Continual Learning of Object Classification in the Real World

Jie Mei

A dissertation
submitted in partial fulfillment of the
requirements for

Doctor of Philosophy

University of Washington

2024

Reading Committee:
Jenq-Neng Hwang, Chair
Radha Poovendran
Rania Hussein

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Continual Learning of Object Classification in the Real World

Jie Mei

Chair of the Supervisory Committee:
Jenq-Neng Hwang
Electrical and Computer Engineering

Technological advances in deep learning have brought remarkable performance in the object classification task but only when all the training data of classes to be learned are available at the same time. However, real-world data continually evolve through time, resulting in ever-changing learning configurations, e.g., new classes are added continuously. When a deep learning model loses access to previously trained classes data (e.g., due to privacy issues, storage limitations, or data transfer difficulties) and can only be fine-tuned based on new classes data, it could overfit on new classes and catastrophically forget the previously trained classes due to the end-to-end training strategy. Therefore, a class incremental learning (CIL) model should be able to learn more and more new classes over time from a stream of continuously arriving data, i.e., only the training data for a small number of classes have to be present at the beginning of training and new classes can be added progressively.

Traditional object classification models based on deep learning have a fixed number of output classes from the final softmax layer, which requires the total number of training classes at the beginning of training. This is impractical in the real-world CIL scenario because we can not know how many classes are going to be added in the future. Another feature of traditional image recognition models based on deep learning is the flat classification outputs from the standard softmax layer, i.e., all training classes are on the same level and the summation of confidence scores over all classes equals to one. Therefore, traditional image recognition models are not able to perform

the hierarchical classification task. For example, 'Dog' class and 'Golden Retriever' class could be in the dataset at the same time and both confidence scores should be one for a 'Golden Retriever' input image. Thus, a hierarchical classifier should be able to predict all confidence scores at different levels in the hierarchical data structure. One obvious advantage is that if the confidence score of a sample is too low at the fine level but very high at the coarse level, then we can use the coarse-level prediction to be the final prediction. In contrast, flat classifiers have no alternative ways if the confidence score is too low at the final prediction. To this end, we construct a hierarchical dataset and propose a CNN-based hierarchical classification architecture, which enforces the hierarchical data structure and introduces an efficient training and inference strategy. Furthermore, taking advantages from both class incremental learning and hierarchical classifier, we first propose a **H**ierarchical **C**lass **I**ncremental **L**earning model (**HCIL**) [101], for continual learning of object classification in the real world.

Most works in class incremental learning (CIL) assume disjoint sets of classes as tasks. Although a few works deal with overlapped sets of classes, they either assume a balanced data distribution or assume a mild imbalanced distribution. Instead, we further explore one of the understudied real-world CIL settings where (1) different tasks can share some classes but with new data samples, and (2) the training data of each task follows a long-tail distribution. We call this setting CIL-LT. We hypothesize that previously trained classification heads possess prototype knowledge of seen classes and thus could help learn the new model. Therefore, we propose a method, i.e., **E**xpert-and-**S**amples-**A**ware (**ESA**) incremental learning under longtail distribution [98], with the multi-expert idea and a dynamic weighting technique to deal with the exacerbated forgetting introduced by the long-tail distribution. Experiments show that the proposed method effectively improves the accuracy in the CIL-LT setup on MNIST, CIFAR10, and CIFAR100.

From the multimodal perspective, text-prompt-based approaches for continual learning leverage pre-trained text encoders and learnable prompts to encode textual features for sequentially arrived classes over time. A common challenge encountered by existing works is how to learn

fine-grained text prompts, which implicitly carry semantic information of new classes, so that the textual features of newly arrived classes do not overlap with those of trained classes, thereby mitigating the catastrophic forgetting problem. To address this challenge, we propose a novel approach **Pro**totype-guided **T**ext **P**rompt **S**election (**ProTPS**) to intentionally increase the training flexibility thus enforcing learning fine-grained text prompts. Specifically, our ProTPS aggregates the image and text encoders to learn class-specific vision prototypes and text prompts. Vision prototypes guide the selection and learning of text prompts that encode exclusive fine-grained textual features of each class. We evaluate our ProTPS in both class incremental (CI) setting and cross-datasets continual (CDC) learning setting. Since our ProTPS achieves performance close to the upper bounds, we further collect a real-world marine species dataset, named Marine112, to bring new challenges to the community. Marine112 is a fine-grained dataset with long-tail distribution and is naturally suited for the class and domain incremental (CDI) learning setting. The results under three continual learning settings show that our approach performs favorably against the recent state-of-the-art methods.

The expected contributions of this proposal for continual learning of object classification in the real world can be concluded as follows:

- We propose the **HCIL** framework, with the ability to jointly perform the hierarchical classification task and class incremental learning without catastrophically forgetting previously trained classes.

- We propose the **ESA** framework with the multi-expert idea and a dynamic weighting technique, for one of the understudied real-world CIL settings where (1) different tasks can share some classes but with new data samples, and (2) the training data of each task follows a long-tail distribution.

- We propose a multimodal transformer-based framework, **ProTPS**, to intentionally increase the training flexibility thus enforcing learning fine-grained text prompts so that the textual

features of newly arrived classes do not overlap with those of trained classes. ProTPS is shown to be effective under three real-world continual learning settings, i.e., class incremental (CI) setting, cross-datasets continual (CDC) learning setting, and class and domain incremental (CDI) learning setting.

- From the dataset perspective, we collect a new hierarchical dataset for continual learning, three under-studied CIL-LT datasets, and a real-world marine species dataset, named Marine112, to bring new challenges to the continual learning community. Marine112 is a fine-grained dataset with long-tail distribution and is naturally suited for the class and domain incremental (CDI) learning setting.

- The proposed and collected datasets present novel real-world challenges to the continual learning community, and the proposed frameworks represent significant advancements toward achieving continual learning of object classification in real-world scenarios.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

My five-year PhD journey at the University of Washington has been nothing short of spectacular. Enrolling in the Information Processing Lab (IPL) to study computer vision and deep learning under the guidance of Prof. Jenq-Neng Hwang was the realization of a long-held dream. The environment at IPL, enriched by inspiring faculty, brilliant colleagues, and wonderful friends, provided the perfect backdrop for my academic pursuits.

Profound gratitude is due to my advisor, Prof. Jenq-Neng Hwang. His role in my life extends far beyond that of an academic guide. He has been a tremendous source of inspiration and support, significantly enriching my understanding and skills in key areas like traditional 3D geometry, object detection, tracking, and classification which formed the cornerstone of my PhD thesis. Collaborating with him has been a privilege, and observing the growth of our research group has been a source of great pride and joy, as depicted in Figure 1.

I am equally grateful to my PhD supervisory committee—Prof. Radha Proovendran, Prof. Rania Hussein, and Prof. Yen-Chi Chen. Their encouragement, insightful feedback, and stimulating discussions have been invaluable throughout my research journey.

Special acknowledgment goes to the researchers from the Alaska Fisheries Science Center at the National Oceanic and Atmospheric Administration, who funded my research. I am particularly indebted to Craig Rose, Suzanne Romain, Keith Fuller, Cindy Tribuzio, Graeme LeeSon, and Kelsey Magrane for their support and belief in the relevance and potential of my work.

My experience was immeasurably enhanced by the brilliant minds and warm hearts of my IPL colleagues. I extend my heartfelt thanks to all current and former members, including Drs. Haotian Zhang, Yizhou Wang, Jiarui Cai, Tsung-Wei Huang, Renshu Gu, Gaoang Wang, and peers like Zhongyu Jiang, Cheng-Yen Yang, Yin Jin, Aotian Zheng, Jen-Hao Cheng, Yudong Li, Alisha

Figure 1: The amazing IPL's research group. (This photo was taken at Prof. Hwang's house with family and friends on 2019 Thanksgiving Day.)

Peng, Henry Yu, and Hsiang-Wei Huang. Their support, camaraderie, and friendship have made my PhD journey a memorable adventure.

Lastly, the unwavering support of my family has been the bedrock of my experience in Seattle. The consistent encouragement from my mother has been a beacon during challenging times, and her wisdom a guiding light. Special thanks to my spouse, Mengyu, for the shared memories—from exploring new cities to the simple joys of raising our dogs and embarking on several road trips across various states. Your love and companionship have been my anchor. And to our beloved fur babies, Amiens and Juven, your constant presence by my side has brought comfort and joy amidst the rigors of PhD life.

DEDICATION

to my Mom, and Spouse

For always encouraging me to pursue my dreams

and for always be my side

Chapter 1

INTRODUCTION

1.1 Class Incremental Learning of Image Recognition

In a supervised image classification task, deep neural networks have been shown to achieve remarkable performances, but only when all the training data of classes to be learned are available at the same time. However, real-world data continually evolve through time, resulting in ever-changing learning configurations, e.g., new classes are added continuously. A class incremental learning (CIL) model should be able to learn more and more new classes over time from a stream of continuously arriving data. When a deep learning model loses access to previous classes data (e.g., due to privacy issues, storage limitations, or data transfer difficulties) and can only be fine-tuned based on new classes data, it could catastrophically forget the old classes, the so-called catastrophic forgetting problem [18, 39, 41, 55, 55, 69, 101, 114, 163, 166]. The main issue behind this problem is due to overfitting the most recent trained classes.

A growing amount of works have emerged to tackle the catastrophic forgetting problem from the rehearsal perspective. Most continual learning strategies [18, 41, 55, 114, 118] applied on large-scale datasets allow storing a fixed number of data from previously trained classes. More specifically, when training the model on new classes, the model can still have access to the stored raw data [18, 22, 114] or compressed data [49, 61] of previously trained classes, which are referred to as **memory** in the CIL scenario. These methods aim at limiting the changes in the model when new classes are learned by applying loss constraints such as knowledge distiallation [53] on the model's predictions for the memory data. However, neither the memory selection strategy nor the data imbalance between the memory and new classes data gets enough attention in these methods. This inevitably leads to classification heads overfitting and further causes the overfitting of the feature extraction backbone through the back-propagation in the end-to-end training. Our proposed

efficient regularization method overcomes the data imbalance problem, along with a proposed comprehensive memory selection module, to mitigate the classification heads overfitting. Besides, thanks to the easy access to large public datasets such as ImageNet-22k [33], we can freeze a pre-trained backbone to completely avoid the feature extraction backbone overfitting.

From the dynamic architecture perspective, recent CIL works [43, 45, 60, 80, 125, 169] dynamically expand classification heads (and feature extraction backbones) for different training tasks, where training classes available at the same time are defined as one 'task'. Unfortunately at inference time, they require a task identifier, i.e., need to know the task to which the test image belongs, to use parameters of the corresponding architecture branch. This is impractical in the real-world class incremental learning scenario. More recently, DyToX [41], DER [166] and Simple-DER [83] avoid the need for this task identifier. DyTox [41] expands an independent classifier for each new task and chooses the output with maximum confidence score as the prediction during inference, on the other hand, DER [166] and Simple-DER [83] learn a single classifier based on the concatenation of all produced embeddings by different subsets of parameters. Yet, previous methods except DyTox [41] encounter dramatic memory overhead when dealing with a large number of tasks and thus need complex pruning as post-processing. Our proposed method is also a dynamic framework, which does not need the task-identifier nor any complex pruning during the inference time. Unlike the previous methods, our proposed scheme expands an independent binary classification head for **each new class**, which can be easily incorporated with the proposed efficient regularization in the training to mitigate the classification heads overfitting.

1.2 Hierarchical Class Incremental Learning

In the real-world image recognition, hierarchical classifiers are more beneficial than flat classifiers. For example, in electronic monitoring (EM) of longline fishing, the goal is to monitor the fish catching activities on fishing vessels, either for the regulatory compliance or catch counting. Hierarchical classification based on videos allows for inexpensive and efficient fish species identification of catches from longline fishing, where fishes are under severe deformation and self-occlusion during the catching process. More importantly, the flexibility of hierarchical classification mitigates the

laborious efforts of human reviews by providing confidence scores in different hierarchical levels. Some related works either use cascaded models for hierarchical classification or make predictions per image or predict one overlapping hierarchical data structure of the dataset in advance. However, with a known non-overlapping hierarchical data structure provided by fisheries scientists, our method in this proposal enforces the hierarchical data structure and introduces an efficient training and inference strategy for video-based fisheries data.

Automated imagery analysis techniques have drawn increasing attention in fisheries science and industry [23, 24, 46, 58, 59, 146, 158, 159, 186], because they are more scalable and deployable than conventional manual survey and monitoring approaches. Especially in the EM systems, a hierarchical classifier is more meaningful for the fisheries than a flat classifier with the standard softmax output layer. The hierarchical classifier can predict coarse-level groups and fine-level species at the same time. If the system predicts some images with high confidence in one coarse-level group but with low confidence in the corresponding fine-level species, then a hierarchical classifier stops predictions of those images at the correct coarse-level group and allows fisheries personnel to assign corresponding experts to review those images and get the correct fine-level labels.

To address the hierarchical classification needs, in this proposal, we construct a hierarchical dataset for the longline fishing monitoring, where fish are caught are caught on hooks and viewed as they are pulled up from the sea and over the rail of the fishing vessel as shown in Fig.1.1. Also, we propose a video-based hierarchical species classification system. The main contributions for hierarchical classification as summarized as follows:

1) A proposed CNN architecture for hierarchical classification task enforces an effective hierarchical data structure with multi-level classification heads and a shared feature extraction backbone. We also construct a dataset with hierarchical data structure.

2) An efficient training strategy with multi-level loss functions applied on multi-level classification heads' outputs. Plus, two robust video-based hierarchical inference schemes based on average confidence scores and majority vote respectively.

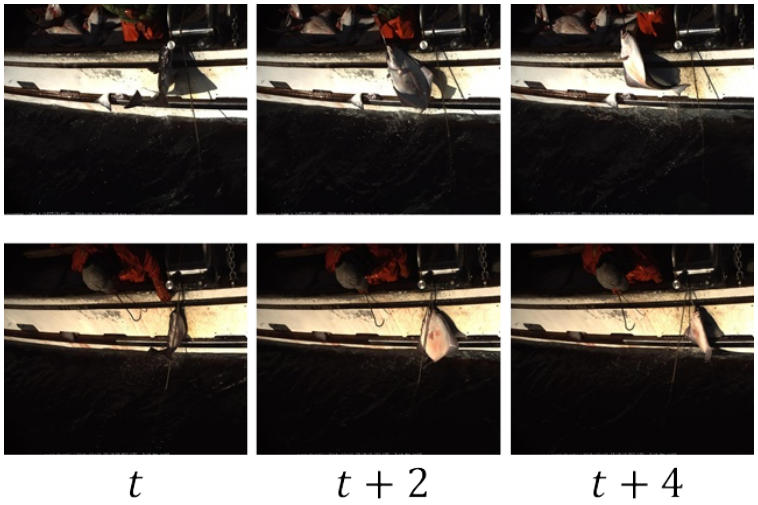


Figure 1.1: Longline Fishing: Each row is a sequence of an individual fish caught on a longline hook, as it is being pulled up from the sea and over the rail of the fishing vessel.

3) The proposed hierarchical classifier allows coarse-level prediction to be the final output if fine-level confidence score is too low and achieves higher accuracy on tail-class species when training data follows a long-tail (imbalanced) distribution than a flat classifier.

4) Taking advantages from both class incremental learning and hierarchical classifier, we further propose a **H**ierarchical **C**lass **I**ncremental **L**earning model (**HCIL**), with the ability to jointly perform hierarchical classification task and class incremental learning without catastrophically forgetting previously trained classes.

1.3 Class Incremental Learning Under Longtail Distribution

In a supervised image classification task, deep neural networks have been shown to achieve remarkable performances. However, real-world data continually evolve through time, resulting in ever-changing learning configurations, *e.g.*, new classes are added incrementally. When a deep learning model loses access to previous classes data (*e.g.*, due to privacy issues, storage limitations, or data transfer difficulties) and can mainly be fine-tuned based on new classes data, it could catas-

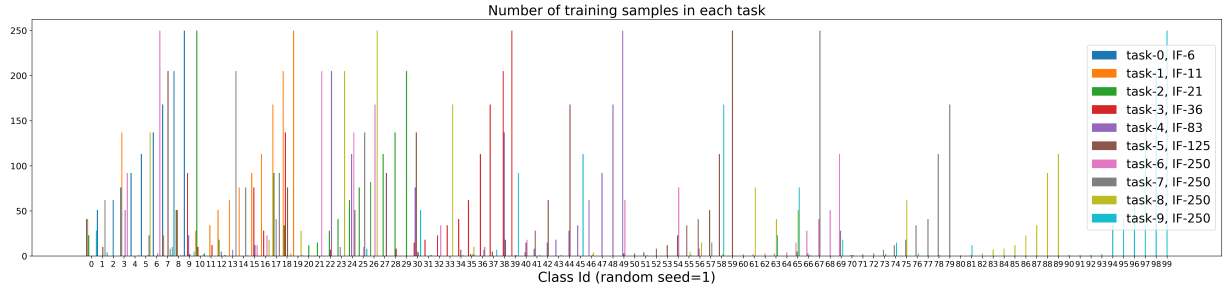


Figure 1.2: **CIFAR100 split under CIL-LT**: Each task includes 10 novel classes and 30% of seen classes that are randomly selected. Within each task, the training data follows a long-tail distribution. The maximum imbalance factor (IF) is 250, which is defined as the number of training samples in the largest class divided by that of the smallest. Thus, we call this split CIFAR100-Overlap30-IF250. Note that: We use 3 different seeds to generate 3 different class orders for evaluation purposes.

trophically forget the old classes, the so-called catastrophic forgetting problem [69, 101, 114, 163]. The main issue behind this problem is overfitting the most recently trained classes. In the real world, this problem can be exacerbated by the long-tail distribution [14, 16] of training data.

