석사 학위논문

# Continual Learning Considering Hierarchical Labels

## 계층적 라벨을 고려한 지속적 학습

2023년 8월

서울대학교 대학원

전기•정보공학부

정 옥 철

# Continual Learning Considering Hierarchical Labels

지도 교수 전 세 영

이 논문을 석사 학위논문으로 제출함
2023년 8월

서울대학교 대학원
전기정보공학부
정 옥 철

정옥철의 석사 학위논문을 인준함
2023년 8월

위 원 장 이종호 (인)

부위원장 전세영 (인)

위 원 문태섭 (인)

## Abstract

# Continual Learning Considering Hierarchical Labels

Okchul Jung
Electrical and Computer Engineering
The Graduate School
Seoul National University

Hierarchical Learning and Continual Learning are two powerful paradigms in machine learning. The former leverages the inherent structure in data through a technique known as Hierarchical Multi-label Classification (HMC), allowing models to capture both broad and specific patterns within the data. The latter enables models to learn from a continuous stream of data over time, essential in adapting to evolving real-world data distributions. Despite their strengths, traditional continual learning approaches often struggle with hierarchical relationships between tasks.

This paper introduces an innovative approach that synergizes hierarchical learning and continual learning, referred to as Hierarchical Label Expansion (HLE). By proposing a multi-level hierarchical class incremental task configuration with an online learning constraint, the paper explores how networks can initially learn coarse-grained classes and then expand to more fine-grained classes across various hierarchy levels. To support this new setup, a rehearsal-based method using hierarchy-aware pseudo-labeling is presented, alongside an effective memory management and

sampling strategy. These components enable the model to better understand and adapt to the hierarchical structure of tasks, efficiently learning from new data while retaining performance on previous tasks. The experimental results validate the effectiveness of this method, showing improved classification accuracy across all hierarchy levels, irrespective of depth or class imbalance ratio. Remarkably, it outperforms existing methods and demonstrates superior performance in various continual learning scenarios.

The paper is structured into chapters, beginning with an introduction to the foundational concepts of hierarchical and continual learning. The main body of the work is dedicated to the integration of these paradigms through 'Continual Learning Considering Hierarchical Labels', representing a novel advancement in modeling complex and evolving environments. The proposed combination paves the way for more robust continual learning frameworks capable of handling the challenges of real-world data.

Keywords : Continual Learning, Hierarchical Learning, Deep Learning, Machine Learning

Student Number : 2021-21407

# 초 록

계층적 학습과 지속적 학습은 기계 학습에서 두 강력한 패러다임입니다. 전자는 계층적 다중 라벨 분류(HMC)라는 기술을 통해 데이터의 본질적 구조를 활용하여 데이터 내의 광범위하고 구체적인 패턴을 포착할 수 있게 합니다. 후자는 시간이 지남에 따라 연속적인 데이터 스트림에서 모델이 학습할 수 있게 하며, 실생활에서의 진화하는 데이터 분포에 적응하는 데 필수적입니다. 하지만 이러한 강점에도 불구하고 전통적인 지속적 학습 접근법은 작업 간의 계층적 관계에서 발생하는 문제를 다루지 않았습니다.

이 논문은 계층적 라벨 확장(HLE)으로 알려진 계층적 학습과 지속적 학습을 결합하는 접근법을 소개합니다. 온라인 학습 제약 조건이 있는 다중 레벨 계층적 클래스 분화 구성을 제안함으로써, 네트워크가 초기에 거친 클래스를 학습하고 다양한 계층 수준에서 더 세밀한 클래스로 확장할 수 있게 하는 방법을 탐구합니다. 이 새로운 설정을 지원하기 위해, 계층을 고려하는 가짜 라벨링을 사용하는 리허설 기반 방법이 제시되며, 효과적인 메모리 관리 및 샘플링 전략과 함께 제시됩니다. 이러한 구성 요소들은 모델이 데이터의 계층적 구조를 더 잘 이해하고 적응하게 하며, 이전 작업에 대한 성능을 유지하면서 새로운 데이터에서 효율적으로 학습할 수 있게 합니다. 실험 결과는 이 방법의 효과를 검증하며, 깊이나 클래스 불균형 비율과 관계없이 모든 계층 수준에서 분류 정확도를 향상시킵니다. 눈에 띄게도 기존 방법을 능가하며 다양한 지속적 학습 시나리오에서 우수한 성능을 보입니다.

논문의 구성은 계층적 학습과 지속적 학습의 기본 개념에 대한 소개로 시작합니다. 논문의 주요 부분은 복잡하고 진화하는 환경을 모델링하는 '계층적 라벨을 고려한 지속적 학습'의 설계 및 이에 적용할 수 있는 지속적 학습 방법의 소개를 다루고 있습니다. 제안된 지속적 학습 환경과 방법은 실생활에서의 데이터를 처리할 수 있는 지속적 학습 프레임워크를 위한 미래를 시사합니다.

**주요어 : 지속적 학습, 계층적 학습, 딥러닝, 머신러닝**
**학 번 :** 2021-21407

# Table of Contents

# List of Tables



# List of Figures

CHAPTER I

# Introduction

## 1.1 Aim Of the Research

The aim of this research is to address the challenges in continual learning, particularly when there are hierarchical relationships between old and new tasks. Traditional continual learning approaches, as discussed in prior works [2–5] do not account for such aforementioned scenario, where data hierarchy is to be considered. There have been Continual Learning works [1, 6] that suggest novel setups aside from the conventional ones, but there have not been any hierarchical Continual Learning setups yet. This research proposes a novel approach that combines the strengths of hierarchical learning, a concept explored extensively [7–9], and Continual Learning. The research introduces a multi-level hierarchical class incremental task configuration with an online learning constraint, termed as Hierarchical Label Expansion (HLE). This innovative configuration allows a network to learn coarse-grained classes initially, with data labels continually expanding to more fine-grained classes across various levels of the hierarchy. The research also introduces a rehearsal-based method that employs hierarchy-aware pseudo-labeling and a simple yet effective memory management and sampling strategy. The ultimate goal of this research is to improve the adaptability and robustness of machine learning models, enabling them to learn and adapt in complex, evolving environments.

## 1.2 Hierarchical Learning

Hierarchical learning is a powerful approach in machine learning that leverages the inherent hierarchical structure present within data. This approach organizes data in a hierarchical manner, allowing models to learn representations at different levels of abstraction [9, 10]. In this context, a key application of hierarchical learning is hierarchical multi-label classification, where data instances are associated with a set of target labels that form a hierarchy [11, 12].

### 1.2.1 Hierarchical Multi-label Classification Problem

Hierarchical multi-label classification (HMC) differs from conventional image classification by incorporating a hierarchical structure into the label assignments [7]. In image classification, a single label is assigned from a flat list of categories, whereas HMC considers labels that form a hierarchy or taxonomy. For example, in animal taxonomy, "Mammal" may be a superclass of "Cat" and "Dog," and an image of a cat would be associated with the path Animal, Mammal, and Cat. This hierarchical structure allows for partially correct predictions, unlike flat classification [7]. HMC is commonly used when classes are organized hierarchically rather than disjointly. The hierarchical structure can take the form of a tree or a Directed Acyclic Graph (DAG) depending on the task, with objects associated with all subclasses or a subset of them [13, 14]. Furthermore, HMC becomes even more challenging when each object can be associated with multiple paths in the class hierarchy. This scenario is encountered in tasks such as text classification, image annotation, and protein function prediction in bioinformatics [15–19].

### 1.2.2 Hierarchical Multi-label Classification Approaches

While addressing the classification methods, there are two primary approaches that hierarchical multi-label classification employs: the local approach [20–22] and the global approach [14, 23]. The local approach trains one classifier per node in the label hierarchy, solving a binary classification problem for each node. On the other hand, the global approach considers the entire hierarchy while training a single model, which, while more computationally demanding, allows the model to exploit the correlations between different labels in the hierarchy.

Algorithms that perform HMC must optimize a loss function either locally or globally. Local learning attempts to discover the specificities dictating the class relationships in particular regions of the class hierarchy, later combining the local predictions to generate the final classification [24]. Global approaches for HMC, however, usually consist of a single classifier capable of associating objects with their corresponding classes in the hierarchy as a whole [16, 25].

Balancing the advantages and disadvantages of these two approaches is crucial. While global approaches are generally cheaper and do not suffer from the error-propagation problem, they may not capture local information from the hierarchy. Conversely, local approaches are more computationally expensive since they rely on a cascade of classifiers, but they are more suitable for extracting information from specific regions of the class hierarchy [7].

One promising direction is a paradigm shift towards a hybrid method capable of simultaneously optimizing both local and global loss functions. This hybrid approach balances the benefits of both local and global strategies, reinforcing the propagation of gradients for proper local information encoding among classes of the corresponding hierarchical level, while also keeping track of the label dependency in the hierarchy as a whole. In addition, a hierarchical violation penalty is introduced to encourage predictions that obey the hierarchical structure, setting a new state-of-the-art for HMC problems [23].

