Online Continual Learning on Hierarchical Label Expansion

Byung Hyun Lee[1,*], Okchul Jung[1,*], Jonghyun Choi[3,†] and Se Young Chun[1,2,†]
[1]Dept. of ECE, [2]INMC & IPAI, Seoul National University, Republic of Korea,
[3]Yonsei University, Republic of Korea
{ldlqudgus756,luckyjung96}@snu.ac.kr jc@yonsei.ac.kr sychun@snu.ac.kr

Abstract

Continual learning (CL) enables models to adapt to new tasks and environments without forgetting previously learned knowledge. While current CL setups have ignored the relationship between labels in the past task and the new task with or without small task overlaps, real-world scenarios often involve hierarchical relationships between old and new tasks, posing another challenge for traditional CL approaches. To address this challenge, we propose a novel multi-level hierarchical class incremental task configuration with an online learning constraint, called hierarchical label expansion (HLE). Our configuration allows a network to first learn coarse-grained classes, with data labels continually expanding to more fine-grained classes in various hierarchy depths. To tackle this new setup, we propose a rehearsal-based method that utilizes hierarchy-aware pseudo-labeling to incorporate hierarchical class information. Additionally, we propose a simple yet effective memory management and sampling strategy that selectively adopts samples of newly encountered classes. Our experiments demonstrate that our proposed method can effectively use hierarchy on our HLE setup to improve classification accuracy across all levels of hierarchies, regardless of depth and class imbalance ratio, outperforming prior state-of-the-art works by significant margins while also outperforming them on the conventional disjoint, blurry and i-Blurry CL setups.

1. Introduction

In real-world continual learning scenarios, new knowledge often augments existing understanding, typically following a hierarchical path from general to specific classes. This hierarchical structure is not an anomaly, but rather an inherent part of many disciplines. The schema theory [9; 43] in cognitive psychology and the conceptual clustering theory [28] in machine learning both emphasize hierarchical organization of knowledge. The COBWEB algorithm [20], a prominent machine learning method, uses hierarchical clustering for grouping related instances into meaningful categories. Hierarchical organization is also observed in biology's taxonomy theory [8], classifying organisms based on shared traits, and in chemistry [27], where elements are arranged hierarchically according to their atomic properties. However, despite the prevalence of hierarchical relationships in these areas, many previous continual learning works [3; 5; 6; 31] do not fully incorporate these relationships. This may be an area that needs more attention, as hierarchical relationships could play a role in knowledge evolution in incremental learning.

Here we introduce a novel CL setup called Hierarchical Label Expansion (HLE), designed to account for hierarchical class relationships in task-free online CL. In HLE, class learning is incremental, with fine-grained classes derived from prior coarse-grained ones, effectively mirroring real-world knowledge accumulation. As our proposed approach is designed for online continual learning, where data is seen only once in the data stream, each task's data is disjoint. We assess our models' performance using any-time inference [31] and evaluate classification accuracy for all levels of hierarchy. This demonstrates the potential of our approach to complement existing CL methods and enhance their evaluation. HLE encompasses both single and multiple hierarchy depths, as well as balanced and imbalanced class data scenarios. To tackle the CL on HLE, we propose a new CL method that utilizes pseudo-labeling based memory management (PL) and flexible memory sampling (FMS). This method effectively exploits hierarchy information between class labels in the dataset, resembling how knowledge is accumulated in real-world scenarios. Extensive experiments demonstrate that our approach outperforms state-of-the-art methods by substantial margins in HLE, while remaining superior in performance on existing CL setups including disjoint, blurry [5] and i-Blurry [31].

We summarize our contributions as follows:

1. We propose new online class-incremental, hierarchy-aware, task-free CL setups called HLE, designed to

∗ Equal contribution, † Corresponding authors.

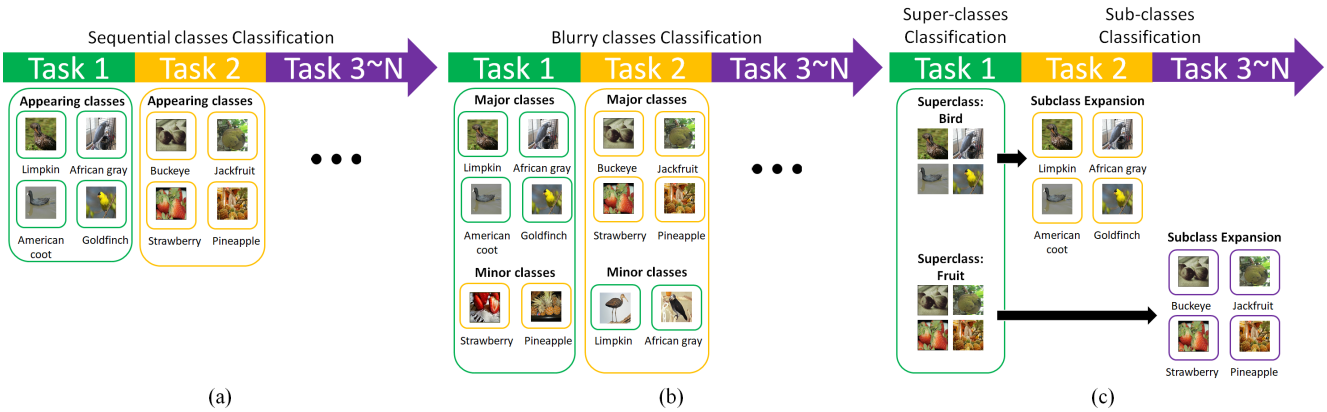


Figure 1. Comparison sketch between conventional, blurry, and our HLE setups. (a) Conventional task-free online CL setup gradually introduces new classes and classifies data without task identification (b) Blurry task-free online CL setup where classes are divided into major and minor categories at each 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.

simulate how knowledge is accumulated in real-world scenarios.