Class incremental learning (CIL) has been widely studied under the setting of end-to-end training from scratch, *i.e.*, incrementally learning informative features. However, most works for class incremental learning assume disjoint sets of classes as tasks, *i.e., disjoint-CIL*, where there are no overlapped classes between different training tasks. Thus a new task only includes novel class training data, e.g. [88]. Even though [12] proposes the setting of *blurry-CIL* where different training tasks share some classes, it assumes a balanced distribution of overlapped classes. A few recent works [10, 104] start to consider overlapped classes and imbalanced distribution in CIL, *i.e., general-CIL*, but their maximum imbalance factor is around 20. But in the real world, training data may follow a longtail distribution where the imbalance factor can be larger than 50 [16]. Thus, we propose the setting, *i.e., CIL-LT* as shown in Fig. 1.2, where (1) different tasks can share some classes but with new data samples, (2) the training data of each task follows a long-tail distribution

with the imbalance factor in a wide range.

1.4 Prototype-Guided Text Prompt Selection for Continual Learning

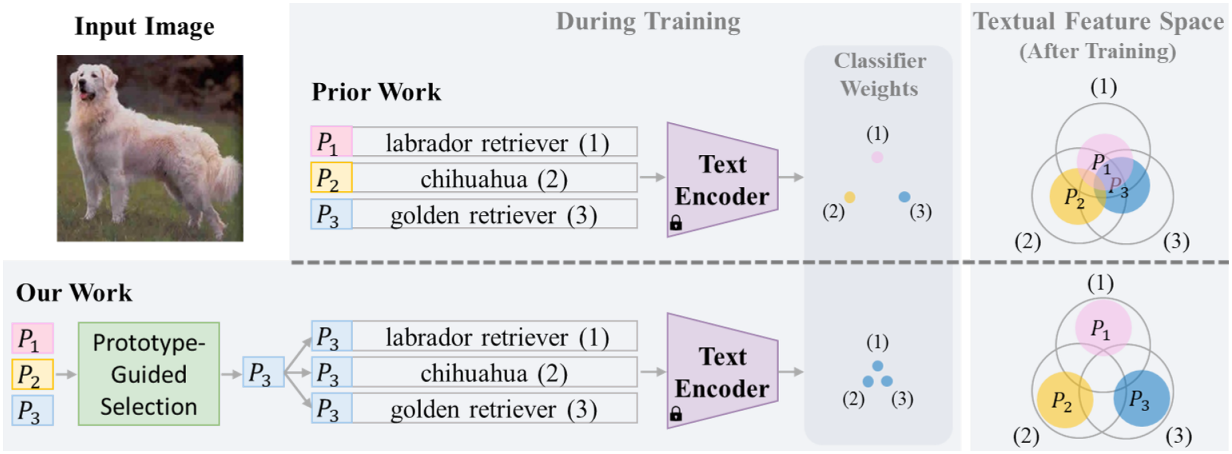


Figure 1.3: An example of our **motivation** for proposing learnable text prompt selection. **Prior work** [180] built upon CLIP [111] proposes learnable class-specific prompts to replace predefined prompt templates. **Our work** introduces a novel text prompt selection approach that enables semantic prompts of each class to interact with other class names. By intentionally increasing the training flexibility, we encourage our text prompts to capture exclusive fine-grained textual features of each class. Note that three big gray circles in the feature space represent all possible textual features for each class.

In supervised image classification, deep neural networks have consistently demonstrated impressive results. Yet, the dynamic nature of real-world data, with its constant evolution and shifting paradigms, such as the incremental addition of new classes, poses significant challenges. When a model's access to historical training data is constrained, due to privacy considerations, data storage and transfer limitations, or energy efficiency concerns, it can only be primarily updated with new data. This scenario lead to a deep learning model's "catastrophic forgetting" of previously learned knowledge, a critical problem documented in seminal works [65, 69, 82, 101, 114, 149, 154, 163] in

the area of continual learning.

To mitigate the catastrophic forgetting problem, recent text-prompt-based methods [133, 149, 151, 180] leverage frozen Contrastive Language-Image Pre-training (CLIP) [111] to achieve outstanding performance. In these methods, text prompts are concatenated with class names as input to the frozen pre-trained CLIP [111] text encoder to get textual features of each class, which then serve as weights of the linear classifier as proposed in the original CLIP [111]. These methods use learnable prompts or predefined prompt templates to encode textual features for sequentially arrived classes over time. However, **a common challenge** encountered by existing works is how to learn text prompts encoded with fine-grained semantic features of different classes so that the textual features of newly arrived classes do not overlap with those of trained classes.

To address this challenge, we propose a novel approach Prototype-guided Text Prompt Selection (ProTPS). Our **motivation** is to intentionally increase the training flexibility thus enforcing the text prompts to encode fine-grained textual features of each class. For example in Fig. 1.3, given an input image of the golden retriever in training, the text prompt belonging to "golden retriever", i.e., P_3, can only interact with "golden retriever" in the prior work, while our ProTPS selects P_3 to concatenate with all class names, increasing the training flexibility. By doing so, our prompts "P_3-labrador retriever" and "P_3-golden retriever" are closer in the feature space at the beginning of training, compared with "P_1-labrador retriever" and "P_3-golden retriever" in the prior work. Then using textual features of our prompts as the linear classifier weights results in a bigger training loss. By minimizing the loss, P_3 is encouraged to capture the exclusive textual features of "golden retriever", thus our text prompts have to learn more fine-grained features of each class. After training, our class-specific text prompts can be more distinct in the feature space as illustrated in Fig. 1.3.

Why use vision prototypes as guidance? Different from existing text-prompts-based works which have a single classifier derived from the text encoder as proposed in the CLIP [111], our ProTPS has an additional linear classifier whose weights are initialized with visual features, i.e., prototypes, of each class, computed from the pre-trained image encoder. At the beginning of training, this linear classifier can provide a reasonable initial prediction. Thus our ProTPS pairs the vision prototype of each class with a text prompt to facilitate learning fine-grained semantic

features. Our **motivation** for using prototypes as guidance for text prompt selection stems from the hypothesis that vision prototypes and text prompts can be complementary features, i.e., vision prototypes can be good at globally distinct categories, such as cows and birds; on the other hand, text prompts can be trained to capture exclusive semantic features corresponding to regional image details in fine-grained categories, such as different dog breeds.

The experimental configurations for existing continual learning methods are idealistic, typically partitioning a single dataset into multiple non-overlapped and data-balanced tasks, i.e., class incremental (CI) learning, or combining two datasets into a long-sequence domain-shift task, i.e., cross-datasets continual (CDC) learning. To handle a more realistic scenario, we collect a real-world challenging dataset, named Marine112, covering 112 fine-grained classes of marine animal species across 6 years. We believe Marine112 is of great importance to the continual learning community, for the following reasons: (1) it features overlapped fine-grained classes among different tasks with varied imaging qualities, i.e., domain shift; (2) it reflects the longtail [91] (imbalanced) distribution observed in nature; and (3) it is naturally suited for the class and domain incremental (CDI) learning. Our contributions are summarized as follows:

- We propose a novel approach ProTPS to continually learn **fine-grained** text prompts so that the textual features of newly arrived classes do not overlap with those of trained classes, thereby mitigating "catastrophic forgetting".

- Our novel ProTPS learns global image-level vision prototypes to guide the selection and learning of text prompts that attend to **region-level** image details. We also propose a novel contrastive loss to align and enhance text prompts and vision prototypes, bringing further boost to the performance.

- In class incremental learning setup, ProTPS attains +1.9% and +10.6% gains on CIFAR100 and ImageNet100 respectively over the current state-of-the-art methods. In cross-datasets continual learning setup, ProTPS secures +2.1% and +7.3% on "ImageNet2CIFAR" and "ImageNet+CIFAR" respectively.

- We collect a real-world **fine-grained** dataset, Marine112, with long-tail distribution and naturally suited for class and domain incremental (CDI) learning. It brings new challenges to the continual learning community.

Chapter 2

RELATED WORK

2.1 Class Incremental Learning

2.1.1 Rehearsal-based Approaches

Class incremental learning models tackle the catastrophic forgetting of previously trained classes [135]. Rehearsal learning has been used in most incremental learning strategies [18, 41, 55, 114, 118] applied on large-scale datasets, i.e., a fixed amount of the training data of previously trained classes is kept as a memory. The memory can be raw images (pixels) [18, 22, 114], compressed images [49, 61] or trimmed data [38] to reduce memory overhead. Another branch of rehearsal-based methods is generative reply [67, 78, 126], which only stores a model to generate new samples of previously trained classes. Most rehearsal approaches use memory data to constrain the changes in the model when new classes are being learned. These constraints can be applied on the model weights [1, 20, 69, 171], feature maps [36, 39, 55], predicted probabilities [18, 19, 82, 114], or on the gradients [21, 93].

2.1.2 Incremental Dynamic Networks

In contrast, dynamic frameworks best handle a growing-class learning configuration [80, 178] by dynamically creating an independent binary classification head for each new class, while other dynamic works [43, 45, 60, 122, 157] in CIL create sub-branches, where each is specialized in one specific task that contains more than one class. Unfortunately, these previous approaches often require the task identifier during inference time to use the corresponding branch. Recently, DER [166] avoids the need for the task identifier by adding a new feature extraction backbone per task. During inference, all backbones' embeddings are concatenated and fed into a unified

classification head. To limit the growing number of parameters in the whole model, it uses the 'Hard Attention to the Task' (HAT) [80] procedure to prune each model. But the pruning is very sensitive to hyperparameters and DER also requires the total number of training classes in advance, which is impractical in the real-world CIL scenario.

DyTox [41] adopts a task-dynamic strategy and keeps a fixed-size memory similar to rehearsal-based methods but without complex pruning. It uses a shared backbone for all tasks and only dynamically expands an independent classification head for each task.

2.2 Long-tail Recognition

Common methods for long-tail recognition are the readjustment of the data distribution during training, *e.g.,* under-sampling of the majority classes, over-sampling of the minority classes, and re-weighting of the loss function based on the frequency or importance of the samples [17, 31, 62, 84, 179]. Another trend to address the long-tailed issue is the use of multi-expert or multi-branch networks, which also show strong performance gains by treating the relatively balanced sub-groups separately. BBN [179] proposes a shared backbone by two branches with normal and reversed sampling, respectively. LFME [165], RIDE [150], and ACE [16] are multi-expert architectures that learn diverse classifiers in parallel, combined with knowledge distillation and distribution-aware expert selection. Their major difference lies in the data split for different experts and optimization strategies.

Similarly, our ESA framework also adopts the multi-expert idea and re-weighting the loss function to the universal-CIL scenario. However, it is non-trivial and challenging to decide on an expert expansion rule to make sure an expert can maintain good ability at certain sub-groups since majority classes may become minority classes in the incoming tasks under the continual learning scenario. Therefore, we propose one expert for one task in the universal-CIL scenario to make each expert relatively good at novel class recognition in its task. To maintain each expert's ability in the continual learning scenario, we present an expert-aware (EA) loss and a samples-aware (SA) loss with a dynamic weighting technique.

2.3 Significance of Pre-trained Backbones

The full ImageNet-22k dataset [33] consists of 14,197,122 images, divided into 21,841 classes. ImageNet-1k, which contains 1,000 mutually exclusive classes, is created by selecting a subset of 1.2M images from ImageNet-22k. Tal et al., 2021 [116] show that ImageNet-22k dataset is underestimated of its significance compared to standard ImageNet-1k pre-training. Their results also show that different models, including small mobile-oriented models, significantly benefit from ImageNet-22k pretraining on numerous other datasets and downstream tasks. Li et al., 2021 [79] takes advantages of a vision backbone (Swin-Transformer [89]) pre-trained on ImageNet-22k and a language backbone (BERT [35]) pre-trained on BooksCorpus [185] and English Wikipedia for the zero-shot object detection task. Given the easy access of large public datasets, more and more recent works [35, 37, 63, 79, 89, 90, 116, 138] start to take advantage of pre-trained backbones, which can then be fine-tuned on various downstream tasks. Besides, the most recent work [140] even shows that a frozen classifier-initialized backbone used to perform object detection task can achieve better performance for many many different detection models.

We can treat the class incremental learning as a downstream task as well. However, the nature of CIL is different from the traditional supervised tasks where all the training data are available at the same time. As a result, in the CIL scenario, fine-tuning a pre-trained model in an end-to-end training scheme on the imbalanced data of new classes and memory inevitably incurs the classification head overfitting and backbone overfitting, resulting in the so-called catastrophic forgetting problem.

2.4 Hierarchical Classification

2.4.1 Traditional Flat Classifiers

We use 'flat classifiers' to represent all deep learning classification systems with softmax as the final layer to normalize the outputs of all classes, without introducing any hierarchical level of prediction. AlexNet [73] is the first CNN-based winner in 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC), which introduces the 1000-way softmax layer for classifying

the 1000 classes of objects. The subsequent ILSVRC winners, VGGNet [128], GoogLeNet [130], and ResNet [51] continue to use softmax as the final layer to achieve good performance. Until now, flat classifiers with softmax operations as the final layer are the dominant design structure for classification tasks.

2.4.2 Hierarchical Classifiers

A hierarchical classifier means the system can output all confidence scores at different levels in the hierarchical data structure. One obvious advantage is that if the confidence score of a sample is too low at the fine level but very high at coarse level, then we can use the coarse-level prediction to be the final prediction. In contrast, flat classifiers have no alternative ways if the confidence score is too low at the final prediction.

Hand-crafted features are used in [57] for hierarchical fish species classification. Hierarchical medical image classification [71] and text classification [70] use cascaded flat classifiers to be their hierarchical classifiers, which use only one flat classifier for each level's prediction. They stack CNN-based models with flat classifiers without considering any hierarchical architecture design and increased computational complexity. HDCNN [167] introduces confidence-score multiplication operations to enforce hierarchical data structure but the model uses the same feature maps for both coarse level and fine level, resulting in learning an overlapping hierarchy of training data. B-CNN [184] uses different feature maps for different levels' predictions without enforcing any hierarchical data structure in the architecture. Deep RTC [161] adopts hierarchical classification to deal with long-tailed recognition, resulting in improved accuracy of tail classes. It adopts a simple confidence-score thresholding method, which is also adopted in our approach, to decide to output fine-level prediction or coarse-level prediction. But Deep RTC predicts an overapping hierarchical data structure in the first place, which is different with our situation. In contrast to all previous works, our proposed CNN based hierarchical classifier uses different level features for different level prediction and share the same backbone for all levels instead of cascading multiple flat classifier. Furthermore, we propose a a Hierarchical Class Incremental Learning model (**HCIL**), with the ability to jointly perform hierarchical classification task and class incremental learning.

2.5 Prompt-based Methods For Continual Learning

2.5.1 Classic Methods.

Rehearsal-based continual learning [18, 41, 55, 114, 118] is highly effective by keeping a fixed amount of previously trained data as a memory. Yet, with potential privacy issues [127], they also tend to underperform as the size or quality of the rehearsal memory decreases. Dynamic architectures [43, 45, 60, 80, 125, 162, 169] expand classification heads and/or backbones for new training tasks. Unfortunately, at inference time, some dynamic architectures need complex pruning [45] as post-processing or require an impractical task identifier [43, 60, 125].

2.5.2 Text-prompt-based Methods.

Continual-CLIP [133] directly applies the pre-trained frozen CLIP [111] to the continual learning scenario by concatenating class names with a single text prompt template, , "a bad photo of a { }", to derive the classifier. Thus, without new parameters, its performance is capped by the pre-trained CLIP [111]. To replace predefined templates, CoOP [180], built upon CLIP [111], proposes learnable class-specific prompts but its prompts cannot interact among different classes. AttriCLIP [149] boosts the performance with an attribute word bank which is shared for all classes, and is continually trained on newly arrived classes. Inevitably its word bank may overfit attributes of newly arrived classes. In contrast, our novel ProTPS continually encourages text prompts to capture class-specific **fine-grained** features so that the textual features of newly arrived classes do not overlap with those of trained classes.

2.5.3 Visual-prompt-based Methods.

Recent visual prompt tuning works [81, 129, 131, 147, 153, 154], built upon a pre-trained image encoder, employ a dynamic query mechanism based on key-value pairs to learn instance-specific prompts tailored to the context. The L2P framework [154] pioneers the amalgamation of visual prompts with continual learning by advocating for a universal prompt repository to facilitate task

adaptation. DualPrompt [153] further innovates by affixing task-invariant and task-specific visual prompts to the pre-trained image encoder. CODA-P [129] introduces decomposed and expanded visual prompts that can be optimized in an end-to-end fashion. HiDe-Prompt [147] explicitly optimizes three hierarchical components for visual prompt tuning. However, unlike text-prompt-based methods, these methods only benefit from pre-trained image encoders.

