tasks, each with 2 classes.

2. **CIFAR-10** (C-5T)[1] [50]: This dataset has 60,000 color images of size 32x32 across 10 classes, with 5,000 training and 1,000 testing images per class. Similar to MNIST, we split it into 5 tasks, each with 2 classes.

3. **CIFAR-100** (CH-τT) [1] [50]: This dataset has 60,000 color images of size 32x32 across 100 classes, with 500 training and 100 testing images per class. We split it into τ tasks so that each task has $100/\tau$ classes.

4. **Tiny-ImageNet** (I-1H-τT and I-2H-τT)[2] [51]: This dataset has 120,000 color images of size 64x64 across 200 classes, with 500 training and 50 testing images per class. For I-1H-τT, we randomly choose 100 classes from the 200 available, whereas for I-2H-τT, we use all 200 classes. We split these classes into τ tasks so that each task has $100/\tau$ or $200/\tau$ classes respectively. Validation images without labels are discarded.

2.3 Data Augmentation Strategies

The combination of data augmentation operations plays a crucial role in obtaining high-quality representations. We apply the same set of augmentations for all baselines, following the detailed specifications provided in [52, 53]. Our initial set of three augmentations includes: (a) Horizontal flip: Mirror the image horizontally with a 50% probability. (b) Color changes: Apply color jitter with an 80% probability to introduce noise and modify brightness, contrast, and saturation. Additionally, perform grayscale adjustments with a 20% probability to convert the image to grayscale. (c) Inception crop: Select a resizing factor uniformly from the range of 0.08 to 1.0 for each image, then crop a portion of the image and resize it back to the original size [54]. This augmentation is accompanied by four rotations of {0°,90°,180°,270°} for OOD detection capability, as shown in Figures 4.1. (a), 5.1. (a), and 6.1. (a).

[0]http://yann.lecun.com/exdb/mnist/
[1]https://www.cs.toronto.edu/ kriz/cifar.html
[2]http://tiny-imagenet.herokuapp.com

2.4 Evaluation Metrics and Settings

2.4.1 Evaluation metrics

To evaluate the utility of the features learned for classification, we apply weighted K-nearest neighbors [52], on the representation and classify the images to evaluate performance. It should be noted that we use labels only to assess the model and not to train it. We consolidate the performance of the model on all tasks $< \tau$ after training on task τ by calculating the average accuracy and forgetting. This is done after training each task and we report the overall average of all these average accuracies and forgettings.

Accuracy: Mathematically, the overall-average accuracy of the model after training τ^{th} task is calculated as:

$$AA_\tau = \frac{1}{\tau}\sum_{i=1}^{\tau} A_{\tau,i} \tag{2.1}$$

$$OAA_\tau = \frac{1}{\tau}\sum_{j=1}^{\tau} AA_j \tag{2.2}$$

where, AA indicates the average accuracy of the model at any given moment, and OAA denotes the overall average accuracy of the model.

Forgetting: It calculates the difference between the highest performance of the model in the past and its current performance.

$$F_{i,\tau} = \max_{k\in\{1,\ldots,\tau-1\}} (A_{k,i} - A_{\tau,i}), \forall i < \tau \tag{2.3}$$

For the τ^{th} task, forgetting is measured by averaging the forgetting of all prior tasks.

$$AF_\tau = \frac{1}{\tau - 1}\sum_{j=1}^{\tau-1} F_{j,\tau} \tag{2.4}$$

The overall average forgetting is calculated by averaging the forgetting across all tasks.

It evaluates the ability of the model to retain experience over time. This remains the same for both the TIL and CIL evaluations. These equations remain same for both the TIL and CIL evaluations.

2.4.2 Evaluation under TIL setting

We follow [52] and apply K-nearest neighbors on the representations and classify the test data. For one sample we calculate its similarity with the entire training data. Then we take the top-K neighbors based on the similarity score, which is used to compute the similarity of the test sample to every class by summing the similarity of all neighbors belonging to each class.

$$P^c(x_i) = \sum_{j=1}^{K} \mathcal{F}_{I_\tau}(x_i) \cdot \mathcal{Q}_j^c \tag{2.5}$$

here, $P^c(x_i)$ is the prediction score for the i^{th} image x_i, $\mathcal{F}_{I_n}$ is the feature extractor that generates prediction score from instance level projection layer for τ^{th} task and $\mathcal{Q}_j^c$ is the j^{th} nearest neighbor for that particular instance, which belongs to class c. This gives us prediction scores for every class and the class with the highest score is taken as the predicted class.

2.4.3 Evaluation under CIL setting

In CIL testing, the model should be able to determine to which task the given sample belongs. At testing time, the model is unaware of which mask to use to embed the nearest neighbors and test sample. Therefore, we generate a separate feature bank for each task τ, where, $\tau \in [\,1, T]$. Then to obtain the prediction score for each class, we follow the same procedure as in the TIL setting. We filter the prediction scores from the K nearest neighbors similarity results, retaining only the scores of classes corresponding to the same mask index as the classes belonging to the particular task index for which it was trained. This filtering is performed to reduce noise in the results.

$$\text{CIL\_Pred}_i = [\ P_1^1(x_i), P_1^2(x_i)..., P_{\tau-1}^{c-1}(x_i), P_\tau^c(x_i)] \tag{2.6}$$

where c is the class, τ is the mask or task index used to calculate the prediction, and i is the sample index. To get the final prediction, we take the maximum.

$$\text{Final\_CIL\_Pred}_i = \text{argmax}([\ P_1^1(x_i), P_1^2(x_i)..., P_{\tau-1}^{c-1}(x_i), P_\tau^c(x_i)]\ ) \tag{2.7}$$

Chapter 3

Literature Review

Our work encompasses the domain of continual learning and unsupervised representation learning. In the following section, we provide an overview of the related literature in both of the aforementioned areas.

3.1 Continual Learning (CL)

It is the process of acquiring new knowledge while retaining previously gained experience from a sequence of tasks [3]. CF hinders deep neural networks from retaining old knowledge while learning new tasks [55]. To balance between plasticity (the capability to learn new tasks) and stability (the ability to retain knowledge from previous tasks), CL research can be classified into five primary categories: Replay-based, Regularization-based, Knowledge Distillation-based, Architecture-based, and Parameter Isolation-based approaches.

3.1.1 Replay-based strategy

Allocates a small memory buffer to store partial old-task data in memory and replays these examples during new-task learning [14, 56]. Gradient Episodic Memory (GEM) [6] and Averaged Gradient Episodic Memory (A-GEM) [7] minimize CF by utilizing episodic memory such that the gradient from the new task data does not deviate too much from the gradient of the previous task. Gradient Coreset Replay (GCR) [57] selects a coreset to estimate gradients for all previously seen data. Graph-based Continual Learning (GCL) [58] addresses forgetting by using a learnable graph to enhance episodic memory, capturing sample similarities to improve task learning and retention. Rainbow Memory (RM) [59] enhances sample diversity in CL by using classification uncertainty and data augmentation. However, in scenarios where buffer size is limited, they suffer from over-fitting [60] and exacerbated representation drifts [61], resulting in increased forgetting of previous tasks. Pseudo-rehearsal methods [8] replace memory modules with generative networks, but training them for sequential tasks or specific data types is challenging. Recent studies [62, 63] suggest that self-supervised learning methods with a replay buffer can mitigate CF in non-iid data. Despite achieving impressive results in downstream tasks, replay-based approaches are memory-intensive, requiring a large number of samples for replay, and may not be viable in applications with privacy constraints.

3.1.2 Regularization-based strategy

Determines the significance of parameters or their gradients in relation to previously learned tasks. It subsequently incorporates a regularization component into the loss function to restrict modifications to these important parameters, thus mitigating CF [64, 16, 65]. Elastic Weight Consolidation (EWC) [15] utilizes the Fisher information matrix to derive parameter importance and applies regularization to penalize the cumulative parameter changes. In a similar approach, Zenke et al. [16] introduced Synaptic Intelligence (SI) by simplifying the computation of the penalty to determine the significance of each parameter by calculating the path integral along the optimization trajectory. Pan et al. [66] proposed a method for training in weight space that identifies key past experiences and a functional prior using a Gaussian process formulation. Despite numerous methods [67, 20] within this category, challenges in effectively preventing CF persist.

3.1.3 Knowledge Distillation (KD)-based strategy

In this strategy, the student model sequentially learns new tasks while retaining past knowledge, guided by a teacher model to prevent CF. Learning Without Forgetting (LWF) [13] uses new samples of outputs of the old model to constrain the outputs of the new model. Many subsequent approaches integrate KD with Experience Replay (ER). For example, Dark Experience Replay (DER) [14] uses both labels and model outputs for distillation. GeoDL [68] distills the geodesic path between features of old and new models, while Distillation of Data Effect (DDE) [69] applies causal inference to distill causal effects between old and new data. In contrast to label distillation, Learning a Unified Classifier Incrementally via Rebalancing (LUCIR) [70] directly enforces consistency between the normalized features of the new and old models while addressing the data imbalance between replayed and training samples. PODNet [71], regulates how each layer's outputs evolve over time. Separated Softmax for Incremental Learning (SS-IL) [72] addresses the bias introduced by data imbalance by applying a separated softmax function in the final layer. Unlike these methods, Complementary Calibration (CoCa) [73] uses a collaborative distillation approach to leverage ensemble dark knowledge from both old and new models, offering richer similarity relationships than those derived from a single model. OCD-Net [74] is a knowledge transfer-based approach similar to [14] and [12]. While it uses an online teacher model instead of an offline one, enabling efficient integration and consolidation of new knowledge through continuous updates to the student model's parameters. The author in [75], like DER, maintains model output consistency for knowledge preservation but improves upon it by decoupling output probabilities to better capture information about old tasks. It also addresses data imbalance. These approaches face several limitations. They rely on replay buffers and require labeled information because they are supervised. Additionally, biases from the initial tasks or teacher models can accumulate and affect the student model's performance on future tasks. The student model might overfit to the teacher's specific patterns, leading to poor knowledge transfer, especially with drastically different tasks or limited model capacity. Moreover, KD encounters other issues, such as scalability and complexity. In contrast, our approach is unsupervised and does not rely on replay buffers. It also simplifies the process compared to complex teacher-student models.

3.1.4 Architecture-based strategy

Prevents forgetting by expanding the network as new tasks are encountered, making it more effective in preventing CF compared to regularization-based and replay-based methods [9, 10, 76]. However, it entails a considerable increase in parameters, sometimes approximating or exceeding the size of the complete model. A notable example is the Progressive Neural Networks (PNN) method by Rusu et al. [9], which allocates a network to each task and leverages previous knowledge to support learning new tasks. It is unsuitable for low-memory devices. Abati et al. [77] introduced a task-aware gating module in each convolutional layer to choose the suitable filter for each task. Yoon et al. [10] introduced Dynamically Expandable Networks (DEN), a flexible model that dynamically incorporates new neurons to accommodate additional tasks. DualNet [78] optimizes memory by utilizing distinct representations for each task. Qin et al. [79] dynamically built a new network to learn new tasks while transferring knowledge between tasks. Selection of Experts for Ensemble Diversification (SEED) [80] selects and fine-tunes the most optimal expert per task using Gaussian distributions, boosting diversity and stability, but requires a predefined maximum number of experts. Other examples of architecture-based approaches include Dynamic Token Expansion (DyTox) [76] and Learn to Grow [81]. While being more effective in preventing CF, the number of parameters grows dramatically with the addition of new tasks. Therefore, it is difficult to use architecture-based approaches in low-memory devices.

3.1.5 Parameter isolation-based strategy

Allocate separate model parameters for each task by masking out parameters from previous tasks and updating only the remaining parameters for learning new tasks [18, 23, 22, 40, 82, 83]. It effectively overcomes the limitations of replay-based strategies by eliminating the need for a memory buffer and addresses the constraints of architecture-based methods by avoiding the expansion of network architecture. Compacting, Picking and Growing (CPG) [84] integrates model compression, weight selection, and progressive network expansion iteratively to create a scalable incremental learning method for multiple sequential tasks. PathNet [85] divides each layer into sub-modules to identify optimal pathways for each task. PackNet [17] freezes crucial weights for each task by identifying them through pruning. While PackNet does not expand the network during CL, it overutilizes the fixed capacity, and it relies heavily on the quality of the pre-trained backbone. Hard Attention to the Task (HAT) [18] uses binary masks on neurons to identify those that are crucial for each task. During training, these masks block gradient flow through the masked neurons, leaving only unmasked neurons trainable. As more tasks are added, fewer neurons remain free, which makes learning subsequent tasks increasingly difficult and leads to a gradual performance decline. Additionally, masking a neuron also masks all parameters feeding into it, leading to significant capacity issues. CAT [23] enhances HAT by identifying task similarities and adjusting masks to facilitate knowledge transfer. However, misclassifying tasks as similar or dissimilar can lead to severe CF. Supermasks in Superposition (SupSup) [22] employs a randomly initialized backbone network and identifies a sub-network specific to each task. The sub-network or mask for each task is preserved. As there are no alterations to the network, SupSup does not suffer from CF and capacity issues. However, due to the independent nature of each mask design, SupSup inherently lacks knowledge transfer it resulting in sub-optimal performance due to the fixed weights constraining the model's representational ability. To tackle this challenge, SpaceNet [86] employs sparse training to conserve parameters for future tasks, albeit at the expense of individual task performance. Winning SubNetworks (WSN) [87], inspired by the lottery ticket hypothesis, learns a compact subnetwork for each task while keeping the weights selected for earlier tasks fixed. It separates the learning process by decoupling the network structure and learning parameters into two distinct, trainable components: weights and weight scores. It then selects the top-k percent of weights based on their ranking scores. SPG [83] uses gradient-based importance scores as soft masks to constrain gradient flow during backpropagation to prevent forgetting, but struggles to scale from small to large networks. Some methods [87, 88] select sub-networks and optimize their parameters within a regularized subspace. Saha et al.[89] employ a technique that divides the weight parameter space into core and residual spaces after each task learning using PCA-based pruning [90]. Gradient Projection Memory (GPM) [36] enables CL by using gradient projection to store knowledge from previous tasks in orthogonal subspaces, minimizing interference with new tasks. While GPM preserves past knowledge in the frozen core space and updates the residual space for new tasks, it does not assign importance to individual parameters. Instead, it imposes constraints on the gradient descent direction. Orthogonal Gradient Descent (OGD) [19] and Orthogonal Weights Modification (OWM) [91] address CF by storing gradient directions, but GPM significantly reduces memory requirements. While OWM uses iterative projector computation, GPM utilizes SVD to identify subspaces in the gradient space efficiently. While most of these parameter-isolation methods have shown impressive performance, they often select parameters based on their magnitude, leading to redundant sub-networks. Information Bottleneck Masked sub-network (IBM) [40] leverages information bottleneck theory to reduce redundancy in sub-networks, resulting in the creation of redundancy-free sub-networks. However, the IB subnetwork masking struggles to balance input compression with pattern preservation in the absence of labels [92, 93]. Without labels, it risks over-compression, which can lead to the loss of crucial hierarchical latent structures. This loss of important structures degrades the model's overall performance. Therefore, it is necessary to employ a mechanism that mitigates the effects of over-compression. Such a mechanism would complement IB sub-network masking by identifying and preserving relevant information, thereby preventing the loss of essential features during the compression process. This further motivates the use of hierarchical prototypical cross-level discrimination with IBM, which is discussed in detail in a later section.

3.2 Unsupervised Representation Learning

Unsupervised representation learning strives to acquire visual representations without incurring the cost of data labeling. Recent advancements show a groundbreaking direction that attains SOTA, such as Barlow Twins[94], Simple Siamese networks (SimSiam) [53], SimCLR [28], Momentum Contrast (MoCo) [30], Infomin [95], Swapping Assignments between multiple Views of the same image (SwAV) [96], Cross-Level instance-group Discrimination (CLD) [33], Non-Parametric Instance Discrimination (NPID) [52], Noise-Contrastive Estimation (NCE) [97] and, Info-NCE [98]. Other recent approaches for unsupervised representation learning are also available, such as autoencoder-based [99, 100] and pretext task-based [101, 102]. Sometimes these methods attain comparable or even superior performance with respect to supervised representation learning [53, 96, 33, 103].

Contrastive loss optimization is a prevalent approach among these methods, aiming to maximize the similarity of representations across different data augmentations by bringing positive pairs closer and pushing negative samples apart [28, 30, 94]. Instance discrimination-based methods assume the presence of distinct and well-separated instances; however, their performance may degrade in real-world scenarios, where data often exhibit high correlation or long-tail distributions. Alternatively, prototypical contrastive learning (grouping of instances) leverages semantic structures by incorporating prototype representations of clusters [104, 105, 106]. This is achieved by contrasting either correlated or uncorrelated pairs of prototypes [31, 107] or by contrasting associated and unassociated pairs of instances with prototypes [32, 33]. Therefore, to enhance the stability of feature learning and discrimination capabilities, we integrate instance-wise and cluster-wise contrastive learning by extending discrimination from individual instances to relationships between instances and batch-level clusters [33]. In Framework-I and Framework-II, we integrate contrastive learning and instance grouping into a unified framework, extending discrimination beyond individual instances to capture relationships between them through the formation of batch-level groups and pseudo-groups (OOD). This approach safeguards feature learning against degeneracy and enhances stability while enabling discrimination at the instance level to extend beyond the finest granularity. These strategies contribute to producing more semantically concise representations. However, these methods often overlook the hierarchical nature of the datasets by representing semantic clusters at a single hierarchy due to a lack of awareness of global semantics across the entire dataset [108]. Therefore, in Framework-III, we explore hierarchical prototypical contrastive learning along with cross-level instance and cluster discrimination. Recognizing this hierarchical structure is essential for a deeper understanding of relationships and structures in the data.

3.3 Unsupervised Continual Learning (UCL)

In UCL, achieving CL without class labels poses a very challenging scenario. Recently, there has been an increasing fascination for UCL, as demonstrated by various related research efforts [46, 11, 109, 110]. Aligning with earlier studies [11, 111], UCL demonstrates its efficacy in generating competitive CL models across various downstream tasks. This empirical evidence suggests that UCL exhibits reduced susceptibility to CF, a characteristic we leverage in our proposed frameworks. Self-Taught Associative Memory (STAM) [112] utilizes an expandable memory structure that necessitates dataset-specific fine-tuning. Lifelong Unsupervised Mixup (LUMP) [11], and LL-UCR [113] use data augmentation and interpolation of new samples with buffered data to tackle CF. Madaan et al. [11] demonstrated that UCL can outperform SCL algorithms in TIL scenarios, leveraging techniques like Mixup [114] to mitigate CF. However, their approach relies on a replay buffer to maintain previous task data. Unlike replay-based methods, our approach does not necessitate the storage of data from previous tasks. Another study [115] conducted a large-scale UCL experiment through a pre-trained model. Knowledge distillation-based methods like CCSL [47] and CaSSLe [116] retain essential knowledge from previous models by capturing snapshots of the model. Approaches such as Projected Functional Regularization (PFR) [48] and CaSSLe incorporate projection heads into SSL frameworks to maintain previous knowledge. However, they face challenges in learning new tasks while maintaining prior knowledge, mainly due to hurdles in regularization. This hampers CL by diminishing flexibility and inducing drift. Continual Unsupervised Representation Learning (CURL) [46] addresses UCL using variational autoencoders combined with a Gaussian mixture model. Some other methods also leverage variational autoencoders and generative replay to mitigate catastrophic forgetting [117, 118, 119]. Plasticity-Optimized Complementary Networks (POCON) [120] trains task-focused experts for high plasticity but faces potential computational overhead during the adaptation-retrospection phase, especially in many-task settings. However, these methods face challenges related to computational cost and scalability, particularly when dealing with large datasets. Despite their benefits, the ability to detect emerging clusters and continually expand the model remains essential. Our approaches align with a parameter isolation-based technique for UCL. However, parameter isolation-based techniques continue to encounter capacity saturation, especially in long task sequences, due to the limitations of hard attention mechanisms used in Framework-I. To mitigate the impact of capacity saturation, we incorporate gradient projection in Framework-II and information bottleneck masking in Framework-III. Previous work [121] in SCL demonstrated that better representation of learning can be achieved by differentiating between OOD samples and ID samples. They claim that it helps in both TIL and CIL setups. We extend their idea in Framework-I and enhance it by cross-level pseudo-instance group discrimination for OOD detection. Framework-II integrates parameter isolation with gradient projection to mitigate CF. Additionally, Framework-II uses evidential deep learning (explained in the next section) to identify pseudo-OOD instances and differentiate class representations. Therefore, it differs from Framework-I by integrating unsupervised contrastive learning framework with cross-level evidential pseudo-OOD detection. This combination provides a robust solution to both TIL and CIL challenges. Framework-I and Framework-II use direct instance grouping along with cross-level discrimination for unsupervised learning. However, they lack sufficient representational capacity for datasets with complex semantic hierarchies. Framework-III addresses this limitation by incorporating hierarchical prototypical cross-level discrimination.

3.4 Evidential Deep Learning (EDL)

The development of uncertainty-aware predictors aligns with the rise of modern Bayesian approaches in machine learning. Gaussian Processes (GPs) [122] are prominent in this field, offering accurate predictions and reliable uncertainty measures. Bayesian Neural Networks (BNNs) [123] also play a role by incorporating prior distributions on model parameters. Despite their predictive power, BNNs require approximation techniques like Variational Bayes (VB) [124, 125] and Stochastic Gradient Hamiltonian Monte Carlo (SG-HMC) [126] due to intractable posterior calculations. In contrast, EDL [37] directly models a Dirichlet posterior using a deterministic neural net, eliminating the need for inferring uncertainty sources. This capability of EDL to identify various sources of uncertainty makes it particularly effective in detecting pseudo-OOD instances, as discussed in §5.1.1.3. In Framework-II, we integrate EDL for OOD detection; however, unlike [127], it is applied in the context of UCL rather than SCL. Notably, the utilization of EDL on pseudo-rotation-augmented samples presents a non-trivial challenge, as explained in §5.1.1.3.

3.5 Research Questions (RQs)

The design of Framework-I is driven by the following core research questions (RQs):

- **RQ1:** Do we need the memory buffer to retain the experience in UCL?
- **RQ2:** Will parameter isolation learn unsupervised representation continually?
- **RQ3:** Does cross-level pseudo-instance group discrimination for OOD detection improve the TIL and CIL by helping in learning clear task boundaries?

The development of Framework-II is guided by the following key RQs:

- **RQ4:** Can unsupervised representation be learned continually using gradient projection along with parameter isolation?
- **RQ5:** Does the implementation of cross-level evidential pseudo-instance group discrimination for OOD detection improve TIL and CIL by learning distinguishable task and class boundaries?

The following RQs guided the design and development of Framework-III:

- **RQ6:** Existing unsupervised contrastive representation learning methods often fail to capture the hierarchical semantic structures inherent in the data. Given this limitation, is there an effective way to address this issue and learn these hierarchical structures in the CL setup?
- **RQ7:** Relying solely on weight magnitude (hard attention) for subnetwork construction can result in retaining unimportant weights and redundant subnetworks, leading to capacity saturation. What is an alternative approach to avoid these issues and enhance subnetwork construction for long task-sequences?

Chapter 4

Replay-free UCL through Hard Attention and Cross-level Discrimination

In this chapter, we propose UCL Framework-I, which combines hard attention with cross-level instance-to-group discrimination. Our methodology strategically leverages the benefits offered by both of these approaches within the UCL paradigm. An abstract view of the Framework-I is presented in Figure 1.5 of Chapter 1. In UCL, it is crucial to preserve the representation learning of previous tasks. This requires the model to distinguish between tasks during training. This can be done using either a task identifier or a guided mechanism. Task information can be encoded by implementing hard attention, wherein each layer of the feature extractor is conditioned accordingly. Hard attention helps in leveraging the learned information to prevent the model from forgetting previous tasks.

To enable unsupervised learning, we use a contrastive loss function. However, instance-level contrastive loss suffers from an imbalance between positive and negative samples. A low positive/negative ratio aids instance discrimination, while a high ratio supports instance grouping. To address this, we combine cross-level instance and group discrimination.

The work presented in this chapter has been published in the paper: "*Unsupervised continual learning by cross-level, Instance-Group and Pseudo-Group Discrimination with Hard Attention*", Journal of Computational Science, 86, 102535–102553 (2025). DOI: 10.1016/j.jocs.2025.102535

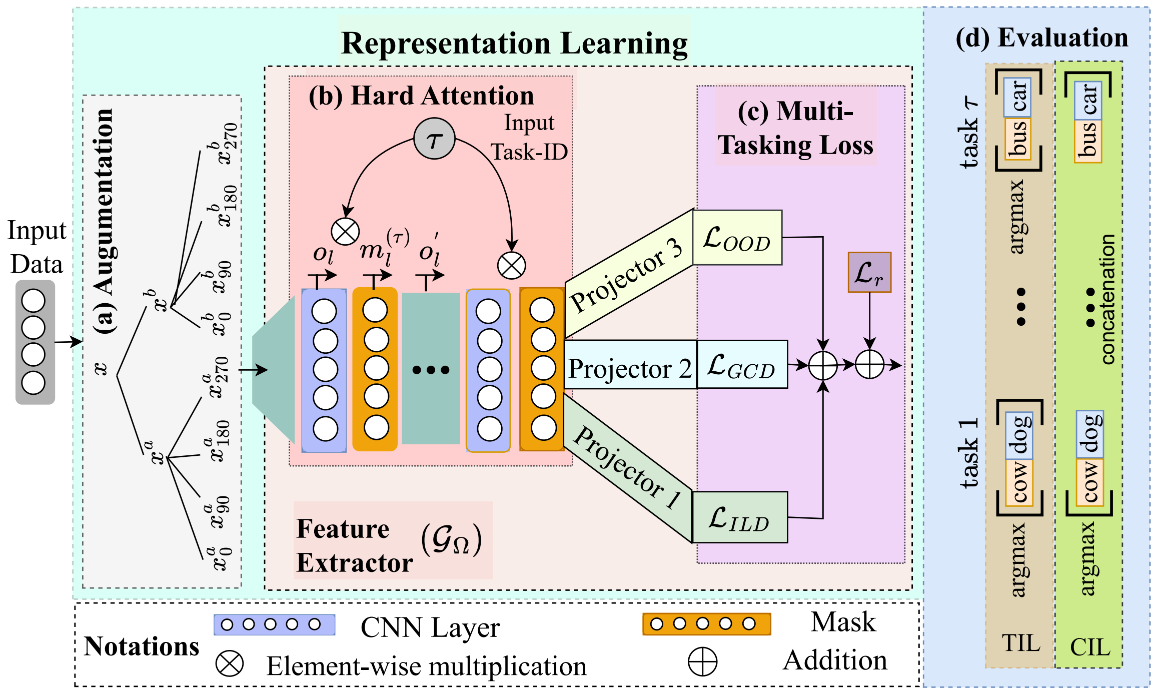


Figure 4.1: Overview of the proposed Framework-I. (a). The augmentation block represents various augmentation strategies of the current batch of samples. (b). Training of feature extractor $\mathcal{G}_\Omega$ using Hard Attention mechanism. Output of Feature Extractor provided to loss function through Projectors, where Projector 1 represents instance-level projection head, Projector 2 represents group-level projection head and Projector 3 represents projection head for pseudo-OOD detection. (c). Multi-Tasking Loss function, further explained in Figure 2. (d). The evaluation block, provides TIL and CIL predictions.

Combining instance-level and group-level discrimination helps distinguish correlated instances while preserving semantic groupings, making it effective for TIL scenarios. However, in OOD cases, it often misclassifies unfamiliar data as the closest in-distribution class, making it vulnerable to adversarial attacks and less suitable for CIL settings. Effective CL requires identifying task boundaries, which can be achieved through OOD detection. While this is manageable in closed-world scenarios, it is challenging in CL, especially UCL, due to the absence of labels. To address this, we explicitly created pseudo-OOD samples using rotation-based augmentations. Our method learns cross-level pseudo-group discrimination, allowing the model to score in-distribution samples higher and assign lower scores to out-of-task samples.

4.1 Proposed Technique

This section elaborates on the UCL Framework-I, which integrates cross-level instance-group discrimination with hard attention, as shown in Figure 4.1. Our methodology strategically leverages the benefits offered by both of these approaches within the UCL paradigm. To facilitate unsupervised learning, we utilize a contrastive loss function. However, a challenge arises concerning instance-level contrastive loss, primarily due to imbalanced ratios between positive and negative samples within batches. We call similar data points positive samples and dissimilar data points negative samples. Imbalanced data with a low positive/negative ratio helps with instance discrimination. On the other hand, a high positive/negative ratio favors instance grouping. Therefore, to mitigate this limitation, we integrate cross-level instance discrimination with group discrimination mechanisms, as shown in Figure 4.2 with green and blue box.

However, this is not enough to achieve CL because the data arrives in a sequence of tasks. Successful CL necessitates the ability of the model to discern task boundaries. This differentiation can be enabled through OOD detection capability, where, samples of one task can be considered as OOD for the other tasks and vice versa. While OOD detection is straightforward in closed-world scenarios [44], it becomes challenging in CL setups, where data arrives incrementally. In the case of UCL setup, it is particularly non-trivial due to unlabeled data. Since in the unsupervised domain labeled information is absent, it is difficult to consider a sample of one task to be OOD for the sample of other tasks. Therefore, as described in §4.1.1.3, we explicitly created pseudo OOD samples through rotation-augmentations. The proposed approach involves detecting pseudo-OOD samples by learning cross-level pseudo-group discrimination from the augmented views. It enables the model to assign a high score to in-distribution samples corresponding to the correct task while assigning a low score to samples from other tasks.

In the following sections, we provide detailed explanations for each component of Framework-I.

4.1.1 Unsupervised contrastive learning with cross-level discrimination and OOD detection

Unsupervised contrastive learning is the basic building block for the UCL framework. Unsupervised contrastive learning maps similar instances close and dissimilar instances farther apart [111, 53, 94, 128, 96, 95, 30, 52, 98, 97]. Therefore, we adapt the concept of discrimination at cross-level [33] within unsupervised contrasting learning. We improve it further for continual feature representation learning.

Given a number of images, let x_k^a and x_k^b represent two distinct augmentations of the same k^{th} image. As shown in Figure 4.1, we construct the model $\mathcal{G}_\Omega$, which consists of a feature extractor and projection heads. It maps the input x onto a d-dimensional hypersphere, with the condition that $||\mathcal{G}_\Omega(x)|| = 1$. Let feature for an instance denote as $\mathcal{F}$, and feature for its positive and negative samples as $\mathcal{F}^{pos}$ and $\mathcal{F}^{neg}$, respectively. We optimize the parameters Ω by minimizing the loss function $\mathcal{L}$ across all instances, ensuring that $\mathcal{F}$ simultaneously pull $\mathcal{F}^{pos}$ closer and pushes away $\mathcal{F}^{neg}$.

$$\mathcal{L}(\mathcal{F}_k, {\mathcal{F}_k}^{pos}, \mathcal{F}_{\neq k}^{neg}) = -\log \frac{\exp \frac{<\mathcal{F}_k, {\mathcal{F}_k}^{pos}>}{\rho}}{\exp \frac{<\mathcal{F}_k, {\mathcal{F}_k}^{pos}>}{\rho} + \sum_{j \neq k} \exp \frac{<\mathcal{F}_k, {\mathcal{F}_j}^{neg}>}{\rho}} \tag{4.1}$$

The hyperparameter ρ serves as a temperature regulator, determining the proximity threshold for defining what distance is considered close. $\mathcal{L}$ represents the instance classification loss [52] using noise contrastive estimation (NCE) [97] with softmax. This can be seen as maximizing a lower bound on the mutual information (MI) among samples that belong to the same instances [129, 98].

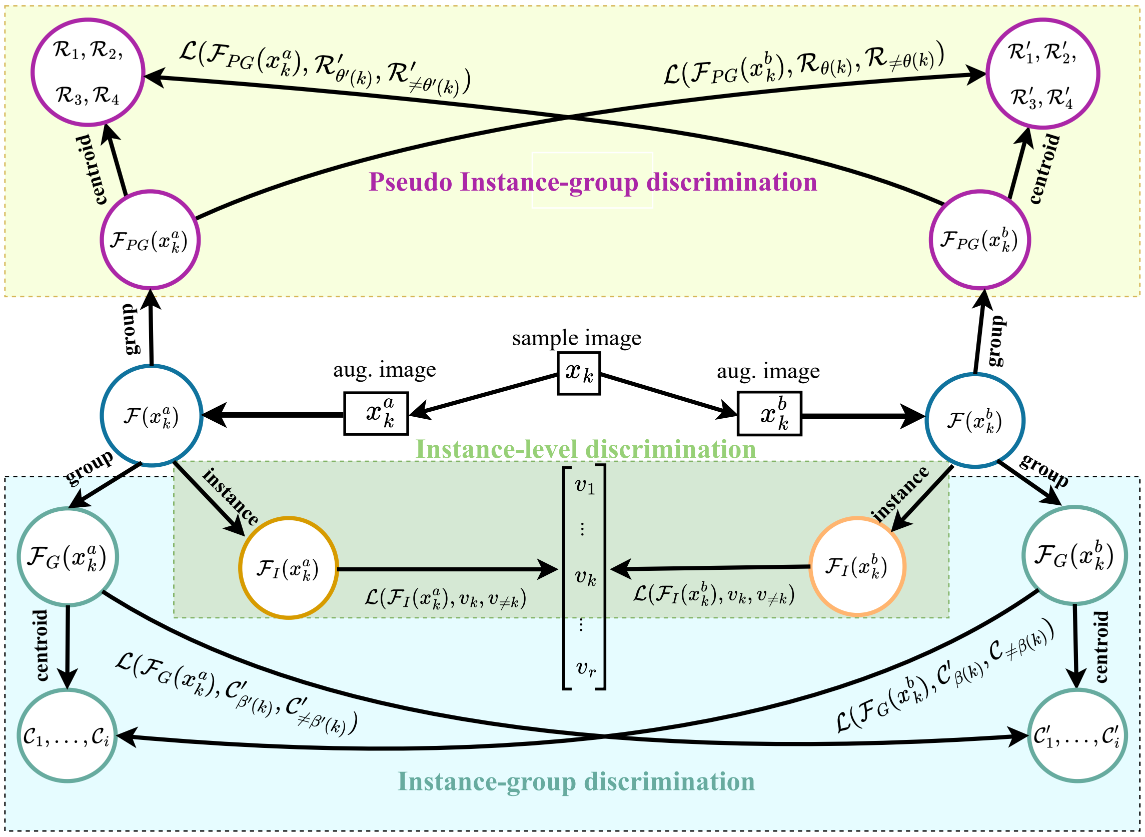

$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$
$\mathcal{L}(\mathcal{F}_{PG}(x_k^a), \mathcal{R}'_{\theta'(k)}, \mathcal{R}'_{\neq\theta'(k)})$
$\mathcal{L}(\mathcal{F}_{PG}(x_k^b), \mathcal{R}_{\theta(k)}, \mathcal{R}_{\neq\theta(k)})$
$\mathcal{R}'_1, \mathcal{R}'_2, \mathcal{R}'_3, \mathcal{R}'_4$
centroid
Pseudo Instance-group discrimination
centroid
$\mathcal{F}_{PG}(x_k^a)$
$\mathcal{F}_{PG}(x_k^b)$
group
sample image
x_k
group
aug. image
x_k^a
aug. image
x_k^b
$\mathcal{F}(x_k^a)$
$\mathcal{F}(x_k^b)$
Instance-level discrimination
group
instance
v_1
v_k
v_r
instance
group
$\mathcal{F}_G(x_k^a)$
$\mathcal{F}_I(x_k^a)$
$\mathcal{L}(\mathcal{F}_I(x_k^a), v_k, v_{\neq k})$
$\mathcal{L}(\mathcal{F}_I(x_k^b), v_k, v_{\neq k})$
$\mathcal{F}_I(x_k^b)$
$\mathcal{F}_G(x_k^b)$
centroid
$\mathcal{L}(\mathcal{F}_G(x_k^a), \mathcal{C}'_{\beta'(k)}, \mathcal{C}'_{\neq\beta'(k)})$
$\mathcal{L}(\mathcal{F}_G(x_k^b), \mathcal{C}'_{\beta(k)}, \mathcal{C}_{\neq\beta(k)})$
centroid
$\mathcal{C}_1, \ldots, \mathcal{C}_i$
$\mathcal{C}'_1, \ldots, \mathcal{C}'_i$
Instance-group discrimination

Figure 4.2: Overview of the Multi-tasking Loss Function in Framework-I. Through the unsupervised contrastive loss function, our objective is to learn representation, denoted as $\mathcal{F}(x)$, for an input image denoted as x_k and its augmentation views referred to as x_k^a and x_k^b. We establish three branches stemming from $\mathcal{F}$: fine-grained instance branch $\mathcal{F}_I$, coarse-grained group branch $\mathcal{F}_G$ and pseudo group branch $\mathcal{F}_{PG}$. All the computation is mirrored and symmetrical across different views of the same instance.

Minimizing the loss function $\mathcal{L}$ effectively increases the similarity of instances of the same semantic category, and decreases the similarity of instances of different semantic categories. In other words, attract positive samples and repel negative samples in the representation space. As shown in Figure 4.1.C and Figure 4.2, the proposed Framework-I is optimized using multitasking loss functions. This multitasking loss function uses three discrimination strategies.

1 **Instance-level discrimination:** Maintains consistency across different perspectives of the same sample while remaining distinct from other samples.

2 **Cross-level instance group discrimination:** Clusters samples belonging to similar semantic categories while being distinctly separated from other instance categories.

3 **Cross-level pseudo-instance group discrimination:** Farther away from any OOD samples in the representation space.

We compute each loss utilizing distinct projection heads. We will discuss the importance of having separate projection heads after explaining each loss function in the subsequent section.