Hierarchical learning, particularly hierarchical multi-label classification, offers a robust and flexible framework for handling complex classification tasks. It provides opportunities to improve prediction accuracy and interpretability by leveraging the inherent hierarchical structure present within the data. Furthermore, advances in combining local and global learning strategies are opening new avenues for the effective application of hierarchical learning in diverse domains.

## 1.3 Continual Learning

Continual Learning (CL) is a learning paradigm that allows models to learn from a continuous stream of data over time. This approach is crucial in real-world scenarios where data is often non-stationary, and models are required to adapt to evolving data distributions. Continual learning enables models to adapt to new tasks and environments without forgetting the knowledge they have previously acquired [4, 5, 26–28]. It addresses the issue of catastrophic forgetting, where a model tends to forget previously learned information when trained on new data.

### 1.3.1 Continual Learning Scenarios

Continual learning, also known as lifelong or incremental learning, is a critical aspect of machine learning models, especially in the realm of deep learning, as it allows models to learn from a continuous stream of data over time. This process can be categorized into three typical learning scenarios [2, 29]: task-incremental, domain-incremental, and class-incremental.

Task-incremental learning scenario involves learning a series of tasks sequentially, with each task having its own distinct training data. The primary goal of this scenario is to retain knowledge acquired from previous tasks while effectively learning new tasks without a significant loss of previously acquired knowledge, a phenomenon known as catastrophic forgetting. Domain-incremental learning scenario focuses on learning from different domains or environments, each representing a different distribution of data. The objective here is to adapt the model to new domains without forgetting the knowledge acquired from previous domains. Class-incremental learning scenario involves learning to recognize an increasing number of classes over time. The model needs to handle the addition of new classes without forgetting the previously learned classes. The field of continual learning is a critical and rapidly evolving area in deep learning, with a wide array of techniques and algorithms being developed to tackle the three aforementioned scenarios. Each of these techniques builds upon the knowledge from prior work, refining and improving upon the existing approaches to handle the continual learning tasks more effectively and efficiently.

### 1.3.2 Continual Learning Settings

Continual Learning encompasses a variety of setups that mirror different real-world learning situations. These setups can be categorized primarily along two axes: online versus offline and task-free

versus task-based. Continual Learning (CL) setups can be classified as either online [30–33] or offline [3, 27, 34–36], depending on how often streamed samples are utilized to train the model. CL setups can also be categorized as task-free [37–39] or task-based [28, 36, 40, 41], depending on the existence of explicit task boundaries or identifiers. These different setups represent varying degrees of complexity and difficulty in continual learning, each with its unique challenges and corresponding methodologies to tackle them.

**Online and Offline Continual Learning setup.** In an online CL setup, data is encountered as a continuous stream, where each data instance is utilized once and only once for training the model. This framework emulates real-world situations where data is generated continuously over time and where the model needs to learn and adapt 'on the fly.' Each incoming data instance is processed individually for model training, after which it is discarded, leaving no opportunity for it to be revisited in the learning process. On the other hand, an offline CL setup provides a different approach to handling data. Unlike its online counterpart, offline Continual Learning allows data from each task to be revisited and utilized multiple times during the training process. Data in this setup is not handled on an individual basis. Instead, it is managed as a batch or a set of multiple data samples. This arrangement enables repeated exposure of the model to the same data instances within a single training batch, offering multiple opportunities for learning and refinement. The above differences in handling data make online CL setup more realistic, but also challenging setup in Continual Learning, relative to offline CL setup.

**Task-free and Task-based Continual Learning setup.** In task-free setups, the learning system encounters a continuous stream of data without explicit task boundaries or task identifiers [37]. This represents the most common way humans and animals experience the world - as a continuous stream of information without clear demarcations of 'tasks'. On the other hand, in task-based setups, the data stream is divided into distinct tasks, and the model might have access to task identifiers during both training and testing [36]. Task boundaries give the model a cue to consolidate the current knowledge before moving to a new task, reducing the chances of catastrophic forgetting.

### 1.3.3 Continual Learning Approaches

There exist three common strategies employed to tackle the problem of catastrophic forgetting in Continual Learning: regularization methods, dynamic architecture or parameter isolation methods, and memory-based methods. Each of these methods offers a unique approach to managing the trade-off between retaining old knowledge and incorporating new information, also known as stability-plasticity dilemma [2].

**Regularization methods.** Regularization-based techniques typically introduce constraints on the update process of model parameters and hyperparameters to reinforce previously acquired knowledge while learning new tasks. The objective is to alleviate catastrophic forgetting in continual learning. One notable method is the Elastic Weight Consolidation (EWC) algorithm proposed by Kirkpatrick et al. [5]. EWC works by adding a regularization term to the loss function that constrains important parameters of the model, defined through the Fisher Information Matrix, from drastically changing while learning new

tasks. This way, EWC allows the model to keep important knowledge from old tasks. However, EWC's assumption that this matrix is diagonal is generally inaccurate. To counter this problem, Liu et al. [42] suggested a strategy to approximate the diagonalization of the Fisher information matrix through the rotation of the model's parameter space, thereby preserving the forward output. While EWC necessitates a quadratic penalty term for each learned task, leading to a linear rise in computational cost, Schwarz et al. [43] proposed an online version of EWC to resolve this issue by focusing only on the most recent task. Further, Chaudhry et al. [44] introduced an efficient alternative to EWC, called EWC++, which employs a single Fisher information matrix for all previously learned tasks, updating the matrix using a moving average approach. Another noteworthy approach is Synaptic Intelligence (SI) proposed by Zenke et al. [45]. SI introduces a surrogate quantity to track the importance of a parameter for the loss and then imposes a quadratic penalty during parameter updates to preserve important parameters. Aljundi et al. [3] introduced the Memory Aware Synapses (MAS) approach. In MAS, the importance of the parameters is computed with respect to the effect of a parameter change on the learned examples. This importance is then used to guide the regularization process.

**Parameter isolation and Dynamic structure methods.** Dynamic architectures or parameter isolation approaches involve either expanding the model architecture or isolating the parameters for each task to tackle the catastrophic forgetting problem. These approaches can involve architectural modifications like allocating new neurons or layers to new tasks or completely isolating the parameters for different tasks. In the field of image classification, several dynamic architecture methods for continual learning have been proposed. One such method is the Progressive Network proposed by Rusu et al. [26] In this method, a new neural subnetwork is trained for each task with lateral connections allowing for feature transfer from previously learned tasks. Aljundi et al. [46] developed a network of experts, where an expert gate is used to select the most relevant previous task to aid the learning of the new task. This gate also selects the most suitable model for a given data instance at test time. Yoon et al. [47] introduced the Dynamically Expandable Network (DEN) that utilizes previously acquired knowledge and expands the network structure when the prior knowledge isn't sufficient for the new task. Hung et al. [48] proposed the Compacting Picking Growing (CPG) method, where parameters trained for all previous tasks are frozen to prevent forgetting. Lee et al. [49] proposed a Continual Neural Dirichlet Process Mixture (CN-DPM) model which uses different expert subnetworks for different data instances and decides to create a new expert subnetwork based on a Bayesian non-parametric framework. Various methods have been proposed to facilitate continual learning in image classification, utilizing the methodology of dynamic architecture and parameter isolation. These methods seek to balance the retention of knowledge from previous tasks with the learning of new tasks, thus mitigating the problem of catastrophic forgetting.

**Memory-based methods.** Memory-based methods [4, 27, 50] typically utilize a storage buffer to retain data and related information from earlier tasks. This information is used during the learning of subsequent tasks, allowing the system to strengthen its retention of prior knowledge and limit the impact of catastrophic forgetting. Various strategies exist to achieve this objective, which are outlined below. One of the primary methods, known as iCARL, was first proposed by Rebuff et al. [27] The iCARL

methodology employs stored data from past tasks and new data for training. However, this method has its limitations, as it necessitates all data from new tasks to be trained concurrently. To address this, Chaudhry et al. [50] introduced Experience Replay (ER), which leverages reservoir sampling [51] to randomly select a specified quantity of data from a stream of unknown length for storage in the memory buffer. However, this approach has its drawbacks if tasks have uneven numbers of instances. To handle this, a range of other sampling algorithms have been proposed. For example, Aljundi et al. [52] treated data selection as a constraint selection problem and opted for instances that minimized the solid angle formed by their corresponding constraints. Liu et al. [53], on the other hand, trained exemplars using image-size parameters to capture the most representative instances from previous tasks.