2.5.4 Prototype Learning.

Prototype learning aims to learn prototype vectors to represent each class. Classic continual learning methods like iCaRL [114], LUCIR [55], PODnet [39] and DER [166] rely on the nearest-prototype classifier [103, 155]. Recent methods [137, 181, 183] use prototypes to generate synthetic samples of trained classes or use synthetic samples to calculate prototypes [144]. In contrast, we hypothesize that prototypes derived from the image encoder are mainly global image-level representations. Thus, we propose to jointly learn fine-grained text prompts capturing exclusive semantic features of each class.

2.5.5 Contrastive Learning.

Recently, the contrastive learning field has shown notable progress within self-supervised representation learning [52, 54, 106, 136, 164]. The shared concept across these studies involves attracting an anchor to a single "positive" sample within the feature space while concurrently repelling the anchor from numerous "negative" samples. SupCon [68] extends the self-supervised batch contrastive approach to the supervised setting, allowing the model to leverage label information to construct many positives per anchor in addition to many negatives. CLIP [111] applies supervised contrastive learning between images and text pairs to align image and text to a common feature space. These methods are designed to enhance the feature extraction ability of encoders. In contrast, we propose a novel *prompt-prototype contrastive loss* to align and enhance text prompts and vision prototypes for continual learning scenarios.

Chapter 3

HIERARCHICAL CLASS INCREMENTAL LEARNING FOR LONGLINE FISHING VISUAL MONITORING

In the real-world image recognition, hierarchical classifiers are more beneficial than flat classifiers. For example, in electronic monitoring (EM) of longline fishing, the goal is to monitor the fish catching activities on fishing vessels, either for the regulatory compliance or catch counting. Hierarchical classification based on videos allows for inexpensive and efficient fish species identification of catches from longline fishing, where fishes are under severe deformation and self-occlusion during the catching process. More importantly, the flexibility of hierarchical classification mitigates the laborious efforts of human reviews by providing confidence scores in different hierarchical levels. Some related works either use cascaded models for hierarchical classification or make predictions per image or predict one overlapping hierarchical data structure of the dataset in advance. However, with a known non-overlapping hierarchical data structure provided by fisheries scientists, our method enforces the hierarchical data structure and introduces an efficient training and inference strategy for video-based fisheries data. Our experiments show that the proposed method outperforms the classic flat classification system significantly and our ablation study justifies our contributions in CNN model design, training strategy, and the video-based inference schemes for the hierarchical fish species classification task. Furthermore, we propose **H**ierarchical **C**lass **I**ncremental **L**earning model (**HCIL**), with the ability to jointly perform hierarchical classification task and class incremental learning without catastrophically forgetting previously trained classes. Our experiments show that the proposed **HCIL** achieves better performance under class incremental learning scenario than the above proposed hierarchical classifier.

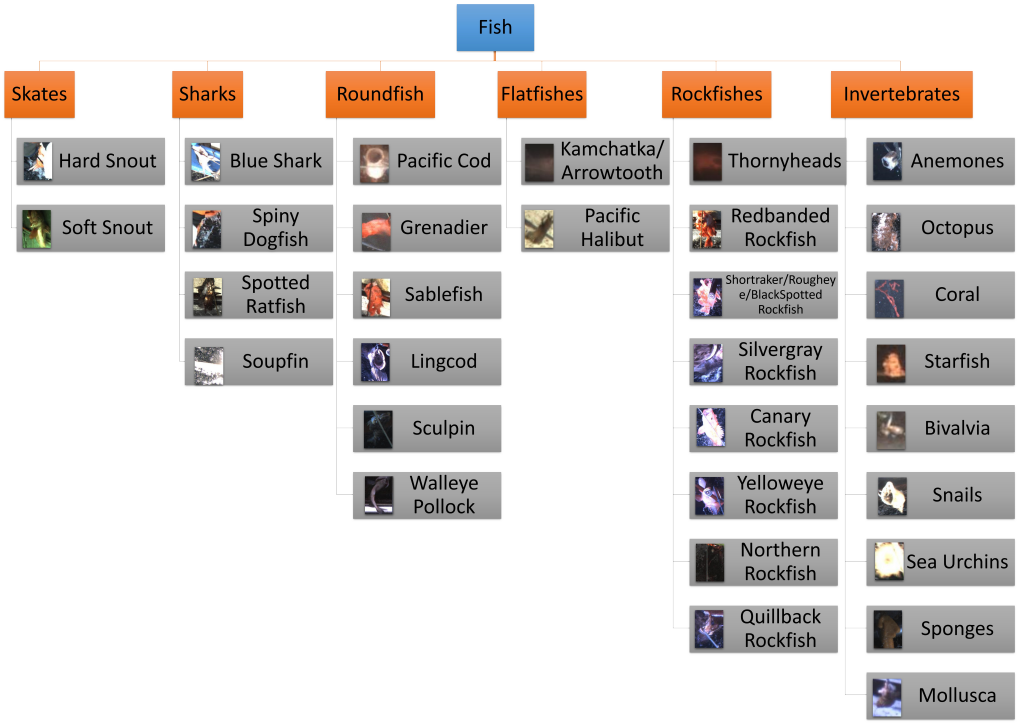


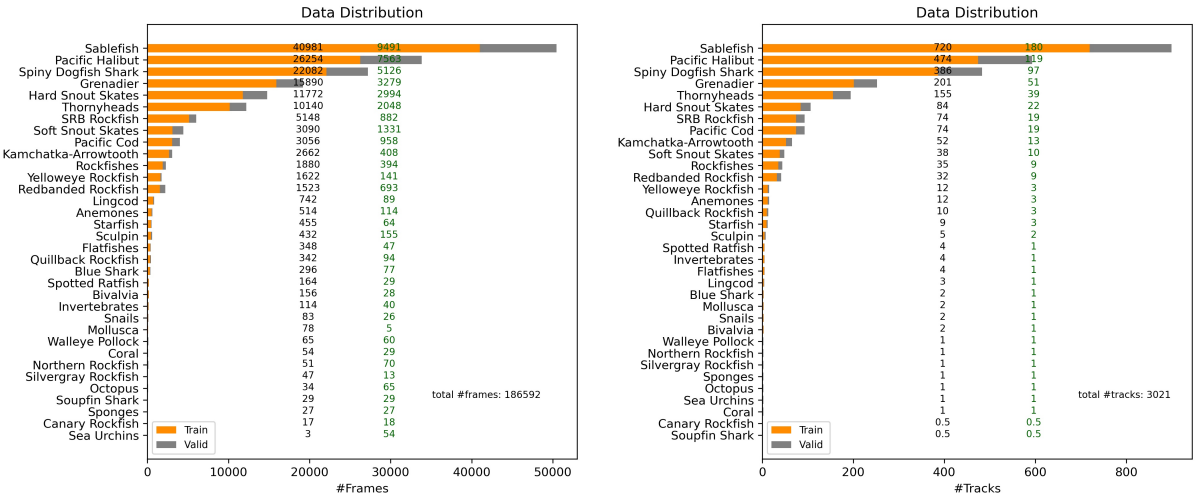
Figure 3.1: Hierarchical Data Structure: The dataset, labeled and provided by NOAA fisheries scientists, includes frames and corresponding labels which are bounding box location, start and end frames' IDs of each individual fish, coarse-level group ground truth, and fine-level species ground truth. The sample images shown here are randomly chosen from the dataset.

3.1 Hierarchical Classification Framework

3.1.1 Hierarchical Dataset

The hierarchical dataset utilized for training our system is professionally labeled and provided by the Fisheries Monitoring and Analysis (FMA) Division, Alaska Fisheries Science Center (AFSC) of NOAA, researchers can contact AFSC directly about getting permission to access this dataset and the corresponding hierarchical data structure.

To construct the dataset used for our system, we use labels of bounding box location to crop objects from raw videos and use labeled start and end frames' IDs for each individual fish to divide



(a) Image-Species Distribution

(b) Track-Species Distribution

Figure 3.2: Dataset Distribution: The left column black numbers in both figures are the number of images or tracks for training while the green numbers are for evaluation. There is a 0.5 in (b) because of only one track in the whole dataset, therefore we have to split it into 2 tracks and denote each as 0.5 track. 'SRB Rockfish' represents Shortraker/Rougheye/BlackSpotted Rockfish. Our dataset follows a long-tail (imbalanced) distribution.

raw videos into individual tracks (video clips). There are 6 coarse-level groups and 31 fine-level species in this hierarchical dataset (see Fig. 3.1). Our dataset is challenging because some fine-level species are very similar to one another. The total number of frames is 186,592 (see Fig. 3.2(a)). The total number of video clips/tracks is 3,021 (see Fig. 3.2(b)). Each video clip contains one individual fish pulled up from sea surface to the fishing vessel during the longline fishing activities.

3.1.2 Hierarchical Architecture

Instead of using cascaded flat classifiers in our species identification in the longline fishing, as inspired by the success of Mask-RCNN [50], which feeds shared feature maps extracted from the backbone to different heads for object classification and instance segmentation at the same time, our proposed architecture is also an end-to-end training network including two parts: a backbone and several hierarchical classification heads (see Fig. 3.3). Inspired by B-CNN [184], we use ResNet101 as our backbone to extract shallow feature maps from images for 'Head-1' and shared deeper feature maps for the other 6 classification heads. We use Head-1 for coarse-level (6 groups) classification and Head-2 to Head-7 for fine-level (31 species in total) predictions.

3.1.3 Enforcing Hierarchical Data Structure

We use confidence-score multiplication operations to enforce the hierarchical data structure in our system. The final confidence score of one specific species is the product of the confidence score in the corresponding coarse group and the confidence score in that specific (fine) species as shown in the following equation, i.e.,

$$P'_{j,i} = P_{1,j-1} \cdot P_{j,i}, \quad j \in [2, 7], \tag{3.1}$$

where $P_{1,j-1}$ represents the confidence score in the $(j-1)^{th}$ group in coarse level, $P_{j,i}$ represents the confidence score in the i^{th} species of the $(j-1)^{th}$ group, $P'_{j,i}$ represents the final confidence score in the i^{th} species of the $(j-1)^{th}$ group. As a result, the final confidence score, $P'_{j,i}$, includes both scores from coarse level and fine level so that this CNN architecture enforces a hierarchical

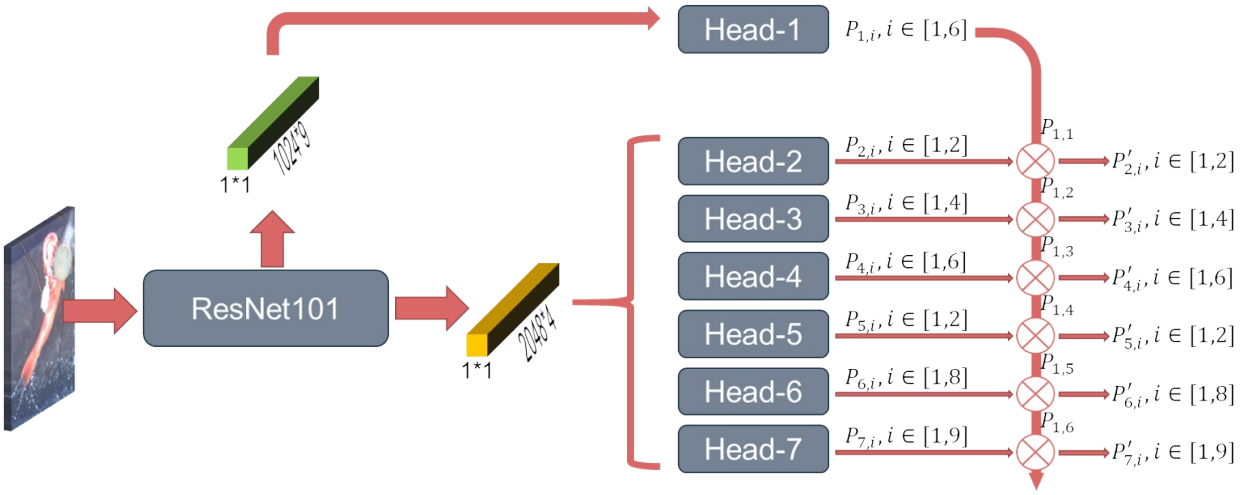


Figure 3.3: Hierarchical Architecture [100]: We call our 7 classification heads 'Hierarchical Heads'. Head-1 is for 6 groups in coarse-level and uses shallower feature maps extracted from the backbone for predictions, while the rest of 6 heads are for fine levels, i.e., Head-2 for 'Skates' group, Head-3 for 'Sharks' group, Head-4 for 'Roundfish' group, Head-5 for 'Flatfishes' group, Head-6 for 'Rockfishes' group, and Head-7 for 'Invertebrates' group, respectively. All fine-level heads use the shared deeper feature maps from the same backbone for predictions. Head-1 has two fully connected layers followed by a sofmax layer. The rest 6 fine-level heads have one fully connected layer followed by a sofmax layer.

data structure when using the final confidence score product to calculate the training loss. Training loss is meaningful when using $P'_{j,i}$ because the final layers in all heads are softmax outputs, which satisfy the following equations:

$$\begin{cases} \sum_i P_{j,i} = 1, \\ \sum_{j=2}^{7} \sum_i P'_{j,i} = 1. \end{cases} \tag{3.2}$$

3.1.4 Efficient Training Strategy

During the image-based training, one input image has both labeled coarse-level ground truth as well as fine-level ground truth. For our architecture, there are two options experimented about how to use these two ground truths.

The first option is training the 'Head-1' and the fine-level head corresponding to the ground truth coarse-level group. Since the corresponding fine-level head is picked by coarse-level group ground truth, therefore the losses are only calculated based on these two heads.

$$Loss_1 = -\sum_i y_{1,i} \cdot log(P_{1,i}) - \sum_i y_{2,i} \cdot log(P'_{j,i}), \tag{3.3}$$

where the first summation is the cross entropy loss in the 'Head-1' and $y_{1,i}$ is coarse-level ground truth. The second summation is the cross entropy loss in the corresponding fine-level head using final predictions, $P'_{j,i}$, after confidence-score multiplication operations. Note that $y_{2,i}$ is the ground-truth among species within this fine-level head. This regular loss does not involve P'_{ji} from other heads, therefore, it does not fully enforce hierarchical data structure during training and only trains two heads each time.

The second option is training the 'Head-1' and all the other fine-level heads using final predictions P'_{ji} after confidence-score multiplication operations, resulting in a more efficient training strategy because it enables confidence-score multiplication operations to fully enforce hierarchical data structure during training and all heads can be trained simultaneously.

$$Loss_2 = -\sum_i y_{1,i} \cdot log(P_{1,i}) - \sum_{j=2}^{7} \sum_i y'_{2,i} \cdot log(P'_{j,i}), \tag{3.4}$$

where $y'_{2,i}$ denotes the ground-truth among 31 species. The reason why given one input image we can calculate cross entropy on all final predictions, $P'_{j,i}$, is that after the confidence-score multiplication operations, summation of these products is still 1.

3.1.5 Video-based Inference Schemes

Although we use image-based training, where training loss is calculated on each individual input image, two video-based (track-based) inference methods are implemented and compared. Since for each input image frame, our system outputs confidence scores of 31 species, $P'_{j,i}$, therefore the first way is to pick the species with the maximum average confidence score of all frames in each track to be the prediction for each track.

The second way is to pick the species with maximum confidence score for every frame in each track, then uses majority vote to select one species as the prediction for that track. Finally, we calculate the average confidence scores among frames corresponding to the selected species. We report performance under these two video-based inference schemes and calculate their average confidence scores with image-based confidence scores in the following section. These two inference schemes can be summarized into the following equations:

$$\begin{cases} p_{1,i} = 1/T \cdot \sum_t P'_{1,i,t}, \\ p_{2,i} = 1/T \cdot \sum_t P'_{j,i,t}, \quad j \in [2,7], \end{cases} \tag{3.5}$$

where t is frame index. $P'_{j,i,t}$ is $P'_{j,i}$ at the t^{th} frame. In the first way, T is the total number of frames in one video clip (a track from the start-frame to the end-frame of one catching), while in the second way, it is the total number of frames corresponding to the selected species in one video clip. As a result, $p_{1,i}$ is video-based average confidence scores in 6 groups and $p_{2,i}$ is video-based average confidence scores in 31 species.

3.2 Experiments on Hierarchical Classifier

3.2.1 Data Split

We use the video-based data split, i.e., each short video clip (a track) is associated with one individual fish and all frames from 80% of all tracks are used as training data for image-based training. All frames from the rest 20% tracks are the evaluation data (see Fig. 3.2). As a result, images for training and evaluation are totally from tracks of different individual fishes. All hyper-parameters like training epochs, learning rate, data augmentation, and so on are kept the same in the following different competing approaches.

3.2.2 Baseline

The dominant species classification architecture is extracting deep features using CNN followed by a flat classifier. As a result, for the baseline, we use ResNet101 as the backbone and two fully connected layers followed by a 31-way softmax layer as the flat classifier head, which is a classic deep learning classification architecture. During training, we only use fine-level ground truth to calculate the cross-entropy loss based on the flat classifier output confidence scores in 31 species without any coarse-level predictions. *From Table 3.1, we can see the accuracy of the baseline is far below our hierarchical method.*

3.2.3 Evaluation Methods and Quantitative Results

Using all frames from the rest 20% tracks for evaluation, we try the following evaluation methods, where we calculate both image-based accuracy as well as video-based (track-based) accuracy, denoted in the 'Unit' column in Table 3.1. We also calculate classification accuracy on the coarse level based on coarse-level ground truth, denoted as 'Level-1' in Table 3.1.