2. We propose a new online CL method, PL-FMS, that consists of pseudo-labeling (PL) based memory management and flexible memory sampling (FMS) to better exploit hierarchy information and address the HLE setup.

3. We evaluate our approach on CIFAR100, Stanford-Cars, iNaturalist-19, and a novel dataset named ImageNet-Hier100, demonstrating that our method outperforms prior state-of-the-art works by significant margins on HLE while still outperforming them on the existing disjoint, blurry and i-Blurry CL setups.

2. Related Work

Continual learning setups. Continual Learning (CL) setups can be classified into three categories: task-incremental, class-incremental, and domain-incremental learning setups [15; 46]. Our work focuses on the class-incremental learning setting proposed by [40], where task identity is not given during inference, and the model is required to solve each task seen so far and infer which task it is presented with. CL setups can be classified as either online [4; 19; 24; 26] or offline [2; 12; 38; 40; 44]. Our work focuses on the more challenging online CL setup where streamed samples are only used once, compared to the offline CL setup where data from each task can be used multiple times to train the model. CL setups can also be categorized as task-free [1; 3; 34] or task-based [18; 35; 44; 45]. Our work focuses on the former, where the model continuously learns and adapts to incoming data without explicit task information, unlike the latter where the model is informed about the tasks it must learn and adapt to.

Despite the considerable attention given to enhancing CL methods, their evaluation has been limited to rather restricted CL settings. To address this, novel CL setups with blurry task boundaries and corrupted labels in data stream [5; 6; 31] have been proposed. A CL setup where classes are shared across tasks and presented sequentially as a stream with limited access to previous data was proposed by [5], while [6] suggested an online blurry CL setup with noisy labels. Recently, a new setup called 'i-Blurry' [31] has been proposed, which combines the advantages of both blurry and disjoint setups by allowing continuous encounters of overlapping classes without suffering from restrictions of blurry and disjoint. However, earlier works all assumed independent class labels, which is often not the case in reality. Our work proposes a complementary CL setup that models hierarchically correlating relationships between labels for online learning depicted in Figure 1.

Hierarchical classification. Various studies have utilized data's hierarchical structure to enhance tasks like image classification [7; 10; 29], multi-label classification [49], object recognition [41], and semi-supervised approaches [22; 48]. The hierarchical taxonomy is typically employed through label-embedding, hierarchical architecture-based, and hierarchical loss-based methods.

The label-embedding method maps class labels to vectors to represent semantic relationships and optimizes a loss on these embedded soft vectors. DeViSE [21] maximizes the cosine similarity between image and label embeddings. It maps target classes to a unit hypersphere and penalizes the output that is more similar to false label embeddings using a ranking loss. Liu *et al.* [36] use hyperbolic geometry to learn hierarchical representations and minimize the Poincaré distance between Poincaré label embeddings and image feature embeddings, similar to DeViSE.

Hierarchical architecture-based methods incorporate

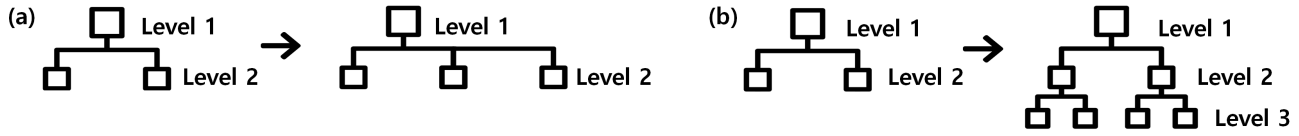


Figure 2. An illustration of two HLE scenarios. (a) In single-depth scenario, fine-grained classes grow horizontally from coarse-grained ones within the same level. (b) In multiple-depth scenario, classes grow vertically from coarse to fine across different hierarchy levels.

class hierarchy into the classifier architecture. Wu *et al.* [50] jointly optimize a multi-task loss function with cross-entropy loss applied at each hierarchy level. Redmon *et al.* [41] propose a probabilistic model, YOLOv2, for object detection and classification, with softmax applied at every coarse-category level to address the mutual exclusion of all classes in conventional softmax classifier. Chang *et al.* [10] propose a multi-granularity classification architecture that uses level-specific classifiers to optimize fine-grained and coarse-grained recognition separately and improve fine-grained classification performance.