Some methodologies focus on the selection of data instances for retraining. For instance, Aljundi et al. [54] presented Maximally Interfered Retrieval (MIR), a method that selects a subset of data instances that experience an increase in loss if the model parameters are updated based on new data. Shim et al. [55] proposed an Adversarial Shapley (AS) scoring method, which selects previous data instances that can mostly maintain their decision boundaries during the training of the new task. Memory buffers can also be divided into sections, as suggested by methods like Bias Correction (BiC) [56] and separation into episodic and semantic memory [57]. There are also methods like Gradient Episodic Memory (GEM) [4] and Averaged GEM (A-GEM) [58], which propose to prevent the parameter update from increasing the loss of each individual previous task during the learning process of a new task. These memory-based methods provide diverse strategies for tackling catastrophic forgetting, utilizing stored instances from previous tasks to help the model retain the old knowledge while learning new tasks.

All three strategies come with their unique strengths and weaknesses, and choosing the right approach depends on the specific requirements of the continual learning problem at hand.

## 1.4 Contribution of Thesis

Traditional continual learning approaches often encounter difficulties when there are hierarchical relationships between old and new tasks, especially when there are small or non-existent overlaps between tasks. This challenge has been the focus of several research efforts, with various strategies proposed to mitigate the effects of catastrophic forgetting, a phenomenon where the model tends to forget previously learned tasks when learning new ones [59, 60]. In light of these challenges, the integration of hierarchical learning and continual learning presents a promising direction for research. By combining these two approaches, it is possible to develop models that can effectively learn and adapt in complex, evolving environments, handling the challenges of real-world data. The integration of these two approaches presents a promising direction for research, offering the potential to develop models that can effectively learn and adapt in complex, evolving environments. This introduction provides a foundation for the concepts and methodologies that will be explored in depth in the subsequent chapters of the thesis.

## 1.5 Organization of Thesis

The paper's first chapter provides a brief introduction to hierarchical learning and continual learning, setting the groundwork for the explored concepts and methodologies. In the second chapter, the thesis work, titled 'Continual Learning Considering Hierarchical Labels,' is introduced. This work presents a novel advancement by integrating hierarchical learning and continual learning, enabling models to learn and adapt in complex and evolving environments.

CHAPTER II

# Online Continual Learning on Hierarchical Label Expansion

In this research, a new Continual Learning (CL) configuration named Hierarchical Label Expansion (HLE) is introduced, which focuses on the hierarchical relationships between classes in task-free online CL scenarios. In HLE, classes are incrementally learned, where fine-grained classes expand from previously learned coarse-grained ones. The study introduces a new CL methodology called PL-FMS, which combines Pseudo-Labeling(PL) based memory management and Flexible Memory Sampling (FMS). This strategy effectively leverages hierarchy information between class labels, mirroring real-world knowledge accumulation. The performance of the models was assessed using any-time inference and measured the classification accuracy across all hierarchy levels. The proposed HLE model caters to single and multiple hierarchy depths and handles balanced and imbalanced class data. The experiments conducted on CIFAR100, Stanford-Cars, iNaturalist-19, and a new dataset, ImageNet-Hier100, showed the superior performance of the proposed model, both in HLE and other existing CL setups such as disjoint, blurry [1], and i-Blurry [61].

## 2.1 Hierarchical Label Expansion

### 2.1.1 Hierarchical CL Configurations

The Hierarchical Label Expansion (HLE) setup we propose implements task-free online learning, enabling the model to gradually learn classes from different hierarchies, vertically and horizontally, regardless of task limits. This setup anticipates the model learning broader parent classes prior to encountering more detailed child classes that emerge from them. An overview of the HLE setup is provided in Figure 2.1(c).

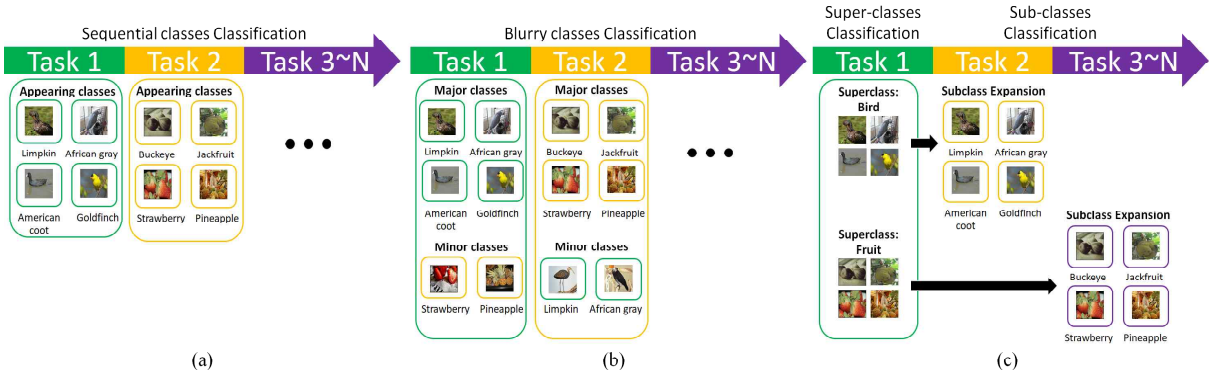


Figure 2.1: Comparison sketch between conventional, blurry [1], and our HLE setups. (a) Conventional task-free online CL setup gradually introduces new classes and classifies data without task identification (b) Blurry task-free online CL setup where classes are divided into major and minor categories at each task, with varying proportions, leads to unclear task boundaries (c) Proposed HLE CL setup features class label expansion where child class labels are added to parent class labels throughout the learning process.

We visualize the model encountering a stream of data points as $\mathcal{T} = ((x_1, y_1), (x_2, y_2), \cdots)$, where each $(x_j, y_j)$ comes from a data distribution $\mathcal{D}_{\mathbb{X}\times\mathbb{Y}}$, $x_j \in \mathbb{X}$ is the $j$th model input (image), and $y_j \in \mathbb{Y}$ is $x_j$'s class label. Sequential tasks, indexed as $k$, can segregate this data stream $\mathcal{T}$ into disjoint sub-sequences, $\mathcal{T}_1, \mathcal{T}_2, \cdots$, where each $\mathcal{T}_k = ((x_j, y_j))_{j=t(k)}^{t(k+1)-1}$ and $t(k)$ is the initial sample index for the $k$-th task. We represent the subset of classes the model encounters during the $k$th task as $\mathbb{Y}_k = \{y_j | j = t(k), \cdots, t(k+1) - 1\}$. Traditional CL assumes the sampling distribution changes over time and the sampling distributions for tasks don't intersect, i.e., $\mathbb{Y}_k \cap \mathbb{Y}_l = \emptyset$ for $k \neq l$. However, more practical contexts often require consideration, as with the i-Blurry CL setup [61] which assumes each task has a shared subset of classes $\mathbb{Y}^s$ that are continually trained, and a separate subset $\mathbb{Y}_k^d$ that is only trained at a specific task. In this case, $\mathbb{Y}_k$ is defined as $\mathbb{Y}_k = \mathbb{Y}^s \cup \mathbb{Y}_k^d$, implying that $\mathbb{Y}_k \cup \mathbb{Y}_l = \mathbb{Y}^s \neq \emptyset$.

Our HLE offers additional structures to $\mathbb{Y}$ by creating a label relationship between classes in $\mathbb{Y}$. We consider $\mathbb{Y}$ to comprise classes from $H$ levels, therefore $\mathbb{Y} = \bigcup_{h=1}^{H} \mathbb{Y}^h$ and $\mathbb{Y}^h \cap \mathbb{Y}^{h'} = \emptyset$, with $\mathbb{Y}^h$ being the label subset at hierarchy level $h$. A smaller $h$ value indicates more coarse-grained classes. In the HLE setup, each task expands the labels for a class subset at level $h$ to their more detailed classes at level $(h+1)$. This means labels expand one level during each task. For the $(k+1)$-th task, a subset $\bar{\mathbb{Y}}_{k+1}^h$ of $\mathbb{Y}_k^h$ is chosen to be expanded into a set of more detailed classes $\mathbb{Y}_{k,new}^{h+1}$, which results in $\mathbb{Y}_k = \mathbb{Y}_{k,new}^{h+1}$.

To include multiple hierarchy levels, we consider a model that encompasses an encoder $\mathcal{F}$ for feature embedding and several classifiers $\{\mathcal{G}^h\}_{h=1}^{H}$ corresponding to each hierarchy. Particularly, $\mathcal{G}^h(\mathcal{F}(x))$ predicts the classes within the $h$-th level that have been encountered until the current iteration. During training, a single label is assigned to each input irrespective of its hierarchical position in the data stream, and the model remains unaware of the hierarchy relationship among classes. Instead, the model is provided with the hierarchy level as a vague indication of its position.