Moreover, with confidence scores in the coarse level as well as the fine level, we can pick the species with maximum fine-level confidence score under the group with maximum coarse-level confidence score as the final prediction, which is denoted as 'Level-2 A' in Table 3.1. While,

with final confidence scores in 31 species, we can directly pick the species with the maximum confidence score product of coarse and fine levels as the final prediction, which is denoted as 'Level-2 B' in Table 3.1. For these two metrics ('Level-2 A' and 'Level-2 B') in video-based schemes, we further use either maximum average confidence score (denoted as '$video$') or majority vote (denoted as '$video^*$') to report the performance, as discussed in the 'Video-based Inference Schemes' in Section 3.1.5.

Finally, with these final confidence scores, $P'_{j,i}, \quad j \in [2, 7]$, in 31 species, for image-based inference, we can also decide to go back to the coarse level if the maximum confidence score product is below a threshold and calculate the accuracy on the coarse level for that input, otherwise stay at the fine level and calculate the accuracy for that input. For video-based inference methods, we compare the average confidence score mentioned in 'Video-based Inference Scheme' section with the threshold. This metric, being able to stay at a coarse level, is denoted as 'Level-2 C' in Table 3.1. Theoretically, the ceiling limit of 'Level-2 C' is 'Level-1' if all samples stop at the coarse level. Therefore we use the greedy search to find a threshold for each scheme in Table 3.1 to make sure that after stopping at the coarse level, the overall video-based inference accuracy will not degrade. We fix these thresholds in image-based inference for every competing scheme.

From Table 3.1, we can see video-based inference is always better than image-based inference in all competing schemes. And these two video-based inference methods, average confidence and majority vote, are comparable with each other.

Scheme-3 is our full system demonstrated in Fig. 3.3, which includes confidence-score multiplication operations to enforce hierarchical data structure and uses the efficient training strategy ($Loss_2$). Scheme-2 only removes the efficient training strategy and uses $Loss_1$ instead. Scheme-1 removes confidence-score multiplication operations in the architecture but keeps 7 heads. It also removes the efficient training strategy and instead uses standard cross-entropy losses on 'Head-1' and the fine-level head corresponding to the ground truth coarse-level group. Scheme-1 shares the same architecture as B-CNN [184]. When evaluating under 'Level-2 B' and 'Level-2 C' for Scheme-1, we have to multiply the coarse-level confidence scores with the fine-level confidence score in advance to get the final confidence scores.

Model	Unit	Level-1	Level-2 A	Level-2 B	Level-2 C
Baseline	img	-	-	78.3	-
Scheme-1	img	86.3	77.4	77.4	82.0 (8567, 27393)
	$video^*$	93.2	86.5	86.6	93.2 (298, 319)
	$video$	93.4	86.5	86.8	93.4 (293, 324)
Scheme-2	img	88.4	79.9	80.0	84.6 (8660, 27300)
	$video^*$	94.3	88.6	88.9	94.3 (329, 288)
	$video$	94.9	88.9	88.8	94.9 (328, 289)
Scheme-3	img	91.0	82.3	82.3	86.3 (5830, 30130)
	$video^*$	96.3	90.6	90.3	96.3 (286, 331)
	$video$	**96.4**	**90.9**	**90.9**	**96.4** (293, 324)

Table 3.1: Comparison with Flat Classifier and Ablation Study: '$video$' denotes video-based inference by using average confidence score among 31 species to pick one predicted species for each track. '$video^*$' denotes video-based inference through majority vote to pick one species for each track. Two numbers under 'Level-2 C' column following the accuracy value are total number of stopping at coarse-level and total number of proceeding to fine-level respectively.

Detailed accuracy on the coarse level and fine level of Scheme-3 (our proposed complete system), based on the maximum average confidence score (denoted as '$video$' in Table 3.1) is in Fig. 3.4.

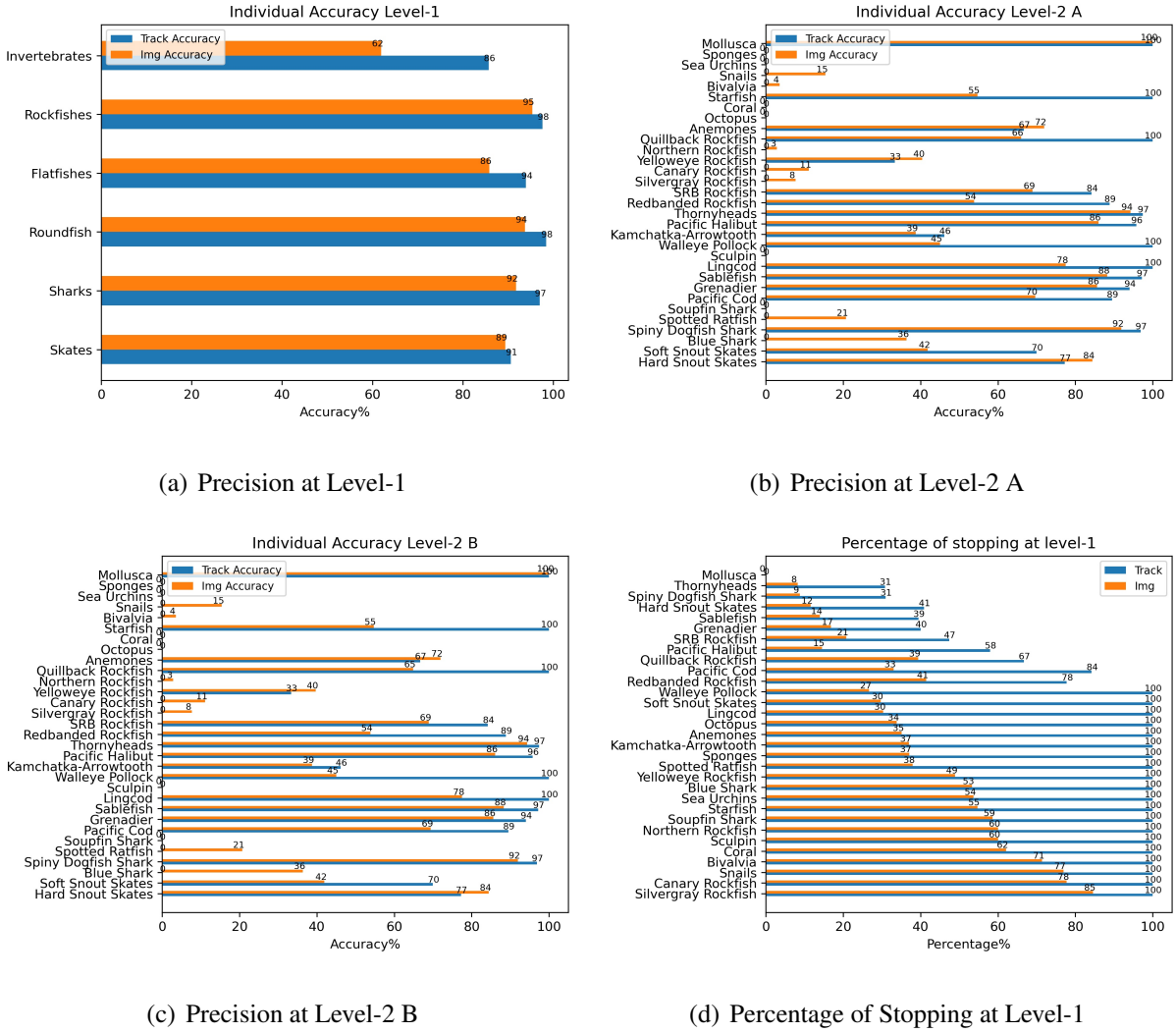
3.2.4 Ablation Study

Scheme-1 and Scheme-2 are experimented mainly for ablation study purposes. *Comparing Scheme-1 with Scheme-2 from Table 3.1, we can see confidence-score multiplication operations can effectively enforce hierarchical data structure and improve the performance even when Scheme-2 only trains two heads each time. Comparing Scheme-2 with Scheme-3, we can see our efficient training strategy ($Loss_2$) improves the performance by fully enforcing hierarchical data structure during training.*

Under 'Level-2 C', the completing systems' final predictions can stop at a coarse level if the final confidence score is lower than the greedy-searched threshold mentioned in the previous section. We call 'Level-2 C' as hierarchical prediction, which is one big advantage of the hierarchical classifier over flat classifiers, which allows fisheries managers to assign corresponding experts to review those images in a certain group and get the correct fine-level labels. *Besides, from Fig. 3.4(d), we can see most tail-class species identification stop at a coarse level, resulting in a significantly higher overall accuracy in 'Level-2 C' over that of 'Level-2 B' in Table 3.1. Also, our full system, Scheme-3, has the greatest number of images or tracks proceeding to fine level and at the same time achieves the best performance.*

3.3 Hierarchical Class Incremental Learning Framework

Hierarchical predictions allow images that cannot be confidently classified at the fine level to be confidently classified at a coarse level for experts future examination, which especially improve overall accuracy on tail-class species identification significantly by stopping at coarse-level predictions. Although the hierarchical classification mitigates the laborious efforts of human reviews by providing confidence scores in different hierarchical levels, its performance drops dramatically



(a) Precision at Level-1

(b) Precision at Level-2 A

(c) Precision at Level-2 B

(d) Percentage of Stopping at Level-1

Figure 3.4: Detailed Accuracy on Coarse Level and Fine Level of the complete proposed system: The orange bar is image-based inference and the blue bar is video-based inference. (d) is under 'Level-2 C' evaluation method. We can see most tail species stop at coarse-level prediction, which makes 5.5% improvement in overall video-based accuracy showed in Table 3.1.

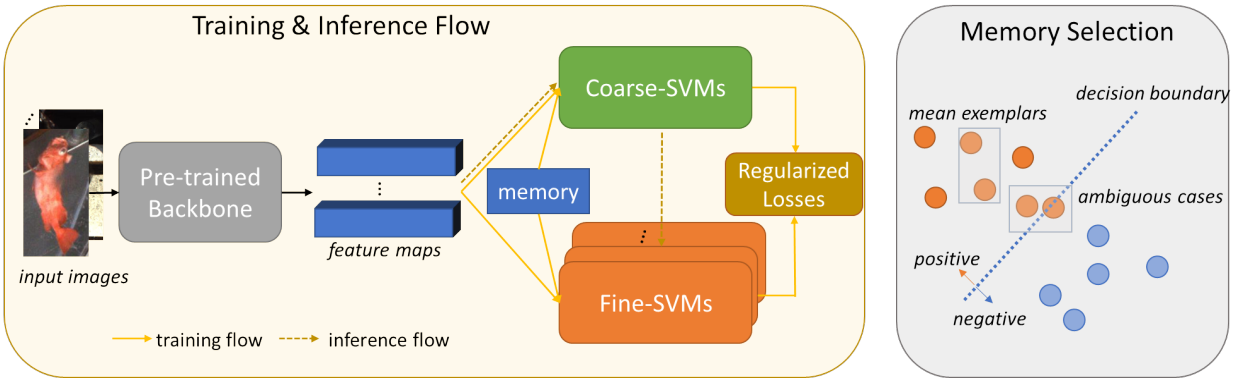


Figure 3.5: HCIL: New classes' feature maps from the fixed pre-trained backbone and old classes' feature maps from CIL memory are used to train coarse-level group SVMs and corresponding fine-level SVMs. After SVMs training, based on the distance between SVMs' decision boundary and each training data, CIL memory keeps adding new classes' features, i.e., hard cases, and adding positive exemplars from new classes based on herding [156]. During inference, the extracted feature map first goes into Coarse-SVMs. And based on the coarse-level group prediction, the extracted feature map goes into corresponding Fine-SVMs for species classification.

under the class incremental learning (CIL) scenario. In this section, we introduce a Hierarchical Class Incremental Learning (**HCIL**) model, which can jointly perform hierarchical classification and class incremental learning without catastrophically forgetting previously trained classes.

Our hierarchical class incremental learning (HCIL) method consists of a fixed pre-trained feature extraction backbone, a comprehensive memory selection module, regularization losses and hierarchical dynamic support vector machines (SVMs), i.e., SVMs are continually added with appearing of newly acquired classes as shown in Fig. 3.5. Since the first three parts are exactly same as **MSDHR** framework in Chapter 4, we only discuss hierarchical dynamic support vector machines (SVMs) in the following.

Hierarchical Dynamic SVMs Expansion During the training of a class incremental learning scenario, the total number of seen classes is increasing. Only the training data for a small number of classes have to be present at the beginning and new classes are added progressively. Thus, the growing-class classifier can be used in the incremental learning models. Our Coarse-SVMs consist of six SVMs for group-level prediction because there are six groups in the fish dataset as shown in Fig. 3.1. We use one-vs-all training strategy, where for each SVM only one group data are treated as positive and all rest of groups data are treated as negative. Each Coarse-SVM has its own corresponding Fine-SVMs. For example, the Coarse-SVM for the 'Sharks' group has four corresponding Fine-SVMs. And these four Fine-SVMs are added into the model sequentially with the availability of newly acquired shark species, which is called 'hierarchical dynamic SVMs expansion' in this section. During the inference, based on the Coarse-SVMs prediction, the feature map goes to corresponding Fine-SVMs for species classification so that the model can provide hierarchical predictions as shown in Fig. 3.5.

3.4 Experiments on HCIL

We compare our method with the above propose hierarchical classifier on NOAA's dataset under both 'hierarchical classification setting' and 'hierarchical class incremental learning setting'.

3.4.1 Hierarchical Classification Setting

This setting serves as a baseline for the next hierarchical class incremental learning setting. More specifically, in this setting, all training data for all classes are available at the same time, i.e., no incremental learning scenario is assumed. Training and testing data split is the same as Sec 3.1, where each individual fish has its own video data. Training and testing fish are totally different individual fish.

In this setting, our proposed **HCIL** model directly uses the fixed ResNet-101 backbone pre-trained on ImageNet-1k as the feature extractor and only trains our SVMs on NOAA's dataset. No CIL memory is involved, thus noted as **'HCIL w/o m'** in Table 3.2. As mentioned in Section 3.3, during inference, based on the Coarse-SVMs prediction, the feature map goes to corresponding Fine-SVMs for species classification so that the model can provide hierarchical predictions.

For a fair comparison, we allow the proposed hierarchical classifier in Sec 3.1 to utilize the same pre-trained ResNet-101 as the initial backbone, which is also further finetuned along with its classifier heads on NOAA's hierarchical fish dataset. In Table 3.2, we report image-based accuracy and video-based accuracy, noted as img and $video$ respectively, on both coarse (group) level and fine (species) level, noted as subscript C and F respectively.

Results are in Table. 3.2. Even though our proposed **'HCIL w/o m'** method fixes the pre-trained backbone, which is not finetuned by the NOAA's fish dataset, the performance is still comparable with the proposed hierarchical classifier in Sec 3.1, that allows updating the backbone and calculating cross-entropy loss functions on both levels. This shows the backbone pre-trained on large public datasets without finetuning can indeed possess the strong ability to extract discriminative features even on new classes or datasets.

3.4.2 Hierarchical Class Incremental Learning Setting

In this setting, both our method, HCIL, and the proposed hierarchical classifier in Sec 3.1 still utilize ResNet-101 [51] backbone pre-trained on ImageNet-1k. Testing data are still the same as the previous setting and cover all species. However, training data are divided into three **tasks**. The

Table 3.2: Hierarchical Classification Setting

Method	img_C	img_F	$video_C$	$video_F$
[97]	92.0	82.9	96.5	91.2
HCIL w/o m	91.8	81.2	95.9	90.7

Table 3.3: Hierarchical Class Incremental Learning Setting

Method	img_C	img_F	$video_C$	$video_F$
[97] w/ m	80.5	65.7	81.1	70.4
HCIL w/o m	82.8	69.9	86.2	75.3
HCIL	91.0	80.4	92.1	82.8
HCIL w/ Swin	**92.6**	**83.5**	**93.2**	**84.3**

first task includes one-third of the species within each group. The second task includes another one-third of the species within each group. The third task includes the rest species' data. There are no overlapping classes between the three tasks. However, the 'Skates' and 'Flatfish' groups have only two species each so one Fine-SVM for each group is enough. As a result, all data from these two groups are included in the first task.

When training our HCIL model, on the first task data, the feature maps from the fixed pre-trained backbone are used to train six Coarse-SVMs and corresponding Fine-SVMs. Based on SVMs confidence scores and the comprehensive memory selection introduced in Section **??**, HCIL constructs the memory. Next, when training on the following tasks, the memory's feature maps are also used to train Coarse-SVMs and newly added Fine-SVMs. We set the total number of hard cases n to 200, and the total number of positive exemplars m to 1800 so that the memory size won't increase as the incremental learning goes.

The proposed hierarchical classifier in Sec 3.1 does not have memory selection or classifiers expansion. For a fair comparison, when finetuning both pre-trained backbone and classifiers of the proposed hierarchical classifier in Sec 3.1 on later two tasks, we allow it to use the same size memory but randomly sampled from previously trained classes, denoted as **‘ [97] w/ m’** in Table 3.3. When training on each task, its classifiers always output predictions over 31 species and calculate loss functions. We evaluate the final trained models on all testing data.