Hierarchical loss-based method incorporates hierarchical class relationships into the loss function and penalizes incorrect predictions while encouraging those that follow the hierarchy. Deng *et al.* [16] directly minimized the expected WordNet LCA height using kNN- and SVM-based classifiers, while Zhao *et al.* [53] modified multi-class logistic regression and added an 'overlapping-group lasso penalty' to encourage the use of similar features for closely related classes. Bertinetto *et al.* [7] proposed the hierarchical cross-entropy approach, where the loss function is based on conditional probabilities given parent-class probabilities.

3. Hierarchical Label Expansion

In this section, we introduce our proposed HLE setup and present its configurations. Section 3.1 details the setup formulation, where the model is provided with samples only for the classes belonging to a single hierarchy level for each task. Section 3.2 describes the construction of single and multiple hierarchy depth scenarios in HLE to observe knowledge expansion at different levels.

3.1. Hierarchical CL Configurations

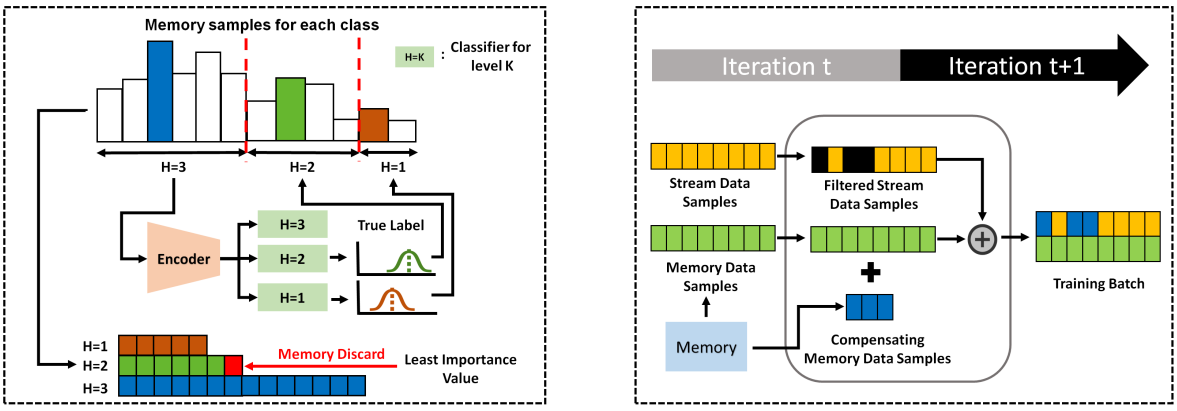
Our HLE setup involves task-free online learning, where the model incrementally learns classes from various hierarchies both vertically and horizontally, agnostic to the task boundaries. The model is presumed to first learn coarse-grained classes, followed by fine-grained classes. Figure 1(c) provides an overview of the HLE setup.

Formally, we consider the model encounters a stream of data points denoted by $\mathcal{T} = ((x_1, y_1), (x_2, y_2), \cdots)$, where (x_j, y_j) is sampled from a data distribution $\mathcal{D}_{\mathbb{X}\times\mathbb{Y}}$, $x_j \in \mathbb{X}$ is the jth input (image) for the model, and $y_j \in \mathbb{Y}$ is the class label of x_j. Often, the sequential tasks with the index k can divide the data stream $\mathcal{T}$ into disjoint sub-sequences $\mathcal{T}_1, \mathcal{T}_2, \cdots$, where $\mathcal{T}_k = ((x_j, y_j))_{j=t(k)}^{t(k+1)-1}$ and $t(k)$ is the start sample index for the k-th task. We define the class subset for the k-th task as $\mathbb{Y}_k \subseteq \mathbb{Y}$, which represents the set of classes that the model encounters during the kth-task. The conventional CL usually assumes that the sampling distribution varies over time and the sampling distributions for tasks are mutually exclusive, *i.e.*, $\mathbb{Y}_k \cap \mathbb{Y}_l = \emptyset$ for $k \neq l$. However, there exist numerous scenarios where more practical contexts need to be taken into account for reality. For example, the i-Blurry CL setup [31] assumes that each task has both shared subset of classes $\mathbb{Y}^s$, trained throughout the learning process, and disjoint subset of classes $\mathbb{Y}_k^d$, trained only at a specific task. For this case, the class subset $\mathbb{Y}_k$ is defined as $\mathbb{Y}_k = \mathbb{Y}^s \cup \mathbb{Y}_k^d$, which implies that $\mathbb{Y}_k \cap \mathbb{Y}_l = \mathbb{Y}^s \neq \emptyset$.