4.1.1.1 Learning invariant mapping through instance-level discrimination

In the supervised learning paradigm, it is observed that models trained to discriminate between classes can also capture similarities between them [52]. For instance, the second-highest prediction logit of a classifier for a class such as Tiger is likely to be similar to the class of Jaguar. This shows that training a model for discrimination also makes it capable of recognizing similarities. The key to this approach is to treat every single training instance to be its own class. Then, if a model is trained to discriminate between each instance, it follows that the model will also capture semantic similarities between each instance. This means that such a model can also map these features in a way that instances of the same semantic class will be mapped close together. To train such a model, we use the instance-level discrimination loss (refer to the green box in Figure 4.2).

$$\mathcal{L}_{ILD} = \mathcal{L}(\mathcal{F}_I(x_k^a), v_k, v_{\neq k}) \tag{4.2}$$

here, $\mathcal{F}_I$ is the instance-level projection head, which is shown as projector 1 in Figure 4.1, v_k is the representation of the k^{th} instance and it acts as a positive sample, $v_{\neq k}$ denotes the representations of the remaining instances which serve as negative samples. This loss is calculated for both the augmented views of the instance k; x_k^a and x_k^b. Training the model to generate similar representations for augmented images enables it to focus solely on capturing the inherent characteristics of the image without being affected by any transformation. Invariant mapping, in this fashion, is the core strength of contrastive learning methods.

Implementation details: It is not computationally efficient to recalculate the representations of all the training data for every training batch. Hence, we maintain a memory bank of all representations of instances belonging to the τ^{th} task. For the loss calculation of any given training batch, we draw the positive and negative samples from the memory bank. The memory bank contains the moving average of the representations and is updated along with every forward pass. We initialize the memory bank randomly for the first epoch. The memory bank generated in a certain task is discarded when we switch tasks. We do not retain any training data from one task to another.

Limitations: While ILD does yield high-quality representations, it fails to address two major issues when we deal with classification. One, forming discriminative representations of instances that are correlated but belong to different classes. Two, a purely discriminative approach does not address between-instance similarities or latent groups in the distribution. Therefore, it is required to capture the group-level similarity as well.

4.1.1.2 Learning grouping through cross-level group discrimination

To establish similarity between instances, it is imperative to organize instances into groups rather than concentrating solely on the differentiation of individual instances. This is primarily due to the underlying assumption that each instance is unique, making the instance-level discrimination approach less stable and less effective without instance grouping. Conversely, relying solely on features obtained from instance grouping can lead to getting stuck in undesirable states. For example, dissimilar instances might be grouped together because grouping can arise not only from attraction but also from shared repulsion. In other words, in our endeavor of identifying the most discriminative features while also adhering to the inherent instance grouping, we aim for each instance to draw closer to the nearest group it is associated, through augmentation and simultaneously push away groups of other instances that are distant from it.

As shown in Figure 4.2 in instance-group discrimination branch, we apply K-means on $\mathcal{F}_G(x_k^a)$ for a batch of instances to identify i centroids, labelled as $\{\mathcal{C}_1, ..., \mathcal{C}_i\}$, with instance k assigned to centroid $\beta(k)$. Their corresponding elements in the alternative view are $\mathcal{F}_G(x_k^b)$, $\{\mathcal{C}'_1, ..., \mathcal{C}'_i\}$, and $\beta'(k)$ respectively. For cross-level discrimination between instances, we employ a contrastive loss $\mathcal{L}$ (blue area in Figure 4.2) between feature $\mathcal{F}_G(x_k^a)$ and centroids $\mathcal{C}'$ based on grouping β', and the flip side for x_k^b. When dealing with two similar instances, such as x_k^a and x_j^a, they would be separated by the instance-level contrastive loss ($\mathcal{L}_{ILD}$) but drawn closer by the cross-level contrastive loss ($\mathcal{L}_{GCD}$), as they distance themselves from common negative groups. The Cross-level Instance Group Discrimination becomes :

$$\mathcal{L}_{GCD} = \mathcal{L}(\mathcal{F}_G(x_k^a), C'_{\beta(k)}, C'_{\neq\beta(k)}) \tag{4.3}$$

$\mathcal{F}_G$ is the group-level projection head, which is shown as projector 2 in Figure 4.1 and Figure 4.2, $C'_{\beta(k)}$ is the centroid of the cluster $\beta(k)'$ derived from the clustering of $\mathcal{F}_G(x_k^b)$, this acts as the positive sample to be attracted. The rest of the cluster centroids $C'_{\neq\beta(k)}$ act as negative samples to be repelled.

Implementation details: From the formulation of $\mathcal{L}_{GCD}$ in (4.3), a possible interpretation is that the loss is equivalent to a normalized prediction score of $\mathcal{F}_G(x_k^a)$ belonging to the $C_{\beta(k)}$ cluster. Hence, minimizing this loss is equivalent to minimizing the cross-entropy between the hard cluster assignment of $C_{\beta(k)}$ and the soft cluster assignment of $C'_{\beta(k)}$. In this perspective, the training objective is to embed an instance in such a way that makes it similar to a group of other related instances. Therefore, this particular branch is referred to as coarse-grained.

Limitations: The combination of instance-level discrimination and instance-group-level discrimination is effective at generating representations that differentiate correlated instances while preserving semantic groupings within a task. This is particularly advantageous for a TIL setting, where the model is informed about the task-ID of the particular test sample.

4.1.1.3 Learning OOD detection through cross-level pseudo-instance group discrimination

In a closed-world setting, identifying OOD instances is relatively easier. However, this task becomes more complex in CL setups, particularly in UCL scenarios where labeled information is lacking. In the contrastive learning framework, one way to make OOD data easier to distinguish would be to map those samples farther away from ID data. We work under the assumption that only the data belonging to the current task is accessible during the training process for that specific task. Within this constraint, we use rotation augmentations of training images to serve as OOD data. We acknowledge that these samples are not true

OOD samples, but we still use them to represent OOD data, to train the model to distinguish between ID and OOD samples at test time for task inference. We refer to the rotated samples as pseudo-OOD in the rest of the places.

To repel pseudo-OOD samples from ID samples, an additional projection head $\mathcal{F}_{PG}$ is added to calculate the pseudo-OOD level discrimination loss, similar to the loss given in (4.1). To generate pseudo-OOD samples, we rotate each sample by an angle θ, effectively quadrupling the total number of samples. Samples rotated by the same angle θ are put into the same cluster and as such we have four clusters : $\{\mathcal{R}_{0^\circ}, \mathcal{R}_{90^\circ}, \mathcal{R}_{180^\circ}, \mathcal{R}_{270^\circ}\}$. We directly compute the centroids of these clusters. We rely on the instance-level discrimination loss to maintain distinct representation within a cluster. While pseudo-group level cross discrimination ensures repulsion between clusters to make sure pseudo-OOD samples occupy different locations in the representation space. The loss function becomes,

$$\mathcal{L}_{OOD} = \mathcal{L}(\mathcal{F}_{PG}(x^a_{\theta,i}), \mathcal{R}'_\theta, \mathcal{R}'_{\neq\theta}) \tag{4.4}$$

This loss is similar to (4.3), where instead of procuring cluster labels by applying K-means, we use the pseudo-labels. As shown in Figure 4.2 in the pseudo-instance-group discrimination (OOD) branch (yellow area in Figure), we apply the contrastive loss between $\mathcal{F}_{PG}(x^a_{\theta,i})$ and the distribution-level clusters $\mathcal{R}'_\theta$ from the projection of the augmented view.

Implementation details: Based on a similar argument, we can interpret this loss as a cross-entropy loss between the hard assignment of the pseudo cluster $\mathcal{R}_\theta$ and the soft assignment $\mathcal{R}'_\theta$ predicted from $\mathcal{F}_{PG}$.

Advantage: This addition of $\mathcal{L}_{OOD}$ to the final loss function $\mathcal{L}$ improves the TIL and CIL accuracy. From an intuitive perspective, a model that can learn features that can distinguish OOD and ID data can also learn more descriptive features. This is more effective in CIL accuracy, as shown in §4.2.4.

Thus far, discussion has centered on learning discrimination at the representation level. However, the model still needs the ability for CL to learn each sequential task while retaining the accumulated knowledge. An isolation-based approach emerges as a promising strategy in this context, as it does not rely on a replay buffer and facilitates model updates at the parameter level. The following section will further support the proposed Framework-I to retain the accumulated knowledge.

4.1.2 Protecting feature representation using hard attention

In UCL, it is crucial to preserve the representation learning of previous tasks. This requires the model to distinguish between tasks during training. This can be done using either a task identifier or a guided mechanism. Task information can be encoded by implementing hard attention, wherein each layer of the feature extractor is conditioned accordingly. Hard attention helps in leveraging the learned information to prevent the model from forgetting previous tasks. In order to condition the current task τ, we employ a layer-wise attention mechanism in the model $\mathcal{G}_\Omega$, as shown in Figure 4.1.(b). Each task-ID τ, has an associated embedding $h_l^{(\tau)}$ for layer l consisting of learnable parameters obtained during training. An individual task embedding is trained for each layer except the last one. To produce the task specific hard attention mask $m_l^{(\tau)}$ from $h_l^{(\tau)}$, a sigmoid gate is used. Hard attention mask $m_l^{(\tau)}$ is computed as follows:

$$m_l^{(\tau)} = \sigma(s \cdot h_l^{(\tau)}) \tag{4.5}$$

where $m_l^{(\tau)}$ represents the attention mask derived from the single-layer task embedding at layer l and $\sigma(z) \in [0, 1]$ stands a gate function. The pseudo-binary value of the attention masks dictates the extent to which information can propagate both forward and backward between neighboring layers. The sigmoid gate function is chosen for two reasons: one is that it is differentiable, unlike the step function. And second, it can be used to control the flow of information through the network. A positive scaling parameter denoted as s is introduced to help in the training process. The value of s decides how much the gate resembles a step function. Scalar s controls the amount of information that flows through the network and the magnitude of the final gradients. As the variable s tends toward infinity, the sigmoid function σ approximates a step function that ranges between 0 and 1, which behaves similarly to a hard attention mechanism and blocks and unblocks the information flow to safeguard the parameters learned for each prior task. We follow [18] and employ a linear annealing strategy to calculate s dynamically for each training batch. This induces a gradient flow during the entire training epoch. We set $s = s_{max}$ during testing, which approximates a unit step function as hard attention $m_l^{(\tau)} \rightarrow 0, 1$, when $s \rightarrow \infty$. Each training epoch commences with all units being equally active, and as the epoch progresses, these units gradually become more polarized. In particular, the annealing of s proceeds as follows:

$$s = \frac{1}{s_{max}} + (s_{max} - \frac{1}{s_{max}})\frac{b-1}{B-1} \tag{4.6}$$

where b represents the batch index, and B stands for the total number of batches in a single epoch.

As shown in Figure 4.1.(b), the mask $m_l^{(\tau)}$ is applied through element-wise multiplication to the output o_l of layer l to produce the output of that layer as

$$o_l^{'} = m_l^{(\tau)} \otimes o_l \tag{4.7}$$

here, $\otimes$ denotes element-wise multiplication. Mask applied to all layers $l = 1, \ldots, L-1$, except for the final layer L, where no masking is applied because it is a task-specific layer.

More precisely, when training for the new task τ, we update the parameters based on the attention mechanism, so that the important parameters related to previous tasks $(1, ..., \tau-1)$, remain unchanged. To achieve this, the gradient has to be conditioned according to the cumulative attention from all the previous tasks. We calculate the accumulated attention masks of all previous tasks as

$$m_l^{<\tau} = max(m_l^{<\tau-1}, m_l^{\tau-1}) \tag{4.8}$$

where l denotes the layer and τ the task id, max stands for element-wise maximum. To adapt the training process for task $\tau + 1$, we alter the gradient $\delta'_{l,i,j}$ at layer l by utilizing the reciprocal of the minimum value between the cumulative attention in the current layer and that in the previous layer as follows:

$$\delta'_{l,i,j} = [1 - min(m_{l,i}^{\leq\tau}, m_{l-1,j}^{\leq\tau})]\delta_{l,i,j} \tag{4.9}$$

where $m_{l,i}^{\leq\tau}$ indicate i_{th} unit of $m_l^{\leq\tau}$ such that i corresponds to the output layer (l), and l represents all layers $l = 1, ...L - 1$, and j corresponds to input layer $(l - 1)$. This results in a reduction of the gradient if the attentions of the corresponding units at layers l and $l - 1$ are not zero. It is crucial to understand that the hard attention values $m_{l,i}^{\tau}$ for active neurons is $m_{l,i}^{\tau} \to 1$. So the values of these masks directly determine the units that will be dedicated to task τ. Therefore, to allocate model capacity for future tasks, we encourage sparsity on the set of attention vectors $\mathcal{M}^{\tau} = m_l^1, ...m_{L-1}^{\tau}$. To achieve this, we add a regularization term denoted as $\mathcal{L}_r$ to the loss function $\mathcal{L}$, which considers the cumulative attention vectors encompassing tasks up to $\tau - 1$, $\mathcal{M}^{<\tau} = m_{<l}^1, ...m_{L-1}^{\tau}$:

$$\mathcal{L}_r = \frac{\sum_l \sum_i m_{l,i}^{\tau}(1 - m_{l,i}^{<\tau})}{\sum_l \sum_i (1 - m_{l,i}^{<\tau})} \tag{4.10}$$

4.1.3 Overall objective function

By combining (4.2),(4.3),(4.4), and (4.10), the resulting objective function to be minimized for task τ is formulated as follows (see in Figure 4.1.(c) and Figure 4.2):

$$\mathcal{L} = \mathcal{L}_{ILD} + \gamma_{GCD}\,\mathcal{L}_{GCD} + \gamma_{OOD}\,\mathcal{L}_{OOD} + \gamma_{R(\tau)}\,\mathcal{L}_{r} \tag{4.11}$$

here, $\mathcal{L}_{ILD}$, $\mathcal{L}_{GCD}$, $\mathcal{L}_{OOD}$ is the loss function for instance-level discrimination, group-level discrimination, and pseudo-group level discrimination for OOD detection, respectively. $\mathcal{L}_r$ is the regularization loss. $\mathcal{L}$ is the overall loss function or can be referred to as a multi-tasking loss function. γ_{GCD}, γ_{OOD}, and $\gamma_{R(\tau)}$ represents positive scaling parameter for the respective loss function. In practice, γ_R is same for all tasks, where $\tau \neq 1$.

4.2 Experimental Analysis

In this section, we explain the (1) baselines, (2) training and evaluation setup, (3) hyper-parameters setting, and (4) results and comparative analysis

4.2.1 Baselines

For baselines, we consider four unsupervised methods, i.e., SI [16], DER [14], PNN [9], LUMP [11] and one supervised method, i.e., Co2L [12]. All these methods are representative of different CL strategies such as SI for regularization-based strategy, DER for replay-based strategy, PNN for architecture-based strategy, LUMP for unsupervised representation learning with buffer, and Co2L for supervised contrastive learning. We denote the proposed **Framework-I** as M-CLD-OOD, which represents masked CLD with OOD detection capability. We compare Framework-I with the mentioned baselines in the TIL and CIL setup. In the ablation study, preceding variants are included, such as N_M-CLD-OOD, M-CLD-N_OOD, and N_M-CLD-N_OOD, which represents non-masked CLD with OOD detection capability, masked CLD without OOD detection capability and non-masked CLD without OOD detection capability respectively.

Table 4.1: Optimal hyperparameter values for Framework-I (M-CLD-OOD) and its variants on benchmark datasets.

Dataset	ρ_{ILD}	ρ_{GCD}	ρ_{OOD}	γ_{GCD}	γ_{OOD}	$\gamma_{R(1)}$	$\gamma_{R(2)}$	$\mathcal{C}_i$
M-5T	0.1	0.1	0.1	0.5	0.5	0.25	0.1	128
C-10	0.1	0.2	0.2	1	1	1	0.75	128
CH-5T	0.1	0.2	0.2	1	1	1.5	1	150
CH-10T	0.1	0.2	0.2	1	1	2.5	1.5	200
I1H-5T	0.1	0.1	0.1	1	1	1	0.75	160
I1H-10T	0.1	0.1	0.1	1	1	1	0.75	180
I1H-20T	0.1	0.2	0.1	1	1	1.5	0.75	180
I2H-40T	0.1	0.2	0.2	1	1	1.5	1	190
I2H-50T	0.1	0.2	0.2	1	1	2.5	1.5	200
I2H-100T	0.1	0.2	0.2	1	1	3	2	200

4.2.2 Training and evaluation setup

We modify the existing ResNet-18 to create a feature extractor capable of supporting hard attention. For CIFAR-10, we configure the network to have 64 input planes while for more complex datasets, i.e., CIFAR-100 (CH) and Tiny-ImageNet (I1H & I2H), we widen the network by increasing input planes to 128. We train with a 256 batch size for each dataset. The inclusion of OOD samples results in the effective batch size being quadrupled. To reduce computational costs, we resize the images of all benchmark datasets to 32 x 32 x 3. For all baselines along with the proposed Framework-I, we train the model for 200 epochs, except for Tiny-ImageNet. We train all the models for 500 epochs on Tiny-ImageNet due to its complexity. For Framework-I (M-CLD-OOD) along with its variants N_M-CLD-OOD, M-CLD-N_OOD and N_M-CLD-N_OOD, we use LARS [130] as an optimization function to minimize the loss, with initial learning rate 0.03. We employ a linear increase in the learning rate of 0.1 per epoch for the initial 10 epochs until it reaches 1.0. After 10 epochs, we decay the learning rate using a cosine scheduler [131] without restarts as in [28, 132].

For the baseline UCL techniques, we use SGD optimizer with a learning rate of 0.03, weight decay of $5e^{-4}$, and a momentum of 0.9. In the case of replay-based strategies, we set the buffer size to 256 for all datasets. The evaluation of all learned representations is carried out using a KNN classifier [52] across three distinct runs. We conduct experiments on a DGX-A100 system equipped with eight 40 GB GPUs. The hyperparameter settings are explained in the subsequent section.

Table 4.2: Optimal hyperparameters for baseline methods on M-5T, C-5T, CH-5T, and CH-10T. (while Table 4.3 presents the values for the CH-20T, I1H-5T, I1H-10T, and I1H-20T datasets, and Table 4.4 contains the values for the I2H-40T, I2H-50T, and I2H-100T datasets.)[4]

Method	M-5T	C-5T	CH-5T	C-H-10T
SI	$c : 0.4,\ \xi : 1$	$c : 0.4,\ \xi : 1$	$c : 0.4,\ \xi : 1$	$c : 0.01,\ \xi : 1$
PNN	$\alpha : 0.1$	$\alpha : 0.1$	$\alpha : 0.1$	$\alpha : 0.2$
DER	$\alpha : 0.4$	$\alpha : 0.01$	$\alpha : 0.01$	$\alpha : 0.1$
LUMP	$\lambda : 0.1$	$\lambda : 0.1$	$\lambda : 0.01$	$\lambda : 0.3$
Co2L	$\eta : 0.01, \tau : 0.1,$ $\kappa : 0.2, \kappa^* : 0.01$	$\eta : 0.5, \tau : 0.5,$ $\kappa : 0.2, \kappa^* : 0.01$	$\eta : 0.5, \tau : 0.5,$ $\kappa : 0.2, \kappa^* : 0.01$	$\eta : 0.5, \tau : 0.5,$ $\kappa : 0.2, \kappa^* : 0.01$

4.2.3 Hyperparameters setting

In any machine learning algorithm, it is essential to evaluate the impact of hyperparameters on the performance. In the proposed Framework-I, we have a set of hyperparameters that influence performance in various ways, such as temperature parameter ρ in the loss function for instance-level, group level, and pseudo-group level contrastive losses, we denote them as ρ_{ILD}, ρ_{GCD} and ρ_{OOD} respectively for further reference. The positive scalling hyperparameter γ_R, γ_{GCD} and γ_{OOD} are for regularization, group-level and pseudo-group-level losses respectively. As explained in (4.10), we consider only two values of γ_R, such as $\gamma_{R(1)}$ for first task and $\gamma_{R(2)}$ for all the remaining tasks. As detailed in §4.1.1.2, the number

[4]Note: The notations presented in these tables are not connected to the existing notations in the other part of this paper; they have been directly adopted from the referenced research papers for the purpose of referring to their original definitions. For more detailed information about these notations, directly refer to the respective research papers.

of clusters $\mathcal{C}_i$ assigned for group-level instance discrimination is also an important hyper-parameter. As discussed in §4.1.2, the value of s is calculated dynamically during training using a linear annealing strategy followed by [18]. During inference, we use a fixed value of s_{max} = 700, which helps in the approximation of a unit step function as a mask. To ensure a fair comparison for training all UCL baselines, we use the same set of hyperparameters as those used in SimSiam [53] to evaluate the sensitivity of UCL to hyperparameters. We initially adopted hyperparameters from [11] for all UCL experiments, but fine-tuned them to fit in the our dataset settings. Similarly, for Co2L, we initially used hyperparameters from [12], then fine-tuned them. On the other hand, we tune the hyperparameters for CL strategies for the proposed Framework-I. In order to find the optimal hyperparameters for each approach, we conduct a grid search while utilizing a task sequence determined by a single seed. The optimal hyperparameter values for Framework-I (M-CLD-OOD) and its variants on benchmark datasets is presented in Table 4.1. In Table 4.2, 4.3, and 4.4, optimal hyperparameter values for baseline models are presented.

Table 4.3: Optimal hyperparameters for baseline methods on CH-20T, I1H-5T, I1H-10T and I1H-20T.

Method	CH-20T	I1H-5T	I1H-10T	I1H-20T
SI	$c: 0.1,\ \xi: 1$	$c: 0.01,\ \xi: 1$	$c: 0.01,\ \xi: 1$	$c: 0.01,\ \xi: 1$
PNN	$\alpha: 0.2$	$\alpha: 0.1$	$\alpha: 0.2$	$\alpha: 0.4$
DER	$\alpha: 0.1$	$\alpha: 0.01$	$\alpha: 0.01$	$\alpha: 0.01$
LUMP	$\lambda: 0.1$	$\lambda: 0.1$	$\lambda: 0.1$	$\lambda: 0.3$
Co2L	$\eta: 0.3, \tau: 0.5,$ $\kappa: 0.1, \kappa^*: 0.01$	$\eta: 0.1, \tau: 0.5$, $\kappa: 0.1, \kappa^*: 0.1$	$\eta: 0.1, \tau: 0.5,$ $\kappa: 0.1, \kappa^*: 0.1$	$\eta: 0.2, \tau: 0.5,$ $\kappa: 0.1, \kappa^*: 0.01$

Table 4.4: Optimal hyperparameters for baseline methods on I2H-40T, I2H-50T, and I2H-100T.

Method	I2H-40T	I2H-50T	I2H-100T
SI	$c: 0.01,\ \xi: 1$	$c: 0.01,\ \xi: 1$	$c: 0.01,\ \xi: 1$
PNN	$\alpha: 0.4$	$\alpha: 0.4$	$\alpha: 0.4$
DER	$\alpha: 0.01$	$\alpha: 0.01$	$\alpha: 0.01$
LUMP	$\lambda: 0.3$	$\lambda: 0.3$	$\lambda: 0.4$
Co2L	$\eta: 0.5, \tau: 0.5$, $\kappa: 0.1, \kappa^*: 0.01$	$\eta: 0.5, \tau: 0.5,$ $\kappa: 0.1, \kappa^*: 0.01$	$\eta: 0.6, \tau: 0.5,$ $\kappa: 0.1, \kappa^*: 0.01$

Table 4.5: Overall-average TIL Accuracy (in %) for Framework-I and baselines on benchmark datasets across all tasks. Where, -τ is the number of tasks in the name of the datasets, present in the first column. The last row shows the average accuracy of each method over all datasets. SI, DER, PNN, LUMP, and Framework-I represent the UCL methods and Co2L represents the SCL method. Additional details about the table are available in §4.2.4.

Methods → Datasets ↓	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Co2L(Sup)**
M-5T	99.46	99.28	99.23	99.74	99.68	**99.85**
C-5T	91.4	91.64	91.05	91.23	**94.12**	92.02
CH-5T	61.11	61.60	58.50	60.04	65.3	**69.46**
CH-10T	68.21	69.37	62.47	69.15	73.63	**73.86**
CH-20T	75.60	76.47	65.17	78.74	**80.45**	78.59
I1H-5T	54.62	53.44	48.09	52.90	56.38	**56.40**
I1H-10T	55.39	58.29	49.36	63.68	**65.4**	60.20
I1H-20T	67.16	64.32	57.29	71.18	**75.16**	69.61
I2H-40T	61.37	61.47	55.38	71.38	**73.47**	66.34
I2H-50T	64.05	64.23	58.13	73.20	**74.59**	69.78
I2H-100T	79.52	78.00	69.54	85.89	**86.61**	79.87
Average	70.72	70.74	64.93	74.28	**76.79**	74.18

4.2.4 Results and comparative analysis

Tables 4.5, 4.6, 4.7, and 4.8 show the evaluation results for baselines and the proposed framework on M-5T, C-5T, CH-5T, CH-10T, CH-20T, I1H-5T, I1H-10T, I1H-20T, I2H-40T, I2H-50T, and I2H-100T datasets. The row header contains the names of the baseline methods. The last row of the tables shows overall-average performance of the respective models. The remaining rows show the performances of each model on the dataset indicated in the first column. We observe that all the mentioned baselines, which represent various CL strategies, yield lower performance compared to Framework-I with negligible or nearly zero forgetting.

4.2.4.1 Comparison of overall-average TIL accuracy across all tasks

The method of calculating overall-average accuracy across all tasks is explained in §2.4. In Table 4.5 and Figures 4.3, 4.4, 4.5 we show the overall-average TIL accuracy of the models. In all cases, Framework-I achieves significantly better overall-average accuracies than representative baselines. It can be observed from the last row of the table, which contains the average accuracy of all models, that Framework-I achieves a 6.07%, 6.05%, 11.86%, and 2.51% improvement compared to SOTA baselines such as SI, DER, PNN, and LUMP respectively (thus answering the **RQ2** [§3.5] in TIL setup). Outperforming both DER and LUMP indicates that there is no need of a replay buffer anymore to improve the performance (thus answering the **RQ1** [§3.5] in TIL setup). The improved accuracy of our model, in comparison to PNN, underscores the effectiveness of our approach over architecture-based strategies. Moreover, it achieves performance that is on par with SCL methods like Co2L and even surpasses it in terms of average performance across all datasets by 2.61%. It can also be noted that Framework-I consistently outperforms Co2L, particularly as the number of tasks increases. This trend is evident across various datasets, with significant performance improvements. Specifically, for tasks such as C-5T, CH-20T, I1H-10T, I1H-20T, I2H-40T, I2H-50T, and I2H-100T, Framework-I achieves improvements of 1.92%, 1.86%, 5.2%, 5.55%, 7.13%, 4.81%, and 6.74%, respectively. This demonstrates the proposed framework is comparable with SCL methods. This is noteworthy, as it highlights the gains made without the need for labeled data. We compare our method with Co2L due to its contrastive learning foundation. While we do not claim to surpass all SOTA SCL baselines, our aim is to show comparable performance, with further validation needed through additional experiments.

4.2.4.2 Comparison of overall-average TIL forgetting across all tasks

The method of calculating overall-average forgetting across all tasks is explained in §2.4. In Table 4.6 and Figure 4.6, we show the overall-average TIL forgetting of the models. It can be observed that both UCL and SCL baselines exhibit substantial forgetting in comparison to Framework-I. In contrast, Framework-I demonstrates **almost zero** or significantly negligible CF, as shown by the green downward arrow in Figure 4.6. while SCL baselines like

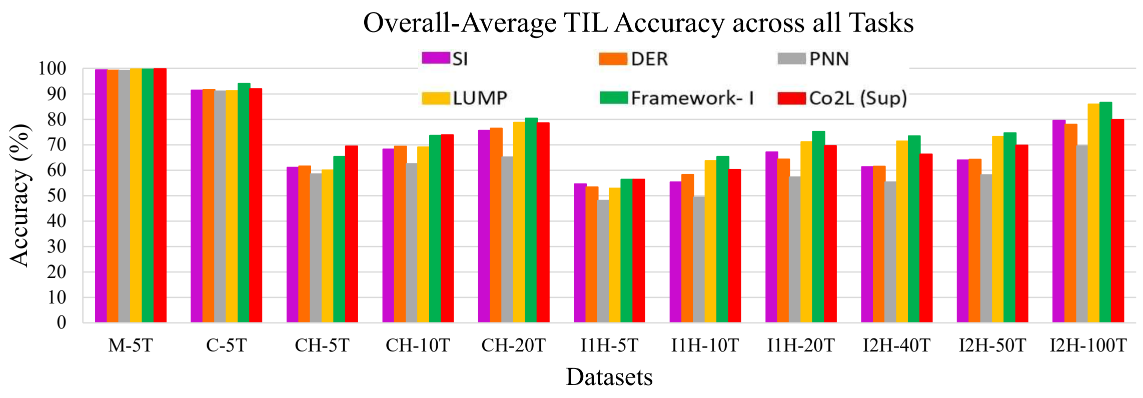


Figure 4.3: Overall-average TIL Accuracy for Framework-I and baselines on benchmark datasets.

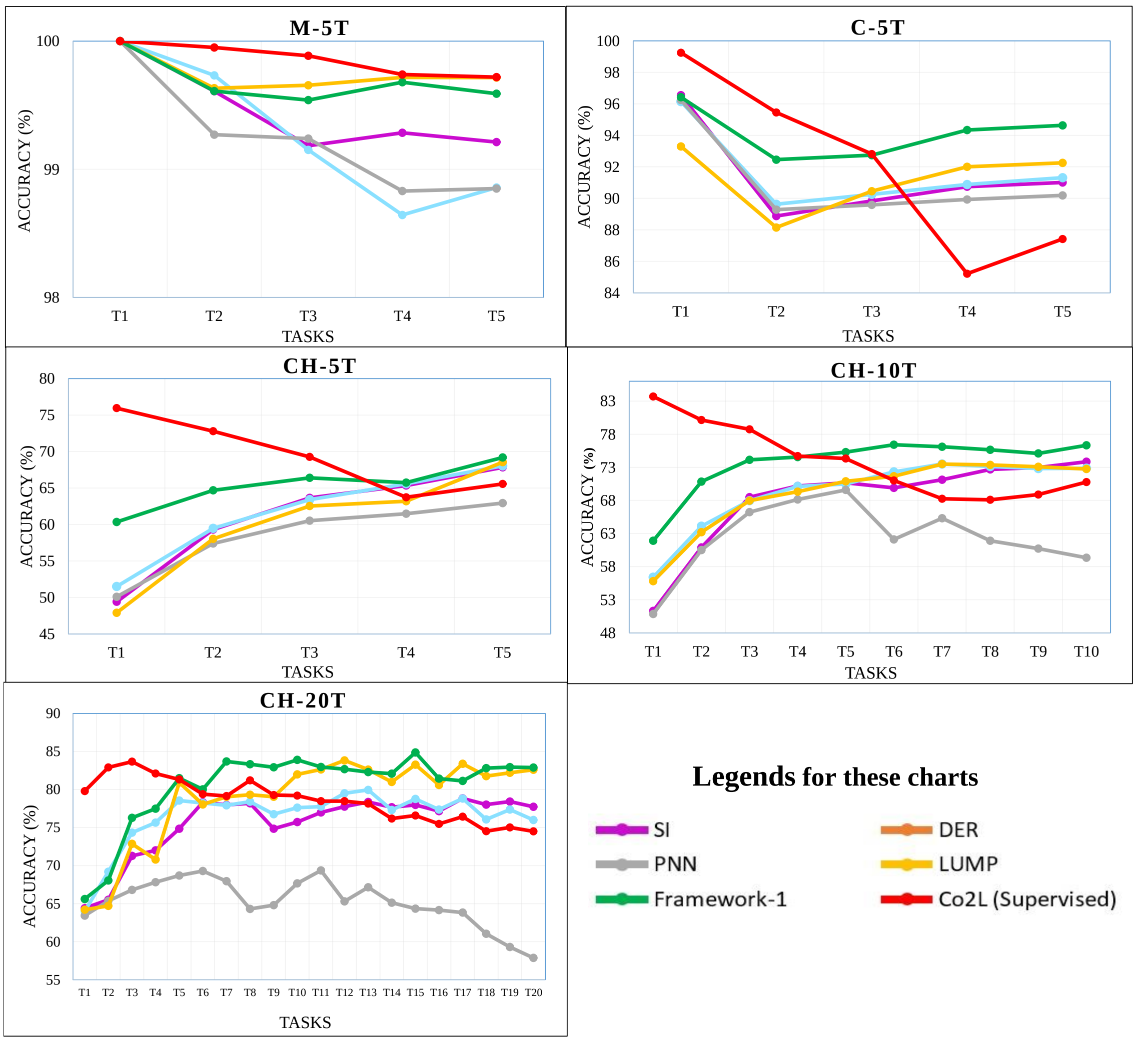


Figure 4.4: TIL Performance comparison for Framework-I and baselines on benchmark datasets across all tasks.

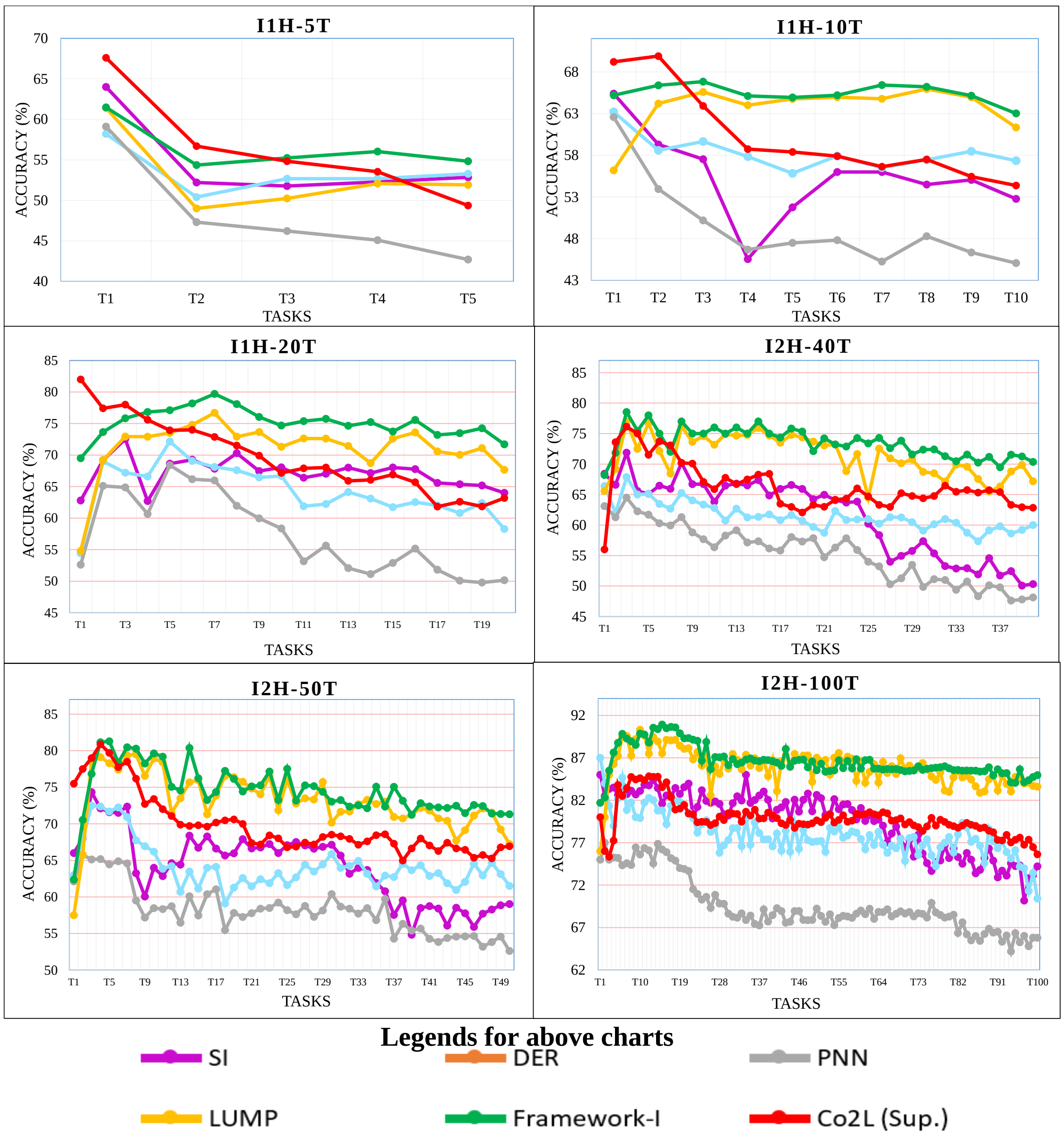


Figure 4.5: TIL Performance comparison for Framework-I and baselines on variants of Tiny-ImageNet dataset across tasks ranging from 5 to 100.

Table 4.6: Overall-average TIL Forgetting (in %) for Framework-I and baselines on benchmark datasets across all tasks.

Methods → **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Co2L(Sup)**
M-5T	0.45	0.72	0.0002	0.17	0.001	0.07
C-5T	1.85	3.72	0.0001	2.27	0.002	7.35
CH-5T	2.36	3.47	0.0004	3.18	0.001	14.02
CH-10T	2.57	3.63	0.0008	2.76	0.001	15.46
CH-20T	5.86	5.67	0.003	1.98	0.029	16.18
I1H-5T	7.71	5.33	0.0005	3.78	0.002	14
I1H-10T	11.66	8.41	0.0008	4.47	0.001	16.79
I1H-20T	7.89	8.51	0.009	5.47	0.06	17.38
I2H-40T	8.73	10.65	0.012	5.61	0.0481	18.56
I2H-50T	11.31	11.61	0.057	7.2	0.084	19.22
I2H-100T	13.04	12.48	0.14	7.49	0.1482	20.53
Average	6.68	6.75	0.02	4.03	0.0343	14.51

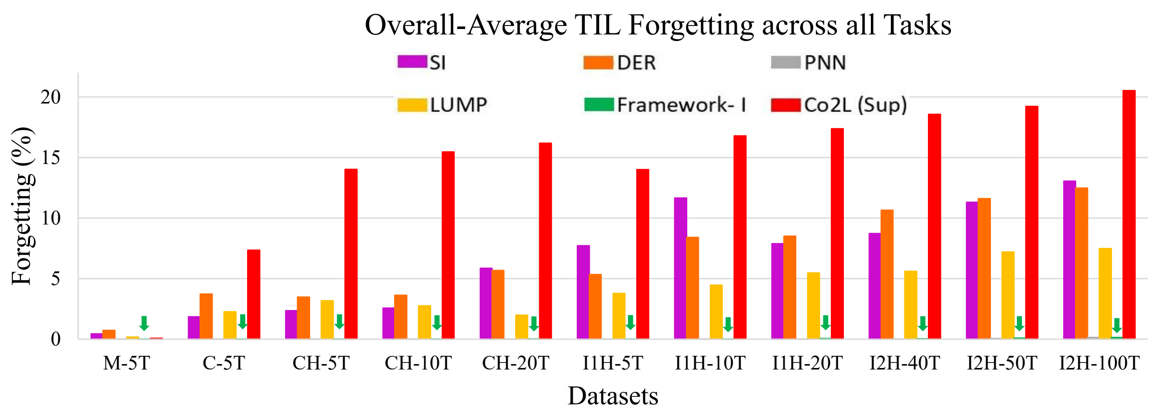


Figure 4.6: Overall-average TIL Forgetting for Framework-I and baselines on benchmark datasets. Additionally, in Framework-I, almost zero CF is represented by a vertical green arrow.