Results are in Table. 3.3. Compared with the proposed hierarchical classifier in Sec 3.1 which is not designed for incremental learning, our HCIL model achieves better performance. For the comprehensive memory ablation study, we replace it with random selection, noted as **‘HCIL w/o m’** in Table 3.3, and the performance drops dramatically but is still better than the proposed hierarchical classifier in Sec 3.1 with randomly sampled memory. This ablation study shows the benefits of the comprehensive memory module. It also tells that under incremental learning setting, updating the backbone and classifiers even with some randomly sampled memory, makes the deep model suffer from catastrophic forgetting. For the backbone ablation study, we replace ResNet-101 [51] with Swin-Transformer [89], noted as **‘HCIL w/ Swin’**, and get the best performance.

3.5 Summary

We construct a hierarchical dataset and propose an efficient **Hierarchical Classifier** to enforce hierarchical data structure for fish species identification, combined with an efficient training strategy, and two video-based inference schemes. Our experiments show that the integrated use of these three main strategies in our hierarchical classifier indeed achieves higher accuracy than the baseline method, a flat classifier. Additionally, hierarchical predictions allow images that cannot be confidently classified at the fine level to be confidently classified at a coarse level for experts future examination, which especially improve overall accuracy on tail-class species identification significantly by stopping at coarse-level predictions. Furthermore, our proposed **HCIL** model can jointly perform hierarchical classification and class incremental learning. The proposed HCIL achieves better performance under class incremental learning scenario than the above proposed hierarchical classifier, without catastrophically forgetting previously trained classes.

Chapter 4

EXPERT-AND-SAMPLES-AWARE INCREMENTAL LEARNING UNDER LONGTAIL DISTRIBUTION

Most works in class incremental learning (CIL) assume disjoint sets of classes as tasks. Although a few works deal with overlapped sets of classes, they either assume a balanced data distribution or assume a mild imbalanced distribution. Instead, in this section, we explore one of the understudied real-world CIL settings where (1) different tasks can share some classes but with new data samples, and (2) the training data of each task follows a long-tail distribution. We call this setting CIL-LT. We hypothesize that previously trained classification heads possess prototype knowledge of seen classes and thus could help learn the new model. Therefore, we propose a method with the multi-expert idea and a dynamic weighting technique to deal with the exacerbated forgetting introduced by the long-tail distribution. Experiments show that the proposed method effectively improves the accuracy in the CIL-LT setup on MNIST, CIFAR10, and CIFAR100.

4.1 Proposed Method

4.1.1 Motivation

Most previous works reinitialize classification heads when new tasks come in while some work such as [41] concatenate logits from non-overlapped heads to keep a unified classification head. However, we hypothesize that previously trained classification heads possess prototype knowledge of seen classes and will help the continual learning of the new model. In general, our proposed model dynamically creates an overlapped expert, *i.e.,* **a light-weight linear head**, for each task as shown in Fig. 4.1. The remaining questions are (1) how to merge their decisions to get a final prediction for each sample, and (2) how to maintain their expertise as more novel classes come in and more seen classes wither away.

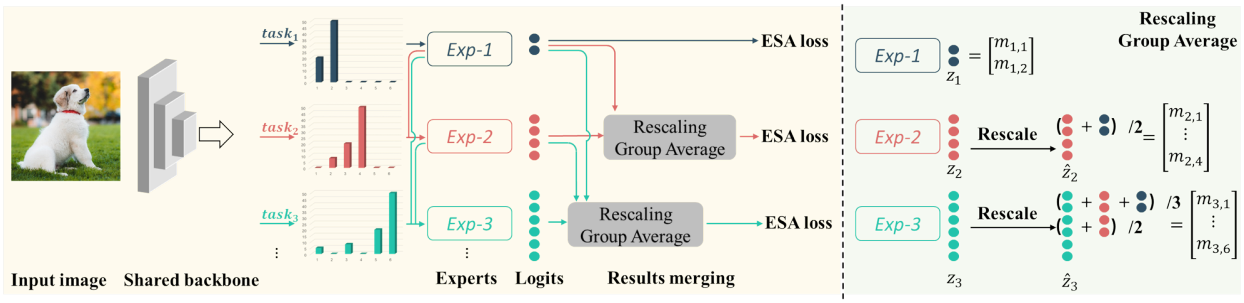


Figure 4.1: **Pipeline**: When training a model on a new task, our model first expands a new expert, *i.e.,* a linear classification head, which covers all seen classes and novel classes in the new task. To utilize previously learned prototypes of seen classes saved as the weights in previous experts, all the training data are also inferred by previous experts and get corresponding logits (before softmax). To fairly merge logits from different experts, we propose the rescaling of group average to integrate logits among overlapped classes. Finally, the proposed experts-and-samples-aware (ESA) loss is applied to find the balance between the number of experts and the number of training samples. Note that: the inference flow is exactly the same as the training flow of the last task. This ensures consistency between the training targets and testing evaluation.

4.1.2 Multi-Expert Strategy

To tackle the first question, we make our experts output their knowledge, i.e., logits (before softmax), on all novel classes and classes seen so far as shown in Fig. 4.1. Thus, overlapped classes among different tasks have corresponding outputs from different experts which gives experts the opportunity to communicate with each other via the proposed rescaling-group-average operation as shown in the right of Fig. 4.1. This is similar to the aggregation method in [16] but our method deals with dynamically expanded experts. Specifically, we denote task ID as $i = 1...T$, where T is the current training task ID and denote logits from expert E_i as $z_i \in R^{|C_i|}$, where C_i is the total number of classes learned so far after task i. Since each logit in logits $z_i \in R^{|C_i|}$ is from the same expert, they are comparable with each other. Thus, we only need to scale logits from different experts. To fairly merge the logits $z_i \in R^{|C_i|}$ from different experts E_i, we first scale all logits into

a comparable range with the norm of learned weights:

$$\hat{\mathbf{z}}_i = \frac{\left\|\mathbf{w}_{1,1}\right\|^2}{\left\|\mathbf{w}_{i,1}\right\|^2} \cdot \mathbf{z}_i, \tag{4.1}$$

where $w_{i,1}$ are the weights of expert E_i for class 1.

Furthermore, we use group average to get the merged logit $m_{T,c}$ of each class c for each sample in the current task T, *i.e.,*

$$\mathbf{m_{T,c}} = \frac{1}{|S_c|} \sum_{i \in S_c} \hat{\mathbf{z}}_{i,c}, \tag{4.2}$$

where S_c is the set of expert IDs that have been trained with class c and $\hat{z}_{i,c}$ is the c^{th} entry in logits $\hat{z}_i$ from expert E_i as shown in Fig. 4.1. Finally, the softmax operation is applied on the vector m_T to get the probability scores in $task - T$.

Figure. 4.1 shows how to merge logits when training the model on $task - 1$, $task - 2$, and $task - 3$. Our inference flow is exactly the same as the last training flow.

To tackle the second question, *i.e.,* how to maintain experts' expertise as more novel classes come in and more seen classes wither away, we propose to apply an experts-and-samples-aware (ESA) loss to the merged logits, *i.e.,* m_T, in each task.

4.1.3 Experts-Aware Loss

We first use the expert-aware (EA) cross-entropy loss as the classification loss on task T:

$$L_{EA}^{T} = -\sum_{j} y \log\left(\sigma\left(\mathbf{m}_T\right)\right), \tag{4.3}$$

where $\sigma(\cdot)$ is the softmax operation, y is the ground-truth one-hot vector, j represents the jth sample in the current task. Since this loss is applied on the merged logits m_T, the gradient will back-propagate via all experts to the backbone. The trained weights of previous experts can be treated as feature prototypes of each seen class. Thus, we give previous experts a smaller learning rate than the newly expanded expert and the backbone to mitigate forgetting of seen classes. This

is because when the backbone gets updated, the seen classes' features will change and maybe drift away from the learned prototypes, i.e., previous experts' weights, causing a big loss to limit the change of backbone.

4.1.4 Samples-Aware Loss

Besides, long-tail distribution in the CIL-LT scenario exacerbates the forgetting by overfitting the majority classes, which have the most training samples. This difference between traditional long-tail recognition and CIL-LT also makes it challenging to find an optimal re-weighting technique for the CIL-LT scenario. Thus, we further propose the following sample-aware (SA) cross-entropy loss with a dynamic weighting technique to give more emphasis on samples of minority classes.

$$
\begin{aligned}
L_{SA}^{T} &= -\sum_{j_T^*} y \log\left(\sigma\left(\mathbf{m}_T\right)\right) \cdot w_{T,c_{j_T^*}}, \\
w_{T,c_{j_T^*}} &= log\left(\frac{N_{T,c_{j_T^*}} \cdot P_T}{M_{T,c_{j_T^*}}} + 1\right) + 1,
\end{aligned}
\tag{4.4}
$$

where j_T^* represents the jth samples of seen classes before task T, $c_{j_T^*}$ is the corresponding class, $N_{T,c_{j_T^*}}$ is the total number of experts before task T covering class $c_{j_T^*}$, and $M_{T,c_{j_T^*}}$ is the total number of training samples of class $c_{j_T^*}$ in task T, P_T is the total number of samples in task T. As the number of tasks increases, the seen classes have fewer and fewer samples for training thus $M_{T,c_{j_T^*}}$ becomes smaller and smaller; but more and more expanded experts cover seen classes thus $N_{T,c_{j_T^*}}$ becomes bigger and bigger, causing the weight $w_{T,c_{j_T^*}}$ bigger and bigger. Thus, this dynamic weighting can give more emphasis on seen class samples with more tasks coming in by considering both the number of experts and the number of samples.

Finally, our experts-and-samples-aware (ESA) loss on task T is as follows:

$$
L_{ESA}^{T} = L_{EA}^{T} + L_{SA}^{T}. \tag{4.5}
$$

4.2 Experiments

We empirically validate the efficacy of our proposed ESA method by comparing it with the state-of-arts under the CIL-LT scenario. For each of MNIST [34], CIFAR10 [72], and CIFAR100 [72] benchmarks, we create three random data splits, i.e., random class order under the CIL-LT scenario.

Table 4.1: **Comparison** with two metrics (A{5, 10} and I{5, 10}: %) in MNIST-Overlap50-IF100, CIFAR10-Overlap50-IF100, and CIFAR100-Overlap30-IF250 under CIL-LT setup. K represents memory size.

	MNIST10 (K=200)		CIFAR10 (K=200)		CIFAR100 (K=2,000)	
Methods	**A5(↑)**	**I5(↓)**	**A5(↑)**	**I5(↓)**	**A10(↑)**	**I10(↓)**
Joint	88.49	-	91.40	-	62.16	-
EWC [69]	$39.23_{\pm 0.98}$	$49.26_{\pm 0.84}$	$63.16_{\pm 1.02}$	$28.24_{\pm 0.78}$	$27.67_{\pm 0.85}$	$34.49_{\pm 0.54}$
Rwalk [20]	$39.73_{\pm 0.88}$	$48.76_{\pm 0.69}$	$64.38_{\pm 0.87}$	$27.02_{\pm 0.98}$	$32.56_{\pm 0.99}$	$29.60_{\pm 0.77}$
iCaRL [114]	$59.78_{\pm 0.73}$	$28.71_{\pm 0.63}$	$52.16_{\pm 0.78}$	$39.24_{\pm 0.83}$	$33.74_{\pm 0.84}$	$28.42_{\pm 1.33}$
GDumb [110]	$41.71_{\pm 0.71}$	$46.78_{\pm 0.62}$	$37.22_{\pm 0.63}$	$55.18_{\pm 0.65}$	$28.33_{\pm 0.68}$	$33.83_{\pm 1.45}$
BiC [163]	$33.38_{\pm 0.77}$	$55.11_{\pm 0.59}$	$64.23_{\pm 0.66}$	$27.17_{\pm 0.76}$	$37.68_{\pm 0.69}$	$24.48_{\pm 0.99}$
DER++ [15]	$84.55_{\pm 0.56}$	$3.94_{\pm 0.55}$	$65.10_{\pm 0.54}$	$26.30_{\pm 0.89}$	$35.57_{\pm 0.56}$	$26.59_{\pm 1.52}$
RM [12]	$86.50_{\pm 0.43}$	$1.99_{\pm 0.49}$	$65.97_{\pm 0.33}$	$25.43_{\pm 0.56}$	$40.53_{\pm 0.49}$	$21.63_{\pm 0.76}$
CLS-ER [10]	$86.08_{\pm 0.40}$	$2.41_{\pm 0.37}$	$66.88_{\pm 0.35}$	$24.52_{\pm 0.62}$	$41.52_{\pm 0.61}$	$20.64_{\pm 0.51}$
ESA (ours)	$\mathbf{88.51}_{\pm 0.31}$	$\mathbf{-0.02}_{\pm 0.23}$	$\mathbf{71.32}_{\pm 0.44}$	$\mathbf{20.08}_{\pm 0.32}$	$\mathbf{44.21}_{\pm 0.68}$	$\mathbf{17.95}_{\pm 0.54}$

4.2.1 Experimental Setup

Benchmark CIL-LT Setup. As the naming format mentioned in Fig. 1.2, we create MNIST-Overlap50-IF100, CIFAR10-Overlap50-IF100, CIFAR100-Overlap20-IF250, CIFAR100-Overlap30-IF250, and CIFAR100-Overlap50-IF250. For each of them, we create three random class orders. For MNIST and CIFAR10, we use 5 tasks (2 major novel classes per task). For CIFAR100, we use

10 tasks (10 major novel classes per task). Our split also makes sure the samples of seen classes in the new task are different from samples in any previous tasks. Our experimental results are the average values on three random data splits. We consider an offline learning setup where a model can be updated multiple times.

4.2.2 Evaluation Metrics.

We use two popular metrics in this work: *Last Accuracy (A5)* and *Intransigence (I5)* used in [12]. 'Last' means the metric is calculated after all tasks are trained and '5' represents 5 tasks in MNIST and CIFAR10. Accordingly, the metrics are denoted as 'A10' and 'I10' for CIFAR100. Finally, we use various memory sizes for different datasets denoted as K in each table. Please note that we repeat each experiment three times on different class orders to report means and standard deviations for both metrics.

4.2.3 Baselines and Implementation Details.

We compare our proposed ESA with classic and state-of-the-art methods including EWC [69], Rwalk [20], iCaRL [114], BiC [163], GDumb [110], DER++ [15], RM [12], CLS-ER [10]. Note that CLS-ER [10] is designed for general continual learning and does not make any assumption about the distribution of the data. RM [12] and GDumb [110] are specifically designed for blurry-CIL setup. All methods in the table use the same MLP400, ResNet18, and ResNet32 as their feature extraction backbones for MNIST, CIFAR10, and CIFAR100, respectively, and are trained from scratch.

Following [12], for the training hyperparameters on MNIST and CIFAR10/100, we use 32 as batch size, 256 epochs, and a cosine annealing learning rate scheduler ranging from 0.05 to 0.0005. We use CutMix [170] and AutoAug [30] as data augmentation for all methods. We use 0.1 as a factor on the learning rate of previous experts in the new tasks. EWC [69] and Rwalk [20] don't consider any memory but for fair comparison, following [12], we use an episodic memory, which is updated through reservoir sampling [117]. We also report 'Joint' training in our experiments,

where all data from different tasks are available at the same time for training. For all other methods, we use uncertainty-based memory selection proposed by RM [12].

4.2.4 Quantitative Results

We compare the proposed ESA to other methods in the CIL-LT setup. As shown in Table 4.1, the proposed ESA method outperforms all other methods on MNIST10, CIFAR10, and CIFAR100. We outperform CLS-ER [10], which was originally designed for general-CIL setup. Interestingly, we also noticed on MNIST, that joint training no longer serves as the ceiling performance under the CIL-LT setup. This is because the data distribution is also imbalanced even in joint training and it also suffers from long-tail distribution but our ESA method achieves even better performance.

Table 4.2: **Ablation Study** on each proposed component, *i.e.,* multi-expert architecture, rescaling group average, and dynamic weighting for ESA loss in CIFAR10-Overlap50-IF100. Note: Standard deviations are omitted due to space limitations.

Methods	**Memory**	**Multi-Expert**	**Experts Scaling**	**Dynamic-Weighting**	**Regularization**	CIFAR10 (K=200)	
						A5(↑)	**I5(↓)**
Ablation	rainbow [12]					66.62	24.98
	rainbow [12]	✓				65.93	25.67
	rainbow [12]	✓	✓			67.96	23.64
	rainbow [12]	✓	✓		✓	65.96	26.64
	rainbow [12]	✓	✓	class-wise		66.46	25.14
	rainbow [12]	✓	✓	constant value, 1		69.96	21.64
	herding [156]	✓	✓	samples-aware		69.43	22.17
	reservoir [117]	✓	✓	samples-aware		68.92	22.68
ESA (ours)	rainbow [12]	✓	✓	samples-aware		**71.32**	**20.08**

4.2.5 Ablation Study

Table 4.2 shows when adding dynamically expanded experts without our proposed scaling method, the performance drops. This is because logits from different experts have different scales and are

not comparable with one another. We can also see the proposed dynamic weighting achieves better performance than constant weighting and class-wise weighting. Table. 4.3 shows the comparison of different methods on CIFAR100 under another two different overlap ratios, *i.e.,* 20, and 50. We can see ESA still achieves the best last accuracy in both setups.