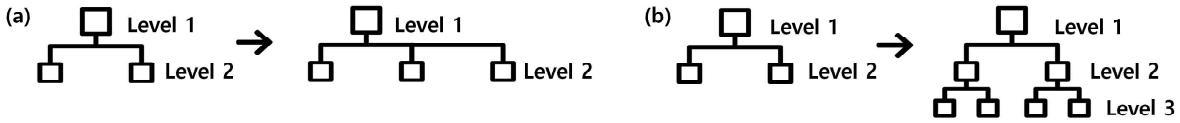


Figure 2.2: The illustration depicts the two hierarchical label expansion (HLE) scenarios. (a) In the single-depth scenario, fine-grained classes increment horizontally from the coarse-grained classes within the same hierarchy level. (b) In the multiple-depth scenario, classes expand vertically from the coarse-grained to the fine-grained classes across multiple hierarchy levels.

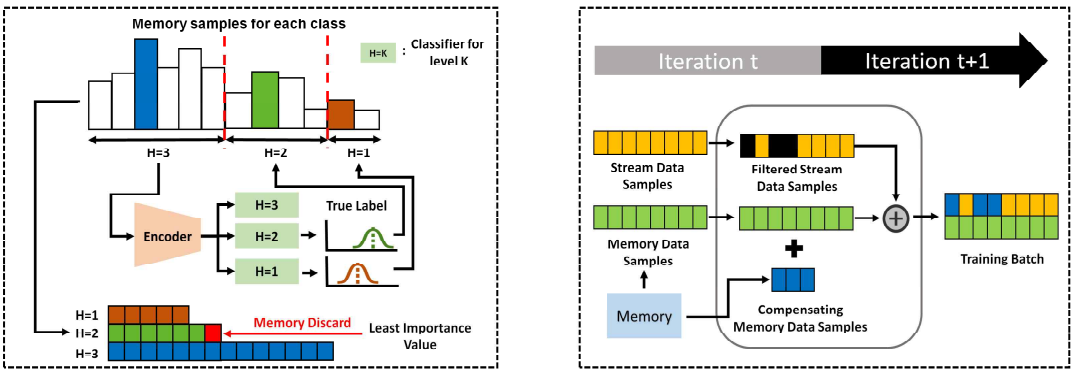


Figure 2.3: Sketch of our proposed method, PL-FMS's two components: PL and FMS. (a) Pseudo-Labeling based memory management (PL) illustrates how to discard data samples based on their contribution to the decrease in loss, irrespective of whether they are associated with pseudo-labels or true labels. (b) Flexible Memory Sampling (FMS) demonstrates how the training batch is constructed by filtering and compensating data samples.

### 2.1.2 Hierarchical CL Depth Scenarios

Our HLE configuration includes two situations: single-depth and multiple-depth scenarios (compared to the non-existent depth in existing setups), as shown in Figure 2.2. In the single-depth scenario, incremental learning occurs horizontally within the same hierarchical level. Conversely, in the multiple-depth scenario, new classes are introduced vertically, each with progressively more detailed characteristics. In the single-depth scenario, the model is trained on all parent classes during the initial task and then expands them partially in subsequent tasks. In the multiple-depth scenario, the model's capacity to learn and expand hierarchical knowledge is assessed as it navigates through a complex hierarchy. This implies that the model learns classes at hierarchy level $h$ during the $h$th task.

### 2.1.3 Hierarchical Label Scenarios

We further explored the single-depth scenario by conducting experiments under two label situations, dual-label (with overlapping data across tasks) and single-label (with disjoint data across tasks), as outlined in Table 2.1. Disjoint data across tasks would lead to only a single-label per data sample, meaning that the data samples are to be appeared in the learning process only once, with only one label for the

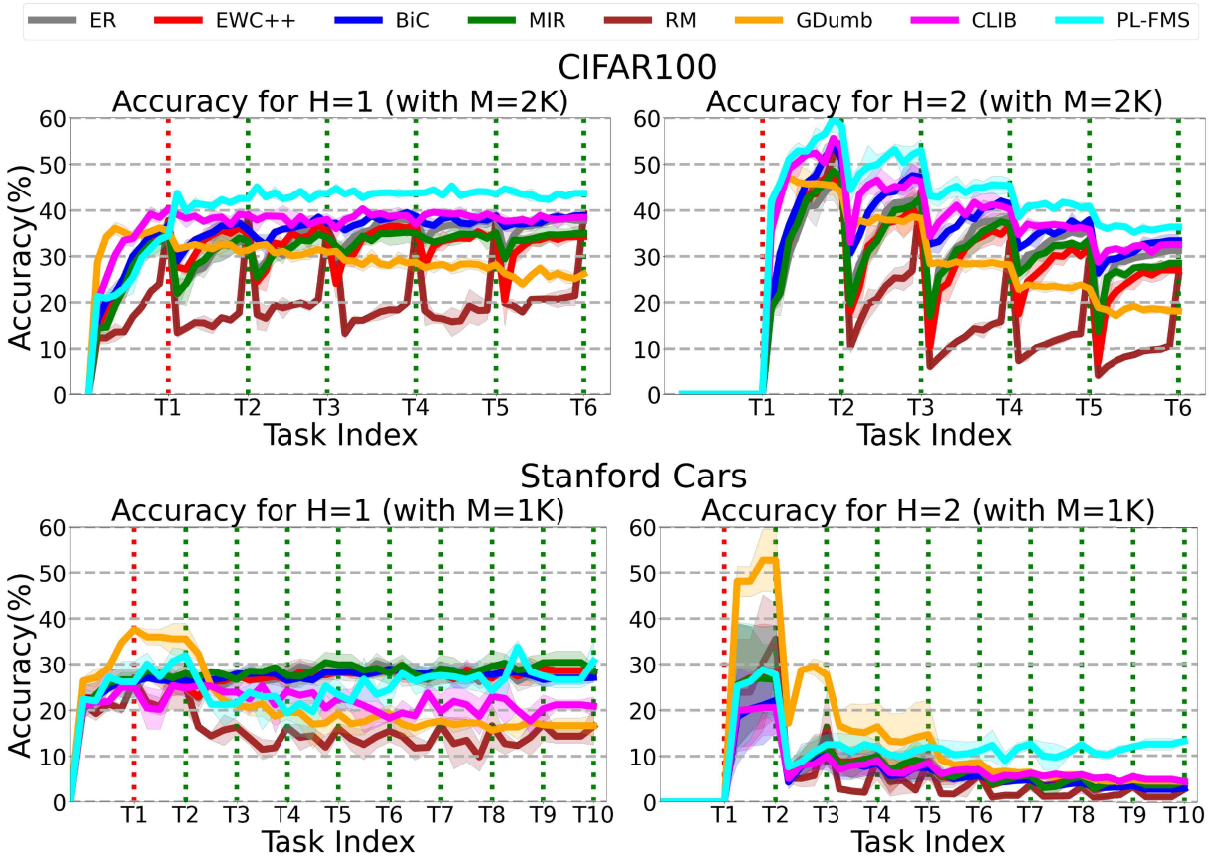


Figure 2.4: Any-time inference results on CIFAR100 and Stanford Cars datasets for single-depth hierarchy. H=1 is parent classes and H=2 child classes. Task index 1 receives parent class labeled data and subsequent indexes receive child class labeled data. Each data point shows average accuracy over three runs ($\pm$ std. deviation).

entire hierarchy. On the other hand, overlapping data across tasks would lead to dual-label per data sample, meaning that data samples are to appear twice in the learning process, but with corresponding labels for every hierarchical label.

## 2.2 Pseudo-Labeling based Flexible Memory Sampling (PL-FMS)

In this part, we introduce our approach which utilizes a rehearsal-based incremental learning strategy. This strategy involves training models by revisiting previously encountered data held in a stream buffer. Our method integrates pseudo-labeling to fully capitalize on the hierarchical relationships between classes, along with a memory sampling approach that flexibly builds the training batch from both stored data and incoming data streams. We will delve into our method, which encompasses Pseudo-Labeling (PL) based Memory Management and Flexible Memory Sampling (FMS), in sections 2.2.1 and 2.2.2 respectively.

### 2.2.1 Pseudo-Labeling based Memory Management (PL)

In this part, we introduce an innovative strategy for memory management that utilizes a model's predictions to generate pseudo-labels for each hierarchical level in our HLE structure. This approach, illustrated in Figure 2.3 (a), is termed Pseudo-Labeling (PL) based memory management.

To explain further, let's denote $\mathcal{M}$ as the memory that retains samples from the data stream and let $\mathcal{M}_y$ represent the subset of the memory where samples belong to class $y$. We also consider a metric, $\mathcal{H}_n$, to quantify the significance of samples within the memory. When the capacity of $\mathcal{M}$ reaches its maximum, we need to make room for new samples, which involves removing an existing sample from the memory. To do this, we identify $\bar{y}$, which represents the class with the highest number of samples in the memory. Contrary to previous works, instead of only considering samples from $\mathcal{M}_{\bar{y}}$, we propose to include samples from other classes hierarchically related to $\bar{y}$. This is achieved by using the network's class probability predictions.