In a different direction to complement existing CL setups, our HLE allows more structures on $\mathbb{Y}$ by constructing a label relation between classes in $\mathbb{Y}$. Specifically, we consider that $\mathbb{Y}$ consists of classes from H levels, so $\mathbb{Y} = \bigcup_{h=1}^{H} \mathbb{Y}^h$ and $\mathbb{Y}^h \cap \mathbb{Y}^{h'} = \emptyset$ where $\mathbb{Y}^h$ is the label subset whose hierarchy level is h. By h, the smaller value of h represents the hierarchy level for more coarse-grained classes. In the HLE setup, each task conducts the label expansion for a subset of classes in level h to their fine-grained classes in level $(h+1)$. That is, the labels are expanded by one level during a task. Let $\mathbb{Y}_k^h \subseteq \mathbb{Y}^h$ be the label subset for level h that has been trained by the model until the k-th task. For the $(k+1)$-th task, a subset $\bar{\mathbb{Y}}_{k+1}^h$ of $\mathbb{Y}_k^h$ is selected to be newly expanded to a set of their fine-grained classes $\mathbb{Y}_{k,\text{new}}^{h+1}$, resulting in $\mathbb{Y}_k = \mathbb{Y}_{k,\text{new}}^{h+1}$. To handle multiple hierarchy levels, our model consists of an encoder f for feature embedding and multiple classifiers $\{g^h\}_{h=1}^{H}$, each corresponding to a hierarchy level. Specifically, $g^h(f(x))$ predicts the classes within level h encountered until the current iteration. Regardless of its hierarchical position, each input is assigned a single label during training, and the model remains unaware of the hierarchy among classes. The hierarchy level is instead given as a soft hint to the model.

3.2. Hierarchical CL Depth Scenarios

Our HLE setup includes two scenarios: single-depth and multiple-depth scenarios (existing setups are 0-depth), for hierarchical label expansion as depicted in Figure 2. In the single-depth scenario, incremental learning is observed horizontally within the same hierarchy level, while in the multiple-depth scenario, new classes are introduced with in-



(a) Pseudo-Labeling (PL) based Memory Management

(b) Flexible Memory Sampling (FMS)

Figure 3. Sketch of our proposed method, PL-FMS's two components: PL and FMS. (a) Pseudo-Labeling based memory management (PL) outlines the method of discarding a data sample, which will be replaced with incoming data, based on its effect on reducing loss, irrespective of its label's nature (true or pseudo)..(b) Flexible Memory Sampling (FMS) shows formation of the training batch by filtering and compensating data samples.

creasing levels of specificity vertically. For the single-depth scenario, the model learns for all parent classes at the first task and partially expands them through subsequent tasks. The single-depth scenario involves horizontal incremental learning within the same hierarchy level, starting with parent classes and broadening them in following tasks. This scenario is further explored through dual-label (overlapping data) and single-label (disjoint data) setups, as detailed in Table 1. In the multiple-depth scenario, the model's ability to learn and expand hierarchical knowledge is tested while navigating complex vertical hierarchies by increasing the hierarchy level of classes to be learned for subsequent tasks, meaning that the model learns for classes of hierarchy level h at the hth-task.

4. Pseudo Labeling-based Flexible Memory Sampling (PL-FMS)

In this section, we introduce our method which employs a rehearsal-based incremental learning approach, where models are trained using previously seen data from a stream buffer. Our method incorporates pseudo-labeling to fully utilize the hierarchical class relationship and a memory sampling strategy to flexibly build the training batch from stored and incoming data. Further details on our method's two main components, Pseudo-Labeling (PL) based Memory Management and Flexible Memory Sampling (FMS), are followed in sections 4.1 and 4.2, respectively.

4.1. Pseudo-Labeling based Memory Management

We introduce a novel memory management strategy that uses the model's predictions to generate pseudo-labels for each hierarchy level in our HLE setup, as shown in Figure 3(a). This strategy is referred to as Pseudo-Labeling (PL) based memory management.

Basically, it first finds the modal label that are the most frequent in memory for class balance [5; 31; 38]. Let $\mathcal{M}$ be the memory that stores samples from the data stream and $\mathcal{M}_y = \{(x_n, y_n) \in \mathcal{M} | y_n = y\}$ be the subset of the memory whose samples belong to the class y. For rehearsal-based method, we need to remove a sample from the memory to accept a new sample once $|\mathcal{M}|$ reaches the maximum memory size. To achieve this, we identify the class with the highest number of samples in the memory, which we denote as $\bar{y} = \arg\max_y |\mathcal{M}_y|$. Prior works [5; 31; 38] have typically removed samples only from $\mathcal{M}_{\bar{y}}$. To further improve the efficiency, we propose to consider samples from other classes hierarchically related to $\bar{y}$. To do so, we use the class probability predicted by the network, denoted as $p^h(x) = \sigma(g^h(f(x))) \in \mathbb{R}^{|\mathbb{Y}^h|}$ for level h, where $\sigma(\cdot)$ is the soft-max function.

We use the model to predict classes that are hierarchically related to $\bar{y}$. We do this by accumulating the model's predictions for samples in $\mathcal{M}_{\bar{y}}$ for all levels, except for the level of $\bar{y}$. The classes with the most predictions for each level are then identified, defined as:

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

where $\mathbf{1}_y(x)$ is an indicator function defined as:

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

In other words, the class at level h that has the most predictions in $\mathcal{M}_{\bar{y}}$ is deemed as the class hierarchically related to $\bar{y}$. By using the predicted classes for the other levels, we

construct an index set of candidate samples to be removed from the memory as:

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

To determine the index of a sample to remove, we adopt the sample-wise loss importance value, $\mathcal{H}_n$, introduced by [31]. Specifically, $\mathcal{H}_n$ is computed as:

$$\mathcal{H}_n = L(\theta) - L(\theta_n),$$

where $L(\theta) = \sum_{(x,y)\in\mathcal{M}} l(x, y; \theta)$ is the averaged loss in the memory and $\theta_n = \theta - \nabla_\theta l(x_n, y_n; \theta)$. By using the loss importance value, we find the index $\hat{j}$ of the sample to remove whose measured importance is the least:

$$\hat{j} = \arg\min_{j\in\mathcal{I}_{\bar{y}}} \mathcal{H}_j. \quad (3)$$

That is, we measure the decrease in loss for each sample during training and subsequently removes the data from the memory whose loss decrease is the least.