Co2L exhibit an overall average forgetting of 14.51%, and UCL baselines such as SI, DER, PNN, and LUMP demonstrate 6.68%, 6.75%, 0.02%, and 4.03% overall average forgetting, respectively (thus again confirming the **RQ2** [§3.5] in TIL setup). Achieving negligible forgetting in comparison to both DER and LUMP indicates that there may no longer be a need for a replay buffer to alleviate CF (thus again confirming the **RQ1** [§3.5] in TIL setup).

Table 4.7: Overall-average CIL Accuracy (in %) for Framework-I and baselines on benchmark datasets across all tasks.

Methods $\rightarrow$ Datasets $\downarrow$	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Co2L(Sup)**
M-5T	97.57	96.33	97.39	99.16	98.87	**98.95**
C-5T	83.46	83.67	83.54	83.38	**87.03**	82.53
CH-5T	53.7	53.82	50.06	52.56	57.14	**65.38**
CH-10T	53.83	53.81	49.2	55.95	56.26	**61.7**
CH-20T	52.69	52.86	46.28	**57.84**	57.59	57.28
I1H-5T	46.51	43.5	44.14	42.27	48.39	**51.41**
I1H-10T	40.42	42.48	39.64	**49.28**	46.8	46.81
I1H-20T	40.61	37.55	36.53	46.028	**51.63**	45.63
Average	58.60	58.0	55.85	60.68	62.96	**63.71**

4.2.4.3 Comparison of overall-average CIL accuracy across all tasks

The method of calculating overall-average CIL accuracy is explained in §2.4.3. Overall-average CIL Accuracy of the models is presented in Table 4.7 and shown in Figures 4.7, 4.8. In the average column of Table 4.7, it can be seen that Framework-I achieves an increase of 4.36%, 4.96%, 7.11%, and 2.28% in overall average CIL accuracy across all datasets in the UCL setup compared to SI, DER, PNN, and LUMP, respectively (thus answering the **RQ1** and **RQ2** [§3.5] in CIL setup as well). However at the individual level, LUMP exhibits performance gain on M-5T, CH-20T, and I1H-10T datasets, but Framework-I also outperforms all the UCL baselines on C-5T, CH-5T, CH-10T, I1H-5T, and I1H-20T datasets. The overall-average CIL accuracy of the SCL method, i.e., Co2L, is also comparable to Framework-I on all the datasets (see Figure 4.7). Surprisingly, Framework-I even outperforms Co2L on the C-5T, CH-20T, and I1H-20T datasets.

4.2.4.4 Comparison of overall-average CIL forgetting across all tasks

The method of calculating overall-average CIL forgetting is explained in §2.4.3. Overall-average CIL forgetting of the models is presented in the Table 4.8 and shown in Figure 4.9. It can be observed that both UCL and SCL baselines exhibit considerable CIL

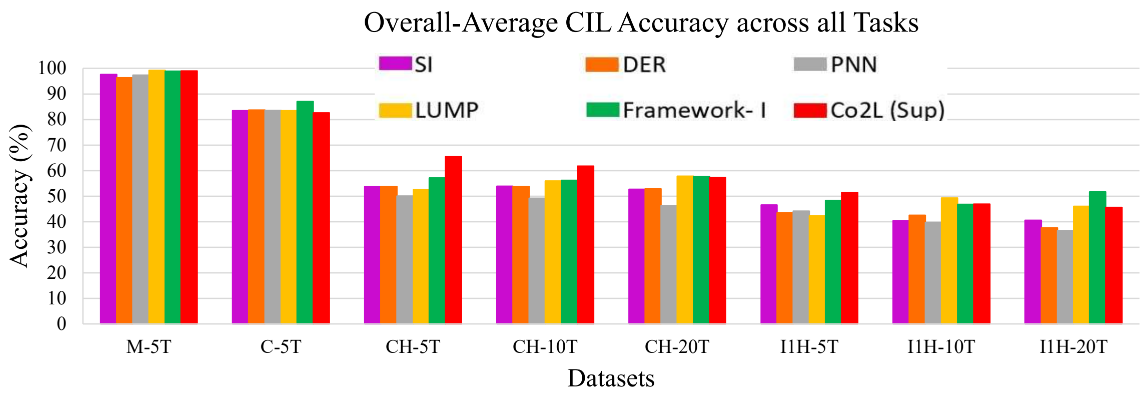


Figure 4.7: Overall-average CIL Accuracy for Framework-I and baselines on benchmark datasets.

Table 4.8: Overall-average CIL Forgetting (in %) for Framework-I and baselines on benchmark datasets across all tasks.

Methods → **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Co2L(Sup)**
M-5T	0.16	1.02	0.0008	0.04	**0.0005**	0.24
C-5T	3.82	3.69	0.005	1.54	**0.001**	12.78
CH-5T	1.57	2.82	0.009	2.67	**0.001**	17.48
CH-10T	2.31	2.77	0.013	1.861	**0.001**	24.77
CH-20T	4.71	1.76	0.063	2.5	**0.01**	27.53
I1H-5T	7.54	3.78	0.007	2.93	**0.002**	18.03
I1H-10T	11.21	6.09	0.017	3.65	**0.001**	25.59
I1H-20T	5.86	3.11	0.094	5.18	**0.03825**	30.09
Average	4.65	3.13	0.0261	2.55	**0.00685**	19.56

forgetting compared to Framework-I in the CIL setting as well. Framework-I, on the other hand, shows almost zero or significantly negligible CF, as indicated by the green downward arrow in Figure 4.9. In comparison, SCL baselines like Co2L exhibit an overall-average forgetting of 19.56%, while UCL baselines like SI, DER, PNN, and LUMP demonstrate 4.65%, 3.13%, 0.0261%, and 2.55% overall average forgetting, respectively (thus again confirming the **RQ1** and **RQ2** [§3.5] in CIL setup as well).

4.2.5 Ablation study

In this section, we study the effectiveness of multi-tasking loss function along with OOD detection capability and hard attention.

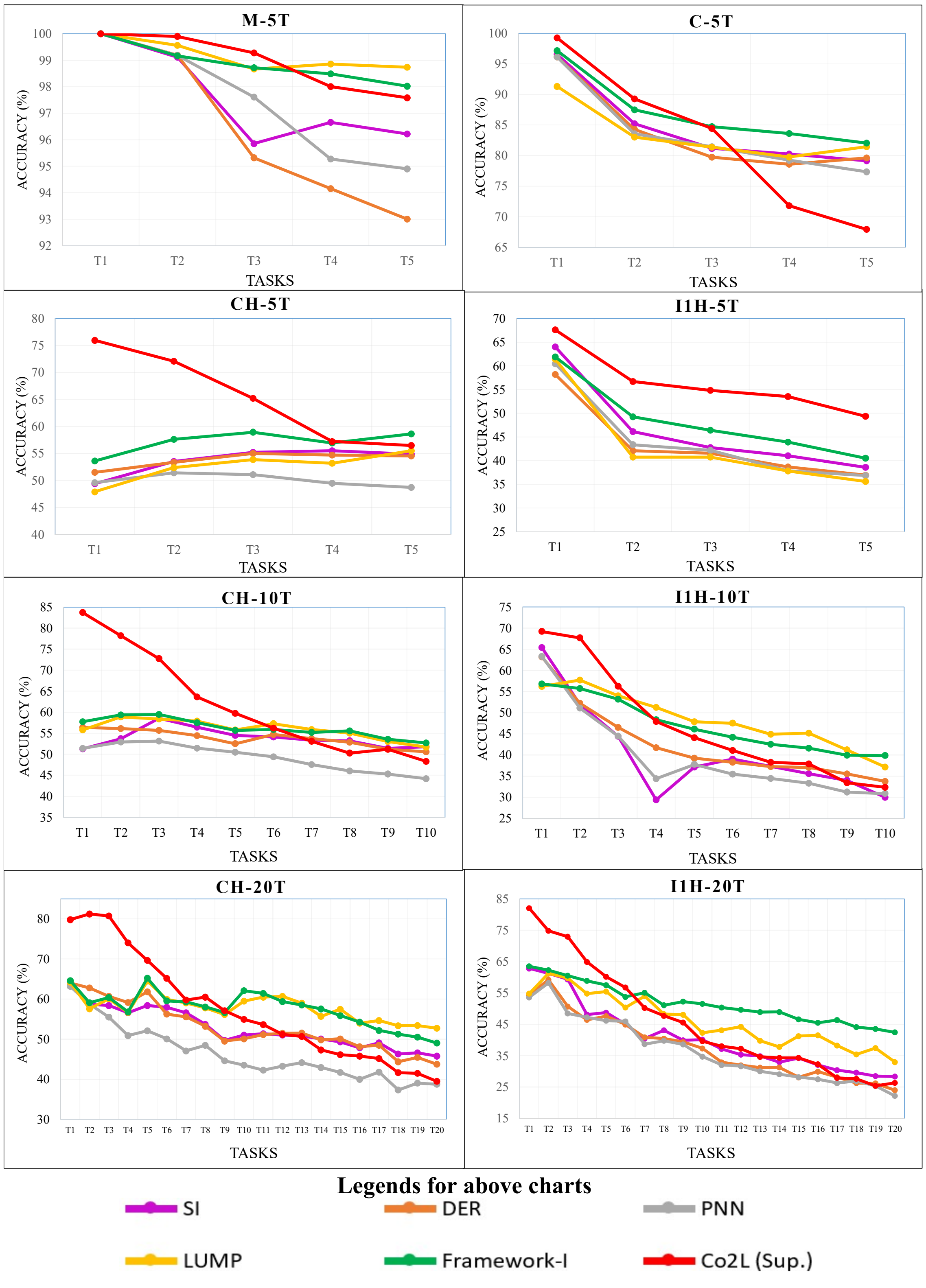


Figure 4.8: CIL Performance comparison for Framework-I and baselines on benchmark datasets across all tasks.

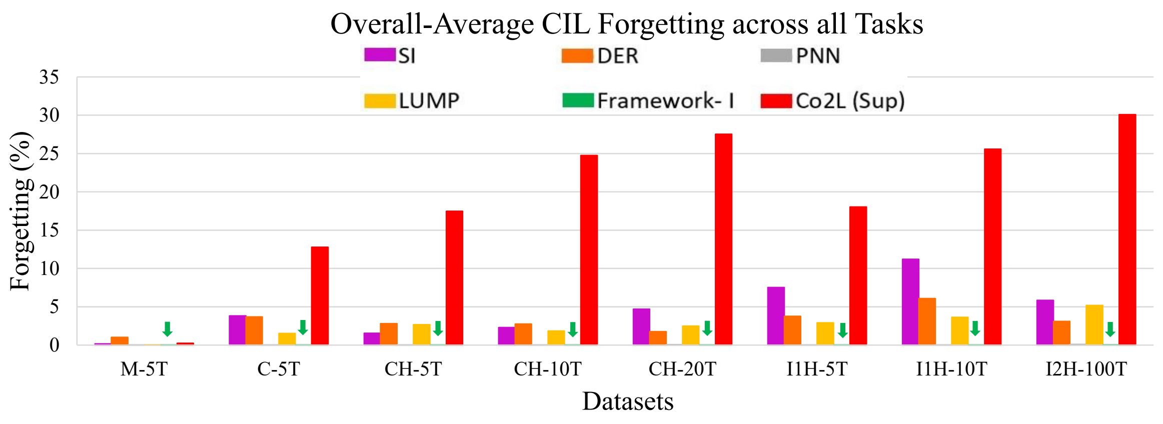


Figure 4.9: Overall-average CIL Forgetting for Framework-I and baselines on benchmark datasets.

4.2.5.1 Effect of multi-tasking loss function along with OOD detection capability

Tables 4.9, 4.13 and Figures 4.10, 4.13 demonstrate that the successive variants of the proposed Framework-I (M-CLD-OOD), show gradual improvements in terms of overall-average accuracy. Figure 4.12, shows the presence of well-defined task boundaries within the representation space, primarily due to the incorporation of task-specific projection layers. Moreover, the high quality of clusters remains consistent throughout incremental training, signifying that the Framework-I effectively retains knowledge. These results indicate that the multi-tasking loss function with the pseudo-OOD detection capability helps in UCL (thus answering the **RQ3** [§3.5] in both TIL and CIL setup).

4.2.5.2 Effect of hard attention

Tables 4.10, 4.14, and Figures 4.11, 4.14 show that successive variants of the Framework-I (M-CLD-OOD), exhibit gradual improvements in terms of achieving nearly zero forgetting. It can also be inferred that when we do not utilize hard attention, the model experiences significant forgetting. For example, a notable overall average TIL forgetting of 8.62% and CIL forgetting of 7.70% across all the datasets in the case of N_M-CLD-N_OOD. However, when we do not use hard attention but still use OOD discrimination capability, the

Table 4.9: Overall-average TIL Accuracy (in %) of successive variants of Framework-I.

Methods → **Datasets ↓**	**N_M-CLD-N_OOD**	**N_M-CLD-OOD**	**M-CLD-N_OOD**	**M-CLD-OOD (Framework-I)**
M-5T	98.18	98.83	99.66	**99.68**
C-5T	87.06	87.57	91.01	**94.12**
CH-5T	57.56	58.2	59.13	**65.3**
CH-10T	64.96	65.83	69.61	**73.63**
CH-20T	71.97	72.86	77.81	**80.45**
I1H-5T	52.28	53.04	53.17	**56.38**
I1H-10T	57.38	59.79	61.29	**65.4**
I1H-20T	69.36	71.33	74.11	**75.16**
I2H-40T	68.04	67.75	71.12	**73.47**
I2H-50T	70.84	69.89	72.81	**74.59**
I2H-100T	84.04	83.462	85.22	**86.61**
Average	71.09	71.69	74.08	**76.80**

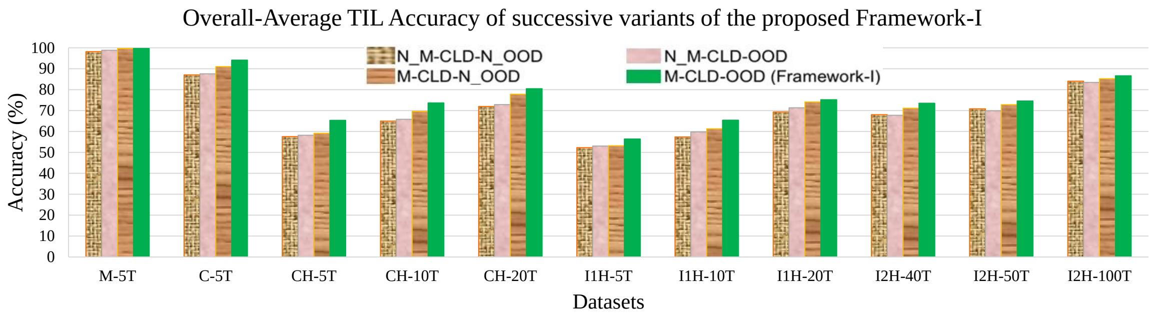


Figure 4.10: Overall-Average TIL Accuracy of successive variants of Framework-I.

model experiences forgetting, although to a lesser extent, with an overall-average TIL forgetting of 5.45% and CIL forgetting of 4.19% across all the datasets as seen in N_M-CLD-OOD. Nevertheless, when we use hard attention without OOD discrimination capability in the UCL setting, the forgetting is even less, with an average TIL forgetting of 0.0208% and CIL Forgetting of 0.0158% across all datasets, as observed in the case of M-CLD-N_OOD. These findings suggest that the incremental improvements in the Framework-I, especially the inclusion of hard attention with OOD detection capabilities, lead to increased performance compared to the SOTA baselines with only 0.0343% average TIL forgetting and 0.0068% average CIL forgetting across all the tasks. This indicates that the influence of hard attention on the performance of the Framework-I enables this behavior.

It is also shown in Figures 4.11, 4.14 that Framework-I (M-CLD-OOD) surpasses its

Table 4.10: Overall-average TIL Forgetting (in %) of successive variants of Framework-I.

Methods → Datasets ↓	N_M-CLD-N_OOD	N_M-CLD-OOD	M-CLD-N_OOD	M-CLD-OOD (Framework-I)
M-5T	0.74	0.18	0.01	**0.001**
C-5T	7.01	0.5	0.013	**0.002**
CH-5T	8.22	0.42	0.02	**0.001**
CH-10T	10.34	0.78	0.002	**0.001**
CH-20T	12.03	14.6	0.048	**0.029**
I1H-5T	6.05	0.39	0.003	**0.002**
I1H-10T	7.79	0.74	0.002	**0.001**
I1H-20T	15.63	9.91	0.065	**0.06**
I2H-40T	10.06	11.37	0.0196	**0.0481**
I2H-50T	9.59	11.92	0.0132	**0.084**
I2H-100T	7.32	9.1	0.033	**0.1482**
Average	8.62	5.45	0.0208	**0.0343**

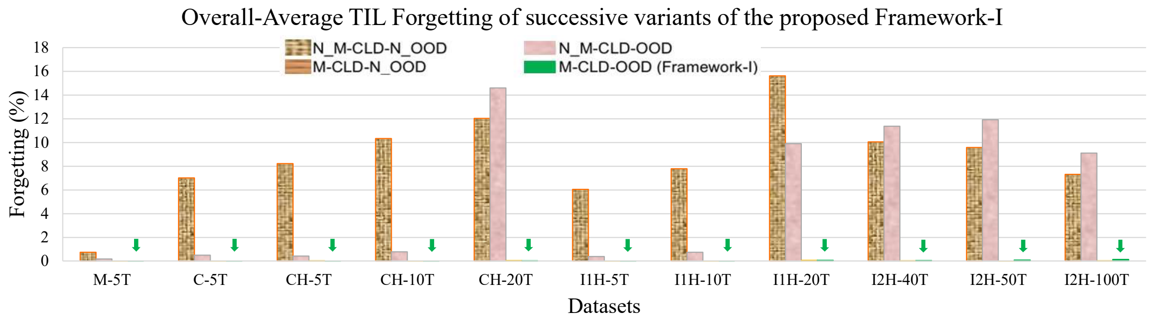


Figure 4.11: Overall-Average TIL Forgetting of successive variants of Framework-I.

variants N_M-CLD-N_OOD, N_M-CLD-OOD and M-CLD-N_OOD with considerable difference in forgetting. We have presented results on datasets containing task counts ranging from 5 to 100, and class counts from 2 to 20. In all of these scenarios, we surpass the SOTA baselines. This indicates the effectiveness of the hard attention mechanism, which is capable of handling varying task complexities, encompassing situations with a smaller number of classes and extending to scenarios with a larger number of classes.

4.2.5.3 Comparison of hard attention and soft attention

SPG [133] assigns an importance score to each weight using gradients. These scores act as soft masks to control gradient flow during backpropagation, helping to prevent CF of previous knowledge. For comparison, we modified SPG by changing the loss function

Table 4.11: TIL Performance comparison of Hard Attention and Soft Attention.

Datasets →	**CH-10T**		**I1H-10T**	
Methods ↓	**Accuracy** (%)	**Forgetting** (%)	**Accuracy** (%)	**Forgetting** (%)
Framework-I (Hard Attention)	73.63	0.001	65.4	0.001
SPG Unsupervised (Soft Attention)	45.74	21.35	32.57	23.68

Table 4.12: Number of network parameters in Framework-I after learning the final task.

Methods → **Datasets ↓**	**Without Mask**	**With Mask**	**Difference (%)**
M-5T	12151872	12209792	0.0047
C-5T	12151872	12209792	0.0047
CH-5T	46618752	46734592	0.0025
CH-10T	48584832	48835712	0.0051
CH-20T	52516992	53037952	0.0098
I1H-5T	46618752	46734592	0.0025
I1H-10T	48584832	48835712	0.0051
I1H-20T	52516992	53037952	0.0098
I2H-40T	60381312	61442432	0.0173
I2H-50T	64313472	65644672	0.0203
I2H-100T	83974272	86655872	0.0309

to make it unsupervised and compared it with hard attention in the proposed setup. Comparison of soft attention and hard attention reveals that soft attention struggles to scale with deeper model architectures like ResNet [134]. As presented in Table 4.11, Framework-I achieves 73.63% and 65.4% TIL accuracy with nearly zero forgetting on the CH-10T and I1H-10T datasets, respectively. In contrast, SPG in an unsupervised setup with the ResNet architecture achieves only 45.74% and 32.57% accuracy, along with 21.35% and 23.68% CF on the same datasets. This demonstrates the superior performance of Framework-I in terms of both accuracy and memory retention because of the hard attention mechanism. These observations, regarding the degrading performance of soft attention, are consistent with findings in IBM [40], which also highlight similar challenges in scaling soft attention mechanisms.

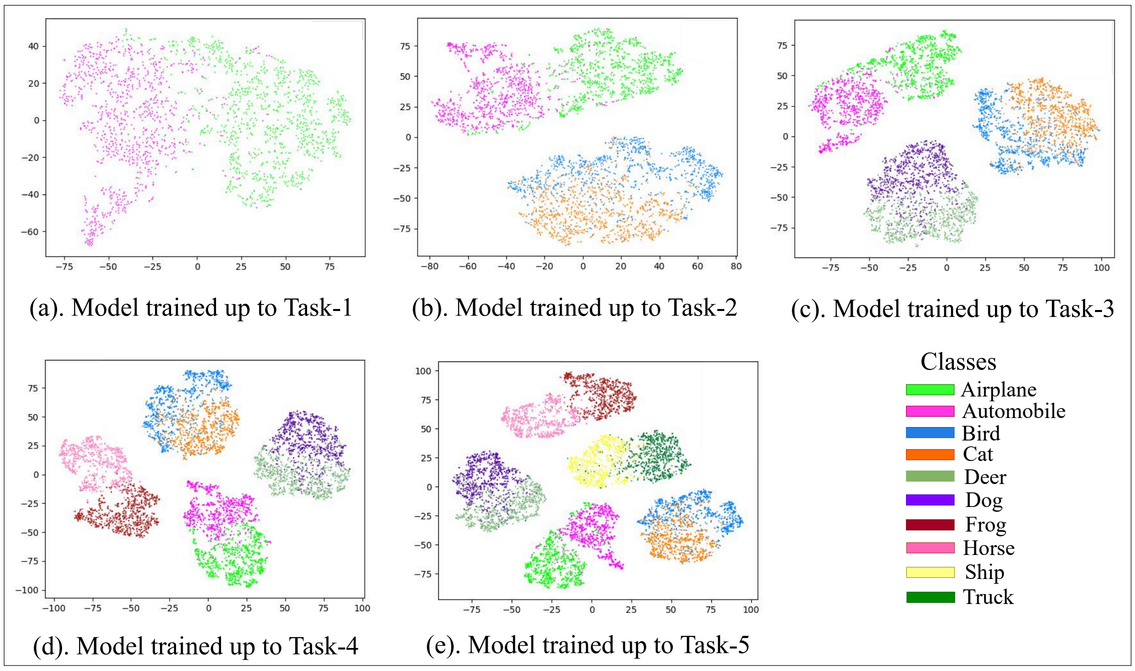


Figure 4.12: Visualization of TSNE embeddings of the representations across tasks in CIFAR-10 (test-set).

4.2.5.4 Analysis of the number of model parameters

For all experiments, we use ResNet-18 as the backbone architecture. For CIFAR100 and Tiny-ImageNet, we adopt the same ResNet-18 structure used for M-5T and C-5T, but with modifications to accommodate the increased complexity of the data. Specifically, we double the number of channels in each convolutional layer to enhance the model's capacity for learning multiple tasks and handling the richer data distributions. Due to hard attention embeddings and task-specific heads, Framework-I requires task-specific parameters for each task. In Table 4.12, we report the network parameter counts after the final task in each experiment has been trained. For the datasets M-5T, C-5T, CH-5T, CH-10T, CH-20T, I1H-5T, I1H-10T, I1H-20T, I2H-40T, I2H-50T, and I2H-100T, we incorporate task-specific parameters, resulting in the following percentage differences in parameter count after each task: 0.0047%, 0.0047%, 0.0025%, 0.0051%, 0.0098%, 0.0025%, 0.0051%, 0.0098%, 0.0173%,

Table 4.13: Overall-average CIL Accuracy (in %) of successive variants of Framework-I.

Methods → Datasets ↓	N_M-CLD-N_OOD	N_M-CLD-OOD	M-CLD-N_OOD	M-CLD-OOD (Framework-I)
M-5T	94.91	95.33	97.89	**98.87**
C-5T	76.97	78.84	79.16	**87.03**
CH-5T	48.81	50.01	51.13	**57.14**
CH-10T	51.19	51.47	51.85	**56.26**
CH-20T	47.1	48.52	50.32	**57.59**
I1H-5T	42.03	44.67	44.96	**48.39**
I1H-10T	42.49	44.96	45.16	**46.80**
I1H-20T	45.97	37.39	43.89	**51.63**
Average	56.18	56.40	58.05	**62.96**

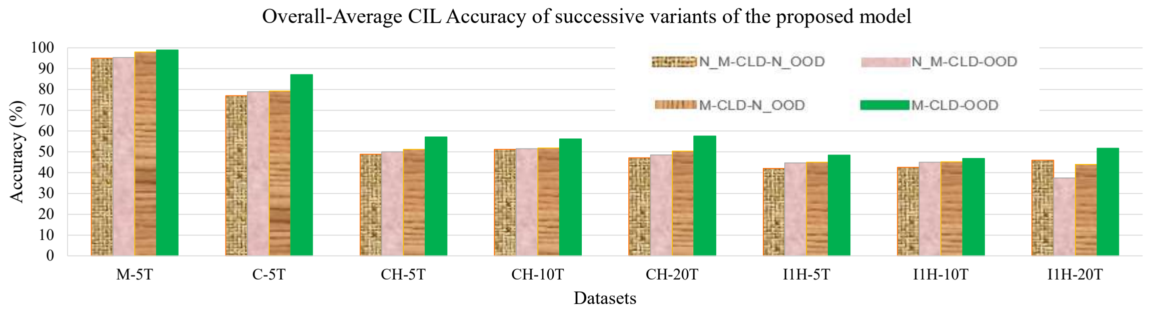


Figure 4.13: Overall-Average CIL Accuracy of successive variants of Framework-I.

0.0203%, and 0.0309% respectively. These small increments in parameter count ensure minimal growth in model complexity while accommodating the addition of new tasks. Contrastive learning also introduces task-specific parameters through the projection functions. However, these parameters can be discarded during deployment, as they are not required for inference.

4.2.5.5 Reason for rotation augmentation used in OOD detection

For OOD detection, traditional methods rely on large labeled datasets, which are costly and difficult to scale. Unsupervised semantic representation learning addresses this challenge by eliminating the need for manual annotations, enabling more efficient use of abundant visual data, and making OOD detection more scalable. In this work, for OOD detection, we train the model to learn image features by identifying the 2D rotations

{0°,90°,180°,270°} applied to the input image. This approach enables the Framework-I to capture useful representations that assist in distinguishing in-distribution from OOD data. Gidaris et al. conducted comparisons with various rotation angles, including {0°,180°}, {90°,270°}, and {0°,45°,90°,135°,180°,255°,270°,315°}. However, they observed that the best performance was achieved using only the set of angles {0°,90°,180°,270°}. Similarly, the authors in [135, 132, 136, 137] demonstrated that self-supervised learning, particularly through a rotation or distributionally shifted instance prediction task, enhances the model's ability to detect challenging OOD examples. Golan et al. [138] chose geometric transformations over non-geometric ones, as the latter was found to degrade performance and fail to preserve important image features. *These insights guided the design of our approach, where we focus exclusively on rotation augmentations (a type of geometric transformation) for OOD detection.* It can be observed that Framework-I (M-CLD-OOD) achieves higher performance in comparison to its variants, including N_M-CLD-N_OOD and M-CLD-N_OOD, where pseudo-OOD detection capability is absent. It can be inferred from Tables 4.9, 4.13, and Figures 4.10, 4.13, that Famework-I (M-CLD-OOD) achieves incremental gains of 5.71%, and 2.72% in terms of overall-average TIL accuracy and 6.78%, and 4.91% in terms of overall-average CIL accuracy across all tasks, when compared with its variants such as N_M-CLD-N_OOD, and M-CLD-N_OOD respectively. This can be attributed to the inclusion of cross-level pseudo-instance (rotation augmented) group discrimination for OOD Detection in multi-tasking loss function. This effect is noted in both TIL and CIL settings, but it is more relevant to the CIL setting.

4.3 Discussion

As explained in §4.2.4, Framework-I showcases performance improvement by incorporating cross-level pseudo OOD detection with hard attention. For learning the unsuper-

Table 4.14: Overall-average CIL Forgetting (in %) of successive variants of Framework-I.

Methods → **Datasets ↓**	**N_M-CLD-N_OOD**	**N_M-CLD-OOD**	**M-CLD-N_OOD**	**M-CLD-OOD (Famework-I)**
M-5T	0.51	0.14	0.001	**0.0005**
C-5T	6.9	0.38	0.003	**0.001**
CH-5T	8.05	0.25	0.03	**0.001**
CH-10T	11.5	0.4	0.001	0.001
CH-20T	13	16.51	0.082	**0.01**
I1H-5T	6.37	0.25	0.002	0.002
I1H-10T	8.22	0.25	0.001	0.001
I1H-20T	6.99	15.39	0.0065	**0.03825**
Average	7.70	4.19	0.0158	**0.0068**

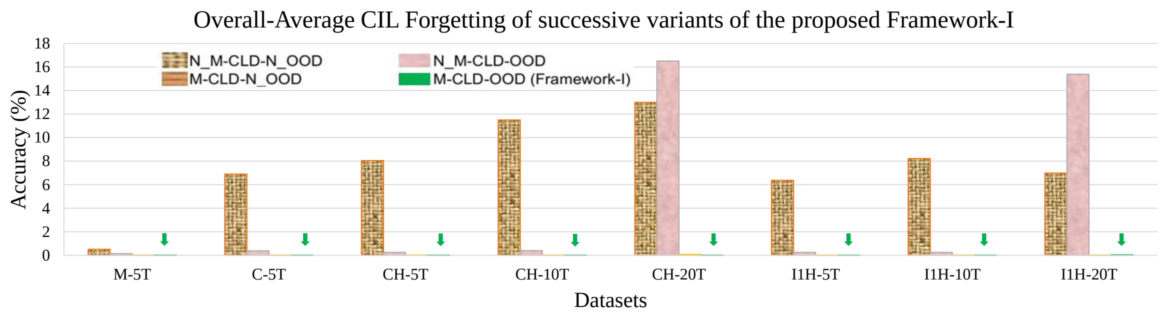


Figure 4.14: Overall-Average CIL Forgetting of successive variants of Framework-I.

vised representations, we initially experimented using only instance-level contrastive loss function like in LUMP [11] and DER [14] instead of opting for a multilevel loss function. However, this approach introduced biases towards instance discrimination over invariant grouping, thus compromising the robustness of the model. To address this limitation, we integrated cross-level instance and group discrimination within the contrastive loss function. This combination effectively overcomes the drawbacks associated with using instance-level and group-level contrastive loss individually, as explained in §4.1.1.

Furthermore, we experimented with WSN [87], SupSup [22] and SPG [83] as alternatives to hard attention. SupSup employs a fixed, randomly initialized base network and finds a supermask for each new task, which selectively activates or deactivates specific weights to form a subnetwork. This approach avoids forgetting by not updating the network weights. However, while this prevents forgetting, the performance of SupSup is sub-optimal because the fixed weights constrain its representational capacity in an unsupervised setting. SPG [83] assigns importance scores to individual weights based on gradient information. These scores function as soft masks during backpropagation, regulating gradient flow to mitigate CF. While SPG shows promising results with compact neural architectures like AlexNet [139], its efficacy diminishes when applied to more complex networks such as ResNet [134]. The challenges in scaling parameter-level soft-masking to larger architectures hindered our research goals and resulted in decreased performance.

In summary, Framework-I introduces a novel unsupervised framework with task-specific hard attention and OOD detection for effective TIL and CIL with minimal forgetting. It outperforms SOTA baselines, eliminates the need for replay buffer, and narrows the gap with supervised methods.

Chapter 5

Addressing Capacity Saturation in Hard Attention and Improving Accuracy with Evidential OOD Detection

The previously presented Framework-I based on hard attention and cross-level discrimination has performed pretty well compared to existing SOTA approaches. However, it suffers from the "capacity saturation problem". When hard attention is used independently, it imposes excessive constraints on parameter updates, leading to suboptimal learning performance for new tasks. This exacerbates the capacity problem by restricting the availability of free neurons and potentially increases forgetting. Therefore, instead of solely relying on the importance of individual parameters, in Framework-II, we also focus on constraining the direction of gradients. This ensures that the neural network learns new tasks by updating its parameters in directions orthogonal to the gradient subspaces considered crucial for previous tasks. Through the empirical study, we found that integrating hard attention with gradient projection can complement the advantages and drawbacks of each other when

The work presented in this chapter has been published in the paper: "*Unsupervised Continual Learning using Cross-Level Discrimination and Evidential Pseudo Out of Distribution Detection along with Gradient Projected Hard Attention*", IEEE Access, 12, 171143-171165 (2024). DOI: 10.1109/ACCESS.2024.3435555 given equal importance.

Unsupervised contrastive learning is effective in maintaining representation continuity but faces challenges with imbalanced positive-to-negative ratios, leading to biased instance discrimination. As discussed in Chapter 4, to address this issue in Framework-1, we proposed an approach that combines direct instance grouping with cross-level discrimination for both real and pseudo groups, enhancing unsupervised representation learning. In this Framework-II, we additionally utilize the capabilities of Evidential Deep Learning (EDL) to improve pseudo-OOD detection by identifying different sources of uncertainty. Cross-entropy is commonly used for image classification but is known to be biased towards the training data, often assigning high probabilities even to incorrect predictions [45]. Instead of adopting a traditional loss function, we employ a novel approach that combines cross-entropy with evidential KL-divergence for rotation-augmented pseudo labels. This strategy enables the quantification of uncertainty, a vital aspect for ensuring correct predictions [45]. Including evidential cross-entropy loss for OOD detection in the final loss function improves both TIL and CIL accuracy. Intuitively, a model capable of distinguishing between OOD and ID data learns more descriptive features, leading to improved TIL accuracy. For CIL, it is evident that any task inference mechanism necessitates discriminative features between ID and OOD. A detailed discussion of the various steps of the proposed technique and the experimental analysis is provided below.

5.1 Proposed Technique

This section presents the novel UCL Framework-II. Since contrastive learning's effectiveness in maintaining representation continuity was discussed in Chapter 4, this chapter focuses on addressing two key aspects. Firstly, it aims to improve pseudo-OOD detection, and secondly, to address the capacity saturation problem of hard attention that leads to CF.

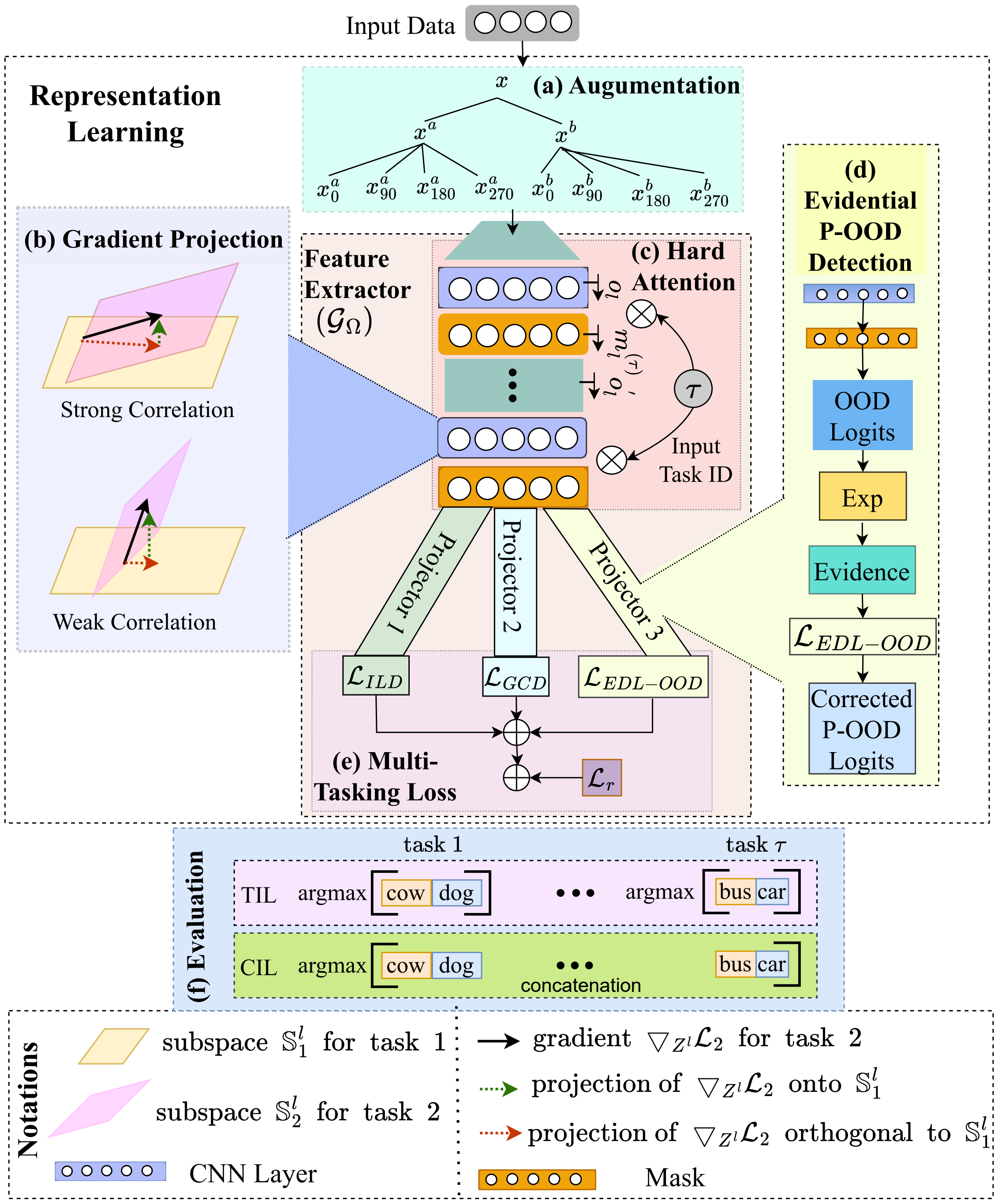

Input Data
Representation Learning
(a) Augmentation
x
x^a
x^b
x_0^a x_{90}^a x_{180}^a x_{270}^a x_0^b x_{90}^b x_{180}^b x_{270}^b
(b) Gradient Projection
Strong Correlation
Weak Correlation
Feature Extractor ($\mathcal{G}_\Omega$)
(c) Hard Attention
o_1
$m_1^{(\tau)}$
o_1'
τ
Input Task ID
(d) Evidential P-OOD Detection
OOD Logits
Exp
Evidence
$\mathcal{L}_{EDL-OOD}$
Corrected P-OOD Logits
Projector 1
Projector 2
Projector 3
$\mathcal{L}_{ILD}$
$\mathcal{L}_{GCD}$
$\mathcal{L}_{EDL-OOD}$
$\mathcal{L}_r$
(e) Multi-Tasking Loss
(f) Evaluation
task 1
task τ
TIL argmax cow dog ••• argmax bus car
CIL argmax cow dog ••• bus car
concatenation
Notations
subspace $\mathbb{S}_1^l$ for task 1
subspace $\mathbb{S}_2^l$ for task 2
CNN Layer
gradient $\nabla_{Z^l}\mathcal{L}_2$ for task 2
projection of $\nabla_{Z^l}\mathcal{L}_2$ onto $\mathbb{S}_1^l$
projection of $\nabla_{Z^l}\mathcal{L}_2$ orthogonal to $\mathbb{S}_1^l$
Mask

Figure 5.1: Overview of the Proposed Framework-II. (a) Various augmentation strategies (explained in §2.3). (b) Gradient Projection for strong and weak correlation (explained in §5.1.3). (c) Hard Attention mechanism (explained in §5.1.2). (d) Evidential pseudo-OOD detection (explained in §5.1.1.3). (e) Multi-tasking loss function to train feature extractor $\mathcal{G}_\Omega$ (further demonstrated in Figure 5.2 and §5.1.1). (f). TIL and CIL evaluation process (discussed in §2.4).