To identify classes that are hierarchically linked to $\bar{y}$, we collect the model's predictions for samples in $\mathcal{M}_{\bar{y}}$ across levels, excluding the level of $\bar{y}$. We then pinpoint the classes that receive the highest count of these predictions at each level. This process is represented mathematically by the following equation:

$$\hat{y}^h(\mathcal{M}_{\bar{y}}) = \arg\max_{y \in \mathbb{Y}^h} \sum_{(x,\bar{y}) \in \mathcal{M}_{\bar{y}}} 1_y(x), \tag{II.1}$$

Where $1_y(x)$ serves as an indicator function, defined as follows:

$$1_y(x) = \begin{cases} 1, & y = \arg\max_i \ p_i^h(x) \\ 0, & \text{otherwise.} \end{cases}$$

Subsequently, using the predicted classes for the other levels, we create an index set of candidate samples to be removed from the memory, represented by the following equation:

$$\mathcal{I}_{\bar{y}} = \{j | (x_j, y_j) \in \mathcal{M}_{\bar{y}} \cup \bigcup_{k=0, k \neq h}^{H} \mathcal{M}_{\hat{y}^k}\}. \tag{II.2}$$

Finally, we locate the index $\hat{j}$ of the sample to be removed, which corresponds to the sample with the least measured significance. This process is described by the equation:

$$\hat{j} = \arg\min_{j \in \mathcal{I}_{\bar{y}}} \mathcal{H}_j. \tag{II.3}$$

As a metric to evaluate the significance of samples, we utilize the sample-wise loss importance value, which measures the decrease in loss for each sample during training and prioritizes removing the data from the memory that demonstrates the least reduction in loss.

| Methods | Single-Label Scenario | | | | | | Dual-Label Scenario | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CIFAR100 | | ImageNet-Hier100 | | Stanford Cars | | CIFAR100 | | ImageNet-Hier100 | | Stanford Cars | |
| | $H=1$ | $H=2$ | $H=1$ | $H=2$ | $H=1$ | $H=2$ | $H=1$ | $H=2$ | $H=1$ | $H=2$ | $H=1$ | $H=2$ |
| ER [62] | 37.8±2.06 | 31.3±0.78 | 73.4±1.91 | 55.7±1.87 | 28.4±0.73 | 4.01±0.06 | 42.0±0.57 | 25.5±0.33 | 78.8±0.82 | 57.2±1.89 | 37.8±0.72 | 3.53±0.44 |
| EWC++ [63] | 34.3±0.68 | 27.1±0.80 | 73.4±0.99 | 54.0±1.34 | 27.9±0.74 | 3.42±0.33 | 39.9±2.26 | 23.3±1.93 | 76.3±1.20 | 53.0±3.32 | 38.3±0.47 | 3.17±0.36 |
| BiC [63] | 38.8±0.41 | 33.4±1.41 | 72.5±0.09 | 58.7±0.78 | 27.1±1.08 | 3.05±0.29 | 42.1±1.06 | 28.0±1.01 | 77.7±1.24 | 60.4±0.30 | 36.5±1.04 | 3.26±0.34 |
| MIR [64] | 35.0±1.47 | 28.6±0.18 | 74.5±0.90 | 57.3±1.93 | 28.6±1.09 | 4.50±0.44 | 42.4±0.95 | 26.2±1.79 | 78.5±0.57 | 56.0±2.25 | **43.1±1.18** | 5.02±0.74 |
| RM [1] | 39.3±0.83 | 25.9±0.89 | 69.7±0.27 | 61.0±0.86 | 16.5±4.05 | 2.83±0.64 | 38.2±0.76 | 25.7±1.12 | 71.5±0.73 | 63.1±0.89 | 18.1±2.54 | 3.29±0.28 |
| GDumb [35] | 26.2±0.87 | 18.6±0.09 | 53.4±1.18 | 37.2±0.33 | 16.6±2.31 | 4.50±0.12 | 25.7±0.83 | 18.5±1.11 | 59.2±0.54 | 42.3±0.54 | 15.0±1.40 | 4.06±0.33 |
| CLIB [61] | 38.4±0.58 | 32.6±0.59 | 64.6±0.72 | 49.4±1.32 | 20.8±2.08 | 4.52±0.78 | 44.5±0.87 | 37.1±0.20 | 71.3±0.76 | 55.4±0.35 | 19.1±4.30 | 3.83±0.78 |
| PL-FMS | **43.7±0.13** | **36.4±0.62** | **77.8±1.32** | **64.6±0.97** | **30.7±4.39** | **13.2±0.29** | **49.0±0.19** | **39.5±0.64** | **79.5±0.54** | **67.2±0.41** | 42.0±3.59 | **26.8±3.27** |

Table 2.1: Experimental results of baseline methods and our proposed method evaluated on HLE setup for single-depth hierarchy scenario in CIFAR100, ImageNet-Hier100, and Stanford Cars. Dual-label means overlapping data between tasks, and single-label means disjoint data between tasks. Classification accuracy on hierarchy level 1 and 2 at the final task (%) was measured for all datasets, and the results were averaged over three different random seeds.

### 2.2.2 Flexible Memory Sampling (FMS)

Previous rehearsal-based techniques proposed including the stream buffer directly in training, which can create bias towards the data stream distribution and negatively affect the model's performance. Although training exclusively with memory samples has been suggested, it has been found to restrict adaptability to new classes. To address these issues, we propose a solution known as Flexible Memory Sampling (FMS). This is a straightforward yet effective sampling strategy that adjusts the number of stream samples included in the training batch, which is demonstrated in Figure 2.3 (b).

Experience Replay (ER) is employed to construct a training batch $B_t$ at iteration $t$, which uses all samples in the stream buffer $S_t$ and takes an equal number of samples from the memory, resulting in a batch size of $|B_t| = 2|S_t|$. However, FMS distinguishes itself by randomly excluding samples from $S_t$ during the training process.

Suppose $T_c$ is the iteration at which class $c$ first appears. In this case, we selectively include stream samples of class $c$ with an increasing probability as $t - T_c$ grows larger, slowly incorporating new classes from the stream buffer. The probability to include a stream sample of class $c$ is governed by a Bernoulli distribution, calculated as follows:

$$\rho_t(c) \sim \text{Ber}\left(\min\left(\frac{t - T_c}{T}, 1\right)\right), \tag{II.4}$$

where $T$ is a hyperparameter that controls how quickly the network adopts stream samples for training. Initially, after encountering new classes, the training process looks similar to the memory-only training approach, but as $t - T_c$ increases, it begins to resemble the sampling method of ER.

By combining these two strategies, we have our proposed method, known as Pseudo Labeling-based Flexible Memory Sampling (PL-FMS). A more detailed description of the PL-FMS algorithm is available in the supplementary material.

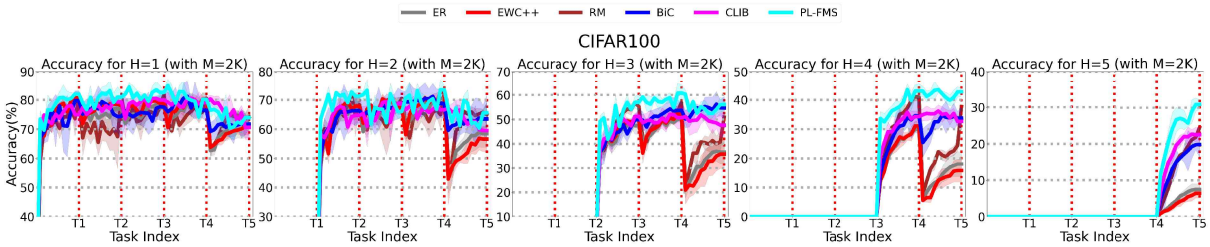


Figure 2.5: Any-time inference results on CIFAR100 dataset for multiple-depth hierarchy. H=1 represents the coarsest level and H=5 represents the finest level of class hierarchy. The dotted line represents the point at which the model is fully given the task data for the corresponding task index. The reported data points represent the average accuracy over three runs ($\pm$ std. deviation).

## 2.3 Empirical Evaluation

This section introduces the datasets, baselines, evaluation metrics, and implementation details on constructing our proposed HLE Continual Learning setup. Each detailed explanation is further explained in the following Section 2.3.1, 2.3.2, !2.3.3, 2.3.4, and 2.3.5, respectively.

### 2.3.1 Datasets

Our Hierarchical Label Expansion (HLE) setup is tested using a single-depth scenario across three datasets: CIFAR100 [65], Stanford Cars [66], and a purpose-built dataset termed ImageNet-Hier100. The CIFAR100 and Stanford Cars datasets have 2 levels of hierarchy with (20,100) and (9,196) classes respectively. We adhere to the hierarchical taxonomy provided in each dataset for our experiments. Furthermore, ImageNet-Hier100, created from a subset of ImageNet [10] based on the WordNet [67] taxonomy, also features 2 hierarchy levels with a total of (10,100) classes. Additional details on how the ImageNet data was curated to construct the ImageNet-Hier100 dataset can be found in the supplementary material.