4.2. Flexible Memory Sampling (FMS)

Prior rehearsal-based methods [13; 26; 39; 51] proposed directly including the stream buffer in training, leading to bias toward the data stream distribution and negatively impacting the model's performance. Using only memory samples for training was also suggested by [31], but it limited adaptability to new classes. To balance the usage of memory and data stream, we propose Flexible Memory Sampling (FMS), a simple yet effective sampling strategy that flexibly adjusts the number of stream samples in the training batch. The apporach is depicted in Figure 3 (b).

To construct a training batch B_t at iteration t, ER utilizes all samples in the stream buffer S_t and takes samples from the memory in an amount equal to $|S_t|$, which results in $|B_t| = 2|S_t|$. Unlike ER, FMS randomly excludes samples from S_t in the training process. Let T_c be the iteration when the class c has been encountered for the first time. Then, we selectively include stream samples of class c with increasing probability as $t - T_c$ gets larger, gradually adopting new classes from the stream buffer. In proportional to the value, the probability to include a stream sample of class c is determined by a Bernoulli distribution for each class as:

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

where T is a hyper-parameter that adjusts how fast the network adopts the stream samples for training. Therefore, it resembles the memory-only training of [31] immediately after encountering new classes, while it becomes more like the sampling approach of ER as $t - T_c$ gets larger.

By combining those two strategies, we call our proposed method Pseudo Labeling-based Flexible Memory Sampling (PL-FMS). A detailed description of the algorithm for PL-FMS can be found in the supplementary material.

5. Experiments

5.1. Experimental Setups

Datasets. We evaluate the Hierarchical Label Expansion (HLE) setup with a single-depth scenario on three datasets: **CIFAR100** [33], **Stanford Cars** [32], and a newly constructed dataset called **ImageNet-Hier100**. CIFAR100 and Stanford Cars datasets each have 2 levels of hierarchy, with a total of (20,100) classes and (9,196) classes, respectively. The hierarchical taxonomy provided in each dataset was followed for the experiments. Additionally, we artificially constructed the ImageNet-Hier100, which is a subset of ImageNet [17] based on the taxonomy of WordNet [37]. This dataset also has 2 levels of hierarchy with a total of (10,100) classes. Details on the curation of ImageNet data to construct ImageNet-Hier100 dataset are available in the supplementary material.

We evaluate the HLE setup with a multiple-depth scenario on two datasets: **CIFAR100** [33] and **iNaturalist-19** [47]. For CIFAR100, we follow the hierarchical taxonomy as described in [23], where the dataset has 5 levels of hierarchy with (2, 4, 8, 20, 100) classes, excluding the root node. For iNaturalist-19, we use the taxonomy in [7], where the dataset has 7 levels of hierarchy with (3, 4, 9, 34, 57, 72, 1010) classes, excluding the root node. Notably, only the iNaturalist-19 dataset is class-imbalanced among the two datasets. Further details regarding the number of classes introduced at each task, dataset characteristics are available in the supplementary material.

Baselines. To provide a baseline for our method, we compare it with a range of previous works. We compare our rehearsal-based methods with previous works that were conducted under conventional CL setup, including **ER** [42], **EWC++** [11], and **MIR** [39]. We also compare our rehearsal-based methods with works that were used in recently proposed CL setup, including **RM** [5] and **CLIB** [31]. For regularization-based methods, we compare our methods with **BiC** [51] and **GDumb** [38]. In the single-depth scenario, we evaluated all baseline methods, while in the multiple-depth scenario, we excluded MIR and GDumb as GDumb had the lowest performance and MIR had similar performance to ER, EWC++, and BiC. Further details about the experimental setup are available in the supplementary material.

Scenarios. We conducted experiments in two scenarios: a single-depth hierarchy level and a multiple-depth hierarchy level, as detailed in Section 3.2 and illustrated in Figure 2. Our HLE setup assumes disjoint data between tasks and is

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	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++	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	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	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	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	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	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 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.