We leverage EDL's capabilities to enhance pseudo-OOD detection by identifying various sources of uncertainty (§5.1.1.3). To handle capacity saturation problem, we integrate hard attention (§5.1.2 with gradient projection §5.1.3). Our approach maintains the benefits of both methods by reducing the constraint on gradient updates for each. The subsequent sections elaborate on each component of Framework-II.

5.1.1 Unsupervised contrastive learning with cross-level discrimination and evidential OOD detection

Building on the contrastive learning loss function introduced in the previous chapter 4, we utilize a feature extractor $\mathcal{G}_\Omega$ to learn discriminative representations by bringing augmented views of the same instance closer while pushing apart others. This is achieved using a noise contrastive estimation (NCE) as defined in (4.1). Given a set of images, we apply transformations such as cropping or rotation to the i^{th} image, resulting in augmented versions x_i^a and x_i^b, which are then used to compute the corresponding loss functions (As shown in Figure 5.1.(e) and Figure 5.2).

5.1.1.1 Learning invariant mapping via discrimination at the instance level

As discussed in 4.1.1.1, training the model to understand similar representations of augmented images enables it to focus solely on capturing the inherent characteristics of the image without being affected by any transformation. Invariant mapping, in this fashion, is the core strength of contrastive learning methods. For learning feature representations, contrastive learning method used in Framework-I often necessitates large batches, pairs of negative samples [28], architectural modifications [30], or non-differentiable operators [96], making them challenging for CL scenarios. In this regard, in Framework-II, our focus centers on SimSiam [53], which overcomes these limitations and achieves SOTA performance

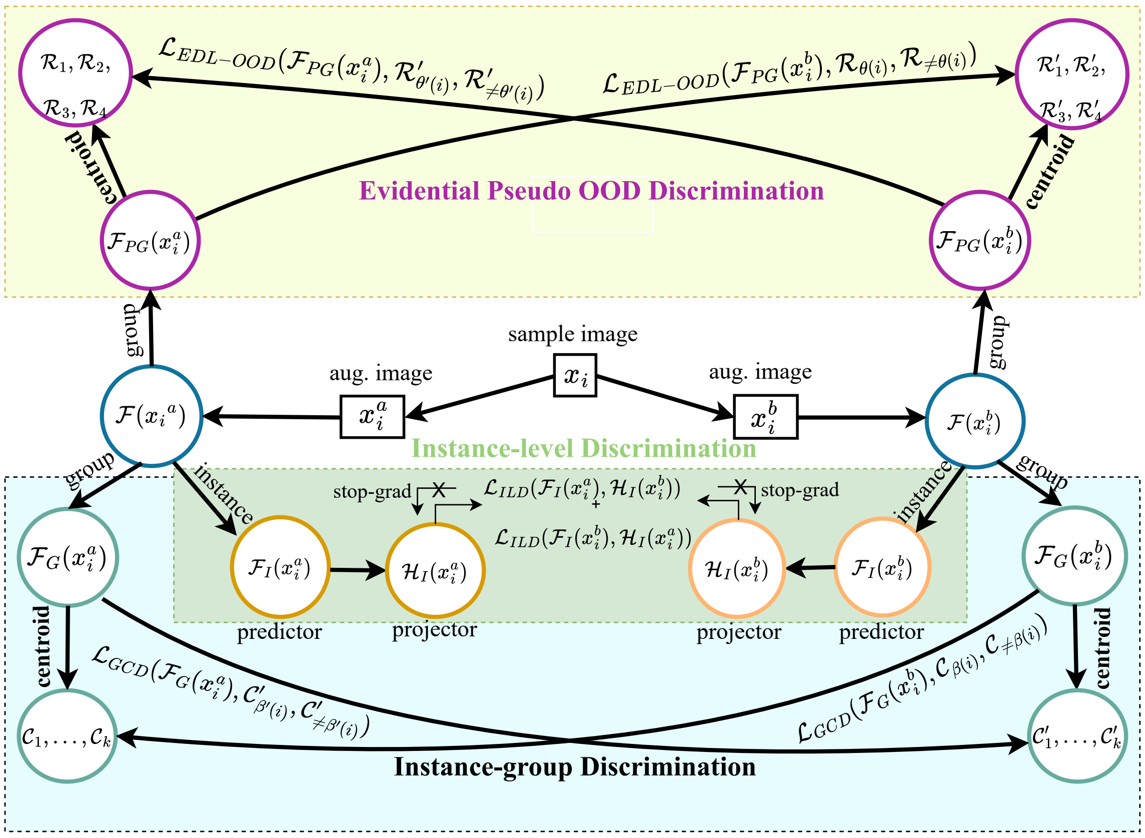

$\mathcal{L}_{EDL-OOD}(\mathcal{F}_{PG}(x_i^a), \mathcal{R}'_{\theta'(i)}, \mathcal{R}'_{\neq\theta'(i)})$
$\mathcal{L}_{EDL-OOD}(\mathcal{F}_{PG}(x_i^b), \mathcal{R}_{\theta(i)}, \mathcal{R}_{\neq\theta(i)})$
$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$
$\mathcal{R}'_1, \mathcal{R}'_2, \mathcal{R}'_3, \mathcal{R}'_4$
centroid
Evidential Pseudo OOD Discrimination
$\mathcal{F}_{PG}(x_i^a)$
$\mathcal{F}_{PG}(x_i^b)$
group
sample image
x_i
aug. image
x_i^a
x_i^b
$\mathcal{F}(x_i{}^a)$
$\mathcal{F}(x_i^b)$
Instance-level Discrimination
instance
stop-grad
$\mathcal{L}_{ILD}(\mathcal{F}_I(x_i^a), \mathcal{H}_I(x_i^b))$
$\mathcal{L}_{ILD}(\mathcal{F}_I(x_i^b), \mathcal{H}_I(x_i^a))$
$\mathcal{F}_G(x_i^a)$
$\mathcal{F}_I(x_i^a)$
$\mathcal{H}_I(x_i^a)$
$\mathcal{H}_I(x_i^b)$
$\mathcal{F}_I(x_i^b)$
$\mathcal{F}_G(x_i^b)$
predictor
projector
$\mathcal{L}_{GCD}(\mathcal{F}_G(x_i^a), \mathcal{C}'_{\beta'(i)}, \mathcal{C}'_{\neq\beta'(i)})$
$\mathcal{L}_{GCD}(\mathcal{F}_G(x_i^b), \mathcal{C}_{\beta(i)}, \mathcal{C}_{\neq\beta(i)})$
$\mathcal{C}_1, \ldots, \mathcal{C}_k$
$\mathcal{C}'_1, \ldots, \mathcal{C}'_k$
Instance-group Discrimination

Figure 5.2: Overview of the Multi-tasking Loss Function in Framework-II. Through the unsupervised contrastive loss function, our objective is to learn representation, denoted as $\mathcal{F}(x)$, for an input image denoted as x_i and its augmentation views referred to as x_i^a and x_i^b. We establish three branches stemming from $\mathcal{F}$: instance-level branch $\mathcal{F}_I$, instance-group branch $\mathcal{F}_G$ and evidential pseudo-OOD branch $\mathcal{F}_{PG}$. All the computation is symmetrical and mirrored across different views of the same instance.

on standard unsupervised representation learning benchmarks by using a variant of Siamese networks [140].

Encoder network $\mathcal{G}_\Omega$ is shared across instance-level prediction head $\mathcal{F}_I$ and projection head $\mathcal{H}_I$ as shown in green box in Figure 5.2 and projector 1 in Figure 5.1. Specifically, SimSiam minimizes the cosine-similarity between $\mathcal{F}_I(x_i^a)$ and $\mathcal{H}_I(x_i^b)$ as follows:

$$\begin{aligned}\mathcal{L}_{ILD} = &\frac{1}{2}\mathcal{D}(\mathcal{F}_I(x_i^a), \text{stopgrad}(\mathcal{H}_I(x_i^b)) \\ &+\frac{1}{2}\mathcal{D}(\mathcal{F}_I(x_i^b), \text{stopgrad}(\mathcal{H}_I(x_i^a))\end{aligned} \tag{5.1}$$

$$\text{where,} \qquad \mathcal{D}(\mathcal{F}_I(x_i^a), \mathcal{H}_I(x_i^b)) = -\frac{\mathcal{F}_I(x_i^a)}{\|\mathcal{F}_I(x_i^a)\|_2} \cdot \frac{\mathcal{H}_I(x_i^b)}{\|\mathcal{H}_I(x_i^b)\|_2}$$

here, $\|\cdot\|_2$ represents the l_2-norm. Notably, the stopgrad function serves a vital purpose in SimSiam by preventing the Siamese networks from converging to trivial solutions. Equation (5.1) builds on (4.2) and addresses its limitations.

5.1.1.2 Learning grouping via cross-level group discrimination

As detailed in the previous chapter, relying solely on instance-level discrimination assumes uniqueness across instances, which can reduce stability [33]. To address this, we adopted a group-level contrastive approach in Framework-I that balances instance separation with semantic grouping through shared repulsion.

Similar to Framework-I in Framework-II also we apply K-means clustering on projected features to define group centroids and compute the cross-level group contrastive loss $\mathcal{L}_{GCD}$, as shown in Figure 5.2 and formulated in Equation (5.2). This encourages each instance to move closer to its corresponding group centroid and away from others, using the group-level projection head $\mathcal{F}_G$. In alignment with the formulation presented in Equation (4.3) under

Framework-I, the Cross-Level Group Discrimination loss in Framework-II is defined as:

$$\mathcal{L}_{GCD} = \mathcal{L}(\mathcal{F}_G(x_i^a), C'_{\beta(i)}, C'_{\neq\beta(i)}) \tag{5.2}$$

$\mathcal{F}_G$ is the group-level projection head, which is shown as projector 2 in Figure 5.1. $C'_{\beta(i)}$ is the centroid of the cluster $\beta(i)'$ derived from the clustering of $\mathcal{F}_G(x_i^b)$, this acts as the positive sample to be attracted. The rest of the cluster centroids $C'_{\neq\beta(i)}$ act as negative samples to be repelled.

5.1.1.3 Learning OOD detection through cross-level evidential pseudo group discrimination

As discussed in chapter 4, Framework-I highlights the importance of detecting OOD and ID samples in CL. The instance-level discrimination loss ensures distinct representations within each distribution-level cluster, while repulsion between clusters ensures that pseudo-OOD samples occupy different locations in the representation space. An additional projection head $\mathcal{F}_{PG}$ is added to calculate the distribution-level discrimination loss.

Cross-entropy is commonly used for image classification but is known to be biased towards the training data, often assigning high probabilities even to incorrect predictions [45]. Instead of adopting a loss function similar to the one mentioned in (5.2), we employ a novel approach that combines cross-entropy with evidential KL-divergence for rotation-augmented pseudo labels. This strategy enables the quantification of uncertainty, a vital aspect for ensuring correct predictions [45]. The formulated loss function is as follows:

$$\mathcal{L}_{EDL-OOD} = \mathcal{L}(\mathcal{F}_{PG}(x_{\theta,i}^a), \mathcal{R}'_\theta, \mathcal{R}'_{\neq\theta}) \tag{5.3}$$

As shown in Figure 5.1. (d), and Figure 5.2 in the pseudo-OOD discrimination branch (yellow section in Figure), we apply loss between $\mathcal{F}_{PG}(x_{\theta,i}^a)$ and the distribution-level clusters

$\mathcal{R}'_\theta$ from the projection of the augmented view. In the context of the evidential framework, it is postulated that the class probability associated with a sample i stems from a prior Dirichlet distribution [37], represented as:

$$Dir(p, \omega) = \frac{\Gamma(\mathcal{Q})}{\prod_{j=1}^{r} \Gamma(\omega_j)} \prod_{j=1}^{r} p_i^{\omega_j - 1}; \omega_j > 0 \tag{5.4}$$

here, p represents a probability mass function, r denotes the number of rotations applied, $\omega = \{\omega_1, \ldots, \omega_j\}$ refers to the Dirichlet parameters, $\Gamma(.)$ stands for the gamma function, and $\mathcal{Q} = \sum_{j=1}^{r} \omega_j$ is termed as the Dirichlet strength. Each ω_j is computed as the sum of the evidence of the j-th class (e_j) with one. The evidence obtained from model learning is derived from the final layer after the application of a non-negative function, such as ReLU [37] or Exponential [141]. The training process for an EDL model for classification closely resembles that of a conventional neural network, with the primary distinction lying in the activation function used in the output layer. Instead of softmax, a non-negative activation is employed. The model is trained by minimizing the loss function known as evidential cross-entropy loss $(\mathcal{L}_{ECE})$ to generate multinomial opinions, which represent the evidence for classifying a given rotation augmentation of sample i into r rotation clusters, following a Dirichlet distribution. The formal definition of $\mathcal{L}_{ECE}$ is provided as follows [37, 141].

$$\mathcal{L}_{ECE} = \sum_{k=1}^{r} y_{i,k} (log(\mathcal{Q}_i) - log(\omega_{i,k})) \tag{5.5}$$

here, y_i represents the one-hot vector encoding of the pseudo-class (rotation) k. It is noteworthy that if the neural network discovers shared patterns among different rotation clusters, it might produce evidence for incorrect pseudo-labels, thereby minimizing the loss function $\mathcal{L}_{ECE}$. To prevent this phenomenon, a regularization loss function, $\mathcal{L}_{EKL}$, to reduce the total evidence to zero for a sample if it cannot be classified correctly. The $\mathcal{L}_{EKL}$ is defined as follows:

$$\begin{aligned}\mathcal{L}_{EKL} &= KL[Dir(p_i \mid \tilde{\omega_i}) \| Dir(p_i \mid 1)] \\ &= \log\left(\frac{\Gamma(\sum_{c=1}^{r} \tilde{\omega_{i,c}})}{\Gamma r) \prod_{c=1}^{r} \prod(\tilde{\omega_{i,c}})}\right) \\ &+ \sum_{c=1}^{r} (\tilde{\omega_{i,c}} - 1) \left[F(\tilde{\omega_{i,c}}) - F(\sum_{j=1}^{r} \tilde{\omega_{i,j}}) \right]\end{aligned} \tag{5.6}$$

here KL[.∥.] represents the Kullback-Leibler divergence, $\tilde{\omega_i}$ denotes the Dirichlet parameters after removing the non-misleading evidence, and $F(.)$ represents the digamma function [37]. The softmax activation can be replaced by an Exponential (Exp) activation for improved stability compared to the ReLU activation. Additionally, minimizing the loss function $\mathcal{L}_{ECE}$, while incorporating evidence from the Exp activation mirrors the standard softmax cross-entropy minimization commonly employed in traditional neural networks [37]. This approach enhances the capability of the model to provide additional information regarding prediction confidence. The loss function $\mathcal{L}_{EKL}$ is introduced to ensure that the model retains its capacity to capture uncertainty. Formally, the loss function introduced in this study is defined as follows:

$$\mathcal{L}_{EDL-OOD} = \lambda_1 \mathcal{L}_{ECE} + \lambda_2 \mathcal{L}_{EKL} \tag{5.7}$$

where, the importance of each loss function is weighted by λ_1 and λ_2. In a CL setting, $\mathcal{L}_{EKL}$ should prioritize newly introduced rotation-labels. Hence, the redefined $\mathcal{L}_{EKL}$ for the τ^{th} task is expressed as follows:

$$\mathcal{L}_{EKL} = KL[Dir(p_{i_\tau} \mid \tilde{\omega_{i_\tau}}) | Dir(p_{i_\tau} \mid 1)] \tag{5.8}$$

here, ω_{i_τ} represent Dirichlet parameters associated with the new classes C_{new}, and p_{i_τ} denotes the probability mass function for sample i of task τ.

Implementation details: Based on a similar argument, we can interpret this loss as a combination of evidential cross-entropy loss and evidential KL-divergence loss between the hard assignment of the distribution-level cluster $\mathcal{R}_\theta$ and the soft assignment $\mathcal{R}'_\theta$ predicted from $\mathcal{F}_{PG}$.

Advantage: Including $\mathcal{L}_{EDL-OOD}$ in the final loss function $\mathcal{L}$ improves both TIL and CIL accuracy. Intuitively, a model capable of distinguishing between OOD and ID data learns more descriptive features, leading to improved TIL accuracy. For CIL, it is evident that any task inference mechanism necessitates discriminative features between ID and OOD.

Up to this point, our discussion has revolved around discrimination at the representation level. Nonetheless, the model necessitates the ability for CL to learn each subsequent task while retaining the accumulated knowledge. As discussed in Chapter 4, an isolation-based methodology emerges as a viable strategy in this context, eliminating the need for a replay buffer and enabling model updates at the parameter level. The following section further substantiates the proposed Framework-II for preserving accumulated knowledge.

5.1.2 Preserving feature representation through hard attention mechanism

The hard attention mechanism employed in Framework-II follows the same design as in Framework-I (see §4.1.2); however, in Framework-II, we additionally address the issue of capacity saturation. When used individually, hard attention can impose excessive constraints on parameter updates, leading to suboptimal learning for new tasks. This intensifies the capacity problem by limiting the availability of free neurons and potentially increasing forgetting.

To mitigate this, we go beyond relying solely on the importance of individual parameters and additionally constrain the direction of gradients. This approach ensures that the network optimizes its parameters in directions orthogonal to those associated with previous tasks, preserving past knowledge while learning new information.

Through the empirical study, we found that integrating hard attention with gradient projection can complement the advantages and drawbacks of each other when given equal importance. Accordingly, we set the constraint parameter $\gamma_R = \frac{1}{2}$ in Equation (4.10), allowing each task to utilize network capacity efficiently while minimizing overall resource usage.

5.1.3 Protecting feature representation via gradient projection

To ensure effective learning of the new task, our approach imposes explicit constraints on accessible gradient directions using Gradient Projection, as shown in Figure 5.1.(b). This technique ensures the gradients for updating parameters related to new tasks are orthogonal to the important gradient subspaces from previous tasks. Unlike existing methods, this approach does not retain previous gradient directions [6] or old examples [11] to generate reference directions. After each task, the complete gradient space of weights is divided into two orthogonal subspaces: the Core Gradient Space (CGS) and the Residual Gradient Space (RGS) [89]. The bases of these gradient subspaces are determined through a Singular Value Decomposition (SVD) analysis of learned representations in a single-shot process, extracting the minimal set of bases for the CGS to preserve previous knowledge and facilitate learning for new tasks. Subsequently, these bases are stored in memory. In the following section, we explain the connection between the gradient and input spaces for identifying the core gradient spaces of previous tasks.

Input and gradient spaces: Proposed Framework-II exploits the observation that updates generated by stochastic gradient descent (SGD) reside within the range of input data points [142]. In the subsequent sections, we will demonstrate this relationship specifically for convolutional layers, which holds true across all layers within the network for any given task.

Consider a convolutional layer with the input tensor $\mathcal{X}$ and filters $\mathcal{Z}$. The convolution operation between $\mathcal{X}$ and $\mathcal{Z}$ which produces an output feature map $\mathcal{O}$, where $\mathcal{X}$ is in $\mathbb{R}^{C_i \times h_i \times w_i}$, $\mathcal{Z}$ is in $\mathbb{R}^{C_o \times (C_i \times k \times k)}$, and $\mathcal{O}$ is in $\mathbb{R}^{C_o \times h_o \times w_o}$ [143]. Here, C_i and C_o represent the number of input and output channels of the convolutional layer, respectively. The dimensions h_i, w_i and h_o, w_o denote the height and width of the input and output feature maps, respectively, while k represents the kernel size of the filters. If $\mathcal{X}$ is reshaped into a matrix R of size $(h_o \times w_o) \times (C_i \times k \times k)$ and $\mathcal{Z}$ is reshaped into a matrix Z of size $(C_i \times k \times k) \times C_o$, the convolution operation can be represented as the matrix multiplication of R and Z, resulting in $O = RZ$, where $O \in \mathbb{R}^{(h_o \times w_o) \times C_o}$. Each row of R represents an input patch vector, denoted as $p_j \in \mathbb{R}^{(C_i \times k \times k) \times 1}$, where $j = 1, 2, \ldots, n$; $(n = h_o * w_o)$. Formulation of convolution in terms of matrix multiplication provides an intuitive picture of the gradient computation during backpropagation. In the backward pass of the convolution layer, an error matrix Δ of the same size as O, specifically $(h_0 \times w_0) \times C_o$, is derived from the subsequent layer. The gradient of the loss with respect to the filter weights is then determined as:

$$\bigtriangledown_Z \mathcal{L} = R^{'} \Delta \tag{5.9}$$

where, $(.)^{'}$ is the matrix transpose and $\bigtriangledown_Z \mathcal{L}$ is of shape $(C_i \times k \times k)C_o$ which is of same size as Z. Since, columns of $R^{'}$ are the input patch vectors (p), the gradient updates of the convolutional filters will lie in the space spanned by these patch vectors.

Now, we demonstrate how gradient descent orthogonal to these spaces enables CL without forgetting.

Training task 1: To begin learning for the initial task $(\tau = 1)$ with dataset $\mathbb{D}_1$, parameter updates are allowed without constraints. Upon completing Task 1, a set of learned parameters, denoted as $\mathbb{Z}_1$, is obtained. To retain the knowledge gained from this task, constraints are imposed on the direction of gradient updates for subsequent tasks, in addition to the constraints applied by hard attention, as detailed in §5.1.2. This process involves partitioning the complete gradient space into two orthogonal subspaces: the Core Gradient Space (CGS) and the Residual Gradient Space (RGS). While gradient steps along the CGS significantly affect the learned tasks, those along the RGS result in minimal or no interference. The bases of the CGS are identified and stored, enabling orthogonal gradient steps for subsequent tasks, with each layer possessing its own CGS. To determine these bases, we construct a representation matrix $\mathfrak{R}_1^l = [((R_{1,1}^l)', (R_{2,1}^l)', \ldots, (R_{\tau_s,1}^l)']$ for each layer after learning Task 1. This matrix concatenates representations received from the forward pass of τ_s randomly selected samples from the current training dataset as they propagate through the network. Subsequently, we perform Singular Value Decomposition (SVD) on $\mathfrak{R}_1^l = V_1^l \sum_1^l (U_1^l)'$, followed by its k-rank approximation $(\mathfrak{R}_1^l)_k$ using a constant threshold of γ_g. This threshold criterion of γ_g is chosen to update the stored gradient projections exclusively for layers where the criterion exceeds it.

Orthogonal projection presents a potential solution for addressing forgetting in CL. However, exclusively adjusting the model in the orthogonal direction to the input space of old tasks may heavily constrain the optimization space for learning the new task, potentially resulting in diminished performance. As discussed in §5.1.2, in this framework, we integrate hard attention and gradient projection to mitigate the disadvantages of each while leveraging their respective advantages to enhance the learning capability of the model across all subsequent tasks. The effective capacity utilization is achieved by setting the threshold of γ_g in the following equation:

$$\|(\mathfrak{R}_1^l)_k\|_F^2 \geq \gamma_g \|\mathfrak{R}_1^l\|_F^2 \tag{5.10}$$

Through the empirical experiments, we discovered that integrating hard attention with gradient projection effectively balances their respective advantages and limitations when equal importance is assigned to both. Hence, during experimentation, we set the constraint parameter γ_g to $\frac{1}{2}$ in (5.10). The space $\mathbb{S}^l$ is define as the span of the first k vectors in V_1^l, denoted as $\mathbb{S}^l = \mathrm{span}\{v_{1,1}^l, v_{2,1}^l, \ldots, v_{k,1}^l\}$. $\mathbb{S}^l$ encompasses the significant representations for Task 1 at layer l, as it includes all directions associated with the highest singular values in the representation. In subsequent tasks, our objective is to perform gradient steps in a manner that preserves the correlation between representation specific to the task and weights within individual layers, as shown in Figure 5.1.(b).

Considering that the inputs encompass the gradient descent space, the bases of $\mathbb{S}^l$ will span a subspace within the gradient space, denoted as the CGS. Gradient descent along the CGS induces the maximum change in the correlation between inputs and weights, whereas gradient steps orthogonal to the CGS, (i.e., the space of low representational significance) cause minimal to no interference with the previous tasks. The subspace orthogonal to the CGS is designated as the RGS. The bases of the CGS are stored in memory denoted as $\mathcal{M} = \{(M^l)_L^{l=1}\}$, where $M_l = [v_{1,1}^l, v_{2,1}^l, \ldots, v_{k,1}^l]$.

Training for tasks 2 to T: Task 2 is learned exclusively using samples from dataset $\mathbb{D}_2$. Before performing a gradient step, the bases of the CGS are obtained from the stored gradient projections. The new gradients ($\bigtriangledown_{Z_2^l}\mathcal{L}_2$) are initially projected onto the CGS, and then the projected components are subtracted from them, ensuring that the remaining gradient components are in the orthogonal space to the CGS. The gradients are then updated as follows:

$$\bigtriangledown_{Z_2^l}\mathcal{L}_2 = \bigtriangledown_{Z_2^l}\mathcal{L}_2 - M^l(M^l)'(\bigtriangledown_{Z_2^l}\mathcal{L}_2) \tag{5.11}$$

Upon the completion of Task 2 training, the store gradient projections are updated with new task-specific bases customized for the CGS. To derive these bases, we create $\mathfrak{R}_2^l =$

$[R^l_{1,2}, R^l_{2,2}, \ldots, R^l_{\tau_s,2}]$, utilizing data solely from task 2. Prior to performing SVD and the subsequent k-rank approximation on $\mathfrak{R}^l_2$, we remove any redundant bases already existing in the stored gradient projections. This ensures that the newly incorporated bases are unique and orthogonal to those already present in memory. To achieve this, we follow these steps:

$$\hat{\mathfrak{R}}^l_2 = \mathfrak{R}^l_2 - M^l(M^l)'(\mathfrak{R}^l_2) = \mathfrak{R}^l_2 - \mathfrak{R}^l_{2,Proj} \tag{5.12}$$

Subsequently, analogous to the approach employed during the learning of task 1, we perform SVD on matrix $\hat{\mathfrak{R}}^l_2(= \hat{V}^l_2 \hat{\sum}^l_2 (\hat{U}^l_2)')$. We then choose k new orthogonal bases, ensuring that the minimum value of k fulfills the prescribed criteria for the threshold γ_g. This threshold is set to $\frac{1}{2}$ during experimentation to give equal importance to both gradient projection and hard attention.

$$||(\mathfrak{R}^l_{2,Proj})||^2_F + ||(\hat{\mathfrak{R}}^l_2)_k||^2_F \geq \gamma_g ||\mathfrak{R}^l_2||^2_F \tag{5.13}$$

The stored gradient projections are updated by appending new bases as $M_l = [M_l, \hat{v}^l_{1,2}, \ldots, \hat{v}^l_{k,2}]$. Consequently, with each new task training, the CGS expands while the RGS shrinks. The maximum size of M^l (and therefore the dimension of the gradient bases) is determined by the initial network architecture. Upon completing the update of stored gradient projections, we proceed on to the next task and repeat the same process as for Task 2.

5.1.4 Overall objective function

The final objective to be minimized for task n is as follows (see in Figure 5.1.(e))

$$\mathcal{L} = \mathcal{L}_{ILD} + \gamma_{GCD}\, \mathcal{L}_{GCD} + \gamma_{OOD}\, \mathcal{L}_{EDL-OOD} + \gamma_R\, \mathcal{L}_r \tag{5.14}$$

here, $\mathcal{L}$ is the overall loss function, or can be referred to as a multi-tasking loss function.

$\mathcal{L}_{ILD}$, $\mathcal{L}_{GCD}$, $\mathcal{L}_{EDL-OOD}$ denote the loss function for instance-level discrimination, group-level discrimination, and pseudo-group level discrimination for evidential OOD detection, respectively. $\mathcal{L}_r$ is the regularization loss. γ_{GCD}, γ_{OOD}, and $\gamma_{R(\tau)}$ denote the positive scaling parameter for the respective loss function. In practice, γ_R is the same for all tasks, where $\tau \neq 1$.

5.2 Experimental Analysis

In this section, we explain the (1) baselines, (2) training and evaluation setup, (3) hyper-parameter setting, and (4) results and comparative analysis.

5.2.1 Baselines

For the naming convention the proposed Framework-II, denoted as GP-M-CLD-S-EOOD, incorporates gradient projection, hard attention, cross-level discrimination, Sim-Siam loss, and evidential OOD detection capabilities. We compare Framework-II with both classic and most recent SOTA TIL and CIL methods. Similar to Framework-I, for the baselines of Framework-II as well, we consider four unsupervised methods, i.e., SI [16], DER [14], PNN [9], LUMP [11], along with one supervised method, Co2L [12]. Additionally, we included Framework-I [144] as the fifth UCL baseline. We also include several preceding methods that we explored before finalizing the proposed solution, such as N_M-CLD-N_OOD, N_M-CLD-OOD, M-CLD-N_OOD, M-CLD-OOD (Framework-I), M-CLD-S-OOD, and GP-M-CLD-S-OOD. An explanation of each model name is provided in the list of abbreviations in Table 1.

5.2.2 Training and evaluation setup

The overall training and evaluation setup largely follows the configuration detailed in §4.2.2. The primary differences lie in the proposed model architecture and variants used in Framework-II. Specifically, we incorporate both hard attention and gradient projection mechanisms into the ResNet-18 backbone. For the proposed Framework-II (GP-M-CLD-S-EOOD) and its variants (N_M-CLD-N_OOD, N_M-CLD-OOD, M-CLD-N_OOD, M-CLD-OOD, M-CLD-S-OOD, GP-M-CLD-S-OOD) we employed the LARS optimization function [130] with the same learning rate schedule as before in Framework-I. Baseline methods are trained using SGD with identical settings, and all evaluations are conducted using a KNN classifier [52]. Additional hyperparameter details are provided in the following section.

Table 5.1: Optimal hyperparameter values for Framework-II (GP-M-CLD-S-EOOD) and its variants across datasets determined via grid search

Dataset	ρ_{ILD}	ρ_{GCD}	ρ_{OOD}	γ_{GCD}	γ_{OOD}	γ_R	$\mathcal{C}_i$
M-5T	0.1	0.1	0.1	0.5	0.5	0.5	128
C-10	0.1	0.2	0.2	0.5	0.5	0.5	128
CH-5T	0.1	0.2	0.2	0.5	0.5	0.5	150
CH-10T	0.1	0.2	0.2	0.5	0.5	0.5	180
CH-20T	0.1	0.1	0.1	0.5	0.5	0.5	200
I1H-5T	0.1	0.2	0.2	0.5	0.5	0.5	160
I1H-10T	0.1	0.1	0.1	0.5	0.5	0.5	180
I1H-20T	0.1	0.1	0.1	0.5	0.5	0.5	200

5.2.3 Hyperparameters setting

The hyperparameter configuration mostly follows the setup discussed in §4.2.3. In Framework-II, we use similar sets of temperature parameters (ρ_{ILD}, ρ_{GCD} and ρ_{OOD}), scaling factors (γ_R, γ_{GCD} and γ_{OOD}), the number of clusters $\mathcal{C}_i$, and positive scaling parameter (s and $s_{\max}$), with minor tuning based on the gradient projection and evidential OOD detction introduced in this chapter. As explained in §5.1.2, we set the constraint parameter γ_R of (4.10) to $\frac{1}{2}$ after hyperparameter tuning, which allows each task to efficiently utilize network capacity while minimizing overall capacity usage. To maintain consistency in training all

Table 5.2: Overall-average TIL accuracy (in %) for Framework-II and baselines on benchmark datasets across all tasks. The last row displays the average TIL Accuracy of each method. Further information about the table can be found in §5.2.4.

Methods **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Framework-II**	**Co2L (Sup.)**
M-5T	99.46	99.28	99.23	99.74	99.68	**99.91**	99.85
C-5T	91.4	91.64	91.05	91.23	94.12	**94.54**	92
CH-5T	61.11	61.6	58.5	60.04	65.3	66.32	**69.46**
CH-10T	68.21	69.37	62.47	69.15	73.63	**75.57**	73.86
CH-20T	75.6	76.47	65.17	78.74	80.45	**82.39**	78.59
I1H-5T	54.62	53.44	48.09	52.9	56.38	**58.07**	56.4
I1H-10T	55.39	58.29	49.36	63.68	65.4	**68.35**	60.2
I1H-20T	67.16	64.32	57.29	71.18	75.16	**77.41**	69.61
Average	71.62	71.8	66.4	73.33	76.27	**77.82**	75

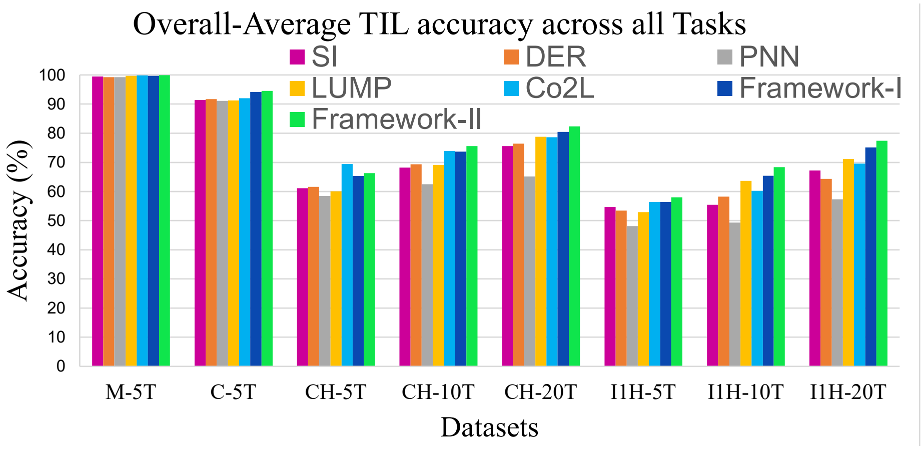


Figure 5.3: Overall-average TIL Accuracy for Framework-II and baselines on benchmark datasets.

UCL baselines, we adopt the identical set of hyperparameters utilized in SimSiam [53]. The configuration of hyperparameters for all datasets is provided in Table 5.1. Optimal hyperparameter values for all baseline models across all benchmark datasets are available in Tables 4.2 and 4.3.

5.2.4 Results and comparative analysis

Tables 5.2, 5.3, 5.4, 5.5, and Figures 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 show the evaluation results for Framework-II and baselines on M-5T, C-5T, CH-5T, CH-10T, CH-20T, I1H-5T, I1H-10T, and I1H-20T. In Tables 5.2, 5.3, 5.4, and 5.5, first row contains the names of the baseline methods. The last row of the tables shows overall-average performance of the respective models. We observe that all the mentioned baselines, which represent various CL strategies, yield lower performance compared to the proposed Framework-II with negligible or nearly zero forgetting.

5.2.4.1 Comparison of overall-average TIL accuracy across all tasks

The method of calculating overall-average accuracy across all tasks is explained in §2.4. In Table 5.2, Figure 5.3 and 5.4 we show the overall-average TIL accuracy of the models. In all cases, Framework-II achieves significantly better overall-average accuracies than representative baselines. It can be observed from the last column of Table 5.2, which contains the average accuracy of all models, that Framework-II achieves a 6.20%, 6.02%, 11.42%, 4.49%, and 1.55% improvement compared to SOTA UCL baselines such as SI, DER, PNN, LUMP, and Framework-I respectively, it can also be verified from Figure 5.4. These results suggest that the combination of gradient projection and hard attention effectively preserves prior knowledge, thus addressing **RQ4** [§3.5] in the TIL setup. Outperforming both DER and LUMP indicates that there is no need for a replay buffer anymore to improve the performance (thus, **RQ1** [§3.5] is answered in Framework-II as well). The improved accuracy compared to PNN also indicates the effectiveness of Framework-II over architecture-based strategies. Moreover, it achieves performance that is on par with SCL methods like Co2L and even surpasses it in terms of average performance across all datasets by 2.82%. As shown in Figure 5.4, Co2L initially demonstrates higher accuracy on certain initial tasks owing to its supervised nature, Framework-II showcases better accuracy on subsequent tasks due to its effective handling of CF compared to Co2L. Importantly, this is noteworthy as it highlights the gains made without the need for labeled data.