Table 4.3: **Comparison** with two metrics (A{10} and I{10}: %) in CIFAR100 for another two overlap ratios.

	Overlap20		Overlap50	
Methods	**A10(↑)**	**I10(↓)**	**A10(↑)**	**I10(↓)**
Joint	64.05	-	64.71	-
EWC [69]	$32.45_{\pm 0.78}$	$31.60_{\pm 0.96}$	$33.95_{\pm 0.63}$	$30.76_{\pm 0.98}$
Rwalk [20]	$36.22_{\pm 0.67}$	$27.83_{\pm 0.69}$	$38.05_{\pm 0.96}$	$26.66_{\pm 0.65}$
iCaRL [114]	$37.11_{\pm 0.57}$	$26.94_{\pm 0.91}$	$37.57_{\pm 1.22}$	$27.14_{\pm 1.63}$
GDumb [110]	$29.53_{\pm 0.96}$	$34.52_{\pm 0.87}$	$31.86_{\pm 1.53}$	$32.85_{\pm 0.75}$
BiC [163]	$42.08_{\pm 0.88}$	$21.97_{\pm 0.82}$	$43.51_{\pm 0.63}$	$21.20_{\pm 0.68}$
RM [12]	$45.52_{\pm 0.56}$	$18.53_{\pm 0.58}$	$48.01_{\pm 0.78}$	$16.70_{\pm 0.57}$
CLS-ER [10]	$45.86_{\pm 0.51}$	$18.19_{\pm 0.68}$	$47.57_{\pm 0.54}$	$17.14_{\pm 0.72}$
ESA (ours)	$\mathbf{47.22}_{\pm 0.50}$	$\mathbf{16.83}_{\pm 0.53}$	$\mathbf{50.06}_{\pm 0.53}$	$\mathbf{14.65}_{\pm 0.65}$

4.3 Summary

In this section, we investigated an understudied CIL setting, called CIL-LT. Different from existing CIL setups, in the CIL-LT,(1) different tasks can share a limited and random number of classes, (2) the training data of each task follows a long-tail distribution with the imbalance factor in a wide range. We propose a multi-expert architecture together with rescaling group average operation to fully utilize the learned prototypes saved in the trained weights of previous experts. The model gathers 'opinions' from different light-weight experts before making the final prediction.

We further present an experts-and-samples-aware (ESA) loss to deal with the exacerbated forgetting introduced by the long-tail distribution by giving more emphasis on seen classes with more tasks coming in. The dynamic weighting in ESA is designed for the CIL-LT.

Chapter 5

PROTOTYPE-GUIDED TEXT PROMPT SELECTION FOR CONTINUAL LEARNING

Text-prompt-based approaches for continual learning leverage pre-trained text encoders and learnable prompts to encode textual features for sequentially arrived classes over time. A common challenge encountered by existing works is how to learn fine-grained text prompts, which implicitly carry semantic information of new classes, so that the textual features of newly arrived classes do not overlap with those of trained classes, thereby mitigating the catastrophic forgetting problem. To address this challenge, we propose a novel approach **Pro**totype-guided **T**ext **P**rompt **S**election (ProTPS) to intentionally increase the training flexibility thus enforcing learning fine-grained text prompts. Specifically, our ProTPS aggregates the image and text encoders to learn class-specific vision prototypes and text prompts. Vision prototypes guide the selection and learning of text prompts that encode exclusive fine-grained textual features of each class. We evaluate our ProTPS in both class incremental (CI) setting and cross-datasets continual (CDC) learning setting. Since our ProTPS achieves performance close to the upper bounds, we further collect a real-world marine species dataset, named Marine112, to bring new challenges to the community. Marine112 is a fine-grained dataset with long-tail distribution and is naturally suited for the class and domain incremental (CDI) learning setting. The results under three settings show that our approach performs favorably against the recent state-of-the-art methods.

5.1 Methodology

We start with a pilot study in Sec. 5.1.1, establishing the feasibility of using vision prototypes for initial predictions in continual learning. These vision prototypes can then be used as guidance for our text prompt selection in our ProTPS framework, which will be introduced in Sec. 5.1.2. We

Table 5.1: **Pilot Study.** Comparison of different CLIP-based methods in the class incremental (CI) setting on CIFAR100 [72] with ViT-L/14 [37] backbone. After learning each task $T-i$, the accuracy on exposed classes from all the trained tasks (i.e., $T-1, 2, ..., i$) is reported. The upper-bound method is "Linear Probe CLIP" [111] using all tasks' training data at the same time to optimize a linear classifier. The column "Classifier" denotes the encoder from which the linear classifier weights are derived.

Method	Classifier	T-1	T-2	T-3	T-4	T-5	T-6	T-7	T-8	T-9	T-10
Continual-CLIP [133]	text	95.3	92.3	82.1	79.2	78.5	77.9	75.7	75.2	74.8	73.4
Zero-Shot CLIP [111]	text	95.7	**92.9**	85.2	82.9	82.3	82.1	79.6	79.6	79.2	78.3
Prototypes-CLIP	vision	**97.6**	92.8	**88.1**	**85.3**	**84.3**	**84.3**	**81.9**	**81.9**	**81.2**	**80.0**
Upper-bound	-	-	-	-	-	-	-	-	-	-	86.7

also detail the proposed *prompt-prototype contrastive loss* in Sec. 5.1.3 that can align and enhance text prompts and vision prototypes.

5.1.1 Pilot study

Continual learning formulation. We define a sequence of T training tasks, denoted by $D = \{D_1, ..., D_T\}$, with each task $D_t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$ comprising n_t images x_i^t and their respective labels y_i^t. Training on task D_t occurs without access to data from prior tasks $\{D_1, ..., D_{t-1}\}$. In the conventional class incremental learning scenario, each task contains non-overlapped classes.

In the pilot study, we first assess the performance of two methods that utilize frozen CLIP [111], in the conventional class incremental learning scenarios, on CIFAR100 [72]. Tab. 5.1 shows the setup of 10 training tasks in sequence (10 new classes per task). For each task, Continual-CLIP [133] adopts a single unchanged prompt template, "a bad photo of a {CLS}.", where {CLS} is a placeholder for the class label name, to derive the classifier via the text encoder. Meanwhile, for each task, zero-shot CLIP [111] predefines 18 text prompt templates and uses an ensemble

strategy to construct the classifier through the text encoder. Both methods compute the visual and textual similarities to perform the classification task. In Tab. 5.1, we can see in the continual learning scenario, that prompt engineering and ensembling can bring improvement over a single prompt template.

We further apply the basic nearest-prototype classier to CLIP [111] denoted as Prototypes-CLIP in Tab. 5.1. We characterize each class prototype by calculating the average image embedding for that class via the frozen CLIP image encoder. These prototypes then function as the weights of a linear classifier. As new tasks commence, we retain the prototypes from previous tasks and calculate the new class prototypes to expand the classifier. We find this strategy achieves slightly better performance than text prompt engineering and ensembling. This pilot study motivates us to use prototypes as guidance for our prompt selection.

5.1.2 Framework of ProTPS

Our proposed novel Prototype-guided Text Prompt Selection (ProTPS) is built upon frozen CLIP [111]. In contrast to existing text-prompt-based methods [133, 149, 151, 180], our ProTPS continually encourages text prompts to capture class-specific **fine-grained** semantic features.

Refinement of vision prototypes. Our pilot study inspires us to use the prototype I_i of class i to initialize the weights of the $classifier_1$ at the beginning of Task-1 as shown in Fig. 5.1, which is referred to as ProTPS's "vision classifier". We can see vision prototypes as global image-level representations of each class. Throughout the training of Task-1, we finetune these vision prototypes for refinement. As new tasks commence, while the "vision classifier" is expanded with prototypes of new classes, we ensure previously learned prototypes remain frozen to conserve the integrity of earlier learned representations, as illustrated in Fig. 5.1.

Prototypes-guided text prompts selection. Each class prototype I_i is paired with a learnable text prompt P_i. For a single image, ProTPS selects the text prompt that corresponds to the highest cosine similarity score λ_i as indicated by vision prototypes, , λ_1 in Task-1 of Fig. 5.1. Recognizing that the "vision classifier" may not always yield accurate prediction, we scale the selected text prompt by λ_i as shown in Fig. 5.1 and concatenate it at the embedding level with all class

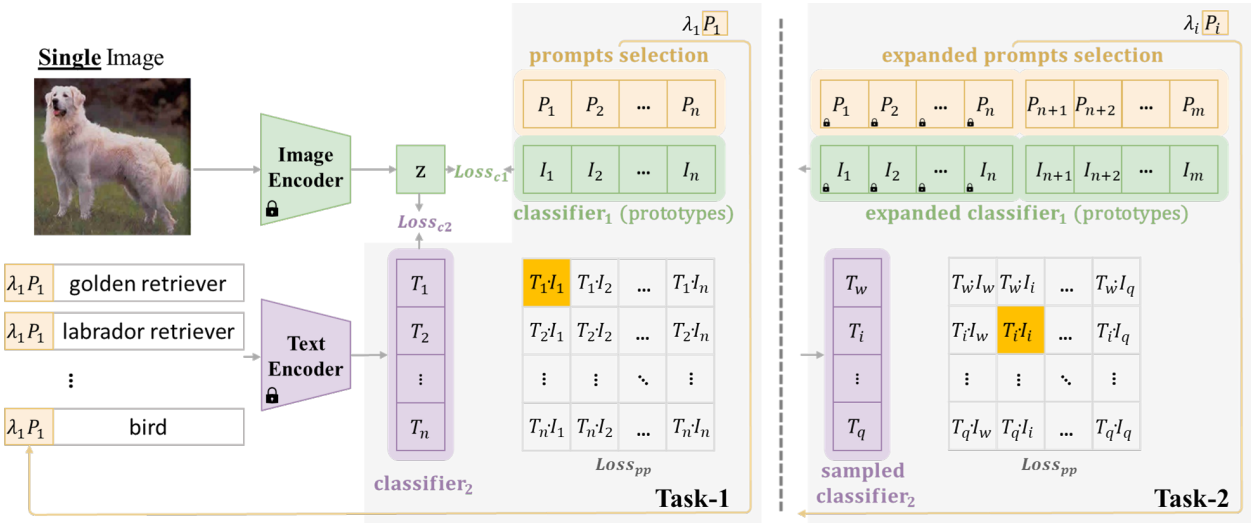


Figure 5.1: **ProTPS framework.** In Task-1, we use the prototype I_i of class i to initialize the weights of the linear $classifier_1$, which is referred to as ProTPS's "vision classifier". We finetune these vision prototypes for refinement. Each class prototype I_i is paired with a learnable text prompt P_i. For a single input image, we select a text prompt to concatenate with all class names in Task-1 to create $classifier_2$, which is referred to as the "text classifier" of ProTPS. In Task-2, prototypes and paired prompts are expanded. We freeze prototypes $I_1, ..., I_n$ and prompts $P_1, ..., P_n$ trained in the previous task. We propose a sampling method to create $sampled\ classifier_2$ to help new text prompts capture exclusive fine-grained features for new classes. For the later tasks, the expansion rule is the same as Task-2.

names in Task-1. These combined textual inputs are processed through the text encoder to derive $classifier_2$, which is referred to as ProTPS's "text classifier" and can be seen as **fine-grained** semantic features of each class.

When new tasks start, we expand text prompts for new classes and ensure previously learned prompts remain frozen to conserve the integrity of earlier learned fine-grained semantic features, as depicted in Fig. 5.1. Besides new class names, we randomly sample a fixed number of class names from previous classes before concatenating with the selected text prompt in the new task, yielding $sampled\ classifier_2$ in Fig. 5.1. This sampling strategy allows the newly expanded text prompts to interact with previous class names during training to help new text prompts learn exclusive fine-grained features for new classes while keeping a constant GPU memory usage.

One aggregated classifier. During inference, we also keep the proposed prompt selection and ProTPS generates predictions by calculating the average cosine similarity score of each class from the "vision classifier" and "text classifier".

5.1.3 Optimization Objectives of ProTPS

Classification losses. Given one single image input of class i, we first compute two cross-entropy losses respectively for dual classifiers of ProTPS based on the image feature $\mathbf{z}$:

$$Loss_{c1} = -\log \frac{e^{\langle \mathbf{z},\ \mathbf{I}_i \rangle / \tau}}{\sum_{k=k_0}^{k_1} e^{\langle \mathbf{z},\ \mathbf{I}_k \rangle / \tau}}, \tag{5.1}$$

$$Loss_{c2} = -\log \frac{e^{\langle \mathbf{z},\ \mathbf{T}_i \rangle / \tau}}{\sum_{k=k_0'}^{k_1'} e^{\langle \mathbf{z},\ \mathbf{T}_k \rangle / \tau}}, \tag{5.2}$$

where τ is the temperature parameter learned by CLIP [111], $\langle \cdot, \cdot \rangle$ denotes the cosine similarity, k_0 and k_1 respectively represent the start and end indices of newly expanded prototypes by default unless otherwise specified in experiments, , $k_0 = n{+}1$ and $k_1 = m$ in Task-2 as depicted in Fig. 5.1. This is to ensure that one of the newly expanded text prompts can be selected and optimized to learn fine-grained features of new classes. k_0' and k_1' respectively represent the start and end indices of weights vectors in the "text classifier", , $k_0' = 1$ and $k_1' = n$ in Task-1 and $k_0' = w$ and $k_1' = q$ in Task-2 as depicted in Fig. 5.1.

Prompt-prototype contrastive loss. Although vision prototypes and text prompts represent respectively global image-level representations and fine-grained textual features of each class, we propose a "prompt-prototype contrastive loss", $Loss_{pp}$, to align and enhance text prompts and vision prototypes, which further boosts the performance. It is different from the well-studied contrastive loss, such as SupCon [68], in two aspects: (1) it refocuses the goal from improving encoders to refining text prompts and vision prototypes, and (2) it is formulated using just a single input image. Given one single image input of class i, the proposed "prompt-prototype contrastive loss", depicted in Fig. 5.1, is as follows:

$$Loss_{pp} = (1 - \langle \mathbf{T}_i,\ \mathbf{I}_i \rangle) + \frac{1}{N} \sum_{j \neq i \text{ or } k \neq i}^{N} \langle \mathbf{T}_j,\ \mathbf{I}_k \rangle, \tag{5.3}$$

where $\langle \cdot, \cdot \rangle$ denotes the cosine similarity. The first term is the cosine distance of the positive pair (yellow colored grids in Fig. 5.1) between prototypes and weights of "text classifier", given the single input image of class i; the second term is the mean cosine similarity of negative pairs (uncolored grids in Fig. 5.1) between prototypes and weights of "text classifier". N is the total number of negative pairs. With λ_{pp} as the balance factor, the whole optimization objective is:

$$Loss = Loss_{c1} + Loss_{c2} + \lambda_{pp} \cdot Loss_{pp}, \tag{5.4}$$

5.2 Experiments

In this section, we first detail the implementation in Sec. 5.2.1. We then compare the proposed ProTPS with other popular methods in the class-incremental (CI) setting in Sec. 5.2.2 and the cross-datasets continual (CDC) learning setting in Sec. 5.2.3. We also introduce a new fine-grained dataset, Marine112, for class and domain incremental (CDI) learning setting in Sec. 5.2.4. Finally, we present ablation studies to evaluate ProTPS's individual components in Sec. 5.2.5.

5.2.1 Implementation Details

Datasets. Our experiments are conducted on ImageNet100, CIFAR100 [93], and Marine112. ImageNet100 is a subset of ImageNet-1K [33], containing images sized 224x224 from 100 classes.

We use the same 100 classes as AttriCLIP [149], with about 1,300 training and 50 testing images per class. Details are provided in the Section 5.3.1. CIFAR100 has 100 classes with 500 training and 100 testing images per class. Both CIFAR100 and ImageNet100 are split into 10 tasks with 10 classes in each task. Besides, we collect a real-world fine-grained Marine112 dataset, which is naturally suited for the class and domain incremental (CDI) learning and reflects the long-tail distribution [91] observed in nature. Its data covers 112 marine species in total across 6 different years.

Metric. After learning each task $T-i$, the accuracy is evaluated on exposed classes of all trained tasks (i.e., $T-1, 2, ..., i$) [149], as detailed in Section 5.3.3.2.

Baselines. We compare the proposed ProTPS with (1) visual-prompt-based methods (L2P [154], DualPrompt [153], CODA-P [129], and HiDe-Prompt [147]), and (2) text-prompt-based methods (CoOp [180] with 1000 memory buffer, ContinualCLIP [133], Zero-Shot CLIP [111], and AttriCLIP [149]). Compared with AttriCLIP [149], our ProTPS introduces a negligible memory increase of 105 KB, i.e., 0.006% of 1.71 GB full model size. The upper-bound method is the Linear Probe CLIP from [111] by using all tasks' training data at the same time to optimize a linear classifier. We use the original CLIP pre-trained ViT-L/14 [37] as the image encoder for the upper-bound method, our ProTPS and all baselines (explanation and training details are in Section 5.3.3.1).

5.2.2 Class-Incremental Learning

CIFAR100. The results are reported in Tab. 5.2, where $text^*$ denotes the selection and learning of our text prompts are guided by our vision prototypes in ProTPS. Our ProTPS achieves the best accuracy compared with recent state-of-the-art (SOTA) prompt-based methods. Specifically, ProTPS without any data buffer outperforms CoOP [180] by +15.7% and beats the previous best model AttriCLIP [149] by +1.9%. ProTPS is only 3.4% distance from the upper bound, indicating the effectiveness of mitigating catastrophic forgetting.