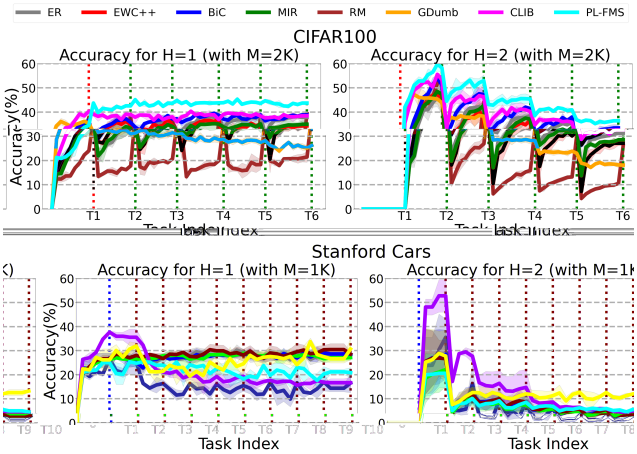


Figure 4. Any-time inference results on CIFAR100 and Stanford Cars datasets for single-depth hierarchy. H=1 is parent classes and H=2 child classes. Task index 1 receives parent class labeled data and subsequent indexes receive child class labeled data. Each data point shows average accuracy over three runs (± std. deviation).

primarily evaluated under the single-label scenario. However, as described in Section 3.2, we also conducted experiments under a dual-label scenario for the single-depth hierarchy level, where data had labels for both hierarchy levels.

Evaluation metrics. We employ two primary evaluation metrics in our study: final classification accuracy for all hierarchy levels and any-time inference. Classification accuracy at the final task is a commonly used metric in evaluating continual learning methods, as demonstrated in previous works [11; 25; 46]. This metric measures the model's accuracy after all tasks have been learned as reported in the experimental tables. We also use any-time inference, as recommended in [31], to assess the model's performance at any given time, crucial for observing knowledge expansion in our task-free setup. We report final accuracy in tables and any-time inference in figures for clarity over time. More details on these metrics are in the supplementary material.

Implementation details. We implemented prior work using the [31] codebase, and applied AutoAugment [14] and CutMix [52] as per their setup, but modified CutMix to mix samples only from the same hierarchy level to preserve the label distribution. We used ResNet34 as the base feature encoder across all methods, and adjusted batch sizes and update rates for each dataset: CIFAR100 (16, 3), ImageNet-Hier100 and iNaturalist-19 (64, 0.25), Stanford Cars (64, 0.5). Memory sizes were 1000, 2000, 5000, and 8000 for Stanford Cars, CIFAR100, ImageNet-Hier100, and iNaturalist-19, respectively. All methods except GDumb, CLIB, and PL-FMS used the Adam optimizer [30] with an initial learning rate of 0.0003 and an exponential learning rate scheduler. CLIB and our method used the same scheduler following the CLIB codebase. GDumb and CLIB adhered to their original optimization configurations.

5.2. Single-Depth Scenario Analysis

In the single-depth hierarchy scenario, knowledge expands horizontally within the same hierarchy level, as depicted in Figure 2(a). The proposed HLE setup was evaluated on three datasets: CIFAR100 and ImageNet-Hier100, both class-balanced, and Stanford Cars, a class-imbalanced dataset, as reported in Table 1 and Figure 4.

Among baseline methods, GDumb consistently showed the worst performance, while other methods showed varying performance depending on the dataset and hierarchy level. In CIFAR100, RM and BiC outperformed other baseline methods in hierarchy level 1 and 2, respectively. EWC++ and MIR demonstrated moderate performance in both hierarchy levels, while CLIB exhibited comparable performance to RM and BiC in hierarchy level 1. In ImageNet-Hier100, MIR showed the best performance in hierarchy level 1, while RM exhibited the best performance in hierarchy level 2. BiC showed moderate performance in hierarchy level 1, while EWC++ and ER demonstrated similar performance in hierarchy level 2. For Stanford Cars, MIR showed the best performance in hierarchy level 1, while CLIB performed well in hierarchy level 2. ER and

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. 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.

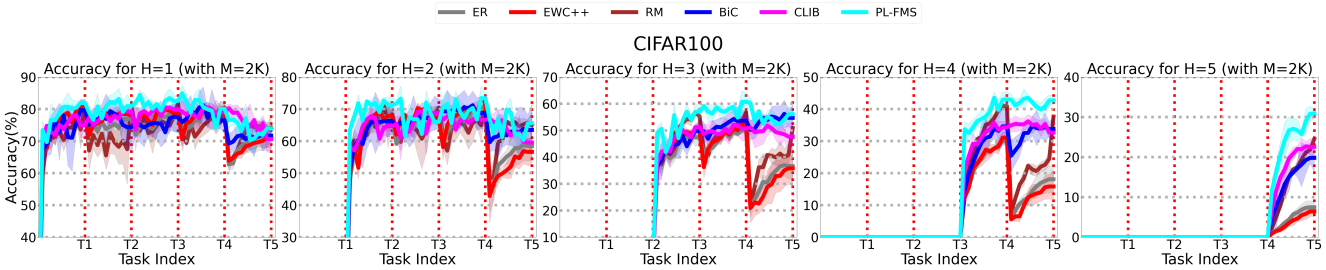


Figure 5. Any-time inference results on CIFAR100 dataset for multiple-depth hierarchy. H=1 represents the coarsest level and H=5 represents the finest level of class hierarchy. The dotted line represents the point at which the model is fully given the task data for the corresponding task index. The reported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

BiC displayed similar performance in hierarchy level 1, while GDumb and RM exhibited the lowest and similar performance. In hierarchy level 2, all baseline methods showed similar performance, with overall accuracy between 3% and 5%. Our proposed method, PL-FMS, outperformed every baseline method in all single-label scenarios, with the largest improvement seen in the class-imbalanced dataset. It is worth noting that RM is a task-aware learning method that has demonstrated high performance under the HLE setup. This is achieved by a two-stage training approach, where the model is first trained on stream data samples and then fine-tuned using memory data samples resulting in an upsurge in performance near task boundaries. BiC includes a bias correction layer that effectively reduces dataset bias, but it does not directly improve performance near task boundaries. Additionally, MIR has shown significant performance by selecting high-loss importance samples, which helps to address the problem of catastrophic forgetting. However, GDumb consistently exhibits performance decay due to its fixed regularization coefficient, which limits its ability to adapt to new tasks.