In addition, we examine the HLE setup under a multiple-depth scenario using two datasets: CIFAR100 [65] and iNaturalist-19 [68]. For CIFAR100, the hierarchical taxonomy outlined in [69] is used, which contains 5 levels of hierarchy with (2, 4, 8, 20, 100) classes, not including the root node. For iNaturalist-19, we employ the taxonomy found in [70], where the dataset has 7 hierarchy levels with (3, 4, 9, 34, 57, 72, 1010) classes, again excluding the root node. Noteworthy is that among these two datasets, only iNaturalist-19 exhibits class imbalance. More details about the number of classes introduced at each task, as well as the characteristics of the dataset, can be found in the supplementary material.

### 2.3.2 Baselines

Our method is evaluated against a variety of established works to establish a baseline. The comparison includes previous rehearsal-based methods designed for traditional continual learning (CL) setups, such as ER [62], BiC [56], and MIR [64]. Our method is also juxtaposed with rehearsal-based methods

that have been employed in more recent CL setups, including RM [1] and CLIB [61]. For methods based on regularization, we compare our approach with EWC++ [63]. All baseline methods were tested in the single-depth scenario. However, in the multiple-depth scenario, we did not include MIR and GDumb. This exclusion was due to GDumb demonstrating the lowest performance and MIR's performance being similar to those of ER, EWC++, and BiC.

### 2.3.3 Scenarios

We carried out our experiments under two conditions: a scenario with a single-depth hierarchical level and another with multiple-depth hierarchical levels, as described in Section 2.1.2 and represented in Figure 2.2. The main focus of our Hierarchical Label Expansion (HLE) setup is on scenarios where the data between tasks do not overlap, which are predominantly evaluated in the context of a single-label scenario. However, for the single-depth hierarchical level, we also conducted investigations under a dual-label scenario where the data carried labels for two hierarchical levels, as explained in Section 2.1.2.

### 2.3.4 Evaluation Metrics

We utilize two key performance indicators in our research: the final classification accuracy across all hierarchical levels and the any-time inference. Final task classification accuracy is a widely accepted metric for assessing the effectiveness of continual learning approaches, as evidenced in past research [63, 71, 72]. This measure evaluates the model's precision after all tasks have been completed, as outlined in our experimental results tables. In addition, gauging the model's capability during the execution of a task is critical for accurately tracking the expansion of knowledge. Given that our framework doesn't have explicit task boundaries and is task-agnostic, we use the any-time inference measure suggested in [61] to assess the model's effectiveness at any given moment. We present the results of any-time inference in graphical form to give a clearer picture of the model's progression over time.

### 2.3.5 Implementation Details

Making use of the codebase provided by [61], we were able to implement previous works, applying AutoAugment [73] and CutMix [74] according to the established experimental protocols. However, to prevent adverse effects on the label distribution for each classifier, we adjusted the use of CutMix to exclusively blend samples from the same hierarchical level. We opted to use ResNet34 as the base feature encoder across all methods. We tailored the batch sizes and update frequencies to each dataset: CIFAR100 was handled with a batch size of 16 and 3 updates per sample, ImageNet-Hier100 and iNaturalist-19 were processed with a batch size of 64 and 0.25 updates per sample, while Stanford Cars was accommodated with a batch size of 64 and 0.5 updates per sample. We allocated memory sizes of 1000, 2000, 5000, and 8000 for Stanford Cars, CIFAR100, ImageNet-Hier100, and iNaturalist-19 respectively. We employed the Adam optimizer [75] with an initial learning rate set to 0.0003, and implemented an exponential learning rate scheduler across all baseline methods with the exception of

GDumb, CLIB, and PL-FMS. For CLIB and our own methodology, we adopted the same learning rate scheduler as used in the original CLIB codebase. The optimization configurations for GDumb and CLIB were derived directly from their respective original papers.

## 2.4 Result Analysis

This section presents an analysis of the conducted experiments based on the guidelines outlined in Section 2.3. The experiments were structured and conducted considering various factors, including single-depth and multiple-depth settings, label regime, prior Continual Learning setups, and an ablation study on the proposed method. Each of these aspects will be discussed in the following subsections.

### 2.4.1 Single-Depth Scenario Analysis

In the scenario involving a single-depth hierarchy, knowledge expansion occurs horizontally within the same hierarchical level, as portrayed in Figure 2.2(a). We assessed the proposed HLE setup using three datasets: CIFAR100 and ImageNet-Hier100, both class-balanced, and Stanford Cars, which is a class-imbalanced dataset. The results are outlined in Table 2.1 and Figure 2.4.

Among the baseline methods, GDumb consistently underperformed, while the effectiveness of other methods varied based on the dataset and hierarchy level. For CIFAR100, RM and BiC were the most successful, outperforming other baseline methods at hierarchy level 1 and 2, respectively. EWC++ and MIR exhibited reasonable performance across both hierarchy levels, and CLIB's performance was similar to RM and BiC at hierarchy level 1. For the ImageNet-Hier100 dataset, MIR was superior at hierarchy level 1, while RM performed best at level 2. BiC demonstrated moderate performance at hierarchy level 1, with EWC++ and ER showing comparable performance at hierarchy level 2. In the Stanford Cars dataset, MIR was the most effective at hierarchy level 1, while CLIB performed commendably at hierarchy level 2. ER and BiC achieved similar results at hierarchy level 1, whereas GDumb and RM performed the worst. At hierarchy level 2, all baseline methods showed similar performance, with an overall accuracy range between 3% and 5%.

Our proposed method, PL-FMS, excelled over all baseline methods across all single-label scenarios, with the most notable improvement observed in the class-imbalanced dataset. It's worth highlighting that RM is a task-conscious learning method that has displayed strong performance under the HLE setup. This performance boost is achieved through a two-stage training strategy, where the model initially trains on stream data samples, then fine-tunes using memory data samples, leading to a performance surge near task boundaries. BiC incorporates a bias correction layer that effectively mitigates dataset bias, but doesn't directly enhance performance near task boundaries. MIR has exhibited impressive performance by selecting samples with high loss importance, which addresses the issue of catastrophic forgetting. However, GDumb consistently demonstrates performance decay due to a fixed regularization coefficient, restricting its adaptability to new tasks.

| Methods | CIFAR100 | | | | | iNaturalist-19 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $H=1$ | $H=2$ | $H=3$ | $H=4$ | $H=5$ | $H=1$ | $H=2$ | $H=3$ | $H=4$ | $H=5$ | $H=6$ | $H=7$ |
| ER | 71.5±4.44 | 58.4±4.58 | 36.6±4.78 | 18.1±4.28 | 7.47±1.61 | 84.9±6.03 | 84.9±0.68 | 59.8±15.5 | 29.3±3.28 | 17.8±3.95 | 13.0±3.95 | 1.50±0.77 |
| EWC++ | 70.9±2.83 | 56.6±4.26 | 35.8±5.93 | 15.8±3.94 | 6.43±1.28 | **87.4±2.38** | 80.7±1.19 | 66.1±9.80 | 29.4±4.48 | 18.1±6.53 | 15.1±5.73 | 1.88±1.15 |
| BiC | 71.6±1.01 | 63.5±2.48 | 54.7±0.61 | 33.8±0.41 | 19.8±0.78 | 79.5±14.4 | 76.3±12.1 | 54.0±27.4 | 22.9±10.3 | 14.8±9.88 | 11.2±7.78 | 1.34±1.41 |
| RM | 74.2±3.99 | 65.0±4.18 | 50.9±1.40 | 37.6±0.60 | 24.5±2.54 | 74.0±5.57 | 69.7±4.21 | 54.4±2.20 | 40.7±1.15 | 37.4±0.85 | 35.1±0.44 | 11.3±0.33 |
| CLIB | 70.6±4.05 | 59.5±1.22 | 47.6±5.06 | 32.6±1.76 | 22.5±2.08 | 87.2±2.26 | 81.3±4.78 | 62.4±4.10 | 41.5±0.97 | 35.3±0.70 | 33.2±1.19 | 8.07±0.94 |
| PL-FMS | **74.5±4.63** | **65.6±3.34** | **56.0±3.66** | **42.7±1.79** | **30.8±1.54** | 86.1±3.15 | **88.4±3.79** | **70.6±3.17** | **49.6±2.42** | **43.9±1.86** | **41.3±2.57** | **13.6±0.28** |

Table 2.2: Experimental results reported for baseline methods and our proposed method evaluated on the HLE setup for the multiple-depth hierarchy scenario in CIFAR100 and iNaturalist-19. The classification accuracy on all hierarchy levels at the final task(%) was measured for all datasets, and the results were averaged over three different random seeds.