ImageNet100. In Tab. 5.3, our ProTPS outperforms the previous SOTA AttriCLIP [149] by +10.6% on ImageNet100. The accuracy of ProTPS is only 1.6% distance to the upper bound. We further report the accuracy of different classifiers in Fig. 5.2 and Tab. 5.4. We can see that **ProTPS's**

Table 5.2: Accuracy of different continual learning methods on CIFAR100 [72] under CI setting. Second-best results are underlined. T-i denotes Task-i.

Prompt	Method	T-1	T-2	T-3	T-4	T-5	T-6	T-7	T-8	T-9	T-10
visual	L2P [154]	60.1	47.8	45.9	47.1	48.9	48.9	49.4	49.0	50.0	49.4
	DualPrompt [153]	73.9	60.2	56.4	55.3	54.5	53.3	53.4	53.1	53.5	54.0
	CODA-P [129]	94.3	87.3	83.7	82.7	81.3	78.2	76.7	74.0	74.1	72.3
	HiDe-Prompt [147]	95.4	93.6	90.7	<u>89.5</u>	<u>87.9</u>	<u>85.5</u>	<u>84.0</u>	<u>82.0</u>	<u>82.2</u>	81.0
text	CoOp [180] (1000)	95.8	90.7	85.2	83.4	80.8	75.8	74.7	71.7	71.3	67.6
	Continual-CLIP [133]	95.3	92.3	82.1	79.2	78.5	77.9	75.7	75.2	74.8	73.4
	Zero-Shot CLIP [111]	95.7	92.9	85.2	82.9	82.3	82.1	79.6	79.6	79.2	78.3
	AttriCLIP [149]	**97.8**	<u>93.7</u>	<u>91.0</u>	87.5	84.7	82.5	82.3	81.9	81.7	<u>81.4</u>
*text**	**ProTPS** (ours)	**97.8**	**94.5**	**91.6**	**90.6**	**89.1**	**87.3**	**85.5**	**84.3**	**84.2**	**83.3**
	Upper bound	-	-	-	-	-	-	-	-	-	86.7

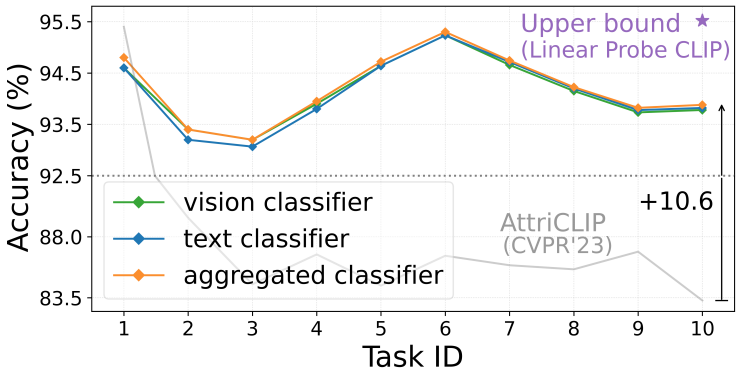


Figure 5.2: Accuracy evolution of ProTPS's different classifiers on ImageNet100 [33] under CI setting. The vision classifier and text classifier of ProTPS are well-aligned.

Table 5.3: Accuracy of different continual learning methods on ImageNet100 [33] under CI setting. Second-best results are underlined. T-i denotes Task-i.

Prompt	Method	T-1	T-2	T-3	T-4	T-5	T-6	T-7	T-8	T-9	T-10
visual	L2P [154]	60.9	53.5	49.8	48.6	47.2	48.1	49.5	49.7	48.1	48.3
	DualPrompt [153]	72.5	60.8	55.7	54.3	53.2	53.6	53.8	53.3	52.8	52.6
	CODA-P [129]	91.1	86.5	82.9	80.3	79.8	78.2	77.1	77.2	75.6	75.2
	HiDe-Prompt [147]	92.3	<u>90.1</u>	<u>89.8</u>	<u>87.5</u>	<u>85.5</u>	85.9	84.5	<u>85.9</u>	83.5	82.9
text	CoOp [180] (1000)	89.2	83.2	76.7	79.8	79.9	82.3	79.7	80.1	80.3	79.3
	Continual-CLIP [133]	93.3	87.6	83.1	81.7	80.5	80.2	79.3	78.5	76.9	75.4
	Zero-Shot CLIP [111]	91.8	89.2	89.3	91.1	92.2	92.6	90.9	79.3	78.5	75.1
	AttriCLIP [149]	<u>95.4</u>	89.4	84.5	86.7	84.4	<u>86.6</u>	<u>85.9</u>	85.6	<u>86.9</u>	<u>83.3</u>
text*	**ProTPS** (ours)	**95.5**	**93.4**	**93.7**	**94.0**	**94.7**	**95.3**	**94.7**	**94.2**	**93.8**	**93.9**
	Upper bound	-	-	-	-	-	-	-	-	-	95.5

Table 5.4: Last accuracy of vision, text, and aggregated classifiers in ProTPS on CIFAR100 [72] and ImageNet100 [33] under CI setting.

Method	Classifier	CIFAR100	ImageNet100
CoOp [180]	text	67.64	79.29
Continual-CLIP [133]	text	73.42	75.43
Zero-Shot CLIP [111]	text	78.28	75.12
AttriCLIP [149]	text	81.38	83.30
ProTPS (ours)	vision	83.16	93.78
	text	83.26	93.82
	aggregated	**83.28** (+1.90)	**93.92** (+10.62)

"vision classifier" and "text classifier" are well aligned. And the aggregated classifier achieves the best performance. Besides, the performance of ProTPS's "text classifier" is **not capped** by its "vision classifier" although text prompts selection is guided by prototypes.

5.2.3 Cross-Datasets Continual Learning

AttriCLIP [149] proposes another continual learning scenario, i.e., combining two datasets into a long-sequence domain-shift task, where continually training a model across different datasets is used to evaluate the ability of knowledge transfer from the previous dataset to the new one.

ProTPS's text prompts are transferable to a new dataset. The models are continually fine-tuned on 10 tasks of CIFAR100 after being trained on 10 tasks of ImageNet100, and finally evaluated on CIFAR100. This performance is denoted as "I2C" in Tab. 5.5. "FT" (forward transfer) is defined as the accuracy of "I2C" minus the accuracy of training on CIFAR100 only. When finetuning our ProTPS on CIFAR100, each class's text prompt is initialized from the prompts learned from ImageNet100 based on the highest cosine similarity between their paired prototypes. "{*Method*}-1" and "{*Method*}-2" are schemes defined by AttriCLIP [149] on how to expand the classifier for a

Table 5.5: Last accuracy of different methods on CIFAR100 [72] under CDC setting. The models are either trained on CIFAR100 only (C), or continually fine-tuned on CIFAR100 after being trained on ImageNet100 [33] (I2C).

Prompt	Method	C	I2C	FT↑
visual	L2P-1 [154]	49.4	45.7	-3.7
	DualPrompt-1 [153]	54.0	49.5	-4.5
	CODA-P-1 [129]	72.3	66.9	-5.4
	HiDe-Prompt-1 [147]	81.0	77.2	-3.8
text	CoOp-1 [180] (1000)	67.6	61.1	-6.5
	Continual-CLIP [133]	73.4	73.4	0
	Zero-Shot CLIP [111]	78.3	78.3	0
	AttriCLIP [149]	81.4	82.3	+0.9
*text**	**ProTPS** (ours)	**83.3**	**84.4** (+2.1)	**+1.1**

new dataset (please refer to AttriCLIP [149] for more details). In Tab. 5.5, our ProTPS outperforms the previous SOTA AttriCLIP [149] by +2.1% in the context of "I2C" and has the highest "FT" score. This indicates that ProTPS learns transferable semantic features encoded in text prompts that can help it to generalize better to a new dataset.

ProTPS is suitable for cross-datasets continual learning. In Tab. 5.6, models are evaluated on both datasets. For ProTPS, when start fine-tuning on CIFAR100, we expand our $classifier_1$ for CIFAR100 classes, and each class's text prompt is still initialized from the prompts learned from ImageNet100 based on the same rule. ProTPS beats visual-prompt-based and text-prompt-based methods, outperforming the previous SOTA AttriCLIP [149] by +7.3%.

Table 5.6: Last accuracy of different methods on ImageNet100 + CIFAR100 (I+C) where each model is continually trained on ImageNet100 and CIFAR100 in sequence under CDC setting.

Prompt	Method	I+C
visual	L2P-2 [154]	41.6
	DualPrompt-2 [153]	45.8
	CODA-P-2 [129]	62.7
	HiDe-Prompt-2 [147]	75.3
text	CoOp-2 [180] (1000)	55.4
	Continual-CLIP [133]	65.9
	Zero-Shot CLIP [111]	68.7
	AttriCLIP [149]	78.3
*text**	**ProTPS** (ours)	**85.6** (+7.3)

Table 5.7: Accuracy on fine-grained Marine112 dataset under CDI setting.

Prompt	Method	2015	2016	2019	2020	2021	2022
visual	CODA-P [129]	71.2	42.1	33.6	13.4	12.7	13.2
	HiDe-Prompt [147]	73.4	50.8	45.2	22.2	22.3	22.5
text	Continual-CLIP [133]	12.7	11.8	10.2	9.6	10.1	9.5
	Zero-Shot CLIP [111]	15.4	13.1	11.6	10.3	10.5	9.6
	CoOp [180] (1000)	65.3	30.2	15.1	10.2	11.4	10.3
	AttriCLIP [149]	76.1	48.3	34.8	11.9	13.6	13.9
*text**	**ProTPS** (ours)	**76.6**	**70.5**	**61.5**	**59.7**	**58.1**	**55.6** (+33.1)
	Upper bound	-	-	-	-	-	78.2

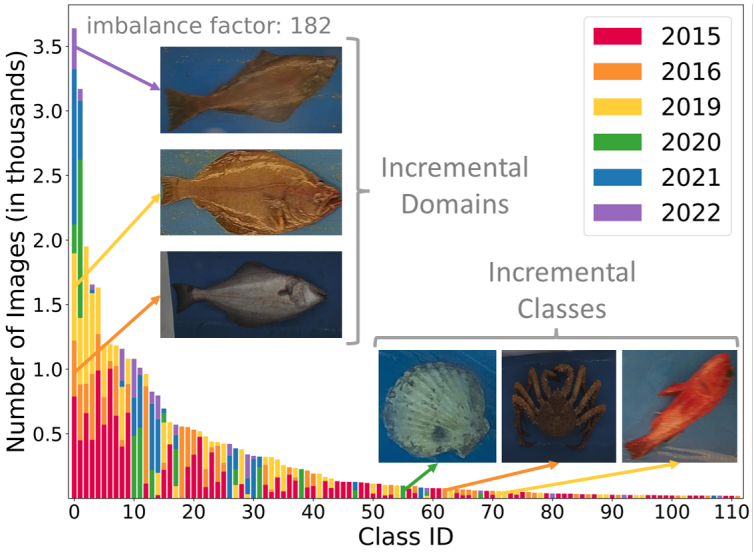


Figure 5.3: **Marine112** distribution. The number of images per class is larger than 20.

5.2.4 Class-and-Domain Incremental Learning

Marine112. It contains 112 **fine-grained** classes across 6 years. Each year, some new classes data are collected and new data of old classes brings domain shifts due to illumination, camera brands and resolutions, etc. as shown in Fig. 5.3. It is naturally suited for class and domain incremental (CDI) learning. Thus we keep $k_0 = 1$ for Eq. 5.1 when continually training ProTPS on each year's data. Marine112 also reflects the longtail [91] (imbalanced) distribution observed in nature. Details such as the test set split are in Section 5.3.2.

ProTPS learns more fine-grained text prompts than existing methods. In Tab. 5.7, ProTPS sets the strongest baseline on this fine-grained dataset. Yet, the potential future research for ProTPS may involve how to unfreeze and finetune the previously learned text prompts and vision prototypes when encountering new data of old classes in later tasks.

Table 5.8: Ablation study on CIFAR100."STC": Sampled Text Classifier strategy.

Prototype-Learning	Prompt-Selection	STC	$Loss_{pp}$	Last acc.
refined	weight-top-1	✓	✓	**83.28**
refined	top-1	✓	✓	83.12 (-0.16)
refined	weight-top-1	×	✓	82.95 (-0.33)
refined	weight-top-1	✓	×	82.87 (-0.41)
frozen	weight-top-1	✓	✓	80.65 (-2.63)
refined	weight-top-3	✓	✓	78.74 (-4.54)
refined	top-3	✓	✓	78.19 (-5.09)
refined	×	✓	✓	77.62 (-5.66)
scratch	weight-top-1	✓	✓	58.42 (-24.86)

5.2.5 Ablation Study

Tab. 5.8 shows the impact of each proposed module in ProTPS. The proposed prototype learning strategy is denoted as "refined", and it outperforms learning prototypes "from scratch" or freezing them after the proposed initialization method, referred to as "frozen". We can see our prototype learning strategy lays the foundation for the success of ProTPS's text prompt learning. The proposed text prompt selection method, denoted as "weight-top-1", brings a **significant improvement** compared to no weighting, selecting top 3 prompts, or no selection in training and inference. Both proposed $sampled\ classifier_2$ strategy and "prompt-prototype contrastive loss" also bring non-negligible improvement.

t-SNE visualization of text prompts. In Fig. 5.4, we use t-SNE [139] to display the distribution of text prompts in the feature space for 24 fine-grained dog breeds from ImageNet100. This is done after the text prompts are processed by the text encoder. We can see our text prompts are more distinct in the feature space than CLIP's text prompt templates.

Visualization of vision prototypes. To further confirm the improvement of our refined vision

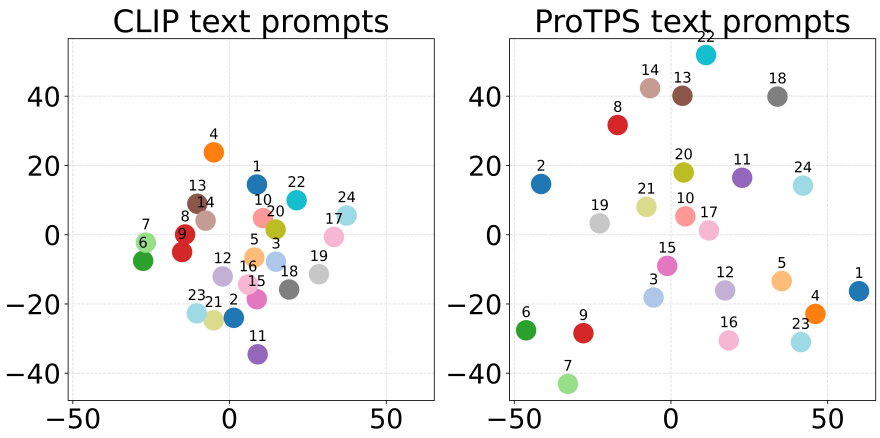


Figure 5.4: Comparison of t-SNE [139] visualization on text prompts for 24 fine-grained dog breeds from ImageNet100 [33].

prototypes compared to the initialization at the beginning of each task, we visualize the cosine similarity matrix among prototypes in Fig. 5.5. Despite freezing earlier learned prototypes in later tasks, ProTPS's vision prototypes become more distinguishable after 10 learning tasks on CIFAR100 with our training recipe, as indicated by the darker color outside the diagonal.

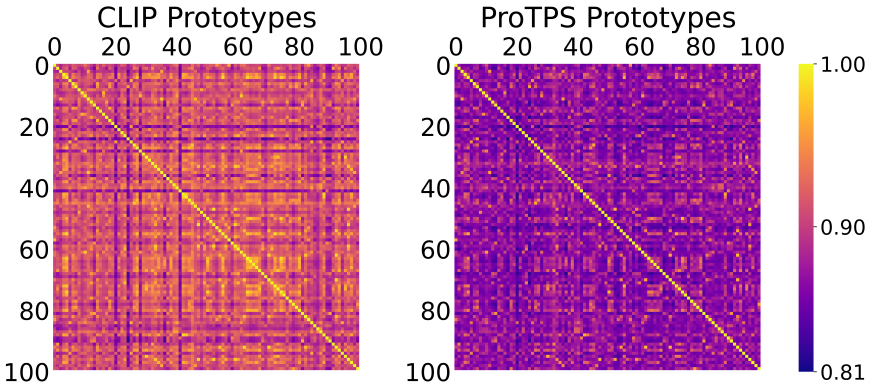


Figure 5.5: Similarity matrix of prototypes on CIFAR100 [72]. Left: Initialized prototypes via the frozen CLIP image encoder. Right: ProTPS's refined vision prototypes.

Prompt-prototype contrastive loss. We ablate our $Loss_{pp}$ and instead use SupCon [68] (de-

Table 5.9: Different loss functions on CIFAR100 [72].

Loss function	Last acc.
$Loss_{pp}$ (ours)	**83.28**
SupCon Loss [68]	82.57 (-0.71)

Table 5.10: Ablation of text selection during **inference** on ImageNet100 [33].