5.3. 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 and Figure 5. The multiple-depth hierarchy scenario involves vertical knowledge expansion across all hierarchy levels, as shown in Figure 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. 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 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, 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 [4]	Blurry [5]	i-Blurry [31]
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 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.

5.4. Label Regime Analysis

Table 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.

5.5. Prior CL Setups Analysis

Table 3 reports the results of our proposed HLE setup and baseline methods evaluated on various CL setups. Figure 1 depicts the difference between HLE and conventional CL setups. We evaluated the methods on disjoint, blurry [5], and i-Blurry [31] setups to check for code reproducibility and to observe whether our method could perform well on different setups. As reported in [31], 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.

Methods	Multiple-Depth					Single-Label		Dual-Label	
	H=1	H=2	H=3	H=4	H=5	H=1	H=2	H=1	H=2
Proposed	**73.8**	**65.6**	**56.0**	**42.7**	**30.8**	**43.7**	**36.4**	**49.0**	**39.5**
w/o PL	73.5	61.7	48.2	34.6	23.3	41.3	33.2	46.1	33.9
w/o FMS	71.4	60.5	45.9	30.7	21.5	39.6	31.5	43.8	32.8

Table 4. Ablation study conducted on CIFAR100 to compare the performance of PL-FMS, with and without the PL and FMS components, as well as their combination. Average accuracy across three runs is reported.

5.6. Ablation Study

In Table 4, we conducted an ablation study to determine the contribution of each component in our proposed method for multi-depth, single-label, and dual label scenarios. The two components, PL and FMS, were evaluated separately to observe 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 for all scenarios.

Methods	Multiple-Depth				
	H=1	H=2	H=3	H=4	H=5
Oracle	94.6±0.7	92.3±1.2	84.1±0.8	73.5±1.2	61.1±1.4
PL-FMS-T	87.7±0.2	81.9±0.5	69.0±1.7	51.5±2.0	35.2±1.3
PL-FMS	74.5±4.6	65.6±3.3	56.0±3.7	42.7±1.8	30.8±1.5

Table 5. CIFAR100 multiple-depth scenario results (%) across three runs. 'Oracle': All-classes-at-once (offline batch learning, assuming unlimited access to true class hierarchy labels during training). 'PL-FMS-T': PL-FMS with true class hierarchy labels.

We also compared our method against an oracle result obtained via offline batch learning on all classes simultaneously and an approach leveraging true class hierarchy labels (PL-FMS-T). As seen in Table 5, our method gains from scenarios where true class hierarchy is available.

Methods	Multiple-Depth					Single-Label		Dual-Label	
	H=1	H=2	H=3	H=4	H=5	H=1	H=2	H=1	H=2
T=500	79.0	62.4	52.1	37.2	27.4	40.0	32.8	47.6	36.7
T=1,500	73.0	65.4	50.2	36.4	28.7	38.3	32.8	46.6	36.9
T=5,000	**74.5**	**65.6**	**56.0**	**42.7**	**30.8**	**43.7**	**36.4**	**49.0**	**39.5**
T=15,000	72.7	64.7	50.5	34.6	28.3	41.4	33.5	48.8	38.6
T=50,000	77.9	67.0	52.9	39.0	28.8	39.0	31.7	46.3	34.4

Table 6. Effect of hyperparameter T in Eq. 4 (%) of PL-FMS on CIFAR100.

Table 6 shows how the hyperparameter T in PL-FMS, controlling the network's adaptation speed during training, affects performance. Choosing a value of 5,000 for T yielded the highest accuracy, especially in fine-grained hierarchy classes across all scenarios.

6. Conclusion

In this work, we propose hierarchical label expansion (HLE), novel hierarchical class incremental task configurations with an online learning constraint, that complement existing CL setups by mimicking knowledge expansion. Then, we propose Pseudo-Labeling (PL) based memory management and Flexible Memory Sampling (FMS) to tackle this newly proposed CL setups for fully exploiting the inherent data hierarchy. Our proposed method outperforms prior state-of-the-art works by significant margins on our HLE setups across all levels of hierarchies, regardless of depth and class imbalance while also outperforming them on the previous disjoint, blurry and i-Blurry CL setups.