### 2.4.2 Multiple-Depth Scenario Analysis

Our proposed HLE setup was evaluated on two datasets: class-balanced CIFAR100 and class-imbalanced iNaturalist-19, with the results reported in Table 2.2 and Figure 2.5. The multiple-depth hierarchy scenario involves vertical knowledge expansion across all hierarchy levels, as shown in Figure 2.2 (b). All baseline methods were included except for GDumb and MIR. GDumb displayed consistently low performance across all datasets and hierarchy levels in single-depth hierarchy. MIR exhibited similar performance to that of ER and EWC++ in most cases, making it redundant to report separately.

Our method, PL-FMS outperforms all baseline methods in CIFAR100, with the performance gap increasing significantly from hierarchy level 4 onwards, as reported in Table 2.2. EWC++ had the lowest performance across all hierarchy levels, while ER performed similarly, but slightly better. RM and BiC had competing performances until hierarchy level 5. Throughout the hierarchy levels, CLIB's performance improved, ranking second among the baselines in the last hierarchy level. Note that most baseline methods suffer from catastrophic forgetting at all task indexes, but the most significant performance drop occurs at task boundary between task 4 and 5, as shown in Figure 2.5. This is due to the fact that the sampling strategy used by baseline methods for training batches fails to consider the biased class distribution induced by sub-categorization. On the other hand, PL-FMS and CLIB exhibit only a mild performance drop by avoiding direct adoption of the stream buffer. PL-FMS outperformed all baseline methods in iNaturalist-19 except for level 1, with RM and CLIB showing the best performance in deeper hierarchy levels. EWC++ performed best only at the coarsest level and rapidly deteriorated thereafter, while BiC exhibited the worst performance overall. ER, EWC++, and BiC exhibited performance decline with increasing hierarchy levels, whereas RM and CLIB demonstrated significant performance improvements in comparison.

In Table 2.2, we observe a similar performance transition across the two datasets. However, at the hierarchy level 7, other baseline methods except for RM and CLIB show performance near 1%, while RM, CLIB, and our method perform much better in the highest hierarchy level with performance above 10%. We believe that ER, EWC++, and BiC exhibit significantly worse performance than RM, CLIB, and our method because they have not been tested under robust conditions, while RM and CLIB were

proposed under more realistic conditions with blurry task boundaries and data streams. These methods are better equipped to deal with hierarchical knowledge formulation, which requires capturing common features throughout hierarchy trees. Overall, we observe that our method performs especially strongly under class imbalance situations, which is more similar to real-world scenarios.

| Methods | Disjoint [30] | Blurry [1] | i-Blurry [61] |
|---|---|---|---|
| ER | 36.6±1.35 | 24.5±1.79 | 38.7±0.51 |
| EWC++ | 36.7±1.04 | 24.3±1.20 | 38.7±1.06 |
| MIR | 34.5±0.97 | 24.0±0.34 | 38.1±0.69 |
| RM | 35.4±1.12 | 37.8±0.81 | 36.7±1.32 |
| GDumb | 26.3±0.43 | 25.9±0.08 | 32.1±0.63 |
| CLIB | 38.0±1.44 | 38.3±0.42 | 43.4±0.44 |
| FMS | **39.2±0.34** | **41.3±1.98** | **45.3±1.02** |

Table 2.3: Experimental results of baseline and FMS evaluated on three CL setups: conventional (disjoint), blurry, and i-Blurry. Test accuracy at the final task (%) was measured for each setup and averaged over three runs with standard deviation reported.

### 2.4.3 Label Regime Analysis

Table 2.1 presents the results of our experiment on a single-depth hierarchy, which we conducted under two scenarios: dual-label and single-label. Our dual-label scenario showed similar trends to the single-label scenario, with GDumb being the worst-performing method. Baseline methods that performed well in the single-label scenario had moderate performance in the dual-label scenario. Notably, incorporating the dual-label scenario resulted in an overall higher performance for the baseline methods in hierarchy level 1, although this was not consistent for hierarchy level 2 and varied among methods. Our proposed method, PL-FMS, consistently showed higher performance in the dual-label scenario across all datasets and hierarchy levels, suggesting that it is more adept at capturing hierarchy information in such scenarios, while still performing well in the single-label scenario against baseline methods.

### 2.4.4 Prior CL Setups Analysis

Table 2.3 reports the results of our proposed HLE setup and baseline methods evaluated on various CL setups. Figure 2.1 depicts the difference between HLE and conventional CL setups. We evaluated the methods on disjoint, blurry [1], and i-Blurry [61] setups to check for code reproducibility and to observe whether our method could perform well on different setups. As reported in [61], CLIB exhibited superior or competitive performance to the other baseline methods across all previous setups, especially with large margin for the i-Blurry setup, since it has design for the i-Blurry setup. Note that our FMS outperformed CLIB for all the prior setups, which indicates that our method is not limited to the suggested HLE setup.

### 2.4.5 Ablation Study

We conducted an ablation study (Table 2.4) to determine the contribution of each component in our proposed method, PL-FMS. The two components, PL and FMS, were evaluated separately to observe

| Methods | CIFAR100(Depth=5) | | | | |
|---|---|---|---|---|---|
| | $H=1$ | $H=2$ | $H=3$ | $H=4$ | $H=5$ |
| Proposed | **73.8±4.63** | **65.6±3.34** | **56.0±3.66** | **42.7±1.79** | **30.8±1.54** |
| w/o PL | 73.5±2.84 | 61.7±2.19 | 48.2±3.40 | 34.6±1.89 | 23.3±2.11 |
| w/o FMS | 71.4±1.83 | 60.5±5.10 | 45.9±2.80 | 30.7±0.87 | 21.5±0.80 |

Table 2.4: PL-FMS was evaluated in an ablation study by comparing its performance with and without each component (PL and FMS) as well as with the combination of both. It indicates the average accuracy across three runs (± std. deviation).

the performance gain achieved by each component. Results indicate that PL contributes more to the overall performance gain compared to FMS. However, when used together, the two components benefit each other and show higher performance gain in hierarchy levels 2-5.

CHAPTER III

# Conclusion

In this thesis, we put forth novel approach to the continual learning (CL) paradigm by introducing Hierarchical Label Expansion (HLE). This novel framework presents hierarchical class incremental task configurations that incorporate an online learning constraint. HLE, as outlined in Section 2.1.1, serves as an extension to conventional CL setups, inspired by the intuitive process of human knowledge expansion. We further propose a Pseudo-Labeling (PL) based memory management scheme and a Flexible Memory Sampling (FMS) methodology to adapt and address the unique challenges posed by our newly devised CL setups. The PL approach, as elaborated in Section 2.2.1, allows for a more effective memory management, leveraging unlabeled data for better retention and transfer of knowledge across tasks. FMS, discussed in depth in Section 2.2.2, enhances the way samples are drawn from memory, ensuring a balanced and diverse representation of prior tasks, thereby alleviating catastrophic forgetting and promoting forward transfer of knowledge.

By implementing these methodologies within our HLE framework, our proposed method demonstrates superior performance compared to the prior works in the field. Our experiments, detailed in Section 2.3, validate our method's effectiveness across all levels of hierarchies, indifferent to the depth and class imbalances inherent in the data. Moreover, the efficacy of our method is not limited to the novel HLE setup. Remarkably, it also outperforms previous approaches on traditional CL setups, such as disjoint, blurry, and i-Blurry setups, further establishing the versatility and robustness of our approach. This suggests that our method, with its innovative memory management and sampling strategies, not only excels in a hierarchical learning scenario but also extends its utility to various CL setups. In conclusion, this work provides a significant contribution to the continual learning domain by not only introducing a new, hierarchical perspective to task configuration but also providing a practical and effective solution for tackling the unique challenges that this perspective presents. This establishes a new paradigm in

continual learning that closely mimics natural learning processes, and opens up new avenues for future research in the field.