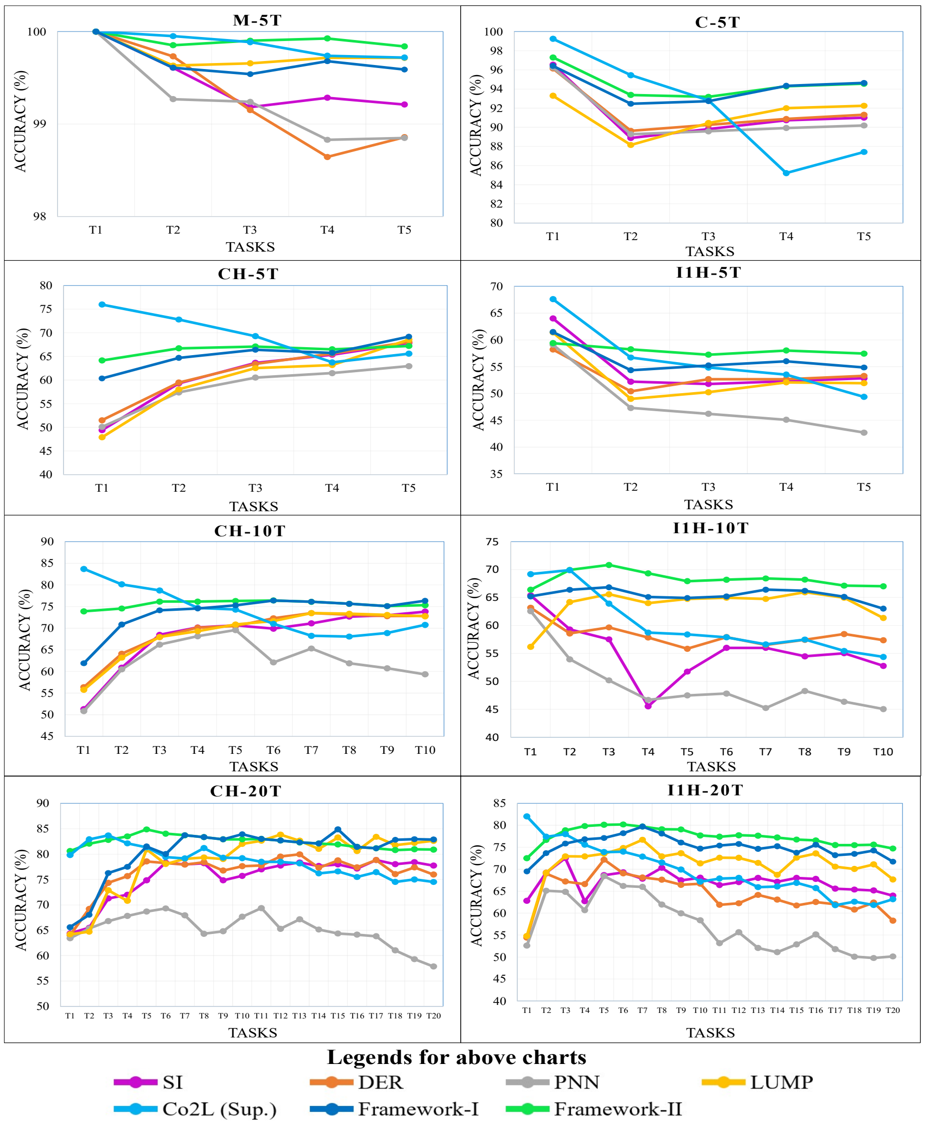


Figure 5.4: TIL Performance comparison for Framework-II and baselines on benchmark datasets across all tasks.

Table 5.3: Overall-average TIL Forgetting (in %) for Framework-II and baselines on benchmark datasets across all tasks.

Methods **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Framework-II**	**Co2L (Sup.)**
M-5T	0.45	0.72	0.0002	0.17	0.001	**0.0005**	0.07
C-5T	1.85	3.72	0.0001	2.27	0.002	**0.0004**	7.35
CH-5T	2.36	3.47	0.0004	3.18	0.001	**0.0006**	14.02
CH-10T	2.57	3.63	0.0008	2.76	0.001	**0.0003**	15.46
CH-20T	5.86	5.67	0.003	1.98	0.029	**0.002**	16.18
I1H-5T	7.71	5.33	0.0005	3.78	0.002	**0.0005**	14
I1H-10T	11.66	8.41	0.0008	4.47	0.001	**0.0001**	16.79
I1H-20T	7.89	8.51	0.009	5.47	0.06	**0.003**	17.38
Average	5.04	4.93	0.0019	3.01	0.0121	**0.000925**	12.66

5.2.4.2 Comparison of overall-average TIL forgetting across all tasks

The method of calculating overall-average forgetting across all tasks is explained in §2.4. In Table 5.3 and Figure 5.5, we present the overall-average TIL forgetting of the models. Framework-II exhibits **nearly zero** or significantly negligible CF compared to UCL and SCL baselines. For instance, the SCL baseline Co2L shows an overall-average TIL forgetting of 12.66%, while UCL baselines SI, DER, PNN, LUMP, and Framework-I demonstrate 5.04%, 4.93%, 0.0019%, 3.01%, and 0.0121% overall average TIL forgetting, respectively. This indicates that gradient projection with hard attention preserves prior knowledge, supporting **RQ4** [§3.5] in the TIL setup. This remarkable reduction in forgetting compared to DER and LUMP suggests that a replay buffer may no longer be necessary to address CF (consequently, again confirming the **RQ1** [§3.5] in TIL setup).

5.2.4.3 Comparison of overall-average CIL accuracy across all tasks

The process for computing overall-average CIL accuracy is detailed in §2.4. Table 5.4 and Figures 5.6, 5.7 display the overall-average CIL accuracy of the models. In the Average column of Table 5.4, it is evident that Framework-II achieves an increase of 8.79%, 9.39%, 11.54%, 6.59%, and 4.43% in overall average CIL accuracy across all datasets in the UCL setup compared to SI, DER, PNN, LUMP, and Framework-I respectively (thus

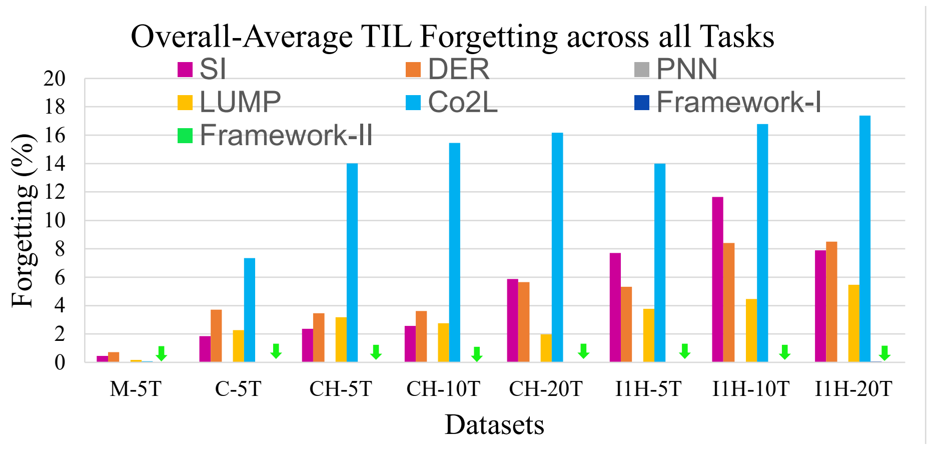


Figure 5.5: Overall-average TIL Forgetting across all tasks for Framework-II and baselines. Additionally, in Framework-II, almost zero CF is represented by a vertical green arrow.

Table 5.4: Overall-average CIL Accuracy (in %) for Framework-II and baselines on benchmark datasets across all tasks.

Methods **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Framework-II**	**Co2L (Sup.)**
M-5T	97.57	96.33	97.39	99.16	98.87	**99.44**	98.95
C-5T	83.46	83.67	83.54	83.38	87.03	**90.11**	82.53
CH-5T	53.7	53.82	50.06	52.56	57.14	61.29	**65.38**
CH-10T	53.83	53.81	49.2	55.95	56.26	**62.27**	61.7
CH-20T	52.69	52.86	46.28	57.84	57.59	**63.06**	57.28
I1H-5T	46.51	43.5	44.14	42.27	48.39	**53.52**	51.41
I1H-10T	40.42	42.48	39.64	49.28	46.8	**55.08**	46.81
I1H-20T	40.61	37.55	36.53	46.028	51.63	**54.32**	45.63
Average	58.6	58	55.84	60.8	62.96	**67.39**	63.71

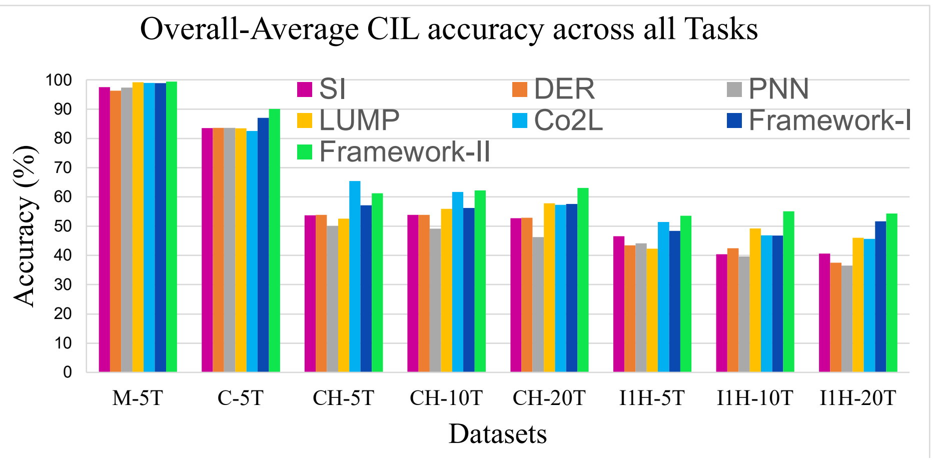


Figure 5.6: Overall-average CIL Accuracy for Framework-II and baselines across all tasks.

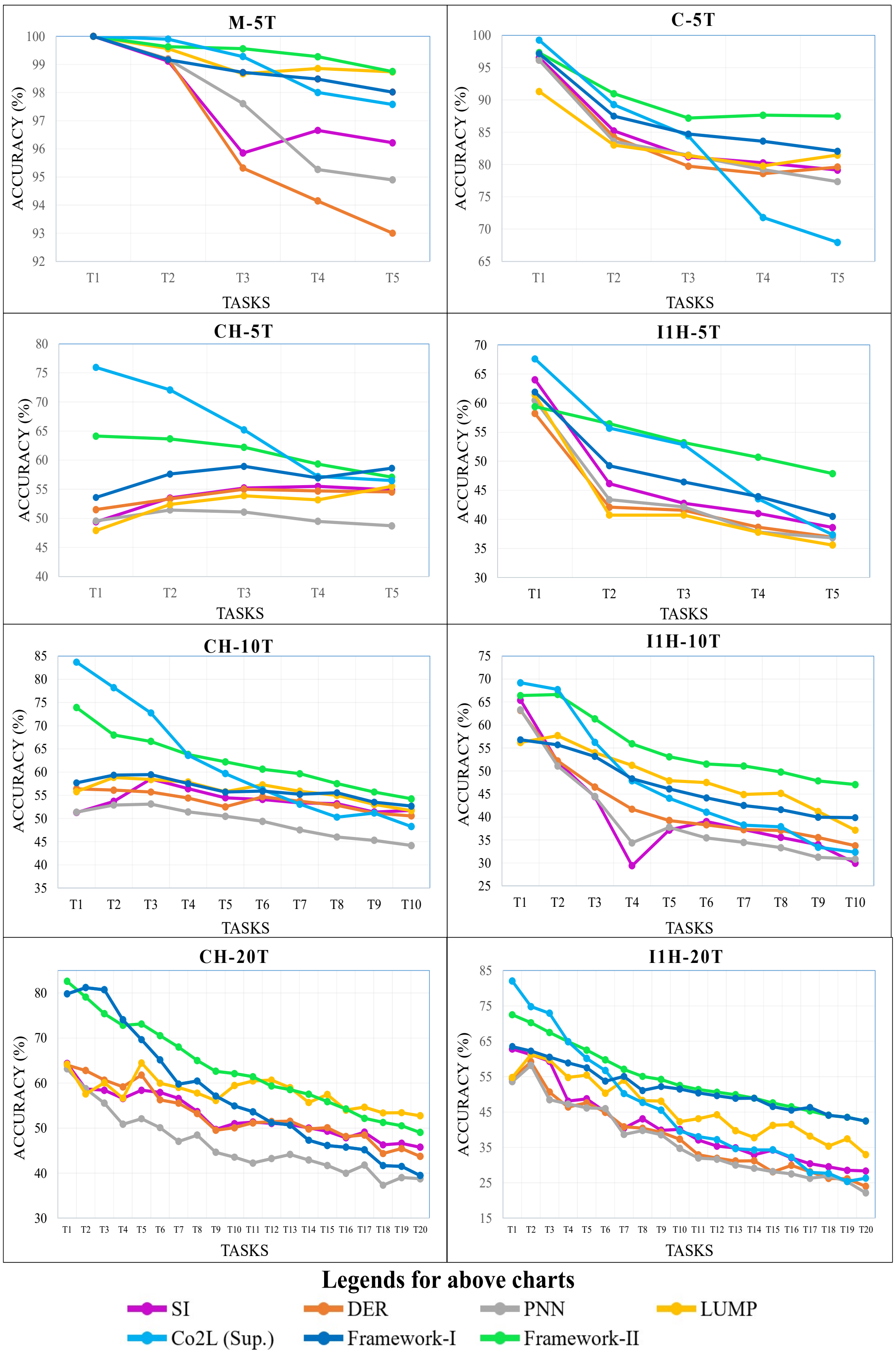


Figure 5.7: CIL Performance comparison for Framework-II and baselines on benchmark datasets across all tasks.

Table 5.5: Overall-average CIL Forgetting (in %) for Framework-II and baselines on benchmark datasets across all tasks.

Methods **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-I**	**Framework-II**	**Co2L (Sup.)**
M-5T	0.16	1.02	0.0008	0.04	0.0005	**0.0001**	0.24
C-5T	3.82	3.69	0.005	1.54	0.001	**0.0002**	12.78
CH-5T	1.57	2.82	0.009	2.67	0.001	**0.0001**	17.48
CH-10T	2.31	2.77	0.013	1.861	0.001	**0.0003**	24.77
CH-20T	4.71	1.76	0.063	2.5	0.01	**0.003**	27.53
I1H-5T	7.54	3.78	0.007	2.93	0.002	**0.0001**	18.03
I1H-10T	11.21	6.09	0.017	3.65	0.001	**0.0002**	25.59
I1H-20T	5.86	3.11	0.094	5.18	0.03825	**0.015**	30.09
Average	4.65	3.13	0.026	2.55	0.0068	**0.0024**	19.56

confirming the **RQ4** [§3.5] in CIL setup as well). Moreover, Framework-II outperforms the SCL method Co2L on all datasets, with a 3.68% increase in overall average CIL accuracy (see Figures 5.6 and 5.7).

5.2.4.4 Comparison of overall-average CIL forgetting across all tasks

The process for computing overall-average CIL forgetting is outlined in §2.4. Table 5.5 and Figure 5.8 present the overall-average CIL forgetting of the models. Both UCL and SCL baselines exhibit considerable CIL forgetting compared to Framework-II in the CIL setting as well. Framework-II however, demonstrates almost zero or significantly negligible CF. In contrast, SCL baselines like Co2L exhibit an overall-average CIL forgetting of 19.56%, while UCL baselines like SI, DER, PNN, LUMP, and Framework-I show 4.65%, 3.13%, 0.026%, 2.55%, and 0.0068% overall average CIL forgetting, respectively (thus, reconfirming the **RQ4** [§3.5] in CIL setup as well). This substantial reduction in forgetting compared to DER and LUMP suggests that a replay buffer may no longer be necessary to address CF (thus, reaffirming the **RQ1** [§3.5] in CIL setup for Framework-II as well).

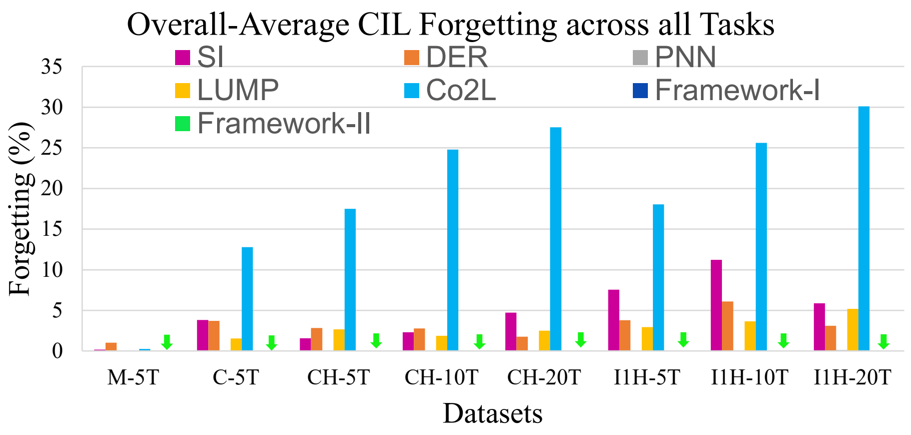


Figure 5.8: Overall-average CIL Forgetting across all tasks for Framework-II and baselines.

5.2.5 Ablation study

In this section, we study the effectiveness of hard attention, gradient projection and multi-tasking loss function along with evidential OOD detection capability. In Tables 5.6, 5.7, 5.8, and 5.9, the first column contains the names of the variants of Framework II. The last column of the tables shows overall-average performance of the respective models. These tables demonstrate that the successive variants of GP-M-CLD-S-EOOD (Framework-II), show gradual improvements in terms of overall-average accuracy, all while maintaining minimal forgetting.

5.2.5.1 Effect of multi-tasking loss function along with evidential OOD detection capability

It can be inferred from the Tables 5.7, and 5.9 that the model experiences significant forgetting when evidential OOD discrimination capability, hard attention, gradient projection, and SimSiam loss are not utilized, as seen in N_M-CLD-N_OOD with a notable overall-average TIL forgetting of 8.48% and CIL forgetting of 7.69% across all datasets. However, incorporating these techniques significantly mitigates forgetting, with an average TIL forgetting of **0.000925%** and CIL forgetting of **0.0024%** across all tasks in GP-M-CLD-

Table 5.6: Overall-average TIL Accuracy (in %) of successive variants of Framework-II.

Methods	M-5T	C-5T	CH-5T	CH-10T	CH-20T	I1H-5T	I1H-10T	I1H-20T	Avg.
N_M-CLD-N_OOD	98.48	87.06	57.56	64.96	71.97	52.28	57.38	69.36	69.88
N_M-CLD-OOD	98.83	87.57	58.2	65.83	72.86	53.04	59.79	71.33	70.93
M-CLD-N_OOD	99.66	91.01	59.13	69.61	77.81	53.17	61.29	74.11	73.22
M-CLD-OOD (F-I)	99.68	94.12	65.3	73.63	80.45	56.38	65.4	75.16	76.27
M-CLD-S-OOD	99.79	94.28	65.39	74.22	81.03	56.81	67.44	76.52	76.94
GP-M-CLD-S-OOD	99.89	94.39	65.73	74.93	81.64	57.39	67.95	76.97	77.36
GP-M-CLD-S-EOOD (F-II)	**99.91**	**94.54**	66.32	**75.57**	**82.39**	**58.07**	**68.35**	**77.41**	**77.82**

Table 5.7: Overall-average TIL Forgetting (in %) of successive variants of the Framework-II.

Methods	M-5T	C-5T	CH-5T	CH-10T	CH-20T	I1H-5T	I1H-10T	I1H-20T	Avg.
N_M-CLD-N_OOD	0.74	7.01	8.22	10.34	12.03	6.05	7.79	15.63	8.48
N_M-CLD-OOD	0.18	0.5	0.42	0.78	14.6	0.39	0.74	9.91	3.44
M-CLD-N_OOD	0.01	0.013	0.02	0.002	0.048	0.003	0.002	0.065	0.0203
M-CLD-OOD (F-I)	0.001	0.002	0.001	0.001	0.029	0.002	0.001	0.06	0.0121
M-CLD-S-OOD	0.0012	0.002	0.002	0.001	0.009	0.006	0.0005	0.058	0.00996
GP-M-CLD-S-OOD	0.0011	0.0006	0.0007	0.0009	0.002	0.004	0.0003	0.014	0.00295
GP-M-CLD-S-EOOD (F-II)	**0.0005**	**0.0004**	**0.0006**	**0.0003**	**0.002**	**0.0005**	**0.0001**	**0.003**	**0.000925**

S-EOOD (Framework-II). Furthermore, other variants such as N_M-CLD-OOD, M-CLD-N_OOD, M-CLD-OOD, M-CLD-S-OOD and GP-M-CLD-S-OOD also demonstrate a gradual decrease in overall-average TIL forgetting of 3.44%, 0.0203%, 0.0121%, 0.00996%, and 0.00295% and CIL forgetting of 4.19%, 0.0158%, 0.0068%, 0.0162%, and 0.0089%, respectively. This can be attributed to the inclusion of Cross-Level Pseudo-Instance Group Discrimination for OOD Detection along with EDL (thus, addressing the **RQ5** [§3.5] in both TIL and CIL setup). This effect is noted in both TIL and CIL settings, but it is more relevant to CIL setting. The Tables 5.6, 5.7, 5.8, 5.9 also indicates that the influence of SimSiam loss on the performance of the model. M-CLD-S-OOD (model with SimSiam loss) shows improvement in performance in compare to its previous variants (model without SimSiam loss, i.e. N_M-CLD-N_OOD, N_M-CLD-OOD, M-CLD-N_OOD, and M-CLD-OOD). This indicates the effectiveness of SimSiam loss, in Framework-II. As shown in Figure 5.4 and Figure 5.7, the multi-tasking loss function with the evidential OOD detection capability helps in UCL, when comparing with SOTA baselines.

Table 5.8: Overall-average CIL Accuracy (in %) of successive variants of Framework-II.

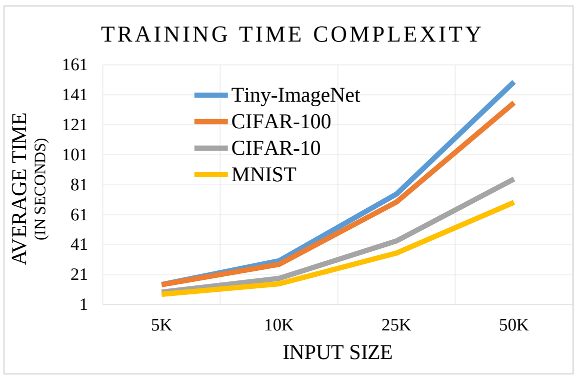
Methods	M-5T	C-5T	CH-5T	CH-10T	CH-20T	I1H-5T	I1H-10T	I1H-20T	Avg.
N_M-CLD-N_OOD	94.91	76.97	48.81	51.19	47.1	42.03	42.49	45.97	56.18
N_M-CLD-OOD	95.33	78.84	50.01	51.47	48.52	44.67	44.96	37.39	56.39
M-CLD-N_OOD	97.89	79.16	51.13	51.85	50.32	44.96	45.16	43.89	58.04
M-CLD-OOD (F-I)	98.87	87.03	57.14	56.26	57.59	48.39	46.8	51.63	62.96
M-CLD-S-OOD	99.07	88.15	58.63	60.85	61.46	51.79	51.06	52.89	65.49
GP-M-CLD-S-OOD	99.38	88.86	60.91	61.58	62.65	52.83	54.37	53.46	66.75
GP-M-CLD-S-EOOD (F-II)	**99.44**	**90.11**	61.29	**62.27**	**63.06**	**53.52**	**55.08**	**54.32**	**67.39**

5.2.5.2 Effect of gradient projection

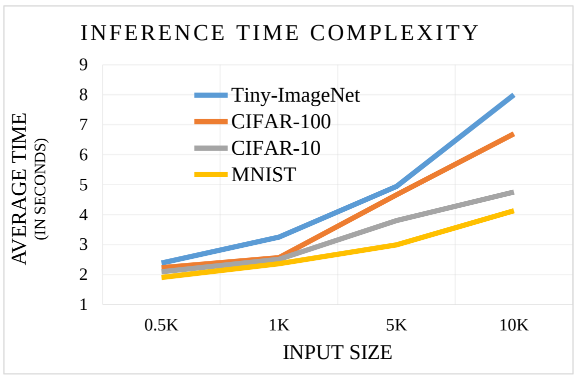
The effectiveness of hard attention has already been observed in Framework-I (see §4.2.5.2); however, it faces the challenge of capacity saturation. To address this, Framework-II incorporates gradient projection alongside hard attention. Through multiple variants of Framework-II, we have demonstrated that incremental enhancements in the framework have allowed us to achieve an increase in performance. The Tables 5.6, 5.7, 5.8, 5.9 and Figures 5.3 , 5.6, 5.5, and 5.8 indicates that the influence of gradient projection on the performance of the model. GP-M-CLD-S-OOD (model with gradient projection) shows gradual improvement in performance and decrease in CF in compare to its previous variants (model without gradient projection, i.e. N_M-CLD-N_OOD, N_M-CLD-OOD, M-CLD-N_OOD, M-CLD-OOD, and M-CLD-S-OOD). This indicates the effectiveness of gradient projection, in the proposed solution (thus, reaffirming the **RQ4** [§3.5]). Furthermore, Framework-II (GP-M-CLD-S-EOOD) demonstrates superiority over its variants N_M-CLD-N_OOD, N_M-CLD-OOD, M-CLD-N_OOD, M-CLD-OOD (Framework-I), M-CLD-S-OOD and GP-M-CLD-S-OOD with incremental gains of 7.94%, 6.89%, 4.6%, 1.55%, 0.88%and 0.46% in TIL setting and 11.21%, 11.0%, 9.35%, 4.43%, 1.9%, and 0.64% in CIL setting, respectively. These findings suggest that the incremental enhancements in the Framework-II, particularly the inclusion of gradient projection along with hard attention and evidential OOD detection capabilities have led to improvements in performance compared to SOTA baselines.

Table 5.9: Overall-average CIL Forgetting (in %) of successive variants of Framework-II.

Methods	M-5T	C-5T	CH-5T	CH-10T	CH-20T	I1H-5T	I1H-10T	I1H-20T	Avg.
N_M-CLD-N_OOD	0.51	6.9	8.05	11.5	13	6.37	8.22	6.99	7.69
N_M-CLD-OOD	0.14	0.38	0.25	0.4	16.51	0.25	0.25	15.39	4.19
M-CLD-N_OOD	0.001	0.003	0.03	0.001	0.082	0.002	0.001	0.0065	0.0158
M-CLD-OOD (F-I)	0.0005	0.001	0.001	0.001	0.01	0.002	0.001	0.03825	0.0068
M-CLD-S-OOD	0.0003	0.0005	0.0007	0.0004	0.032	0.0009	0.0006	0.094	0.0162
GP-M-CLD-S-OOD	0.0002	0.0003	0.0004	0.001	0.006	0.0005	0.0004	0.063	0.0089
GP-M-CLD-S-EOOD (F-II)	**0.0001**	**0.0002**	**0.0001**	**0.0003**	**0.003**	**0.0001**	**0.0002**	**0.015**	**0.0024**



(a)



(b)

Figure 5.9: Training and inference time complexity comparison of Framework-II across all benchmark datasets.

5.2.5.3 Training and inference time complexity across all datasets

Training and inference time complexity of Framework-II across all datasets is shown in Figure 5.9. Each dataset is randomly divided into four subsets corresponding to full, half, one-fifth, and one-tenth of the data size. Stratified sampling based on class labels is applied to ensure representative splits, allowing for comprehensive time complexity testing. As illustrated in Figure 5.9, the complexity of Framework-II is analogous to **linearithmic**. This is evident when transitioning from simpler datasets (e.g., MNIST) to more complex ones (e.g., Tiny-ImageNet), which requires an increase in input planes in the ResNet-18 architecture from 64 to 128, as discussed in §5.2.2. This adjustment is necessary to accommodate the higher complexity of datasets such as CIFAR-100 and Tiny-ImageNet. Both plots (a) and (b) exhibit approximately similar characteristics.

5.3 Discussion

As explained in §5.2.4, Framework-II showcases performance improvement by incorporating cross-level instance-group discrimination and evidential pseudo OOD detection along with gradient projected hard attention. Unlike previous UCL approaches that heavily rely on replay-based strategies to mitigate CF, our novel isolation-based approach leverages the combined benefits of hard attention and gradient projection. A limitation of hard attention is its tendency to saturate the capacity of the network, a challenge we mitigate through the implementation of gradient projection. Rather than solely depending on parameter-level masking, we utilize gradient projection to train the gradient in an orthogonal direction for previous tasks. This combined strategy efficiently mitigates CF without necessitating an external replay buffer, thereby resolving concerns regarding overfitting, representation drifts, and privacy associated with conventional replay-based methods.

We also explored the option of preserving past task samples exclusively for OOD detection, similar to the approach in MORE [145]. Despite being supervised, this approach results in overfitting and performance degradation. To improve OOD detection within the UCL framework, we use EDL, which does not rely on past training data for learning OOD detection. Moreover, EDL enhances pseudo-OOD detection by identifying various sources of uncertainty. Nevertheless, integrating EDL into the UCL framework presents challenges due to its dependence on labeled data. To overcome this obstacle, we utilize hard-coded pseudo-rotation augmented labels, as elaborated in §5.1.1.3. This approach proves to be effective for OOD detection within the UCL setup.

Chapter 6

Improving Network Capacity with Sub-network Masking and Accuracy through Hierarchical Representation Learning

As discussed in chapter 4 and 5, relying solely on weight magnitude (hard attention) for subnetwork construction can result in retaining unimportant weights and redundant sub-networks, leading to capacity saturation. This drives us to explore alternative approaches to circumvent these challenges and refine the construction of subnetworks. Inspired by Information Bottleneck (IB) theory [38, 39, 40], we use subnetwork masking through IB to achieve CL. This approach integrates IB to minimize redundancy while maintaining or freezing essential parameters, thereby mitigating CF. Simultaneously, irrelevant information is channeled into expendable parameters. It optimizes inter-layer mutual information to construct redundancy-free sub-networks, addressing challenges in CL. Although subnet-

The work presented in this chapter has been published in the paper: "*HiProIBM: Unsupervised Continual Learning through Hierarchical Prototypical Cross-Level Discrimination along with Information Bottleneck Subnetwork Masking*", Applied Intelligence, 55(6), 476-497 (2025). DOI: 10.1007/s10489-025-06362-z work masking using IB is explored in SCL [40], however minimizing forgetting through IB in UCL poses inherent challenges [42]. The absence of labels makes it difficult to identify relevant information, increasing the risk of losing essential features and leading to potential overcompression that can obscure important latent structures [43].

As discussed in 4 and 5, grouping instances is essential for capturing similarity, as relying only on individual discrimination can reduce stability and effectiveness. However, clustering based solely on attraction may group dissimilar instances due to shared repulsion effects [33]. To address this, we guide each instance toward its assigned cluster while repelling it from unrelated clusters. While this approach typically compares an image with a singular set of prototypes, this overlooks potential semantic relationships with multiple clusters. Instead of solely associating an image with the most similar prototypes, it is beneficial to consider other clusters as well, which are then regarded as negative samples.

Consequently, we employ hierarchical prototypes to represent semantic structures in data and dynamically update them during training. By leveraging these hierarchical semantic representations, we aim to enhance traditional instance-wise and prototypical contrastive learning. This involves optimizing positive and negative pair selections to better align with the semantic structures.

Hierarchical prototypical cross-level discrimination complements IB subnetwork masking by identifying relevant information and preventing the loss of essential features due to overcompression. Together, these components address each other's limitations, enabling effective management of the trade-off between stability and plasticity in UCL setup. Figure 1.7 shows an abstract view of the proposed Framework-III. A detailed description of the various steps involved in the proposed technique is discussed as follows.

6.1 Proposed Technique

6.1.1 Unsupervised contrastive learning with cross-level instance and hierarchical cluster discrimination with OOD detection

This section presents a novel Framework-III for UCL, emphasizing contrastive learning methods for maintaining continuous representation. However, as discussed in 4 and 5, challenges emerge due to imbalanced ratios of positive and negative samples within batches, which may lead to biasing models toward instance discrimination rather than invariant clustering. To address this issue, we propose an innovative approach that integrates direct instance grouping and cross-level group discrimination along with hierarchical prototypical clustering. It helps in capturing the inherent semantic structures present in the data. In the subsequent sections, we elaborate on each component of the Framework-III.

Building on the contrastive loss $\mathcal{L}$ from Chapter 4, we use a feature extractor $\mathcal{G}_\Omega$ with noise contrastive estimation (NCE) as in (4.1) to learn representations by aligning augmented views of the same instance while separating others, as shown in Figure 6.1(e) and Figure 6.2.

As discussed earlier, this loss function $\mathcal{L}$ comprises three components: (1) instance-level discrimination for distinctiveness, (2) cross-level clustering for semantic grouping, and (3) pseudo-instance discrimination for OOD separation.

6.1.1.1 Learning invariant mapping via instance level discrimination

Training models to identify similar representations of augmented images involves capturing intrinsic characteristics that remain consistent despite various transformations. This section reuses the SimSiam [53] based instance-level discrimination discussed in §5.1.1.1. As shown by a yellow branch in the multi-tasking loss function in Figure 6.1.(c) and yellow

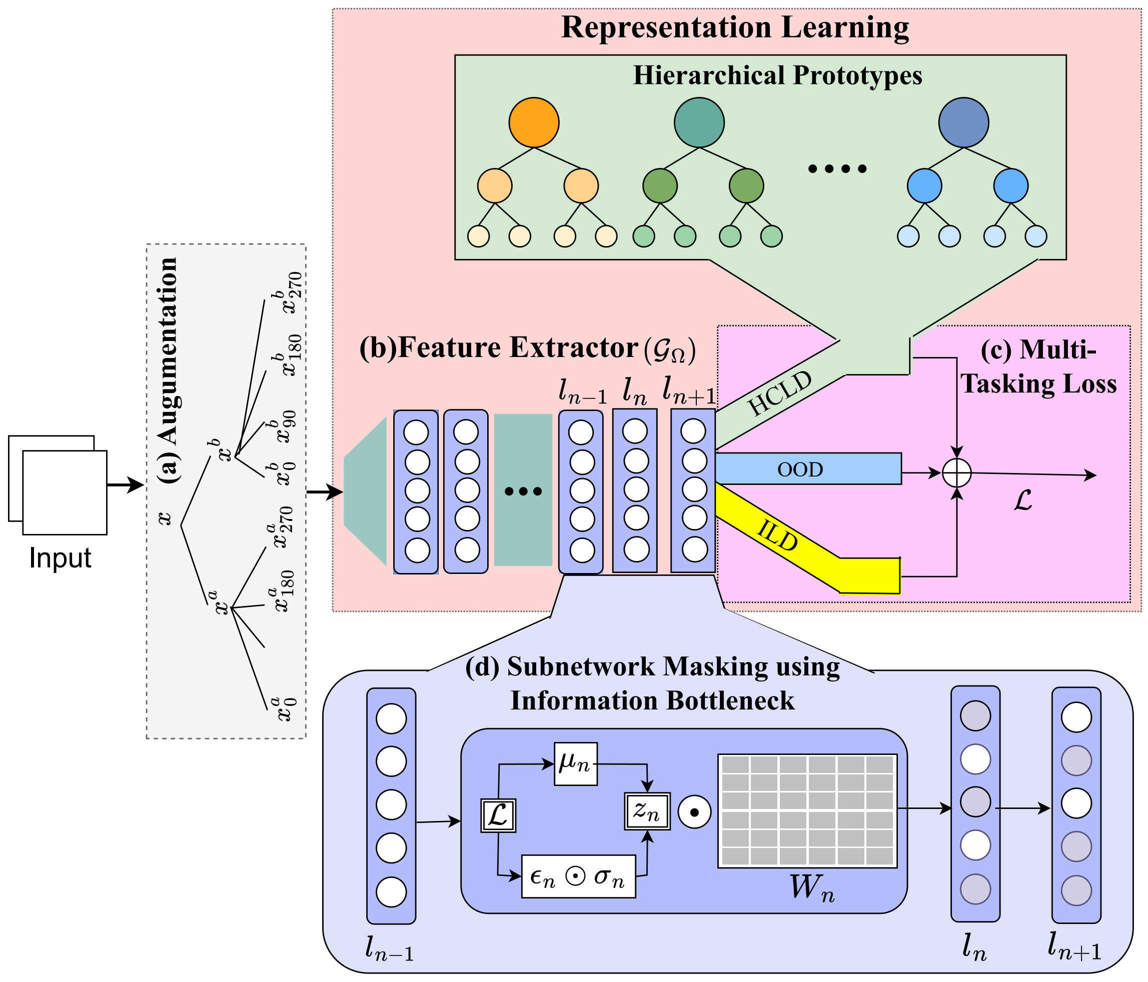

Representation Learning
Hierarchical Prototypes
(a) Augumentation
Input
(b)Feature Extractor ($\mathcal{G}_\Omega$)
l_{n-1} l_n l_{n+1}
HCLD
OOD
ILD
(c) Multi-Tasking Loss
$\mathcal{L}$
(d) Subnetwork Masking using Information Bottleneck
μ_n
z_n
$\epsilon_n \odot \sigma_n$
W_n

Figure 6.1: Overview of the Proposed Framework-III. (a). The augmentation block includes various techniques for augmenting the current batch of samples. (b). Feature extractor $\mathcal{G}_\Omega$ is trained using the information bottleneck principle. Its output is fed into a loss function via projectors: Projector 1 for instance-level discrimination, Projector 2 for prototypical hierarchical cross-level cluster discrimination, and Projector 3 for pseudo-instance group cross-level discrimination. (c). Multi-tasking Loss function, expanded details can be found in Figure 6.2. (d). Subnetwork masking using information bottleneck.

box in Figure 6.2, we use a shared encoder $\mathcal{G}_{\Omega}$ along with prediction and projection heads ($\mathcal{F}_I$, $\mathcal{H}_I$) to minimize the symmetric cosine distance between augmented views, following the formulation in Equation (5.1).