Acknowledgments

This work was partly supported by the National Research Foundation of Korea(NRF) grants funded by the Korea government (MSIT) (NRF-2022R1A4A1030579, NRF-2022M3C1A309202211, NRF-2022R1A2C4002300 5%), IITP grants (No.2020-0-01361, AI GS Program (Yonsei University) 5%, No.2021-0-02068, AI Innovation Hub 5%, 2022-0-00077 5%, 2022-0-00113 5%, 2022-0-00959 5%) funded by the Korea government (MSIT), and Creative-Pioneering Researchers Program through Seoul National University. Also, the authors acknowledged the financial support from the BK21 FOUR program of the Education and Research Program for Future ICT Pioneers, Seoul National University.

References

[1] 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.

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, pages 139–154, 2018.

[3] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *CVPR*, pages 11254–11263, 2019.

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

[5] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, pages 8218–8227, 2021.

[6] Jihwan Bang, Hyunseo Koh, Seulki Park, Hwanjun Song, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on a contaminated data stream with blurry task boundaries. In *CVPR*, pages 9275–9284, 2022.

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

[8] Olaf RP Bininda-Emonds, John L Gittleman, and Andy Purvis. Building large trees by combining phylogenetic information: a complete phylogeny of the extant carnivora (mammalia). *Biological Reviews*, 74(2):143–175, 1999.

[9] William F Brewer and James C Treyens. Role of schemata in memory for places. *Cognitive psychology*, 13(2):207–230, 1981.

[10] Dongliang Chang, Kaiyue Pang, Yixiao Zheng, Zhanyu Ma, Yi-Zhe Song, and Jun Guo. Your" flamingo" is my" bird": Fine-grained, or not. In *CVPR*, pages 11476–11485, 2021.

[11] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, pages 532–547, 2018.

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

[13] Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *ICML*, pages 1952–1961. PMLR, 2020.

[14] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, pages 113–123, 2019.

[15] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[16] Jia Deng, Alexander C Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*, pages 71–84. Springer, 2010.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009.

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

[19] Enrico Fini, Stéphane Lathuiliere, Enver Sangineto, Moin Nabi, and Elisa Ricci. Online continual learning under extreme memory constraints. In *ECCV*, pages 720–735. Springer, 2020.

[20] Douglas H Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172, 1987.

[21] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. *NeurIPS*, 26, 2013.

[22] Ashima Garg, Shaurya Bagga, Yashvardhan Singh, and Saket Anand. Hiermatch: leveraging label hierarchies for improving semi-supervised learning. In *WACV*, pages 1015–1024, 2022.

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

[24] Yiduo Guo, Bing Liu, and Dongyan Zhao. Online continual learning through mutual information maximization. In *ICML*, pages 8109–8126. PMLR, 2022.

[25] 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*, 31, 2018.

[26] Jiangpeng He, Runyu Mao, Zeman Shao, and Fengqing Zhu. Incremental learning in online scenario. In *CVPR*, pages 13926–13935, 2020.

[27] Stephen R Heller and Alan D McNaught. The iupac international chemical identifier (inchi). *Chemistry International*, 31(1):7, 2009.

[28] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177, 2001.

[29] Shyamgopal Karthik, Ameya Prabhu, Puneet K Dokania, and Vineet Gandhi. No cost likelihood manipulation at test time for making better mistakes in deep networks. *arXiv preprint arXiv:2104.00795*, 2021.

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

[31] 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.

[32] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *CVPR*, pages 554–561, 2013.

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

[34] 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*, pages 3925–3934. PMLR, 2019.

[35] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[36] Shaoteng Liu, Jingjing Chen, Liangming Pan, Chong-Wah Ngo, Tat-Seng Chua, and Yu-Gang Jiang. Hyperbolic visual embedding learning for zero-shot recognition. In *CVPR*, pages 9273–9281, 2020.

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

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

[39] Eugene Belilovsky Massimo Caccia Min Lin Laurent Charlin Tinne Tuytelaars Rahaf Aljundi, Lucas Caccia. Online continual learning with maximally interfered retrieval. 32:11849–11860, 2019a.

[40] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017.

[41] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, pages 7263–7271, 2017.

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

[43] David E Rumelhart and Andrew Ortony. The representation of knowledge in memory. In Richard C Anderson, Rand J Spiro, and William E Montague, editors, *Schooling and the Acquisition of Knowledge*, pages 99–135. Erlbaum, Hillsdale, NJ, 1977.

[44] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, pages 4548–4557. PMLR, 2018.

[45] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NeurIPS*, 30, 2017.

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

[47] 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*, pages 8769–8778, 2018.

[48] Yu Wang, Zhou Wang, Qinghua Hu, Yucan Zhou, and Honglei Su. Hierarchical semantic risk minimization for large-scale classification. *IEEE Transactions on Cybernetics*, 52(9):9546–9558, 2021.

[49] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. Hierarchical multi-label classification networks. In *ICML*, pages 5075–5084. PMLR, 2018.

[50] Hui Wu, Michele Merler, Rosario Uceda-Sosa, and John R Smith. Learning to make better mistakes: Semantics-aware visual food recognition. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 172–176, 2016.

[51] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019.

[52] 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*, pages 6023–6032, 2019.

[53] Bin Zhao, Fei Li, and Eric Xing. Large-scale category structure aware image categorization. *NeurIPS*, 24, 2011.