# References


[1] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi, "Rainbow memory: Continual learning with a memory of diverse samples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8218–8227. viii, 1, 8, 9, 13, 15, 18

[2] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter, "Continual lifelong learning with neural networks: A review," *Neural networks*, vol. 113, pp. 54–71, 2019. 1, 3, 4

[3] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 139–154. 1, 4, 5

[4] David Lopez-Paz and Marc'Aurelio Ranzato, "Gradient episodic memory for continual learning," *Advances in neural information processing systems*, vol. 30, 2017. 1, 3, 5, 6

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al., "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017. 1, 3, 4

[6] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi, "Online continual learning on class incremental blurry task configuration with anytime inference," *arXiv preprint arXiv:2110.10031*, 2021. 1

[7] Carlos N Silla and Alex A Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, pp. 31–72, 2011. 1, 2

[8] Wei Bi and James T Kwok, "Multi-label classification on tree-and dag-structured hierarchies," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 17–24. 1

[9] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher, "Ask me anything: Dynamic memory networks for natural language processing," in *International conference on machine learning*. PMLR, 2016, pp. 1378–1387. 1

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255. 1, 14

[11] Lijuan Cai and Thomas Hofmann, "Hierarchical document categorization with support vector machines," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, 2004, pp. 78–87. 1

[12] Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski, and Hendrik Blockeel, "Decision trees for hierarchical multi-label classification," *Machine learning*, vol. 73, pp. 185–214, 2008. 1

[13] Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor, "Kernel-based learning of hierarchical multilabel classification models," *Journal of Machine Learning Research*, vol. 7, pp. 1601–1626, 2006. 2

[14] Ricardo Cerri, Rodrigo C Barros, and André CPLF De Carvalho, "Hierarchical multi-label classification using local neural networks," *Journal of Computer and System Sciences*, vol. 80, no. 1, pp. 39–56, 2014. 2

[15] Rodrigo C Barros, Ricardo Cerri, Alex A Freitas, and André CPLF de Carvalho, "Probabilistic clustering for hierarchical multi-label classification of protein functions," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part II 13*. Springer, 2013, pp. 385–400. 2

[16] Ricardo Cerri, Rodrigo C Barros, André CPLF de Carvalho, and Alex A Freitas, "A grammatical evolution algorithm for generation of hierarchical multi-label classification rules," in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 454–461. 2

[17] Isaac Triguero and Celine Vens, "Labelling strategies for hierarchical multi-label classification techniques," *Pattern Recognition*, vol. 56, pp. 170–183, 2016. 2

[18] Ricardo Cerri, Rodrigo C Barros, André C PLF de Carvalho, and Yaochu Jin, "Reduction strategies for hierarchical multi-label classification in protein function prediction," *BMC bioinformatics*, vol. 17, no. 1, pp. 1–24, 2016. 2

[19] Ricardo Cerri, Rodrigo C Barros, and André CPLF de Carvalho, "Hierarchical classification of gene ontology-based protein functions with neural networks," in *2015 international joint conference on neural networks (IJCNN)*. IEEE, 2015, pp. 1–8. 2

[20] Min-Ling Zhang and Zhi-Hua Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern recognition*, vol. 40, no. 7, pp. 2038–2048, 2007. 2

[21] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank, "Classifier chains for multi-label classification," *Machine learning*, vol. 85, pp. 333–359, 2011. 2

[22] Ying Yu, Witold Pedrycz, and Duoqian Miao, "Multi-label classification by exploiting label correlations," *Expert Systems with Applications*, vol. 41, no. 6, pp. 2989–3004, 2014. 2

[23] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros, "Hierarchical multi-label classification networks," in *International conference on machine learning*. PMLR, 2018, pp. 5075–5084. 2

[24] Eduardo P Costa, Ana C Lorena, André CPLF Carvalho, Alex A Freitas, and Nicholas Holden, "Comparing several approaches for hierarchical classification of proteins with decision trees," in *Advances in Bioinformatics and Computational Biology: Second Brazilian Symposium on Bioinformatics, BSB 2007, Angra dos Reis, Brazil, August 29-31, 2007. Proceedings 2*. Springer, 2007, pp. 126–137. 2

[25] Ricardo Cerri, Rodrigo C Barros, and André CPLF de Carvalho, "Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks," in *2011 11th International Conference on Intelligent Systems Design and Applications*. IEEE, 2011, pp. 337–343. 2

[26] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016. 3, 5

[27] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010. 3, 4, 5

[28] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017. 3, 4

[29] Gido M Van de Ven and Andreas S Tolias, "Three scenarios for continual learning," *arXiv preprint arXiv:1904.07734*, 2019. 3

[30] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio, "Gradient based sample selection for online continual learning," *NeurIPS*, vol. 32, 2019. 4, 18

[31] Enrico Fini, Stéphane Lathuiliere, Enver Sangineto, Moin Nabi, and Elisa Ricci, "Online continual learning under extreme memory constraints," in *ECCV*. Springer, 2020, pp. 720–735. 4

[32] Yiduo Guo, Bing Liu, and Dongyan Zhao, "Online continual learning through mutual information maximization," in *ICML*. PMLR, 2022, pp. 8109–8126. 4

[33] Jiangpeng He, Runyu Mao, Zeman Shao, and Fengqing Zhu, "Incremental learning in online scenario," in *CVPR*, 2020, pp. 13926–13935. 4

[34] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny, "Efficient lifelong learning with a-gem," *arXiv preprint arXiv:1812.00420*, 2018. 4

[35] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania, "Gdumb: A simple approach that questions our progress in continual learning," in *ECCV*. Springer, 2020, pp. 524–540. 4, 13

[36] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," in *ICML*. PMLR, 2018, pp. 4548–4557. 4

[37] Maruan Al-Shedivat, Trapit Bansal, Yura Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel, "Continuous adaptation via meta-learning in nonstationary and competitive environments," in *ICLR*, 2018. 4

[38] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars, "Task-free continual learning," in *CVPR*, 2019, pp. 11254–11263. 4

[39] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong, "Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting," in *ICML*. PMLR, 2019, pp. 3925–3934. 4

[40] Priya Donti, Brandon Amos, and J Zico Kolter, "Task-based end-to-end model learning in stochastic optimization," *NeurIPS*, vol. 30, 2017. 4

[41] Zhizhong Li and Derek Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017. 4

[42] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M. López, and Andrew D. Bagdanov, "Rotate your networks: Better weight consolidation and less catastrophic forgetting," in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2262–2268. 5

[43] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell, "Progress amp; compress: A scalable framework for continual learning," in *Proceedings of the 35th International Conference on Machine Learning*, Jennifer Dy and Andreas Krause, Eds. 10–15 Jul 2018, vol. 80 of *Proceedings of Machine Learning Research*, pp. 4528–4537, PMLR. 5

[44] Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 5

[45] Friedemann Zenke, Ben Poole, and Surya Ganguli, "Continual learning through synaptic intelligence," in *International conference on machine learning*. PMLR, 2017, pp. 3987–3995. 5

[46] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5

[47] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang, "Lifelong learning with dynamically expandable networks," 2018. 5

[48] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen, "Compacting, picking and growing for unforgetting continual learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc. 5

[49] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim, "A neural dirichlet process mixture model for task-free continual learning," 2020. 5

[50] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato, "On tiny episodic memories in continual learning," *arXiv preprint arXiv:1902.10486*, 2019. 5, 6

[51] Jeffrey S. Vitter, "Random sampling with a reservoir," *ACM Trans. Math. Softw.*, vol. 11, no. 1, pp. 37–57, mar 1985. 6

[52] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio, "Gradient based sample selection for online continual learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc. 6

[53] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun, "Mnemonics training: Multi-class incremental learning without forgetting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 6

[54] Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars, "Online continual learning with maximally interfered retrieval," *ArXiv*, vol. abs/1908.04742, 2019. 6

[55] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang, "Online class-incremental continual learning with adversarial shapley value," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, pp. 9630–9638, 2021. 6

[56] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu, "Large scale incremental learning," in *CVPR*, 2019, pp. 374–382. 6, 14

[57] Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven HOI, "Contextual transformation networks for online continual learning," in *International Conference on Learning Representations*, 2021. 6

[58] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny, "Efficient lifelong learning with a-gem," 2019. 6

[59] Michael McCloskey and Neal J Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*, vol. 24, pp. 109–165. Elsevier, 1989. 6

[60] Robert M French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999. 6

[61] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi, "Online continual learning on class incremental blurry task configuration with anytime inference," in *ICLR*, 2022. 8, 9, 13, 15, 18

[62] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne, "Experience replay for continual learning," *NeurIPS*, vol. 32, 2019. 13, 14

[63] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *ECCV*, 2018, pp. 532–547. 13, 15

[64] Eugene Belilovsky Massimo Caccia Min Lin Laurent Charlin Tinne Tuytelaars Rahaf Aljundi, Lucas Caccia, "Online continual learning with maximally interfered retrieval," vol. 32, pp. 11849–11860, 2019a. 13, 14

[65] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009. 14

[66] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei, "3d object representations for fine-grained categorization," in *CVPR*, 2013, pp. 554–561. 14

[67] George A Miller, *WordNet: An electronic lexical database*, MIT press, 1998. 14

[68] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie, "The inaturalist species classification and detection dataset," in *CVPR*, 2018, pp. 8769–8778. 14

[69] Vivien Sainte Fare Garnot and Loic Landrieu, "Leveraging class hierarchies with metric-guided prototype learning," *arXiv preprint arXiv:2007.03047*, 2020. 14

[70] Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A Lord, "Making better mistakes: Leveraging class hierarchies with deep networks," in *CVPR*, 2020, pp. 12506–12515. 14

[71] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," *NeurIPS*, vol. 31, 2018. 15

[72] Gido M van de Ven and Andreas S Tolias, "Three continual learning scenarios and a case for generative replay," 2018. 15

[73] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le, "Autoaugment: Learning augmentation strategies from data," in *CVPR*, 2019, pp. 113–123. 15

[74] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019, pp. 6023–6032. 15

[75] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 15