6.1.1.2 Prototypical hierarchical cross-level cluster discrimination

As discussed in 4.1.1.2, organizing instances into groups is crucial for determining their similarity, rather than just focusing on individual discrimination. This is necessary because discrimination on an individual level may compromise both stability and effectiveness. However, previous research suggests that clustering can result from attraction as well as shared repulsion. Relying solely on instance clustering may also lead to problems like grouping dissimilar instances [33]. Therefore, in Framework-III as well, our endeavors are twofold: first, to direct each instance towards its associated cluster, and second, to simultaneously repel distant clusters associated with other instances.

Prototypical contrastive learning endeavors to create condensed representations of images within a latent space, ensuring close proximity to their respective cluster centers. While traditional approaches [31, 32] typically compare an image with a singular set of prototypes, this overlooks potential semantic relationships with multiple clusters. Instead of solely associating an image with the most similar prototypes, it is beneficial to consider other clusters as well, which are then regarded as negative samples. Consequently, we employ hierarchical prototypes to represent semantic structures in data and dynamically update them during training. By leveraging these hierarchical semantic representations, we aim to enhance traditional instance-wise and prototypical contrastive learning. This involves optimizing positive and negative pair selections to better align with the semantic structures.

Mathematically, for a given image x, semantic similarity $\mathcal{S}$ between its δ-dimensional

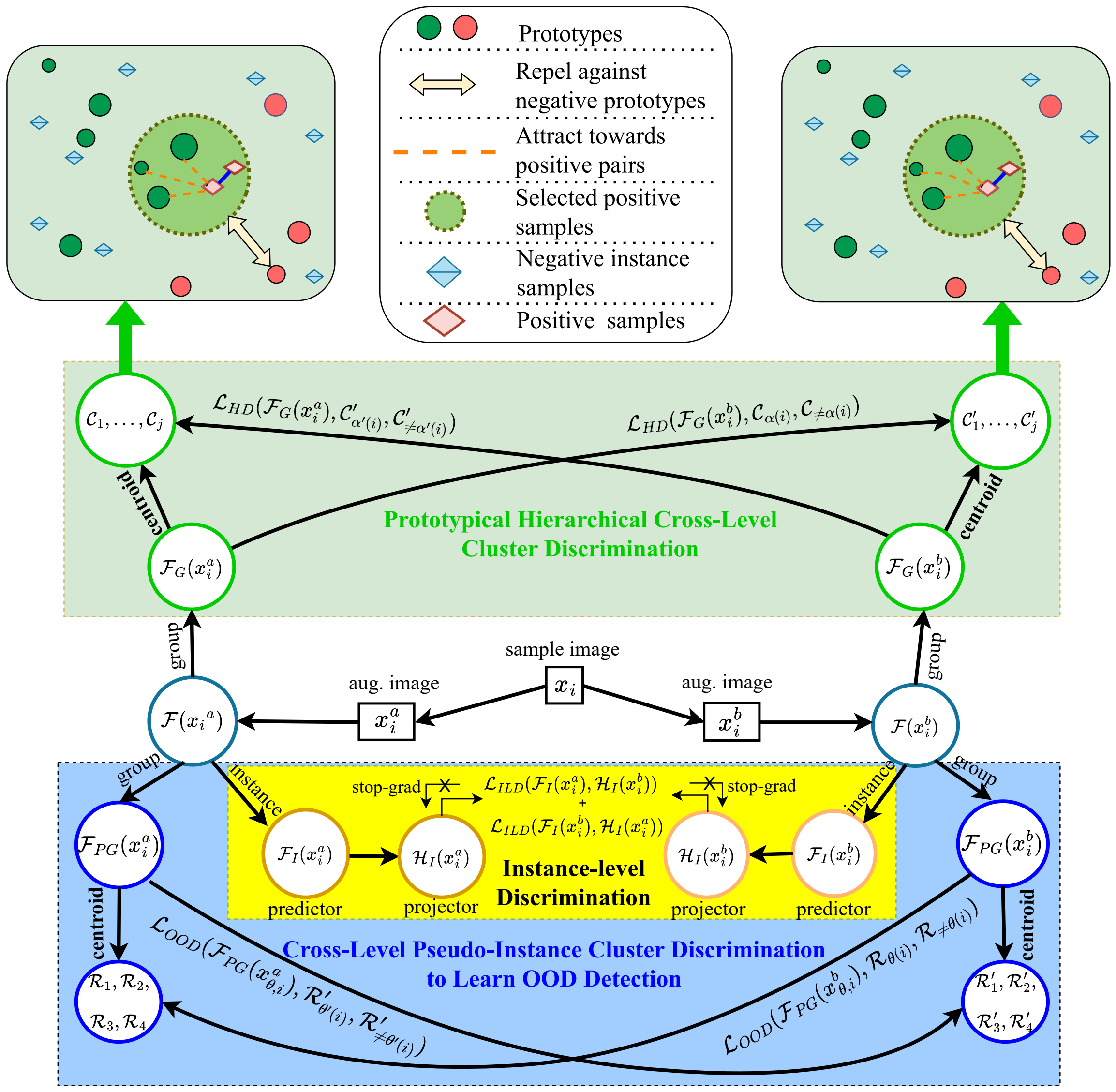

Prototypes
Repel against negative prototypes
Attract towards positive pairs
Selected positive samples
Negative instance samples
Positive samples
$\mathcal{C}_1,\ldots,\mathcal{C}_j$
$\mathcal{L}_{HD}(\mathcal{F}_G(x_i^a), \mathcal{C}'_{\alpha'(i)}, \mathcal{C}'_{\neq\alpha'(i)})$
$\mathcal{L}_{HD}(\mathcal{F}_G(x_i^b), \mathcal{C}_{\alpha(i)}, \mathcal{C}_{\neq\alpha(i)})$
$\mathcal{C}'_1,\ldots,\mathcal{C}'_j$
centroid
Prototypical Hierarchical Cross-Level Cluster Discrimination
$\mathcal{F}_G(x_i^a)$
$\mathcal{F}_G(x_i^b)$
group
sample image
x_i
aug. image
x_i^a
aug. image
x_i^b
$\mathcal{F}(x_i{}^a)$
$\mathcal{F}(x_i^b)$
group
instance
stop-grad
$\mathcal{L}_{ILD}(\mathcal{F}_I(x_i^a), \mathcal{H}_I(x_i^b))$
+
$\mathcal{L}_{ILD}(\mathcal{F}_I(x_i^b), \mathcal{H}_I(x_i^a))$
stop-grad
instance
group
$\mathcal{F}_{PG}(x_i^a)$
$\mathcal{F}_I(x_i^a)$
$\mathcal{H}_I(x_i^a)$
Instance-level Discrimination
$\mathcal{H}_I(x_i^b)$
$\mathcal{F}_I(x_i^b)$
$\mathcal{F}_{PG}(x_i^b)$
predictor
projector
projector
predictor
centroid
Cross-Level Pseudo-Instance Cluster Discrimination to Learn OOD Detection
centroid
$\mathcal{L}_{OOD}(\mathcal{F}_{PG}(x_{\theta,i}^a), \mathcal{R}'_{\theta'(i)}, \mathcal{R}'_{\neq\theta'(i)})$
$\mathcal{L}_{OOD}(\mathcal{F}_{PG}(x_{\theta,i}^b), \mathcal{R}_{\theta(i)}, \mathcal{R}_{\neq\theta(i)})$
$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$
$\mathcal{R}'_1, \mathcal{R}'_2, \mathcal{R}'_3, \mathcal{R}'_4$

Figure 6.2: Overview of the multitasking loss function in Framework-III. To improve representation learning via unsupervised contrastive loss, it creates a representation $\mathcal{F}(x)$ for the augmented views x_i^a and x_i^b of an input image x_i. This representation is used in three branches: instance-level discrimination ($\mathcal{F}_I$), hierarchical cross-level cluster discrimination ($\mathcal{F}_G$), and pseudo-instance cluster discrimination for OOD detection ($\mathcal{F}_{PG}$). Computation is symmetrical across different augmentations of the same instance.

representation vector r and prototype g can be defined as follows, as explained in PCL[32],

$$\mathcal{S}(r,g) = \frac{r \cdot g}{\rho_g}, \quad \rho_g = \frac{\sum_{r_i \in R_g} ||r_i - g||_2}{|R_g| \log(|R_g| + \varphi)} \tag{6.1}$$

where, $r \in R$ such that $R = \{r_1, r_2, \dots, r_n\}$, $g \in G$ such that G is a set of hierarchical prototypes, R_g comprises the representations of the images assigned to cluster g, and φ acts as a smoothing parameter to balance the scale of temperature ρ_g across various clusters.

Out of V_h prototypes, the most similar prototype of a given images x at the h^{th} hierarchy, is defined as

$$g^h(r) = \arg \max_{g \in \{g_i^h\}_{i=1}^{V_h}} \mathcal{S}(r,g) \tag{6.2}$$

To avoid pushing an image too far from the relevant cluster, we use prototypical contrast with candidates that are semantically distant from the positive cluster $g^h(r)$. Hierarchical prototypes help us achieve this by capturing semantic structures.

The parent node of $g^h(r)$ is denoted as $Parent(g^h(r))$. The parent cluster summarizes the common semantics of the child cluster at each hierarchy. The negative clusters for an image share a low semantic correlation. The probability of selecting a negative candidate

$$m^h_{\text{select}}(g_j; g^h(r)) = 1 - \frac{\exp[\mathcal{S}(g_j, Parent(g^h(r)))]}{\sum_{i=1}^{V_{h+1}} \exp[\mathcal{S}(g_j, g_i^{h+1})]} \tag{6.3}$$

Bernoulli sampling is conducted on each negative candidate to select the negative samples that are more semantically distant from $g^h(r)$ for prototypical contrast.

$$\mathcal{M}^h_{\text{select}}(g^h(r)) = \{g_j; \mathcal{B}(m^h_{select}(g_j; g^h(r))) \mid (g_j \in \mathcal{M}^h)\} \tag{6.4}$$

Positive pairs $(r, g^h(r))$ are contrasted with the negative samples, and the hierarchical discriminative loss function is defined as

$$\mathcal{L}_{HD} = \mathbb{E}_{x\sim\mathbb{D}_\tau}\left[\frac{1}{H}\sum_{h=1}^{H}\mathcal{L}_{\text{ProtoNCE}}(r, g^h(r), \mathcal{M}^h_{\text{select}}(g^h(r)))\right] \tag{6.5}$$

here $\mathcal{L}_{\text{ProtoNCE}}$ is same as $\mathcal{L}$ defined in Equation (4.1) and in [32], $\mathbb{D}_\tau$ denoted the data distribution. In the above formulation, the positive pair $(r, g^h(r))$ captures the relationship between the instance and its corresponding prototype at a specific hierarchy level.

As shown by a green color branch in multi-tasking loss function in Figure 6.1.(c) and green box in Figure 6.2, for the prototypical hierarchical cross-level cluster discrimination, K-means is applied to $\mathcal{F}_G(x_i^a)$ to identify j centroids, denoted as $\{\mathcal{C}_1, ..., \mathcal{C}_j\}$, with instance i assigned to centroid $\alpha(i)$. Corresponding elements in the alternative view are $\mathcal{F}_G(x_i^b)$, $\{\mathcal{C}'_1, ..., \mathcal{C}'_j\}$, and $\alpha'(i)$.

For cross-level discrimination between instances, a contrastive loss is employed between feature $\mathcal{F}_G(x_i^a)$ and centroids $\mathcal{C}'$ based on grouping α', and vice versa for x_i^b. When dealing with two similar instances like x_i^a and x_i^b, they are separated by the instance-level contrastive loss ($\mathcal{L}_{ILD}$) but brought closer by the hierarchical cross-level contrastive loss ($\mathcal{L}_{HCLD}$), distancing themselves from common negative groups. The hierarchical cross-level cluster discrimination can be defined as:

$$\mathcal{L}_{HCLD} = \mathcal{L}_{HD}(\mathcal{F}_G(x_i^a), C'_{\alpha(i)}, C'_{\neq\alpha(i)}) \tag{6.6}$$

here, $C'_{\alpha(i)}$ is the centroid of the cluster $\alpha'(i)$ derived from the clustering of $\mathcal{F}_G(x_i^b)$, this acts as the positive sample to be attracted. The rest of the cluster centroids $C'_{\neq\alpha(i)}$ act as negative samples to be repelled.

Implementation details: The loss $\mathcal{L}_{HCLD}$ in equation (6.6) can be interpreted as a normalized prediction score indicating the belongingness of $\mathcal{F}_G(x_i)$ to the $C_{\alpha(i)}$ cluster. Minimizing this loss is equivalent to reducing the cross-entropy between the hard cluster assignment of $C_{\alpha(i)}$ and the soft cluster assignment of $C'_{\alpha(i)}$. Our training objective is to embed an instance to be similar to a group of related instances, hence referred to as coarse-grained.

Limitations: Combining instance-level and prototypical hierarchical cross-level discrimination effectively distinguishes correlated instances and preserves semantic groupings within a distribution. However, it is also susceptible to misclassifying samples from one task as the most similar to samples from another task. To address this, we employ cross-level pseudo-instance cluster discrimination to learn OOD detection for identifying clear task boundaries.

6.1.1.3 Cross-level pseudo cluster discrimination to learn OOD detection

As previously discussed in Chapter 4, cross-level pseudo group discrimination enhances OOD detection by simulating pseudo-OOD samples via rotation augmentations. It supports the model to distinguish between ID and OOD data within a contrastive learning framework.

As shown by the blue branch in Figure 6.1.(c) and the blue box in Figure 6.2, we use an additional projection head $\mathcal{F}_{PG}$ to compute a distribution-level contrastive loss, $\mathcal{L}_{OOD}$. We treat each rotation angle as a separate cluster. Unlike the earlier cluster-based approach discussed in §6.1.1.2, pseudo-label-based cluster guides this loss, enabling more effective OOD separation at the representation level. This loss function $\mathcal{L}_{OOD}$ is the same as discussed in Equation (4.4).

Implementation details: This loss can be interpreted as cross-entropy between the hard assignment of the distribution-level cluster $\mathcal{R}_\theta$ and the soft assignment $\mathcal{R}'_\theta$ predicted by $\mathcal{F}_{PG}$.

Advantage: As discussed in §6.2.5, incorporating $\mathcal{L}_{OOD}$ into $\mathcal{L}$ improves accuracy. Intuitively, a model discerning OOD and ID data also learns more descriptive features.

6.1.2 Subnetwork masking using information bottleneck

A neural network can be conceptualized as a Markov Chain, progressing from input to output through a sequence of hidden layers:

$$x \to l_1 \to l_2 \to \ldots \to l_N \to \hat{o} \to o \tag{6.7}$$

In this architecture, each hidden layer extracts insights from its predecessor, allowing the final layer to estimate the true probability distribution $p(o)$ using output probability distribution $p(\hat{o}|l_N)$. Training improves accuracy in predicting $p(o)$, though internal representations may include unnecessary details [38, 39]. The information bottleneck process introduces a constraint for minimizing redundancy between adjacent layers [39]. This constraint aims to identify the mapping $\sum_{n=1}^{N} f_n$ between layers that learns the maximum information through optimal compression of the adjacent layers, ensuring the extraction of sufficient mapping. This process minimizes the mutual information $I(l_n; l_{n-1})$, while maximizes the mutual information $I(l_n; o)$ simultaneously. The optimal representation of finding l_n is defined by minimizing the following Lagrangian function:

$$\mathcal{L}_n = \varsigma_n I(l_n; l_{n-1}) - I(l_n; o) \tag{6.8}$$

here, the Lagrangian multiplier $\varsigma \geq 0$ acts as a trade-off parameter, determining the balance between compressing the representation and retaining pertinent information.

Variational inference is employed to make equation (6.8) computationally solvable. Particularly, we apply variational distribution of $q(l_n)$ and $q(o|l_N)$ to approximate the distribution of $p(l_n)$ and $p(o|l_N)$. This process yields an upper bound for $\mathcal{L}_n$:

$$\mathcal{L}_n \leq \hat{\mathcal{L}}_n = \varsigma_n \mathbb{E}_{l_{n-1} \sim p(l_{n-1}|x)} \left[\text{KL} \left[p(l_n|l_{n-1}) || q(l_n) \right] \right] - \mathbb{E}_{l \sim p(l|x)} \left[\log q(o|l_N) \right] \tag{6.9}$$

Summing up all the upper bounds of $\mathcal{L}_n$ across the network,

$$\mathcal{L}_{IB} = \sum_n \mathcal{L}_n \tag{6.10}$$

here, the minimum value of $\mathcal{L}_{IB}$ is the final variational information bottleneck (IB) loss, it enables effective information management throughout all layers.

To handle $p(l_n|l_{n-1})$, $q(o|l_N)$ and $q(l_n)$, we require their parametric distributions. We incorporate the concept of the IB in the network by utilizing weight reparameterization. This allows us to select specific weights for each task and helps in integrating IB masking into CL.

$$l_n = f_n(l_{n-1}, (\mu_n + \epsilon_n \odot \sigma_n) \odot W_n) \tag{6.11}$$

here, ϵ_n represents a randomly sampled parameter from $\mathcal{N}(0, I)$, while μ_n and σ_n are the learnable variational parameters. Bias is excluded in all layers to reduce complexity. The parameterized form of $p(l_n|l_{n-1})$ is defined as:

$$p(l_n|l_{n-1}) = \mathcal{N}(l_n; f_n(W_n l_{n-1} \odot \mu_n), \text{diag}[f_n(W_n^2 l_{n-1}^2 \odot \sigma_n^2)]) \tag{6.12}$$

Next, we define $q(l_n)$ as a Gaussian distribution:

$$q(l_n) = \mathcal{N}(l_n; 0, \text{diag}[\xi_n]) \tag{6.13}$$

here, ξ_n is a learnable variance. These Gaussian assumptions enable an interpretable and closed-form estimate for the KL term in equation (6.9), allowing direct optimization of ξ_n. After some algebraic transformations, we arrive at the updated final loss function.

$$\mathcal{L}_{IB} = \sum_{n=1}^{N} \left[\varsigma_n \log \left(1 + \frac{\mu^2}{\sigma^2} \right) \right] - \mathcal{L}\mathbb{E}_{l \sim p(l|x)} \left[\log q(o|l_N) \right] \tag{6.14}$$

where the parameter ς offers flexibility in adjusting the compression ratio across each layer.

A subnetwork devoid of redundancy is formed by pruning expandable weights and retaining selected weights. Certain weights are chosen to build a binary mask, preserving variational parameters for inference. This allows for the reconstruction of the hidden representation of l_n.

$$\begin{aligned} T_n^\tau &= (\mu_n + \epsilon_n \odot \sigma_n) \odot Z_n, \\ l_n &= f(l_{n-1}, T_n^\tau \odot W_n). \end{aligned} \tag{6.15}$$

here, Z_n represents the binary matrix for each layer. Since activated weights are much fewer than the total parameters, the cost of saving these variational parameters is minimal. It helps in creating a non-redundant sub-network for the current task.

Mitigating catastrophic forgetting: During CL, as the network undergoes training for new tasks, prior masks are combined. It retains crucial information pertaining to all preceding tasks.

$$Z = 0, \quad Z_{\text{all}} = \sum_{i=1}^{\tau} Z_i \parallel Z \tag{6.16}$$

here, Z_{all} represents weights that retain valuable information from all past tasks, and $\parallel$ denotes or operation. When training on a new task, we adjust the gradients $\nabla_W \mathcal{L}$ of the weights during backpropagation according to the previously learned binary masks by effectively freezing the weights selected for the prior tasks.

$$\nabla_W \mathcal{L} = \nabla_W \mathcal{L} \odot (1 - Z_{\text{all}}), \tag{6.17}$$

Reinitialization of variational parameters: Freezing the subnetworks related to old tasks can impede knowledge transfer when learning new tasks. To address this, we opt to utilize the variational parameters selected by the previous masks and then reinitialize the remaining ones.

$$\begin{aligned} \mu_n &= \mu_n \odot Z_{\text{all},n} + \mu_{\text{random}} \odot (1 - Z_{\text{all},n}), \\ \sigma_n &= \sigma_n \odot Z_{\text{all},n} + \sigma_{\text{random}} \odot (1 - Z_{\text{all},n}). \end{aligned} \tag{6.18}$$

This re-initialization process allows optimization to avoid local minima and aids in selecting relevant knowledge while disregarding redundant information.

Feature factorization: Since lower layers focus on visual features and higher layers on semantic information, each layer requires different compression ratios. We analyze hidden representations to set these ratios adaptively. In CL, SVD is commonly employed to factorize the hidden representations in these subnetworks. $H = \mathcal{A}\Sigma\mathcal{B}^{\mathrm{T}} \in \mathbb{R}^{i\times j}$ where $\mathcal{A} \in \mathbb{R}^{i\times i}, \quad \mathcal{B} \in \mathbb{R}^{j\times j}$ are orthogonal and Σ consists of the sorted singular values along its main diagonal. We apply SVD on $l_n = \mathcal{A}_n\Sigma_n\mathcal{B}_n^{\mathrm{T}}$ to analyze the distribution of the hidden representation, followed by its k-rank approximation, $(l_n)_k$ based on a specified threshold δ:

$$\|(l_n)_k\|_F^2 \geq \delta\|(l_n)\|_F^2, \tag{6.19}$$

where, δ $(0 < \delta < 1)$ serves as the threshold hyperparameter, and F denotes the Frobenius norm. By applying the k-rank approximation, the first k vectors in $\mathcal{A}$ capture the significant representations, as they include all directions associated with the highest singular values. Hence, the ratio of k across all channels determines the ultimate compression ratio for layer n. As the feature distribution changes during training, we carry out this decomposition at each E epoch to dynamically adjust the ratio, allowing for more accurate compression.

6.1.3 Overall objective function

The final objective to be minimized for task τ is as follows (see in Figure 6.1.(c))

$$\mathcal{L} = \mathcal{L}_{ILD} + \gamma_{HCLD}\,\mathcal{L}_{HCLD} + \gamma_{OOD}\,\mathcal{L}_{OOD} + \gamma_{IB}\,\mathcal{L}_{IB} \tag{6.20}$$

here, $\mathcal{L}_{ILD}$, $\mathcal{L}_{HCLD}$, $\mathcal{L}_{OOD}$, and $\mathcal{L}_{IB}$ denote the loss function for instance-level discrimination, hierarchical cross-level cluster discrimination, pseudo-group level discrimination for OOD detection, and information bottleneck loss respectively. $\mathcal{L}$ represents the overall loss function or can be referred to as a multi-tasking loss function. γ_{HCLD}, γ_{OOD}, and γ_{IB} denotes positive scaling parameter for the respective loss function.

6.2 Experimental Analysis

This section provides an overview of (1) baselines, (2) training and evaluation setup, (3) hyperparameter settings, and (4) results and comparative analysis.

6.2.1 Baselines

For the naming convention, the proposed Framework-III, incorporates cross-level discrimination of instances and hierarchical prototypical clusters along with IB subnetwork masking. We compare Framework-III with both classic and most recent SOTA methods. Similar to Framework-I and Framework-II for the baselines of Framework-III as well, we consider four unsupervised methods, i.e., SI [16], DER [14], PNN [9], LUMP [11], along with one supervised method, Co2L [12]. Additionally, we included Framework-II [2] as the fifth UCL baseline. We also compare Framework-III with its preceding variants that we explored before finalizing it.

6.2.2 Training and evaluation setup

The overall training and evaluation setup is mostly consistent with those described in Section §4.2.2. The primary differences lie in the proposed model architecture and variants used in Framework-III. Specifically, we adjusted the ResNet-18 architecture to incorporate IB subnetwork masking for feature extraction. For the proposed Framework-III and its variants detailed in Table 6.8, we implemented the LARS optimization function [130] with the same learning rate schedule as before in Framework-I and Framework-II. Baseline methods are trained using SGD with identical settings, and all evaluations are conducted using a KNN classifier [52]. Additional information on hyperparameter configurations is presented in the following section.

6.2.3 Hyperparameters setting

Assessing the impact of hyperparameters on performance is crucial in any machine learning algorithm. In the proposed Framework-III, there exists a collection of hyperparameters that influence performance. To determine the optimal hyperparameters for each approach, we performed a grid search using a task sequence determined by a single seed. The configuration of optimal hyperparameters of Framework-III and its variants for all datasets is provided in Table 6.1, 6.2, 6.3, 6.4, 6.5. The hyperparameter values for all baseline models across all benchmark datasets are available in Tables 4.2, 4.3, and 4.4. Framework-III uses a similar set of temperature parameters as Framework-II, ρ_{ILD}, ρ_{HCLD}, and ρ_{OOD}, for instance-level, group-level, and pseudo-group-level contrastive losses, respectively. It also includes scaling factors γ_R,γ_{HCLD}, γ_{OOD}, and γ_{IB} for regularization, hierarchical cross-level, pseudo-group-level, and information bottleneck losses, respectively. In addition, we use cluster counts Ci and threshold values s and $s\max$. These hyperparameters are fine-tuned to accommodate the IB subnetwork masking and hierarchical prototypical cross-level discrimination introduced in this chapter. The sensitivity analysis of hierarchical prototype configurations for Framework-III across datasets is detailed in §6.2.3.3 and presented in Tables 6.4 and 6.5. For IB subnetwork masking, the feature factorization threshold (δ) is set to 0.97, and the epoch interval (E) is set to 50. The sensitivity analysis for the threshold δ is detailed in §6.2.3.1, while the analysis for the feature factorization frequency (E) is provided in §6.2.3.2.

In this section, we present the hyperparameter details for baseline models and variants of Framework-III across all datasets determined via grid search.

6.2.3.1 Sensitivity analysis of Framework-III for varying threshold δ

As stated in (6.19), when δ is large within the range $(0 < \delta < 1)$, the top k singular vectors with the highest singular values can be extracted with minimal performance loss. The ratio of k across all channels determines the overall compression ratio for each layer. As outlined in Equation 4 of [146], when δ is isolated on the left side of the equation, it must be a small to maintain stability. As shown in Table 6.2, we performed a sensitivity analysis on the CH-20T and I1H20T datasets to determine the optimal δ, testing values of 0.90, 0.95, 0.97, and 1.0. The highest performance is observed at $\delta = 0.97$, achieving accuracies of 83.98% and 78.68%, with forgetting rates of 0.0005% and 0.001%, respectively. Based on this analysis, we set $\delta = 0.97$ as the default value for all experiments.

6.2.3.2 Sensitivity analysis of feature factorization frequency

To assess the impact of the feature factorization interval E, we conducted experiments on CH-20T and I1H20T using ResNet-18. Table 6.3 presents the results of the Framework-III with and without the feature factorization module at various training epochs (10, 25, 50, 100, and 150). The results indicate that all variants with the feature factorization module

Table 6.1: Optimal hyperparameter values for variants of Framework-III across datasets determined via grid search. Further details on the other hyperparameters that are not covered here are available in [2].

Dataset	τ_{ILD}	τ_{GCD}	τ_{OOD}	γ_{GCD}	γ_{HCLD}	γ_{OOD}	γ_R	$\mathcal{C}_i$	γ_{IB}
M-5T	0.1	0.1	0.1	0.5	0.5	0.5	0.25	128	0.5
C-5T	0.1	0.2	0.2	0.5	0.5	0.5	0.5	128	0.5
CH-5T	0.1	0.2	0.2	0.5	0.5	0.5	0.5	150	0.5
CH-10T	0.1	0.2	0.2	0.5	0.5	0.5	1	180	0.5
CH-20T	0.1	0.1	0.1	0.5	0.5	0.5	1.5	200	0.5
I1H-5T	0.1	0.2	0.2	0.5	0.5	0.5	0.5	160	0.5
I1H-10T	0.1	0.1	0.1	0.5	0.5	0.5	1	180	0.5
I1H-20T	0.1	0.1	0.1	0.5	0.5	0.5	1.5	180	0.5
I2H-40T	0.1	0.1	0.1	0.5	0.5	0.5	2	190	0.5
I2H-50T	0.1	0.1	0.1	0.5	0.5	0.5	2.5	200	0.5
I2H-100T	0.1	0.1	0.1	0.5	0.5	0.5	3.5	200	0.5

Table 6.2: Sensitivity analysis of Framework-III on CH-20T and I1H-20T for varying values of threshold δ

δ	**CH-20T**		**I1H-20T**	
	Accuracy (%)	Forgetting (%)	Accuracy (%)	Forgetting (%)
0.9	82.79	0.0017	77.54	0.011
0.95	83.56	0.0013	78.06	0.007
0.97	83.98	0.0005	78.68	0.001
1	83.47	0.025	78.15	0.014

consistently outperform the variant without it, highlighting its effectiveness. Furthermore, the factorization interval E should neither be too small nor too large. A smaller interval increases training time due to frequent analysis of hidden representations, reducing efficiency, while a larger interval results in diminished performance. This behavior suggests that an optimal interval allows the module to effectively analyze feature distributions over time and adjust the compression ratio, enhancing overall performance. Based on these findings, we adopt a 50-epoch interval as the default setting for the main experiments, balancing performance and efficiency.

6.2.3.3 Sensitivity of hierarchical prototypes configuration

Table 6.4 provides a comparison between single and multiple prototype hierarchies, ensuring an equivalent total number of prototypes in both cases. In the hierarchical configuration, incorporating two additional prototype hierarchies yields a performance improvement

Table 6.3: Sensitivity analysis of feature factorization frequency at specific epochs

# Epochs	**CH-20T**		**I1H-20T**	
	Accuracy (%)	Forgetting (%)	Accuracy (%)	Forgetting (%)
w/o	80.65	1.39	76.53	1.57
10	81.24	0.43	77.04	0.55
25	82.76	0.12	77.85	0.14
50	83.98	0.0005	78.68	0.001
100	82.63	0.07	77.93	0.09
150	81.80	0.16	77.61	0.20

of 1.12% for CH-20T and 1.05% for I1H-20T, as observed by comparing the second and fourth rows of Table 6.4. The hierarchical configuration of 200-100-25 demonstrates competitive performance, while other hierarchical structures perform less effectively. These findings underscore the effectiveness of utilizing hierarchical prototypes, with consistent patterns of optimal hierarchical structures observed across both datasets. Furthermore, a similar analysis is conducted across all datasets, with the results summarized in Table 6.5.

Table 6.4: Sensitivity analysis on the number of hierarchies and the number of prototypes for Framework-III on CH-20T and I1H-20T.

Configuration of hierarchical prototypes	**CH-20T** Accuracy (%)	**I1H-20T** Accuracy (%)
200	83.33	78.05
400	82.86	77.63
200-50	83.67	78.42
200-100-25	83.98	78.68
200-100-25-5	83.98	78.68

Table 6.5: Optimal Hierarchical prototypes configuration for Framework-III across datasets

Dataset	$\mathcal{C}_i$	**Hierarchical prototypes**	**Accuracy (%)**
M-5T	128	128-64-8	99.46
C-5T	128	128-64-8	91.4
CH-5T	150	150-75-15	61.11
CH-10T	180	180-90-18	68.21
CH-20T	200	200-100-25	75.6
I1H-5T	160	160-80-16	54.62
I1H-10T	180	180-90-18	55.39
I1H-20T	180	180-90-18	67.16
I2H-40T	190	190-95-19	61.37
I2H-50T	200	200-100-25	64.05
I2H-100T	200	200-100-25	79.52

Table 6.6: Overall-average Accuracy (in %) for Framework-III and baselines on benchmark datasets across a range of tasks. Where, names of the datasets are present in the first column. The final row shows the average accuracy of each method across all datasets. SI, DER, PNN, LUMP, and Framework-II refer to the UCL methods, while Co2L corresponds to the SCL method. Additional details are available in §6.2.4.1.

Methods → **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-II**	**Framework-III**	**Co2L(Sup)**
M-5T	$99.46_{\pm 0.13}$	$99.28_{\pm 0.17}$	$99.23_{\pm 0.14}$	$99.74_{\pm 0.06}$	$99.91_{\pm 0.04}$	$\mathbf{99.92}_{\pm 0.02}$	$99.85_{\pm 0.07}$
C-5T	$91.40_{\pm 0.69}$	$91.64_{\pm 0.53}$	$91.05_{\pm 0.87}$	$91.23_{\pm 0.45}$	$94.54_{\pm 0.21}$	$\mathbf{94.58}_{\pm 0.13}$	$92.02_{\pm 1.29}$
CH-5T	$61.11_{\pm 0.91}$	$61.60_{\pm 0.84}$	$58.50_{\pm 1.84}$	$60.04_{\pm 1.57}$	$66.32_{\pm 0.36}$	$\mathbf{66.98}_{\pm 0.24}$	$69.46_{\pm 0.63}$
CH-10T	$68.21_{\pm 1.32}$	$69.37_{\pm 1.25}$	$62.47_{\pm 2.16}$	$69.15_{\pm 1.28}$	$75.57_{\pm 0.48}$	$\mathbf{76.36}_{\pm 0.35}$	$73.86_{\pm 1.98}$
CH-20T	$75.60_{\pm 1.68}$	$76.47_{\pm 0.76}$	$65.17_{\pm 2.47}$	$78.74_{\pm 2.39}$	$82.39_{\pm 0.62}$	$\mathbf{83.98}_{\pm 0.29}$	$78.59_{\pm 2.14}$
I1H-5T	$54.62_{\pm 1.19}$	$53.44_{\pm 1.06}$	$48.09_{\pm 1.69}$	$52.90_{\pm 1.37}$	$58.07_{\pm 0.27}$	$\mathbf{59.20}_{\pm 0.21}$	$56.40_{\pm 1.92}$
I1H-10T	$55.39_{\pm 1.90}$	$58.29_{\pm 0.75}$	$49.36_{\pm 1.81}$	$63.68_{\pm 1.49}$	$68.35_{\pm 0.66}$	$\mathbf{69.46}_{\pm 0.42}$	$60.20_{\pm 1.51}$
I1H-20T	$67.16_{\pm 1.53}$	$64.32_{\pm 1.45}$	$57.29_{\pm 2.76}$	$71.18_{\pm 0.95}$	$77.41_{\pm 0.91}$	$\mathbf{78.68}_{\pm 0.58}$	$69.61_{\pm 1.83}$
I2H-40T	$61.37_{\pm 1.76}$	$61.47_{\pm 1.85}$	$55.38_{\pm 2.94}$	$71.38_{\pm 1.63}$	$76.49_{\pm 0.83}$	$\mathbf{78.27}_{\pm 0.67}$	$66.34_{\pm 2.09}$
I2H-50T	$64.05_{\pm 1.87}$	$64.23_{\pm 2.03}$	$58.13_{\pm 3.14}$	$73.20_{\pm 1.77}$	$78.34_{\pm 0.95}$	$\mathbf{80.19}_{\pm 0.62}$	$69.78_{\pm 2.69}$
I2H-100T	$79.52_{\pm 1.99}$	$78.00_{\pm 2.18}$	$69.54_{\pm 3.36}$	$85.89_{\pm 2.75}$	$88.13_{\pm 0.87}$	$\mathbf{89.37}_{\pm 0.74}$	$79.87_{\pm 2.97}$
Average	70.72	70.74	64.93	74.28	78.68	**79.73**	74.18

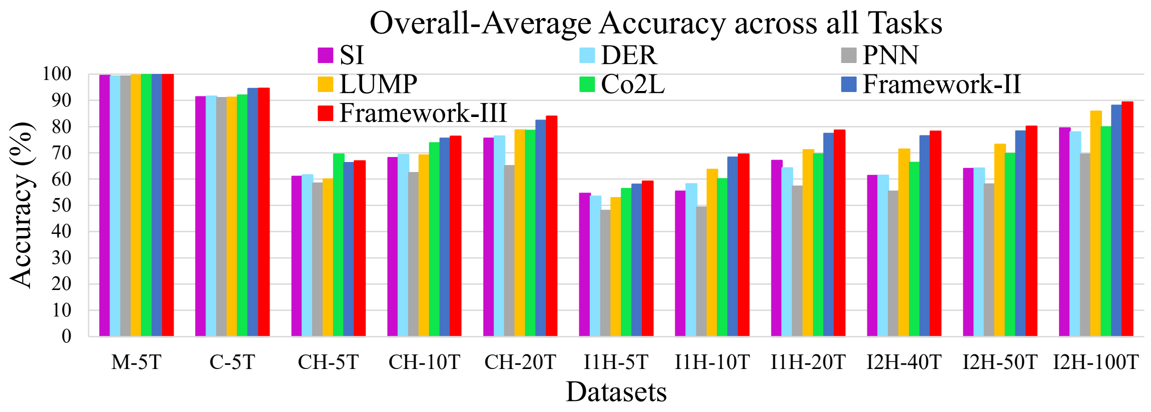


Figure 6.3: Comparison of overall-average Accuracy for Framework-III and baselines on benchmark datasets across all tasks.

6.2.4 Results and comparative analysis

Tables 6.6, 6.7 and Figures 6.3, 6.4, 6.5, 6.6, show the evaluation results for baselines and the proposed Framework-III on M-5T, C-5T, CH-5T, CH-10T, CH-20T, I1H-5T, I1H-10T, I1H-20T, I2H-40T, I2H-50T, and I2H-100T datasets.

6.2.4.1 Overall average-accuracy comparison across tasks

The method of calculating overall-average accuracy across all tasks is explained in §2.4. In Table 6.6 and Figure 6.3, 6.4, 6.5, we present the overall-average accuracy of the baseline models. In Table 6.6, the column headers represent the baseline methods, while the first column contains the names of the benchmark datasets. The last row of this table shows overall-average accuracy of the respective models and the remaining rows show the accuracies of each model on the dataset indicated in the first column. It can be observed from the last row of Table 6.6, which contains the average accuracy of all models, that Framework-III achieves a 9.01%, 8.99%, 14.80%, 5.45% and 1.05% improvement compared to SOTA UCL baselines such as SI, DER, PNN, LUMP, and Framework-II respectively, it can also be verified from Figure 6.3, 6.4, and 6.5. In all cases, we observe that all the mentioned baselines, which represent various CL strategies, i.e. regularization, replay, and architecture, yield lower performance compared to the Framework-III, which is based on parameter-isolation (thus, strengthening the answer in support of **RQ2** [§3.5] in case of Framework-III as well). Outperforming both DER and LUMP indicates that there is no need of a replay buffer anymore to improve the performance. The improved accuracy compared to PNN also indicates the effectiveness of Framework-III over architecture-based strategies. The enhanced accuracy compared to Framework-II demonstrates that Framework-III effectively captures the hierarchical semantic structures inherent in the data (thus, addressing the **RQ6** [§3.5]). Moreover, it achieves performance that is on par with SCL methods like Co2L and even surpasses it in terms of average performance across all datasets by 5.55%. As shown in Figures 6.3, 6.4 and 6.5, while Co2L initially shows higher accuracy on early tasks due to its supervised approach, Framework-III performs better on later tasks by more effectively managing CF compared to Co2L. This is particularly significant because these improvements are achieved without relying on labeled data. It is important to note that our aim is not to claim that our approach surpasses all current SOTA SCL baselines, but rather to show that our method is comparable with these existing baselines.

6.2.4.2 Overall average-forgetting comparison across tasks

The method of calculating overall-average forgetting across all tasks is explained in §2.4. In Table 6.7 and Figure 6.6, we present the overall-average forgetting of the models. Framework-III exhibits **nearly zero** or significantly negligible CF compared to UCL and SCL baselines. For instance, the SCL baseline Co2L shows an overall-average forgetting of 14.51%, while UCL baselines SI, DER, PNN, LUMP, and Framework-II demonstrate 6.68%, 6.75%, 0.02%, 4.03%, and 0.015% overall average forgetting, respectively. Achieving better accuracy with minimal forgetting comparatively Framework-II suggests that the IB subnetwork masking effectively addresses the capacity saturation problem (therefore,

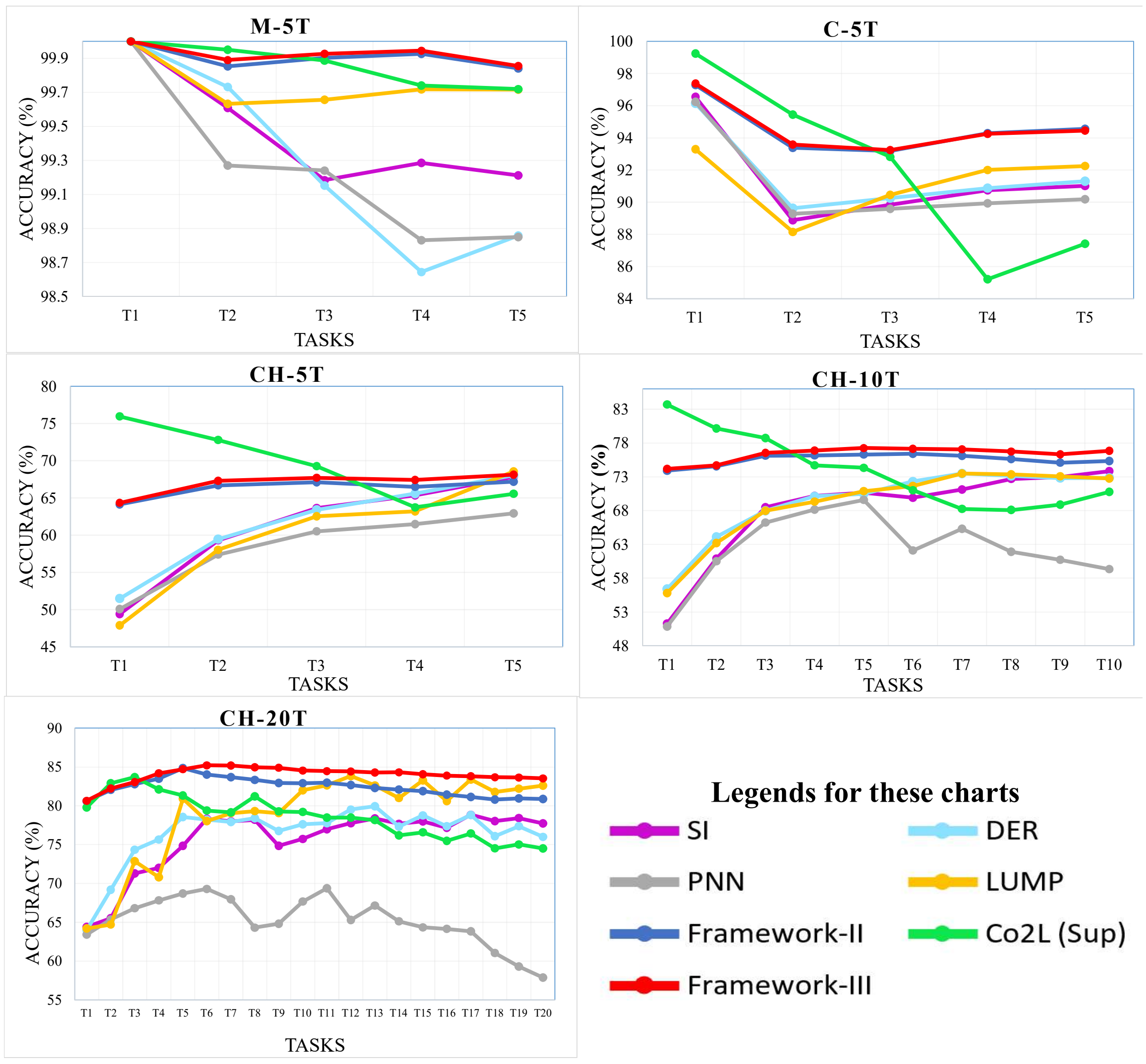


Figure 6.4: Performance comparison for Framework-III and baselines on benchmark datasets across all tasks.

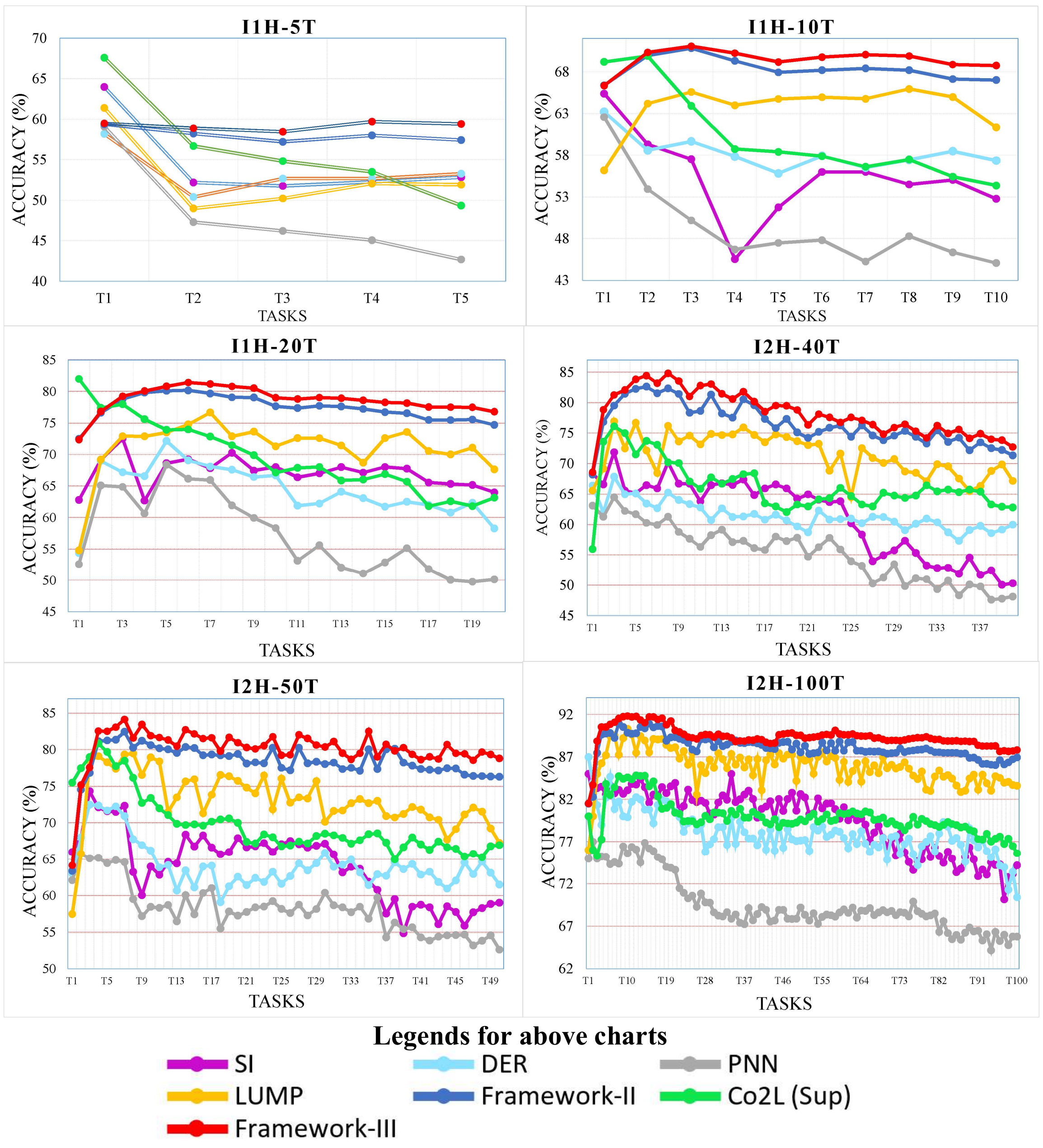


Figure 6.5: Performance comparison for Framework-III and baselines on variants of Tiny-ImageNet (I1H & I2H) datasets across tasks ranging from 5 to 100.

Table 6.7: Overall-average Forgetting (in %) for Framework-III and baselines on benchmark datasets across all tasks.

Methods → **Datasets ↓**	**SI**	**DER**	**PNN**	**LUMP**	**Framework-II**	**Framework-III**	**Co2L(Sup)**
M-5T	$0.45_{\pm 0.12}$	$0.72_{\pm 0.26}$	$0.0002_{\pm 0.0001}$	$0.17_{\pm 0.06}$	$0.0005_{\pm 0.0002}$	$\mathbf{0.0004}_{\pm 0.0001}$	$0.07_{\pm 0.03}$
C-5T	$1.85_{\pm 0.23}$	$3.72_{\pm 0.29}$	$0.0001_{\pm 0.00004}$	$2.27_{\pm 0.62}$	$0.0004_{\pm 0.0003}$	$\mathbf{0.0003}_{\pm 0.0002}$	$7.35_{\pm 1.57}$
CH-5T	$2.36_{\pm 0.87}$	$3.47_{\pm 0.18}$	$0.0004_{\pm 0.0002}$	$3.18_{\pm 0.47}$	$0.0006_{\pm 0.0002}$	$\mathbf{0.0005}_{\pm 0.0003}$	$14.02_{\pm 1.36}$
CH-10T	$2.57_{\pm 1.13}$	$3.63_{\pm 0.16}$	$0.0008_{\pm 0.0003}$	$2.76_{\pm 0.61}$	$0.0003_{\pm 0.0001}$	$\mathbf{0.0007}_{\pm 0.0003}$	$15.46_{\pm 1.58}$
CH-20T	$5.86_{\pm 1.38}$	$5.67_{\pm 0.25}$	$0.003_{\pm 0.002}$	$1.98_{\pm 0.79}$	$0.002_{\pm 0.001}$	$\mathbf{0.0005}_{\pm 0.0002}$	$16.18_{\pm 2.84}$
I1H-5T	$7.71_{\pm 1.29}$	$5.33_{\pm 0.94}$	$0.0005_{\pm 0.0003}$	$3.78_{\pm 0.91}$	$0.0005_{\pm 0.0002}$	$\mathbf{0.0003}_{\pm 0.0001}$	$14.00_{\pm 1.82}$
I1H-10T	$11.66_{\pm 1.47}$	$8.41_{\pm 1.27}$	$0.0008_{\pm 0.0002}$	$4.47_{\pm 1.19}$	$0.0001_{\pm 0.00005}$	$\mathbf{0.00005}_{\pm 0.00002}$	$16.79_{\pm 2.60}$
I1H-20T	$7.89_{\pm 0.74}$	$8.51_{\pm 2.14}$	$0.009_{\pm 0.002}$	$5.47_{\pm 1.28}$	$0.003_{\pm 0.001}$	$\mathbf{0.001}_{\pm 0.0005}$	$17.38_{\pm 3.19}$
I2H-40T	$8.73_{\pm 1.35}$	$10.65_{\pm 2.61}$	$0.012_{\pm 0.006}$	$5.61_{\pm 1.71}$	$0.014_{\pm 0.008}$	$\mathbf{0.003}_{\pm 0.001}$	$18.56_{\pm 3.76}$
I2H-50T	$11.31_{\pm 1.89}$	$11.61_{\pm 2.85}$	$0.057_{\pm 0.014}$	$7.20_{\pm 2.06}$	$0.045_{\pm 0.009}$	$\mathbf{0.018}_{\pm 0.005}$	$19.22_{\pm 3.97}$
I2H-100T	$13.04_{\pm 2.23}$	$12.48_{\pm 3.19}$	$0.14_{\pm 0.05}$	$7.49_{\pm 2.27}$	$0.097_{\pm 0.021}$	$\mathbf{0.042}_{\pm 0.016}$	$20.53_{\pm 4.04}$
Average	6.68	6.75	0.02	4.03	0.015	**0.006**	14.51

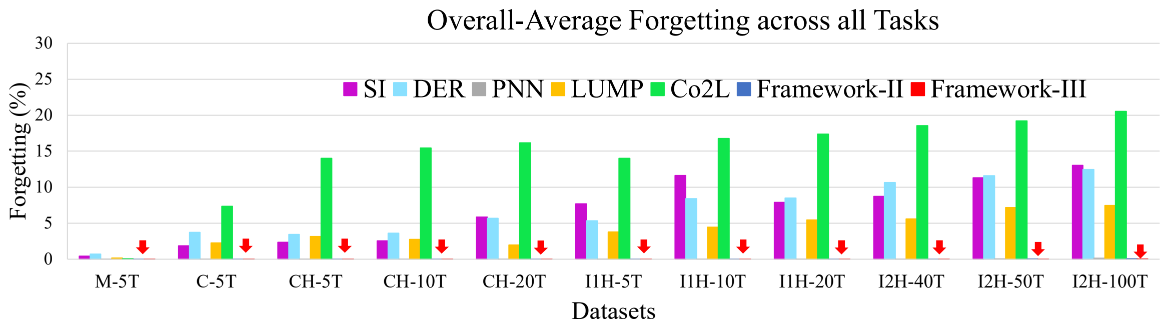


Figure 6.6: Comparison of overall-average Forgetting for Framework-III and baselines on benchmark datasets across all tasks. Additionally, in Framework-III, almost zero CF is represented by a vertical red arrow.

addressing the **RQ7** [§3.5]). The remarkable reduction in forgetting compared to DER and LUMP with long and short task-sequences suggests that a replay buffer may no longer be necessary to address CF (consequently, again confirming the **RQ1** [§3.5] in case of Framework-III as well).

6.2.5 Ablation study

In this section, we evaluate the impact of different components in Framework-III by performing ablation experiments on benchmark datasets to quantify their contributions to overall performance. Tables 6.8 and 6.9, show the ablation results for the variants of the proposed Framework-III on benchmark datasets.

6.2.5.1 Effect of hierarchical prototypical cross-level discrimination

In Table 6.8, the column header lists the variants of Framework-III, while the last row displays their overall-average accuracy. The table demonstrates that, from right to left, each successive variant of Framework-III shows gradual improvements in accuracy while maintaining minimal forgetting in UCL setup. This progression highlights the effectiveness of hierarchical prototypical CLD in the proposed solution. The incremental enhancements across variants of Framework-III have led to better performance and a reduction in CF compared to earlier versions without this feature. From Table 6.8, it is evident that Framework-III shows significant performance drops with each component removed or replaced: a 1.34% decrease when hierarchical prototypical CLD is replaced by standard CLD, a 2% drop when IB masking is substituted with hard attention masking, a 2.93% decrease when the SimSiam loss function is replaced by NPID, and a 5.64% decrease when pseudo-OOD loss is removed while retaining hard attention masking. The accuracy further decreases by 8.04% when hard attention masking is removed but the pseudo-OOD loss is present. Overall, the model experiences a total decrease of 8.64% in accuracy when specific components like attention masking, OOD, and CLD capabilities are removed. This highlights the impact of hierarchical prototypical cross-level discrimination and again addresses the **RQ6** [§3.5].

6.2.5.2 Effect of information bottleneck subnetwork masking

Table 6.9 from right to left shows a consistent reduction in overall-average forgetting across successive variants of Framework-III. This indicates the effect of IB subnetwork masking on the model's performance. The last row shows that without IB masking or hard attention masking, and pseudo-OOD capability, the model exhibits significant forgetting (8.34%). With the inclusion of OOD capability, forgetting slightly decreases to 5.54%, and with the addition of hard attention, it further reduces. However, hard attention leads to capacity saturation, which is addressed by IB masking. IB masking not only minimizes forgetting but also enhances accuracy. These findings indicate that the IB subnetwork masking mechanism is effective in UCL, consistently performing well across task counts ranging from 5 to 100 and class counts from 2 to 20. For long task sequences, the IB masking subnetwork effectively preserves past experiences without encountering capacity saturation issues, resulting in improved accuracy and minimal forgetting compared to hard attention masking. Framework-III, which incorporates IB subnetwork masking, hierarchical proto-

Table 6.8: Overall-average Accuracy (in %) of successive variants of Framework-III on benchmark datasets across all tasks.

Methods→	Framework-III	w	CLD	H.Att.	NPID	-	OOD	-
Datasets↓		w/o	Hi-Pro	IBM	SimS.	OOD	H.Att.	OOD
M-5T	**99.92**$_{\pm 0.02}$		99.84$_{\pm 0.05}$	99.79$_{\pm 0.07}$	99.68$_{\pm 0.08}$	99.66$_{\pm 0.07}$	98.83$_{\pm 0.13}$	98.48$_{\pm 0.26}$
C-5T	**94.58**$_{\pm 0.12}$		94.37$_{\pm 0.18}$	94.28$_{\pm 0.15}$	94.12$_{\pm 0.23}$	91.01$_{\pm 0.78}$	87.57$_{\pm 1.54}$	87.06$_{\pm 1.92}$
CH-5T	**66.98**$_{\pm 0.24}$		65.83$_{\pm 0.46}$	65.39$_{\pm 0.59}$	65.30$_{\pm 0.64}$	59.13$_{\pm 1.14}$	58.20$_{\pm 1.26}$	57.56$_{\pm 2.04}$
CH-10T	**76.36**$_{\pm 0.35}$		75.07$_{\pm 0.53}$	74.22$_{\pm 0.84}$	73.63$_{\pm 1.19}$	69.61$_{\pm 1.81}$	65.83$_{\pm 2.27}$	64.96$_{\pm 2.46}$
CH-20T	**83.98**$_{\pm 0.29}$		81.93$_{\pm 0.59}$	81.03$_{\pm 0.93}$	80.45$_{\pm 1.30}$	77.81$_{\pm 1.98}$	72.86$_{\pm 2.38}$	71.97$_{\pm 2.63}$
I1H-5T	**59.20**$_{\pm 0.21}$		57.68$_{\pm 0.52}$	56.81$_{\pm 0.87}$	56.38$_{\pm 1.15}$	53.17$_{\pm 1.56}$	53.04$_{\pm 2.09}$	52.28$_{\pm 2.31}$
I1H-10T	**69.46**$_{\pm 0.42}$		68.29$_{\pm 0.69}$	67.44$_{\pm 0.95}$	65.40$_{\pm 1.29}$	61.29$_{\pm 1.74}$	59.79$_{\pm 2.23}$	57.38$_{\pm 2.54}$
I1H-20T	**78.68**$_{\pm 0.58}$		77.26$_{\pm 0.76}$	76.52$_{\pm 1.07}$	75.16$_{\pm 1.41}$	74.11$_{\pm 1.88}$	71.33$_{\pm 2.35}$	69.36$_{\pm 2.87}$
I2H-40T	**78.27**$_{\pm 0.67}$		76.13$_{\pm 0.81}$	74.80$_{\pm 1.19}$	73.47$_{\pm 1.59}$	71.12$_{\pm 2.11}$	67.75$_{\pm 2.41}$	68.04$_{\pm 2.73}$
I2H-50T	**80.19**$_{\pm 0.62}$		77.83$_{\pm 0.85}$	77.22$_{\pm 1.26}$	74.59$_{\pm 1.44}$	72.81$_{\pm 2.24}$	69.89$_{\pm 2.69}$	70.84$_{\pm 3.07}$
I2H-100T	**89.37**$_{\pm 0.74}$		88.04$_{\pm 0.96}$	87.50$_{\pm 1.37}$	86.61$_{\pm 1.52}$	85.22$_{\pm 2.40}$	83.46$_{\pm 2.87}$	84.04$_{\pm 3.19}$
Average	**79.73**		78.39	77.73	76.8	74.09	71.69	71.09

typical cross-level discrimination, and a multi-task loss function, outperforms its variants by nearly achieving zero forgetting. The incremental improvements, especially the inclusion of all these capabilities, drive performance gains, reaffirming **RQ7** [§3.5].

6.2.5.3 Capacity for extended task sequence

One of the key challenges in parameter isolation methods is the limited network capacity, which leads to the saturation of free parameters as additional tasks are introduced. This saturation significantly reduces the network's ability to effectively learn new tasks. To evaluate the effectiveness of our method in addressing this challenge, we performed experiments on extended task sequences ranging from 5 to 100 on CIFAR-100 (CH) and Tiny-ImageNet (I1H & I2H). The quantitative results are shown in Tables 6.6, 6.7, 6.8, 6.9, 6.10 and Figures 6.3, 6.4, 6.5, 6.6. Specifically, Figure 6.5 shows the exclusive performance comparison for Framework-III and baselines on variants of Tiny-ImageNet (I1H & I2H) datasets across tasks ranging from 5 to 100. The results demonstrate that Framework-III consistently outperforms other methods while avoiding the challenge of limited capacity. This is achieved through the reduction of redundancy within sub-networks, which preserves additional capacity for future tasks. This provides Framework-III with an increased capacity to learn new tasks, resulting in better overall performance.

6.2.5.4 Impact of reinitialization of variational parameters and feature factorization in information bottleneck masking

Framework-III has two main components: reinitializing variational parameters to transfer valuable knowledge and feature factorization (FF) to adjust compression ratios. A component-wise analysis is conducted by gradually integrating each component into a baseline model that initially excludes feature factorization and reinitialization, using ResNet-18 as the backbone. The experimental findings are presented in Table 6.10. By comparing re-

Table 6.9: Overall-average Forgetting (in %) of successive variants of Framework-III on benchmark datasets across all tasks.

Methods$\rightarrow$ **Datasets** $\downarrow$	**Framework-III**	**w**	**CLD**	**H.Att.**	**NPID**	**-**	**OOD**	**-**
		w/o	**Hi-Pro**	**IBM**	**SimS.**	**OOD**	**H.Att.**	**OOD**
M-5T	$\mathbf{0.0004}_{\pm 0.0002}$		$0.0008_{\pm 0.0004}$	$0.0012_{\pm 0.0008}$	$0.001_{\pm 0.0003}$	$0.01_{\pm 0.005}$	$0.18_{\pm 0.04}$	$0.74_{\pm 0.15}$
C-5T	$\mathbf{0.0003}_{\pm 0.0001}$		$0.001_{\pm 0.0006}$	$0.002_{\pm 0.001}$	$0.002_{\pm 0.0015}$	$0.013_{\pm 0.004}$	$0.5_{\pm 0.02}$	$7.01_{\pm 2.17}$
CH-5T	$\mathbf{0.0005}_{\pm 0.0003}$		$0.0009_{\pm 0.0003}$	$0.002_{\pm 0.0005}$	$0.001_{\pm 0.0005}$	$0.02_{\pm 0.006}$	$0.42_{\pm 0.11}$	$8.22_{\pm 2.58}$
CH-10T	$\mathbf{0.0003}_{\pm 0.0002}$		$0.0001_{\pm 0.00004}$	$0.001_{\pm 0.0003}$	$0.001_{\pm 0.0004}$	$0.002_{\pm 0.0003}$	$0.78_{\pm 0.17}$	$10.34_{\pm 2.93}$
CH-20T	$\mathbf{0.0005}_{\pm 0.0003}$		$0.005_{\pm 0.002}$	$0.009_{\pm 0.003}$	$0.029_{\pm 0.011}$	$0.048_{\pm 0.015}$	$14.6_{\pm 2.39}$	$12.03_{\pm 3.21}$
I1H-5T	$\mathbf{0.0003}_{\pm 0.0002}$		$0.002_{\pm 0.001}$	$0.006_{\pm 0.002}$	$0.002_{\pm 0.001}$	$0.003_{\pm 0.001}$	$0.39_{\pm 0.16}$	$6.05_{\pm 1.76}$
I1H-10T	$\mathbf{0.00005}_{\pm 0.00002}$		$0.0003_{\pm 0.0001}$	$0.0005_{\pm 0.0001}$	$0.001_{\pm 0.0007}$	$0.002_{\pm 0.0008}$	$0.74_{\pm 0.25}$	$7.79_{\pm 2.49}$
I1H-20T	$\mathbf{0.001}_{\pm 0.0005}$		$0.012_{\pm 0.006}$	$0.058_{\pm 0.014}$	$0.06_{\pm 0.02}$	$0.065_{\pm 0.026}$	$9.91_{\pm 3.13}$	$8.63_{\pm 2.97}$
I2H-40T	$\mathbf{0.003}_{\pm 0.001}$		$0.007_{\pm 0.003}$	$0.069581_{\pm 0.026}$	$0.084_{\pm 0.017}$	$0.083_{\pm 0.031}$	$10.1_{\pm 3.35}$	$9.32_{\pm 3.17}$
I2H-50T	$\mathbf{0.018}_{\pm 0.006}$		$0.031_{\pm 0.019}$	$0.096173_{\pm 0.033}$	$0.097_{\pm 0.022}$	$0.11963_{\pm 0.052}$	$11.37_{\pm 3.62}$	$10.59_{\pm 3.35}$
I2H-100T	$\mathbf{0.042}_{\pm 0.012}$		$0.057_{\pm 0.024}$	$0.1911_{\pm 0.07}$	$0.1482_{\pm 0.034}$	$0.233_{\pm 0.09}$	$11.92_{\pm 3.89}$	$11.06_{\pm 3.79}$
Average	**0.006**		0.011	0.040	0.039	0.054	5.54	8.34

Table 6.10: The quantitative results of the ablation studies on feature factorization and reinitialization in Framework-III, using ResNet-18 as the backbone, are presented. The terms "Selected" and "Unselected" refer to the subsets of variational parameters identified by sub-network masks as either included or excluded, respectively.

Variant	**FF**	**Reinitialization**		**CH-20T**		**I1H-20T**	
		Selected	Unselected	Accuracy (%)	Forgetting (%)	Accuracy (%)	Forgetting (%)
①	✗	✗	✗	$80.65_{\pm 0.36}$	$1.39_{\pm 0.48}$	$76.53_{\pm 0.71}$	$1.57_{\pm 0.54}$
②	✓	✗	✗	$81.47_{\pm 0.52}$	$0.12_{\pm 0.06}$	$77.16_{\pm 0.63}$	$0.14_{\pm 0.08}$
③	✓	✗	✓	$83.98_{\pm 0.29}$	$0.0005_{\pm 0.0003}$	$78.68_{\pm 0.58}$	$0.001_{\pm 0.0005}$
④	✓	✓	✓	$80.39_{\pm 0.64}$	$0.43_{\pm 0.25}$	$75.84_{\pm 0.79}$	$0.55_{\pm 0.32}$

sults from ① and ②, the inclusion of feature factorization leads to a notable improvement in average accuracy (e.g., in CH-20T: 0.82% and 1H-20T: 0.63%). Feature factorization identifies the optimal ratios for each layer and adjusts them during training. The results show that the framework dynamically sets these ratios while retaining past knowledge, reducing forgetting by 1.27% for CH-20T and 1.44% for 1H-20T. Furthermore, comparing ② and ③, it is evident that training with reinitialization results in a significant boost in average accuracy (e.g., in CH-20T: 2.51% and 1H-20T: 1.52%) and a decrease in forgetting (e.g., CH-20T: 0.12% to 0.0005%, 1H-20T: 0.14% to 0.001%). This indicates that reinitialization effectively captures important knowledge while minimizing the adverse effects of redundant information. Finally, reinitializing the key parameters (i.e., ③ and ④) leads to the poorest performance on both datasets. This happens because resetting important parameters causes a loss of previously acquired knowledge.

6.2.5.5 Analysis of the number of model parameters

For all experiments, ResNet-18 serves as the backbone architecture. For CIFAR100 (CH) and Tiny-ImageNet (I1H & I2H), we utilize the same ResNet-18 structure as used for M-5T and C-5T, with adjustments made to learn the higher complexity of the data. To increase the model's capacity for handling multiple tasks and learning richer data distributions, we specifically double the number of channels in each convolutional layer.

Table 6.11: Number of network parameters in Framework-III after learning the final task.

Methods → **Datasets ↓**	**# Without IBM**	**# With IBM**	**Difference (%)**
M-5T	69933632	69991552	0.00083
C-5T	69933632	69991552	0.00083
CH-5T	109643392	109759232	0.0011
CH-10T	174634112	174884992	0.0014
CH-20T	304615552	305136512	0.0017
I1H-5T	109643392	109759232	0.0011
I1H-10T	174634112	174884992	0.0014
I1H-20T	304615552	305136512	0.0017
I2H-40T	564578432	565639552	0.00188
I2H-50T	694559872	695891072	0.00191
I2H-100T	1344467072	1347148672	0.00199

Framework-III requires task-specific parameters for each task to integrate information bottleneck masking and task-specific heads. Table 6.11 presents the network parameter counts recorded after training the final task in each experiment. For M-5T, C-5T, CH-5T, CH-10T, CH-20T, I1H-5T, I1H-10T, I1H-20T, I2H-40T, I2H-50T, and I2H-100T, task-specific parameters are incorporated. This results in the following percentage differences in parameter count after each task for the respective dataset: 0.00083%, 0.00083%, 0.0011%, 0.0014%, 0.0017%, 0.0011%, 0.0014%, 0.0017%, 0.00188%, 0.00191%, and 0.00199%. The slight increases in parameter count help to maintain minimal growth in model complexity while supporting the inclusion of new tasks. Contrastive learning adds task-specific parameters through projection functions, but these parameters can be discarded during deployment since they are not needed for inference.

6.2.6 Statistical analysis of the results

To ensure the reliability of our results, we performed three independent runs. As the experiments are conducted on standard datasets with fixed and predefined test sets, no additional dataset splitting is required. The evaluation metrics defined in Section 2.4 are used for statistical analysis.

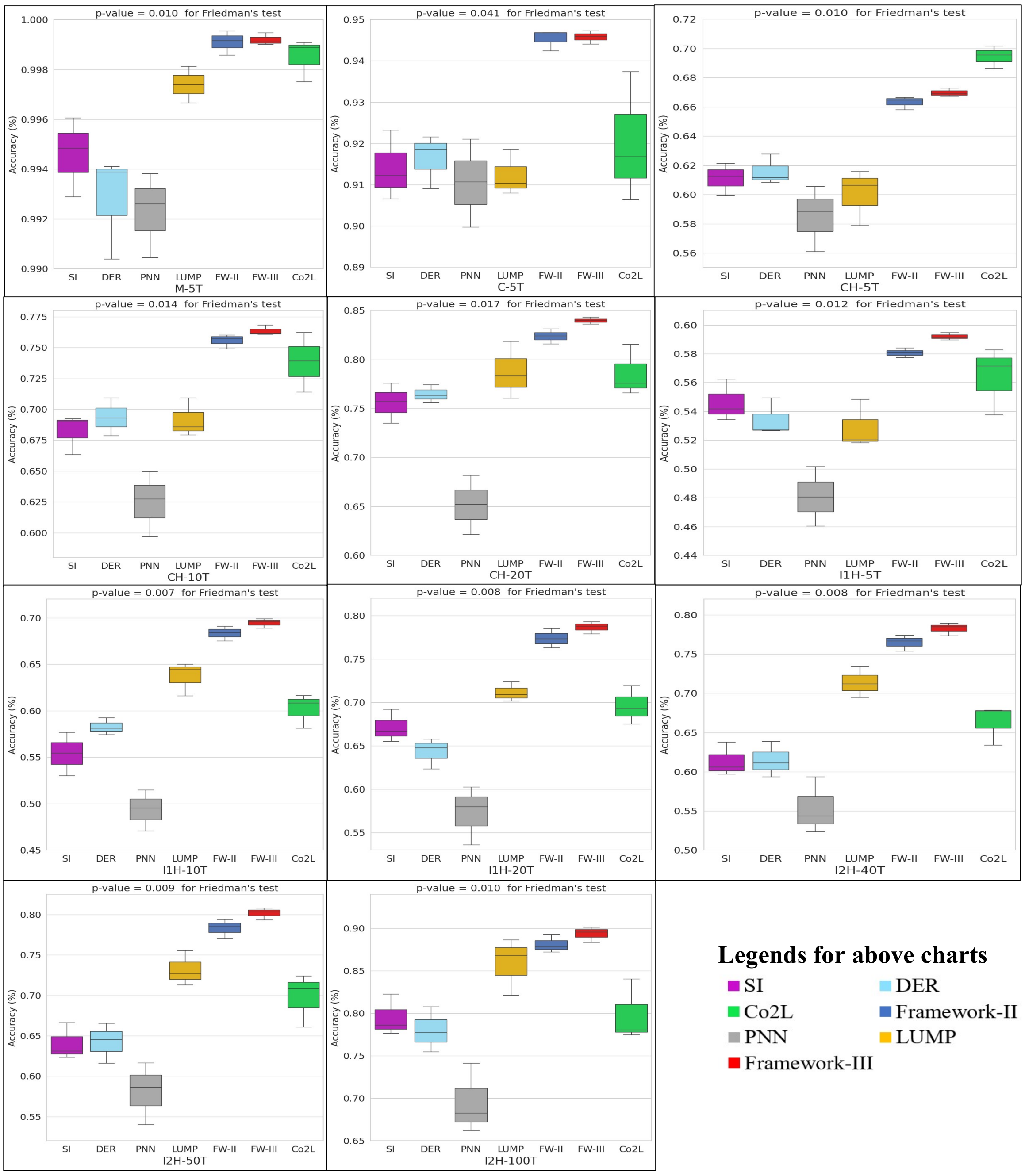


Figure 6.7: Box-plots representing Friedman's test analysis for Framework-III and baselines on benchmark datasets across all tasks.

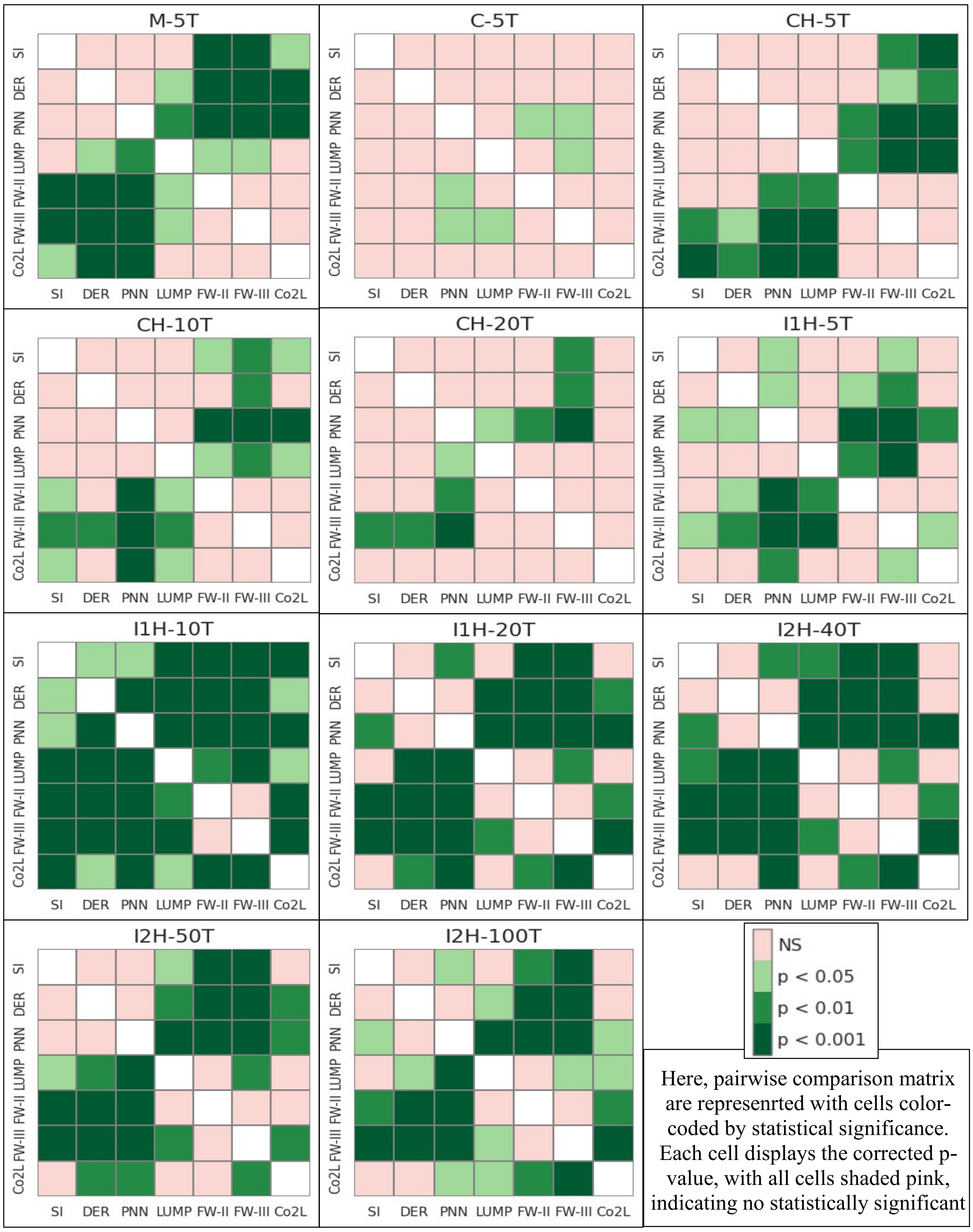
6.2.6.1 Using Friedman's test to compare results

ANOVA [147], or analysis of variance, evaluates statistical significance by comparing group mean differences but relies on assumptions like normality, variance homogeneity, and data independence. For cases where these assumptions are not met, Friedman's test [148] serves as a non-parametric alternative. This test reduces data to rank orders and is particularly useful for dependent or non-normally distributed data. In our analysis, we compared the overall-average accuracy for all baseline methods across three independent runs. Methods are rank-ordered, and Figure 6.7 presents boxplots illustrating overall-average accuracy comparisons for the baseline methods. Above each plot, the p-value from Friedman's test is shown, demonstrating $p < 0.05$ in all subplots, indicating statistically significant differences among the baseline methods.

6.2.6.2 Using post-hoc test to compare results

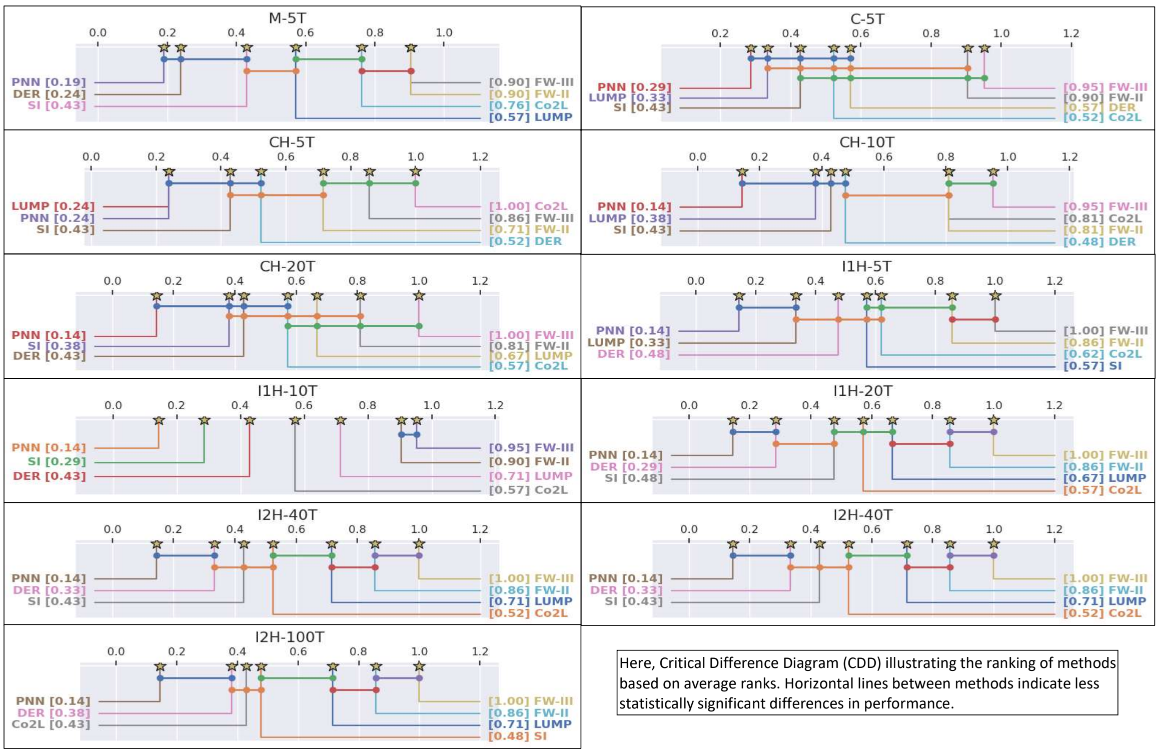
Friedman's test (discussed in §6.2.6.1) determines whether there are significant differences between the methods, but does not indicate where those differences occur. To identify specific differences, post-hoc pairwise tests are conducted. Since comparing multiple methods increases the likelihood of finding a significant difference by chance, the significance threshold is often adjusted using corrections such as the Bonferroni method [149]. This method divides the original significance threshold by the number of comparisons. While some argue that this correction is too strict, it is still widely used. Both tests serve distinct purposes: Friedman's test assesses the overall differences, while post-hoc tests identify specific pairwise differences.

In Figure 6.8, we perform post-hoc tests using the pairwise Conover-Friedman test [150, 151]. The plot shows the significance of the corrected p-values, which indicate the differences between the methods listed in rows and columns. The diagonal cells, represent-



Here, pairwise comparison matrix are represenrted with cells color-coded by statistical significance. Each cell displays the corrected p-value, with all cells shaded pink, indicating no statistically significant

Figure 6.8: Significance-plots representing pairwise Conover-Friedman test Analysis for Framework-III and baselines on benchmark datasets across all tasks.



Here, Critical Difference Diagram (CDD) illustrating the ranking of methods based on average ranks. Horizontal lines between methods indicate less statistically significant differences in performance.

Figure 6.9: Critical difference diagram to rank the Framework-III and baselines on benchmark datasets across all tasks.

ing comparisons of each method with itself, are white. The intensity of the color in the cells reflects the level of statistical significance. Corrected p-values greater than 0.05 (NS = not significant) are shown in pink, while significant p-values appear in progressively darker shades of green. As anticipated from the results of Friedman's test, differences between the baseline methods are observed in Figure 6.8.

6.2.6.3 Using critical difference diagram to compare results

An alternative method for comparing multiple approaches is the Critical Difference Diagram (CDD) [151]. In CDD, methods connected by a horizontal line indicate that their distributions are not significantly different. The vertical line in the diagram represents the mean rank of each method in independent runs. Figure 6.9 shows the CDD diagram for the baseline methods. In all subplots, Framework-III consistently ranks higher than the other methods. Additionally, the horizontal lines between Framework-III and Framework-II suggest a less significant difference between them. However, Framework-III outperforms Framework-II in terms of rank, indicating its superiority.

6.3 Discussion

The primary objective of our research is to tackle CF by using non-redundant sub-networks and learning multiple semantic hierarchies inherent in the dataset. Similar to LUMP [11], we initially explored an unsupervised approach that utilizes an instance-level contrastive loss. However, this method excessively emphasized instance discrimination, leading to weaker group identification. To improve this, we further experimented to integrate both instance-level and group-level discrimination within a unified contrastive loss function, addressing the limitations of each approach when used alone. Despite this enhancement, the approach still lacked the representational capability needed to effectively handle datasets with multiple semantic hierarchies (see §6.2.5). Therefore, we incorporated cross-level discrimination of instances and hierarchical prototypes within the contrastive loss function. The details of this method and its benefits are provided in §6.1.1.2. Similarly, Framework-II uses hard attention and gradient projection to tackle CF in UCL. However, while it helps in capacity saturation issues associated with hard attention, it does not completely resolve them. We observe that as the density of the hard attention mask increases for the longer task sequences such as 40, 50 or 100, the performance gains by the Framework-II method start to decline. This is because denser masks lead to more sparse overlap, reducing the number of free parameters available for updating new tasks. Therefore, to address the aforementioned capacity saturation problem, we integrated the multitasking loss function with IB subnetwork masking. Additionally, Framework-II struggles to effectively learn hierarchical structures present in the data, a challenge that Framework-III successfully addresses. Despite using prototypical cross-level discrimination to address limitations of IB subnetwork masking, we found that random initialization in Framework-III is inadequate for supporting UCL in a class incremental setting, as it hinders effective representation learning in the network. The IB subnetwork masking struggles to balance the trade-off between compressing the input and preserving meaningful patterns, particularly in the absence of labels, making it difficult to define relevant information. Additionally, without label guidance in IB masking, there is a risk of overcompressing the input data, potentially leading to the loss of crucial latent structures and diminishing the model's overall performance in class incremental learning.

Chapter 7

Conclusion and Future Works

CL addresses the key challenge of CF in traditional models, enabling sequential learning without retraining from scratch. Initial strategies to address CF focused on replay-based methods, which reuse past data during retraining. However, concerns about efficiency, capacity, and privacy have led to the development of memory-free alternatives. This motivated the creation of a novel framework that achieves near-zero forgetting without relying on a replay buffer. Regularization-based methods limit changes to important parameters, but still struggle to fully prevent CF. Architecture-based approaches allocate separate parameters per task but lead to linear resource growth by underutilizing network sparsity. Parameter isolation stands out as an effective strategy, assigning distinct sub-networks to tasks within a shared structure using masking. However, most CL research, including parameter isolation, focuses on supervised settings, which limits its applicability in real-world scenarios involving unlabeled or biased data. To bridge this gap, we explore its application in UCL, demonstrating that effective representation learning is achievable without relying on annotations.

This thesis introduces three progressively improved parameter isolation-based approaches for UCL, each improving upon the previous to mitigate CF. Together, they offer replay-free, architecture-preserving solutions with minimal forgetting in an unsupervised setting.

In Chapter 4, we proposed Framework-I that combines hard attention masking with cross-level instance-to-group discrimination to preserve previous task representations and enable task differentiation without labels. Hard attention conditions feature extractor layers on task-specific information to mitigate forgetting. To support unsupervised learning, we use a contrastive loss function, but address its sensitivity to imbalanced positive/negative sample ratios by combining instance-level and group-level discrimination. This joint mechanism improves both fine-grained instance separation and semantic grouping, making it effective for TIL. However, in CIL, where task boundaries are implicit, this method may misclassify OOD instances as similar in-distribution samples, increasing vulnerability to adversarial errors. To address this, we introduce pseudo-OOD detection by applying rotation-based augmentations and training the model to perform cross-level pseudo-group discrimination. This strategy allows the model to assign high scores to in-distribution samples while suppressing out-of-distribution ones, enabling accurate task boundary detection and improved model performance in the absence of labels.

Framework-I performs well against SOTA methods but suffers from capacity saturation. Independent use of hard attention overly constrains parameter updates, limiting learning for new tasks and increasing forgetting. To address this, Chapter 5 introduces Framework-II, which incorporates gradient projection to guide parameter updates in directions orthogonal to the gradient subspaces of previous tasks. This balances hard attention and gradient constraints, mitigating their individual drawbacks. Additionally, Framework-II enhances pseudo-OOD detection using evidential deep learning, combining cross-entropy with evidential KL-divergence on rotation-augmented pseudo labels to better quantify uncertainty. This improves both TIL and CIL accuracy by enabling the model to learn more discriminative and descriptive features to distinguish ID from OOD data.

Relying solely on weight magnitude for hard attention can retain irrelevant weights and redundant subnetworks, leading to capacity saturation. To address this, Chapter 6 presents a subnetwork masking based on IB theory, which reduces redundancy by preserving essential parameters and directing irrelevant information to expendable ones. This optimizes inter-layer mutual information for constructing efficient, redundancy-free subnetworks. Although IB-based masking has shown promise in SCL, its application in UCL is challenging due to the lack of labels, increasing the risk of overcompression and loss of vital features. To overcome this, we emphasize grouping instances over individual discrimination, as individual-level contrast can reduce both stability and effectiveness. However, clustering alone can misgroup dissimilar instances due to shared repulsion. Thus, we guide instances toward their clusters while repelling unrelated ones. We enhance this with hierarchical prototypical contrastive learning, where each image is compared not just to one, but to multiple clusters, treated as positive and negative samples accordingly. These hierarchical prototypes, dynamically updated during training, capture deeper semantic relationships. The combined use of IB subnetwork masking and hierarchical cross-level discrimination ensures retention of relevant information and mitigates overcompression, effectively balancing stability and plasticity in UCL.

Extensive experiments prove that the proposed frameworks show substantial performance gains over standard replay-based, regularization-based, and architecture-based baseline methods, highlighting the effectiveness of the parameter isolation-based approach in the UCL setup. It also bridges the gap in performance and forgetting rates between SCL and UCL.

7.1 Future Scope of the Work

Although the proposed parameter isolation-based frameworks for UCL demonstrate good performance and scalability, several avenues remain open for further exploration and improvement:

☞ **Addressing Network Capacity Saturation**

- Although the proposed frameworks have been validated on up to 100 tasks using hard attention mechanisms, network capacity tends to saturate as task count increases.
- This saturation restricts the model's ability to learn new tasks effectively.
- **Future Work:** Investigate advanced and complementary strategies (e.g., dynamic sparsification, low-rank adaptation, or modular routing) to mitigate capacity limitations without compromising performance.

☞ **Enhancing IB Techniques**

- Existing IB-based UCL methods i.e. Framework-III, often rely on variational approximation and distributional assumptions to estimate the lower bound of mutual information, making them computationally expensive and less effective in hierarchical or long-tailed datasets.
- **Future Work:**
  - Explore differentiable and assumption-free IB formulations (e.g., [152, 153]) to reduce computational overhead and improve performance on long sequence datasets.
  - Investigate task-aware or dynamic IB masking strategies that adjust information compression based on task difficulty or novelty.

☞ **Enabling both Backward and Forward Knowledge Transfer**

- This study aims to mitigate CF by limiting updates to old task parameters, though this may restrict knowledge transfer from new to old tasks, potentially hindering overall learning improvement.
- **Future Work:**
  - To address this, a strategy like CUBER [154] can be used, which first identifies positively correlated old tasks at the layer level, then selectively updates parameters by projecting gradients onto their input subspaces during new task learning.
  - Explore adaptive parameter-sharing mechanisms or meta-regularization strategies that selectively allow beneficial updates.
  - Incorporating unsupervised knowledge distillation or mutual information alignment between tasks could promote bidirectional transfer while still preserving old task knowledge.

☞ **Exploring Task-agnostic and Meta-learning Strategies**

- The current methods are task-specific in their masking and isolation strategies.
- **Future Work:**
  - Investigate task-agnostic UCL approaches that do not rely on clear task boundaries.
  - Integrate meta-learning to enable fast adaptation to new tasks with minimal forgetting.

☞ **Extending to Diverse CL Settings**

- Current validation has been primarily conducted in offline and class-balanced CL setups.
- **Future Work:**
  - Evaluate performance in online CL, where data arrives sequentially and decisions must be made in real-time.
  - Extend experiments to class-imbalanced and few-shot CL scenarios, which are common in real-world applications but challenging for most CL algorithms.

☞ **Generalizing Beyond Classification**

- The current focus of this work is on classification tasks.
- **Future Work:** Adapt and validate the proposed approaches in other domains such as:
  - Object Detection (e.g., [155])
  - Semantic Segmentation (e.g., [156])
  - Multi-modal CL, combining vision, language, and audio modalities (e.g., [157, 158])

These future directions aim to broaden the applicability and robustness of parameter isolation-based UCL methods, moving closer to truly scalable, generalizable, and efficient lifelong learning systems.

Bibliography

[1] 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.

[2] A. Malviya and C. K. Maurya, "Unsupervised continual learning using cross-level discrimination and evidential pseudo out-of-distribution detection along with gradient projected hard attention," *IEEE Access*, vol. 12, pp. 171 143–171 165, 2024.

[3] L. Wang *et al.*, "A comprehensive survey of continual learning: Theory, method and application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 8, pp. 5362–5383, 2024.

[4] M. B. Gurbuz and C. Dovrolis, "Nispa: Neuro-inspired stability-plasticity adaptation for continual learning in sparse networks," in *Proc. International Conference on Machine Learning*, vol. 162, 2022, pp. 8157–8174.

[5] M. De Lange, G. van de Ven, and T. Tuytelaars, "Continual evaluation for lifelong learning: Identifying the stability gap," in *Proc. International Conference on Learning Representations*, 2022.

[6] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Proc. Neural Information Processing Systems*, vol. 30, 2017.

[7] A. Chaudhry *et al.*, “Efficient lifelong learning with a-GEM,” in *Proc. International Conference on Learning Representations*, 2019.

[8] J. Pomponi, S. Scardapane, and A. Uncini, “Pseudo-rehearsal for continual learning with normalizing flows,” in *Proc. 4th Lifelong Machine Learning Workshop at ICML 2020*, 2020.

[9] A. A. Rusu *et al.*, “Progressive neural networks,” in *Proc. NIPS 2016 Deep Learning Symposium Recommendation*, 2016.

[10] J. Yoon *et al.*, “Lifelong learning with dynamically expandable networks,” in *Proc. International Conference on Learning Representations*, 2018.

[11] D. Madaan *et al.*, “Representational continuity for unsupervised continual learning,” in *Proc. International Conference on Learning Representations*, 2022.

[12] H. Cha, J. Lee, and J. Shin, “Co2l: Contrastive continual learning,” in *Proc. International Conference on Computer Vision*, 2021, pp. 9516–9525.

[13] Z. Li and D. Hoiem, “Learning without forgetting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[14] P. Buzzega *et al.*, “Dark experience for general continual learning: a strong, simple baseline,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 15 920–15 930.

[15] J. Kirkpatrick *et al.*, “Overcoming catastrophic forgetting in neural networks,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[16] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. International Conference on Machine Learning*, 2017, pp. 3987–3995.

[17] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proc. Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[18] J. Serra *et al.*, "Overcoming catastrophic forgetting with hard attention to the task," in *Proc. International Conference on Machine Learning*, 2018, pp. 4548–4557.

[19] M. Farajtabar *et al.*, "Orthogonal gradient descent for continual learning," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2020, pp. 3762–3773.

[20] H. Ahn *et al.*, "Uncertainty-based continual learning with adaptive regularization," in *Proc. Neural Information Processing Systems*, vol. 32, 2019.

[21] H. Li, J. Wu, and V. Braverman, "Fixed design analysis of regularization-based continual learning," in *Proc. 2nd Conference on Lifelong Learning Agents*, ser. Proceedings of Machine Learning Research, vol. 232, 2023, pp. 513–533.

[22] M. Wortsman *et al.*, "Supermasks in superposition," in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 15 173–15 184.

[23] Z. Ke, B. Liu, and X. Huang, "Continual learning of a mixed sequence of similar and dissimilar tasks," in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 18 493–18 504.

[24] Y. Fan *et al.*, "Neural sparse representation for image restoration," in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 15 394–15 404.

[25] S. Han *et al.*, “Learning both weights and connections for efficient neural network,” in *Proc. Neural Information Processing Systems*, vol. 28, 2015.

[26] K. Friston, “Hierarchical models in the brain,” *PLoS computational biology*, vol. 4, no. 11, p. e1000211, 2008.

[27] B. Babadi and H. Sompolinsky, “Sparseness and expansion in sensory representations,” *Neuron*, vol. 83, no. 5, pp. 1213–1226, 2014.

[28] T. Chen *et al.*, “A simple framework for contrastive learning of visual representations,” in *Proc. International Conference on Machine Learning*, 2020, pp. 1597–1607.

[29] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” in *Proc. Neural Information Processing Systems*, 2018.

[30] K. He *et al.*, “Momentum contrast for unsupervised visual representation learning,” in *Proc. Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[31] M. Caron *et al.*, “Unsupervised learning of visual features by contrasting cluster assignments,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 9912–9924.

[32] J. Li *et al.*, “Prototypical contrastive learning of unsupervised representations,” in *Proc. International Conference on Learning Representations*, 2021.

[33] X. Wang, Z. Liu, and S. X. Yu, “Unsupervised feature learning by cross-level instance-group discrimination,” in *Proc. Computer Vision and Pattern Recognition*, June 2021, pp. 12 586–12 595.

[34] G. Kim *et al.*, “A theoretical study on solving continual learning,” in *Proc. Neural Information Processing Systems*, vol. 35, 2022, pp. 5065–5079.

[35] X. Luo, J. Xu, and Z. Xu, “Channel importance matters in few-shot image classification,” in *Proc. International Conference on Machine Learning*, 2022, pp. 14 542–14 559.

[36] G. Saha, I. Garg, and K. Roy, “Gradient projection memory for continual learning,” in *Proc. International Conference on Learning Representations*, 2021.

[37] M. Sensoy, L. Kaplan, and M. Kandemir, “Evidential deep learning to quantify classification uncertainty,” in *Proc. Neural Information Processing Systems*, vol. 31, 2018.

[38] B. Dai *et al.*, “Compressing neural networks using the variational information bottleneck,” in *Proc. International Conference on Machine Learning*, 2018, pp. 1135–1144.

[39] N. Tishby and N. Zaslavsky, “Deep learning and the information bottleneck principle,” in *Proc. IEEE Information Theory Workshop*, 2015, pp. 1–5.

[40] C. Chen *et al.*, “Towards redundancy-free sub-networks in continual learning,” *Pattern Recognition*, vol. 171, p. 112020, 2026.

[41] T. Deselaers and V. Ferrari, “Visual and semantic similarity in imagenet,” in *Proc. Computer Vision and Pattern Recognition*, 2011, pp. 1777–1784.

[42] T. Wu and I. Fischer, “Phase transitions for the information bottleneck in representation learning,” in *Proc. International Conference on Learning Representations*, 2020.

[43] K. Ahuja *et al.*, “Invariance principle meets information bottleneck for out-of-distribution generalization,” in *Proc. Neural Information Processing Systems*, vol. 34, 2021, pp. 3438–3450.

[44] Z. Roozbahani, J. Rezaeenour, and A. Katanforoush, “Community detection in multi-relational directional networks,” *Journal of Computational Science*, vol. 67, p. 101962, 2023.

[45] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *Proc. Computer Vision and Pattern Recognition*, 2015, pp. 427–436.

[46] D. Rao *et al.*, “Continual unsupervised representation learning,” in *Proc. Neural Information Processing Systems*, vol. 32, 2019.

[47] Z. Lin, Y. Wang, and H. Lin, “Continual contrastive learning for image classification,” in *Proc. IEEE International Conference on Multimedia and Expo*, 2022, pp. 1–6.

[48] A. Gomez-Villa *et al.*, “Continually learning self-supervised representations with projected functional regularization,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 3867–3877.

[49] Y. LeCun *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[50] Krizhevsky, “Learning multiple layers of features from tiny images,” 2009.

[51] Y. Le and X. Yang, “Tiny imagenet visual recognition challenge,” *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[52] Z. Wu *et al.*, “Unsupervised feature learning via non-parametric instance discrimination,” in *Proc. Computer Vision and Pattern Recognition*, 2018.

[53] X. Chen and K. He, “Exploring simple siamese representation learning,” in *Proc. Computer Vision and Pattern Recognition*, 2020, pp. 15 745–15 753.

[54] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proc. Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[55] A. Robins, “Catastrophic forgetting, rehearsal and pseudorehearsal,” *Connection Science*, vol. 7, no. 2, pp. 123–146, 1995.

[56] A. Chaudhry *et al.*, “Efficient lifelong learning with a-gem,” in *Proc. International Conference on Learning Representations*, 2019.

[57] R. Tiwari *et al.*, “Gcr: Gradient coreset based replay buffer selection for continual learning,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 99–108.

[58] B. Tang and D. S. Matteson, “Graph-based continual learning,” in *Proc. International Conference on Machine Learning*, 2020.

[59] J. Bang *et al.*, “Rainbow memory: Continual learning with a memory of diverse samples,” in *Proc. Computer Vision and Pattern Recognition*, June 2021, pp. 8218–8227.

[60] P. S. Bhat, B. Zonooz, and E. Arani, “Consistency is the key to further mitigating catastrophic forgetting in continual learning,” in *Proc. 1st Conference on Lifelong Learning Agents*, ser. Proceedings of Machine Learning Research, vol. 199, 2022, pp. 1195–1212.

[61] L. Caccia *et al.*, “Reducing representation drift in online continual learning,” *arXiv preprint arXiv:2104.05025*, vol. 1, no. 3, 2021.

[62] S. Cha *et al.*, “Is continual learning truly learning representations continually,” *arXiv preprint arXiv:2206.08101*, vol. 2, no. 3, p. 4, 2022.

[63] S. Purushwalkam, P. Morgado, and A. Gupta, “The challenges of continuous self-supervised learning,” in *Proc. European Conference on Computer Vision*, 2022, pp. 702–721.

[64] R. Aljundi *et al.*, “Memory aware synapses: Learning what (not) to forget,” in *Proc. European conference on computer vision*, 2018, pp. 139–154.

[65] J. Schwarz *et al.*, “Progress & compress: A scalable framework for continual learning,” in *Proc. International Conference on Machine Learning*, 2018, pp. 4528–4537.

[66] P. Pan *et al.*, “Continual deep learning by functional regularisation of memorable past,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 4453–4464.

[67] Z. Li and D. Hoiem, “Learning without forgetting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[68] C. Simon, P. Koniusz, and M. Harandi, “On learning the geodesic path for incremental learning,” in *Proc. Computer Vision and Pattern Recognition*, 2021, pp. 1591–1600.

[69] X. Hu *et al.*, “Distilling causal effect of data in class-incremental learning,” in *Proc. Computer Vision and Pattern Recognition*, 2021, pp. 3957–3966.

[70] S. Hou *et al.*, “Learning a unified classifier incrementally via rebalancing,” in *Proc. Computer Vision and Pattern Recognition*, 2019, pp. 831–839.

[71] A. Douillard *et al.*, “Podnet: Pooled outputs distillation for small-tasks incremental learning,” in *Proc. European conference on computer vision*, 2020, pp. 86–102.

[72] H. Ahn *et al.*, “Ss-il: Separated softmax for incremental learning,” in *Proc. International Conference on Computer Vision*, 2021, pp. 844–853.

[73] Z. Ji *et al.*, “Complementary calibration: Boosting general continual learning with collaborative distillation and self-supervision,” *IEEE Transactions on Image Processing*, vol. 32, pp. 657–667, 2022.

[74] J. Li *et al.*, “Learning from students: Online contrastive distillation network for general continual learning.” in *Proc. International Joint Conferences on Artificial Intelligence*, 2022, pp. 3215–3221.

[75] Z. Ji *et al.*, “Imbalance mitigation for continual learning via knowledge decoupling and dual enhanced contrastive learning,” *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[76] A. Douillard *et al.*, “Dytox: Transformers for continual learning with dynamic token expansion,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 9285–9295.

[77] D. Abati *et al.*, “Conditional channel gated networks for task-aware continual learning,” in *Proc. Computer Vision and Pattern Recognition*, 2020, pp. 3931–3940.

[78] Q. Pham, C. Liu, and S. Hoi, “Dualnet: Continual learning, fast and slow,” in *Proc. Neural Information Processing Systems*, vol. 34, 2021, pp. 16 131–16 144.

[79] Q. Qin *et al.*, “Bns: Building network structures dynamically for continual learning,” in *Proc. Neural Information Processing Systems*, vol. 34, 2021, pp. 20 608–20 620.

[80] G. Rypeść *et al.*, "Divide and not forget: Ensemble of selectively trained experts in continual learning," in *Proc. International Conference on Machine Learning*, 2024.

[81] X. Li *et al.*, "Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting," in *Proc. International Conference on Machine Learning*, 2019, pp. 3925–3934.

[82] I. Osman, A. Eltantawy, and M. S. Shehata, "Task-based parameter isolation for foreground segmentation without catastrophic forgetting using multi-scale region and edges fusion network," *Image and Vision Computing*, vol. 113, p. 104248, 2021.

[83] T. Konishi *et al.*, "Parameter-level soft-masking for continual learning," in *Proc. International Conference on Machine Learning*, 2023, pp. 17 492–17 505.

[84] C.-Y. Hung *et al.*, "Compacting, picking and growing for unforgetting continual learning," in *Proc. Neural Information Processing Systems*, vol. 32, 2019.

[85] C. Fernando *et al.*, "Pathnet: Evolution channels gradient descent in super neural networks," *arXiv preprint arXiv:1701.08734*, 2017.

[86] G. Sokar, D. C. Mocanu, and M. Pechenizkiy, "Spacenet: Make free space for continual learning," *Neurocomputing*, vol. 439, pp. 1–11, 2021.

[87] H. Kang *et al.*, "Forget-free continual learning with winning subnetworks," in *Proc. International Conference on Machine Learning*, 2022, pp. 10 734–10 750.

[88] H. Kang *et al.*, "On the soft-subnetwork for few-shot class incremental learning," in *Proc. International Conference on Learning Representations*, 2023.

[89] G. Saha *et al.*, "Space: Structured compression and sharing of representational space for continual learning," *IEEE Access*, vol. 9, pp. 150 480–150 494, 2021.

[90] I. Garg, P. Panda, and K. Roy, “A low effort approach to structured cnn design using pca,” *IEEE Access*, vol. 8, pp. 1347–1360, 2019.

[91] G. Zeng *et al.*, “Continual learning of context-dependent processing in neural networks,” *Nature Machine Intelligence*, vol. 1, no. 8, pp. 364–372, 2019.

[92] K. Kawaguchi *et al.*, “How does information bottleneck help deep learning?” in *Proc. International Conference on Machine Learning*, 2023, pp. 16 049–16 096.

[93] O. Shamir, S. Sabato, and N. Tishby, “Learning and generalization with the information bottleneck,” *Theoretical Computer Science*, vol. 411, no. 29-30, pp. 2696–2711, 2010.

[94] J. Zbontar *et al.*, “Barlow twins: Self-supervised learning via redundancy reduction,” in *Proc. International Conference on Machine Learning*, 2021, pp. 12 310–12 320.

[95] Y. Tian *et al.*, “What makes for good views for contrastive learning?” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 6827–6839.

[96] M. Caron *et al.*, “Unsupervised learning of visual features by contrasting cluster assignments,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 9912–9924.

[97] M. Gutmann and A. Hyvärinen, “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models,” in *Proc. Thirteenth International Conference on Artificial Intelligence and statistics*, 2010, pp. 297–304.

[98] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv preprint arXiv:1807.03748*, 2018.

[99] P. Vincent *et al.*, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.” *Journal of Machine Learning Research*, vol. 11, no. 12, 2010.

[100] X. Chen *et al.*, “Context autoencoder for self-supervised representation learning,” *International Journal of Computer Vision*, vol. 132, no. 1, pp. 208–223, 2024.

[101] C. Doersch, A. Gupta, and A. A. Efros, “Unsupervised visual representation learning by context prediction,” in *Proc. IEEE international conference on computer vision*, 2015, pp. 1422–1430.

[102] D. Alexey *et al.*, “Discriminative unsupervised feature learning with exemplar convolutional neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1734–1747, 2016.

[103] M. Zhang *et al.*, “Multi-view contrastive learning for multilayer network embedding,” *Journal of Computational Science*, vol. 67, p. 101975, 2023.

[104] J. Xie, R. Girshick, and A. Farhadi, “Unsupervised deep embedding for clustering analysis,” in *Proc. International Conference on Machine Learning*, 2016, pp. 478–487.

[105] J. Yang, D. Parikh, and D. Batra, “Joint unsupervised learning of deep representations and image clusters,” in *Proc. Computer Vision and Pattern Recognition*, 2016, pp. 5147–5156.

[106] C. Zhuang, A. L. Zhai, and D. Yamins, “Local aggregation for unsupervised learning of visual embeddings,” in *Proc. Computer Vision and Pattern Recognition*, 2019, pp. 6002–6012.

[107] Y. Li *et al.*, “Contrastive clustering,” in *Proc. AAAI conference on artificial intelligence*, vol. 35, no. 10, 2021, pp. 8547–8555.

[108] Y. Guo *et al.*, “Hcsc: Hierarchical contrastive selective coding,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 9706–9715.

[109] D. Shenaj *et al.*, “Continual coarse-to-fine domain adaptation in semantic segmentation,” *Image and Vision Computing*, vol. 121, p. 104426, 2022.

[110] E. Fini *et al.*, “Self-supervised models are continual learners,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 9621–9630.

[111] M. Kilickaya and J. Vanschoren, “Are labels needed for incremental instance learning?” in *Proc. Computer Vision and Pattern Recognition*, 2023, pp. 2400–2408.

[112] J. Smith *et al.*, “Unsupervised progressive learning and the stam architecture,” in *Proc. International Joint Conference on Artificial Intelligence*, 2019.

[113] P. Kumar and D. Toshniwal, “Lifelong learning gets better with mixup and unsupervised continual representation,” *Applied Intelligence*, pp. 1–18, 2024.

[114] H. Zhang *et al.*, “Mixup: Beyond empirical risk minimization,” in *Proc. International Conference on Learning Representations*, 2018.

[115] D. Hu *et al.*, “How well does self-supervised pre-training perform with streaming data?” in *Proc. International Conference on Learning Representations*, 2022.

[116] E. Fini *et al.*, “Self-supervised models are continual learners,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 9621–9630.

[117] A. Achille *et al.*, “Life-long disentangled representation learning with cross-domain latent homologies,” in *Proc. Neural Information Processing Systems*, vol. 31, 2018.

[118] J. Ramapuram, M. Gregorova, and A. Kalousis, "Lifelong generative modeling," *Neurocomputing*, vol. 404, pp. 381–400, 2020.

[119] C. Wu *et al.*, "Memory replay gans: Learning to generate new categories without forgetting," in *Proc. Neural Information Processing Systems*, vol. 31, 2018.

[120] A. Gomez-Villa *et al.*, "Plasticity-optimized complementary networks for unsupervised continual learning," in *Proc. Winter Conference on Applications of Computer Vision*, 2024, pp. 1690–1700.

[121] G. Kim *et al.*, "Continual learning based on ood detection and task masking," in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 3856–3866.

[122] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, 2010.

[123] D. J. MacKay, "Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks," *Network: computation in neural systems*, vol. 6, no. 3, p. 469, 1995.

[124] C. Blundell *et al.*, "Weight uncertainty in neural network," in *Proc. International Conference on Machine Learning*, 2015, pp. 1613–1622.

[125] D. Molchanov, A. Ashukha, and D. Vetrov, "Variational dropout sparsifies deep neural networks," in *Proc. International Conference on Machine Learning*, 2017, pp. 2498–2507.

[126] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient hamiltonian monte carlo," in *Proc. International Conference on Machine Learning*, 2014, pp. 1683–1691.

[127] E. Aguilar *et al.*, “Continual evidential deep learning for out-of-distribution detection,” in *Proc. International Conference on Computer Vision*, 2023, pp. 3444–3454.

[128] J.-B. Grill *et al.*, “Bootstrap your own latent-a new approach to self-supervised learning,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 21 271–21 284.

[129] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality reduction by learning an invariant mapping,” in *Proc. Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1735–1742.

[130] Y. You, I. Gitman, and B. Ginsburg, “Large batch training of convolutional networks,” *arXiv preprint arXiv:1708.03888*, 2017.

[131] I. Loshchilov and F. Hutter, “SGDR: Stochastic gradient descent with warm restarts,” in *Proc. International Conference on Learning Representations*, 2017.

[132] J. Tack *et al.*, “Csi: Novelty detection via contrastive learning on distributionally shifted instances,” in *Proc. Neural Information Processing Systems*, vol. 33, 2020, pp. 11 839–11 852.

[133] T. Konishi *et al.*, “Parameter-level soft-masking for continual learning,” in *Proc. International Conference on Machine Learning*, 2023, pp. 17 492–17 505.

[134] K. He *et al.*, “Deep residual learning for image recognition,” in *Proc. Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[135] S. Bulusu *et al.*, “Anomalous example detection in deep learning: A survey,” *IEEE Access*, vol. 8, pp. 132 330–132 347, 2020.

[136] L. Bergman and Y. Hoshen, “Classification-based anomaly detection for general data,” in *Proc. International Conference on Learning Representations*, 2020.

[137] D. Hendrycks *et al.*, “Using self-supervised learning can improve model robustness and uncertainty,” in *Proc. Neural Information Processing Systems*, vol. 32, 2019.

[138] I. Golan and R. El-Yaniv, “Deep anomaly detection using geometric transformations,” in *Proc. Neural Information Processing Systems*, vol. 31, 2018.

[139] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Neural Information Processing Systems*, vol. 25, 2012.

[140] J. Bromley *et al.*, “Signature verification using a” siamese” time delay neural network,” in *Proc. Neural Information Processing Systems*, vol. 6, 1993.

[141] W. Bao, Q. Yu, and Y. Kong, “Evidential deep learning for open set action recognition,” in *Proc. International Conference on Computer Vision*, 2021, pp. 13 349–13 358.

[142] C. Zhang *et al.*, “Understanding deep learning (still) requires rethinking generalization,” *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[143] Z. Liu *et al.*, “Frequency-domain dynamic pruning for convolutional neural networks,” in *Proc. Neural Information Processing Systems*, vol. 31, 2018.

[144] A. Malviya, S. Dhole, and C. K. Maurya, “Unsupervised continual learning by cross-level, instance-group and pseudo-group discrimination with hard attention,” *Journal of Computational Science*, vol. 86, p. 102535, 2025.

[145] G. Kim, B. Liu, and Z. Ke, “A multi-head model for continual learning via out-of-distribution replay,” in *Proc. Conference on Lifelong Learning Agents*, 2022, pp. 548–563.

[146] H. Yang *et al.*, “Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification,” in *Proc. Computer Vision and Pattern Recognition workshops*, 2020, pp. 678–679.

[147] L. St, S. Wold *et al.*, “Analysis of variance (anova),” *Chemometrics and intelligent laboratory systems*, vol. 6, no. 4, pp. 259–272, 1989.

[148] D. W. Zimmerman and B. D. Zumbo, “Relative power of the wilcoxon test, the friedman test, and repeated-measures anova on ranks,” *The Journal of Experimental Education*, vol. 62, no. 1, pp. 75–86, 1993.

[149] J. M. Bland and D. G. Altman, “Multiple significance tests: the bonferroni method,” *Bmj*, vol. 310, no. 6973, p. 170, 1995.

[150] W. J. Conover, “Practical nonparametric statistics,” *John Wiley and Sonss*, vol. 350, 1999.

[151] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[152] X. Yan *et al.*, “Differentiable information bottleneck for deterministic multi-view clustering,” in *Proc. Computer Vision and Pattern Recognition*, 2024, pp. 27 435–27 444.

[153] S. Wang *et al.*, “Self-supervised information bottleneck for deep multi-view subspace clustering,” *IEEE Transactions on Image Processing*, vol. 32, p. 1555–1567, 2023.

[154] S. Lin *et al.*, “Beyond not-forgetting: Continual learning with backward knowledge transfer,” in *Proc. Neural Information Processing Systems*, vol. 35, 2022, pp. 16 165–16 177.

[155] J. Zheng *et al.*, “Towards open-set object detection and discovery,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 3961–3970.

[156] Y. Zhao *et al.*, “Novel class discovery in semantic segmentation,” in *Proc. Computer Vision and Pattern Recognition*, 2022, pp. 4340–4349.

[157] C. Ahuja *et al.*, “Continual learning for personalized co-speech gesture generation,” in *Proc. International Conference on Computer Vision*, 2023, pp. 20 893–20 903.

[158] Y. Guo *et al.*, “Dual mean-teacher: An unbiased semi-supervised framework for audio-visual source localization,” in *Proc. Neural Information Processing Systems*, vol. 36, 2023, pp. 48 639–48 661.