Prompt-Selection	Last acc.
✓	**93.92**
×	93.12 (-0.80)

tailed in Section 5.3.3.5). In Tab. 5.9, the better result is obtained with the proposed "prompt-prototype contrastive loss" ($Loss_{pp}$).

Prompt selection during inference. In Tab. 5.10, we further keep the proposed prompt selection during training but ablate it during inference, which decreases the accuracy by -0.80% on ImageNet100. This highlights the importance of our prompt selection for the inference stage as well.

Visualization of text prompts on images. To verify that our learned fine-grained text prompts can attend to region-level image details among fine-grained classes, we visualize the image contents corresponding to each class's text prompts in Fig. 5.6 on ImageNet100. The "CLIP" row shows the attention of the original CLIP's ensembled prompts on paired images. Details and more visualizations are shown in Section 5.3.4. We can see that our ProTPS's learned text prompts can focus on specific regions, such as face, neck, beak, tail, etc. among **fine-grained** classes.

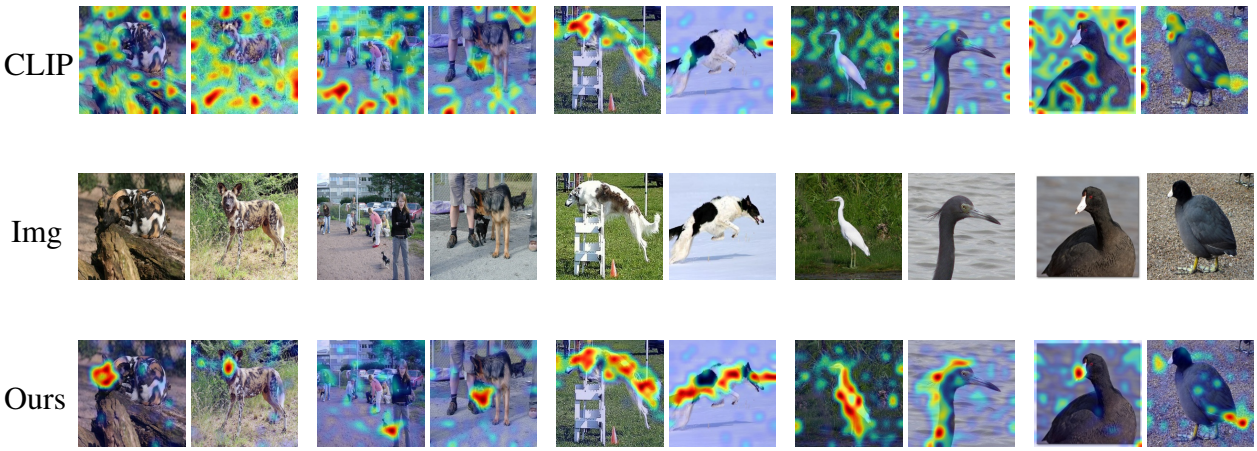


Figure 5.6: Visualization of learned prompts via Grad-CAM [124]. Highlighted image areas are critical for the model to make correct predictions given the learned prompt of each class. Each pair of adjacent columns, moving from left to right, represents the same class: “african hunting dog”, “chihuahua”, “borzoi”, “little blue heron”, “american coot”.

5.3 Supplementary Materials and Experiments

5.3.1 Classes of ImageNet100

Following AttriCLIP [149], we use the same 100 classes from ImageNet [33] to build ImageNet100. The class names are: *'Robin', 'Gila monster', 'hognose snake', 'garter snake', 'green mamba', 'garden spider', 'lorikeet', 'goose', 'rock crab', 'fiddler crab', 'American lobster', 'little blue heron', 'American coot', 'Chihuahua', 'Shih Tzu', 'Papillon', 'toy terrier', 'Walker hound', 'English foxhound', 'borzoi', 'Saluki', 'American Staffordshire Terrier', 'Chesapeake Bay Retriever', 'Vizsla', 'Kuvasz', 'Komondor', 'Rottweiler', 'Doberman', 'Boxer', 'Great Dane', 'Standard Poodle', 'Mexican hairless', 'coyote', 'African hunting dog', 'red fox', 'tabby', 'meerkat', 'dung beetle', 'walking stick', 'leafhopper', 'hare', 'wild boar', 'gibbon', 'langur', 'ambulance', 'bannister', 'bassinet', 'boathouse', 'bonnet', 'bottlecap', 'car wheel', 'chime', 'cinema', 'cocktail shaker', 'computer keyboard', 'Dutch oven', 'football helmet', 'gasmask', 'hard disc', 'harmonica', 'honeycomb', 'iron', 'jean', 'lampshade', 'laptop', 'milk can', 'mixing bowl', 'modem', 'moped', 'shower cap', 'mousetrap', 'obelisk', 'park bench', 'pedestal', 'pickup', 'pirate', 'purse', 'reel', 'rocking chair', 'rotisserie', 'safety pin', 'sarong', 'ski mask', 'slide rule', 'stretcher', 'theater curtain', 'throne', 'tile roof', 'tripod', 'tub', 'vacuum', 'window screen', 'wing', 'head cabbage', 'cauliflower', 'pineapple', 'carbonara', 'chocolate sauce', 'gyromitra', 'mushroom'*

5.3.2 Marine112 Fine-Grained Dataset

We believe our Marine112 dataset is of great importance to the continual learning community, for the following reasons: (1) it features overlapped fine-grained classes among different tasks with varied imaging qualities, i.e., domain shift; (2) it reflects the longtail [91] (imbalanced) distribution observed in nature; and (3) it is naturally suited for the class and domain incremental (CDI) learning. In Marine112, there are 112 classes across 6 different years as shown in Fig. 5.3. More electronic monitoring methods and datasets for the fishing industry can be found in related works [99, 102]

5.3.2.1 Species Names

The labeled class names are: *'arrowtooth flounder', 'rex sole', 'pacific ocean perch', 'flathead sole', 'walleye pollock', 'shortspine thornyhead', 'pacific halibut', 'northern rockfish', 'sablefish', 'pacific cod', 'petrale sole', 'starfish', 'northern rocksole', 'dungeness crab', 'english sole', 'dover sole', 'southern rock sole', 'slender sole', 'kamchatka flounder', 'atka mackerel', 'yellow irish lord', 'eulachon', 'blackspotted rockfish', 'giant grenadier', 'darkfin sculpin', 'dusky rockfish', 'big skate', 'spotted ratfish', 'rougheye rockfish', 'pacific sanddab', 'sea anemone', 'longnose skate', 'shortraker rockfish', 'harlequin rockfish', 'butter sole', 'redstripe rockfish', 'strongylocentrotus', 'sturgeon poacher', 'greenstriped rockfish', 'berryteuthis magister', 'searcher', 'sharpchin rockfish', 'yellowfin sole', 'pacific tomcod', 'podothecus accipenserinus', 'fusitriton oregonensis', 'metridium farcimen', 'bathyraja unidentified', 'actinauge verrillii', 'bairidi tanner crab', 'pacific herring', 'chum salmon', 'red banded rockfish', 'spectacled sculpin', 'silvergray rockfish', 'jellyfish', 'great sculpin', 'prowfish', 'scallop', 'lingcod', 'ctenodiscus crispatus', 'pandalopsis dispar', 'golden king crab', 'popeye grenadier', 'gorgonocephalus eucnemis', 'scissortail sculpin', 'armorhead sculpin', 'paragorgia arborea', 'darkblotched rockfish', 'chlamys', 'pacific octopus', 'longspine thornyhead', 'rosethorn rockfish', 'starry flounder', 'spinyhead sculpin', 'kelp greenling', 'mud skate', 'sawback poacher', 'yellowtail rockfish', 'pandalus', 'shrimp', 'alaska plaice', 'dark rockfish', 'bathymaster signatus', 'ebony eelpout', 'cyanea capillata', 'toad lumpsucker', 'pycnopodia helianthoides', 'northern lampfish', 'pacific sand fish', 'solaster', 'cucumaria fallax', 'eelpout unidentified', 'chrysaora melanaster', 'shortfin eelpout', 'spot shrimp', 'ceramaster', 'bigmouth sculpin', 'capelin', 'rock greenling', 'stegophiura ponderosa', 'leopard skate', 'yelloweye rockfish', 'parastichopus californicus', 'allocentrotus fragilis', 'hippasteria', 'black rockfish', 'sea pen', 'skate egg case', 'serpula', 'crossaster papposus', 'white blotched skate'*

5.3.2.2 Data Distribution

In Marine112, annual data collection includes new data for both newly discovered and previously identified classes. The latter introduces domain shifts due to illumination, pose, imaging qualities

of different types of cameras, etc, as shown in Fig. 5.3. Detailed data distribution of each year is in Fig. 5.9. The imbalance factor, calculated as the ratio of the most common class's instance count to that of the least common class, indicates the degree of class imbalance.

5.3.2.3 Testing Data Split

As mentioned in the previous section, new data of previously seen classes usually bring domain shift. Therefore, each year, no matter the new classes or old classes, we always randomly hold 10 new images from each collected class as testing data. And the accuracy of each year $t, t \in \{2015, 2016, \ldots, 2022\}$ is evaluated on testing images from all the trained years so far (i.e., year 2015, ..., t). Fig. 5.10 shows the number of kept test data from each year.

Justification. There is a real-world application scenario for the above test data split protocol. Initially, in 2015, thousands of images were collected via electronic monitoring from different commercial fishing boats to ensure that fish populations in targeted areas were sustainably managed. The identification of marine species from images is an expensive time-consuming process that requires the consulting of fishery specialists. Therefore, deep learning models serve as valuable tools to efficiently process such data. To train a deep learning model, we ask fishery specialists to label a portion of the collected data. Our model then is expected to identify remaining unlabeled images, reducing the labor cost required to gain an accurate understanding of the species and numbers caught. Our goal is to train a model continually with a subset of each year's data, manually labeled, and apply it to unlabelled data of previous years and the current year. Thus, in our testing protocol, once we train the model on the 2015 labeled training data, we test it on the held 2015 testing data. Subsequently, we continue to train the model on 2016 training data, and then test the model on both 2015 and 2016 testing data. Therefore, our testing split protocol can reflect the performance of the continually trained model on **accumulated** unidentified data from incrementally exposed domains and classes.

5.3.2.4 *Datasets Comparison*

Tab. 5.11 compares the collected fine-grained Marine112 dataset against several public datasets utilized for downstream continual learning. Marine112's distinction lies in its authentic real-world data: it reflects the longtail distribution observed in nature, features varied imaging qualities, i.e., domain shifts, caused by diverse cameras across different years and ships, and is inherently apt for the class and domain incremental (CDI) learning task.

Table 5.11: Comparison of Marine112 with public datasets with domain shift. "DI": Domain incremental learning. "CDI": Class and domain incremental learning.

Dataset	Setting	Distribution	#Classes	Domains	Fine-grained
Rotation MNIST [93]	DI	balanced	10	synthetic	×
Permutation MNIST [69]	DI	balanced	10	synthetic	×
NS-MiniImageNet [94]	DI	balanced	100	synthetic	×
CLEAR [86]	DI	balanced	100	real-world	×
DS-ImageNet-R [129]	CDI[1]	balanced	200	real-world	×
CORe50-NIC [92]	CDI[2]	balanced	50	real-world	×
Marine112 (Ours)	CDI	longtail	112	real-world	✓

5.3.3 *Implementation Details*

5.3.3.1 *Training Details*

We train our ProTPS for 10 epochs on each task for all datasets. SGD is adopted as the optimizer with an initial learning rate of 0.001 and follows a cosine decay schedule to a final learning rate of

[1]DS-ImageNet-R introduces domain shifts that occur between classes, rather than within each class, as reflected in our training data.

[2]CORe50-NIC aims to evaluate the model's generalization to unseen domains, rather than its performance on domains learned incrementally, as demonstrated by our test protocol.

1e-4. The weight decay is 0, and the batch size is 128 on two V100 GPUs. Each text prompt P_i is a learnable embedding with the dimension of [M, 768] where M is the prompt length and 768 is the feature dimension output from the last projection layer of CLIP's image encoder. We observe that the last accuracy is not sensitive to the loss factor λ_{pp} and prompt length M indicated in Fig. 5.7 thus we use λ_{pp}=1.5 and M = 6 for our ProTPS. The average results over 3 runs are reported for all methods in the paper.

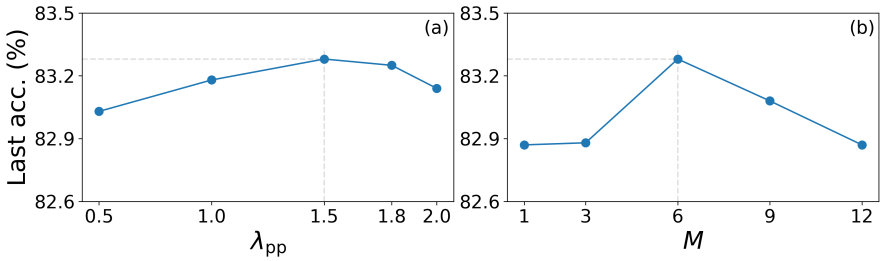


Figure 5.7: Comparison of different (a) loss factor λ_{pp} values and different (b) prompt length M values on CIFAR100 [72].

Other baselines details. For baseline methods, we use their own official released code and follow their own training recipes. CoOp [180] is the only method in all tables storing 1000 data samples during training thus denoted as "CoOp [180] (1000)" in all tables. The prompts of CoOp [180] and AttriCLIP [149] are continually trained in a sequence of tasks. For "Zero-Shot CLIP", "Prototypes-CLIP", and "Linear Probe CLIP", we implement them based on the official CLIP's open-source code. All reported CLIP-based methods use the same original CLIP pre-trained transformer [141] as the text encoder. All reported methods use the original CLIP pre-trained ViT-L/14 [37] as the image encoder.

Why use CLIP pre-trained image encoder for all reported methods? Firstly, our ProTPS is based on CLIP architecture similar to existing text-prompt-based methods [133, 149, 151, 180] which use frozen CLIP pre-trained image encoder and text encoder. Secondly, as mentioned in DualPrompt [153], when visual-prompt-based methods [129,147,153,154] use the image encoder pre-trained on ImageNet, they struggle with fairness when evaluated on the ImageNet-related bench-

marks such as ImageNet100 and ImageNet-R [153]. Thus we use CLIP pre-trained ViT-L/14 [37] as the image encoder for all methods.

Classifiers aggregation methods. Tab. 5.12 shows aggregating ProTPS's dual classifiers via the proposed average operation is better than the max operation.

Table 5.12: Different aggregation methods on ImageNet100.

Aggregation	Last acc.
Average	**93.92**
Max	93.86 (-0.06)

5.3.3.2 Evaluation Metric Details

As emphasized by previous works [129, 153], "Average Accuracy" is the overall evaluation metric for continual learning, which includes two aspects: greater learning capacity and less catastrophic forgetting.

However, real-world continual learning datasets may be imbalanced, Marine112. Thus, we define the *generalized* average accuracy to evaluate continual learning models. Consider a series of T tasks $D = \{D_1, ..., D_T\}$. Each task has its own training data, $D_{t,train} = \{(x^i_{t,train}, y^i_{t,train})\}_{i=1}^{n_{t,train}}$ comprising $n_{t,train}$ instances $x^i_{t,train}$ and their respective labels $y^i_{t,train}$, and testing data, $D_{t,test} = \{(x^i_{t,test}, y^i_{t,test})\}_{i=1}^{n_{t,test}}$ comprising $n_{t,test}$ instances $x^i_{t,test}$ and their respective labels $y^i_{t,test}$. After the model finishes training on the task t, we compute *generalized* average accuracy A_t as follows:

$$A_t = \frac{\sum_{m=1}^{t}\sum_{i=1}^{n_{m,test}} equal(model(x^i_{m,test}), y^i_{m,test})}{\sum_{m=1}^{t} n_{m,test}}, \tag{5.5}$$

where the numerator is the number of true predictions on testing data from all exposed tasks (i.e., task $1, 2...t$) and the denominator is the total number of involved testing data. When testing data from each task is balanced, A_t is exactly the widely-used "Average Accuracy" well-defined in

previous works such as [153] (see its Appendix C). Similarly, we refer to the final task's A_T as the "Last Accuracy".

5.3.3.3 More Results on Class Incremental Learning

Table 5.13: Last accuracy on CIFAR100 [72] and ImageNet100 [33] under CI setting.

Prompt	Method	CIFAR100	ImageNet100
-	SLCA [174]	79.1	56.3
visual	L2P [154]	49.4	48.3
	DualPrompt [153]	54.0	52.6
	APG [131]	54.2	53.1
	CODA-P [129]	72.3	75.2
	HiDe-Prompt [147]	81.0	82.9
text	CoOp [180] (1000)	67.6	79.3
	Continual-CLIP [133]	73.4	75.4
	Zero-Shot CLIP [111]	78.3	75.1
	AttriCLIP [149]	81.4	83.3
*text**	**ProTPS** (ours)	**83.3** (+1.9)	**93.9** (+10.6)

APG [131] is also a visual-prompt-based method that demonstrates its effectiveness when training from scratch and has comparable performance to other existing visual-prompt-based methods when using the same pre-trained image encoder. Since APG [131] does not achieve the best performance among visual-prompt-based methods, we report its performance in this section with the same CLIP pre-trained ViT-L/14 [37] as the image encoder in Tab. 5.13.

Besides, SLCA [174] is also built upon the pre-trained image encoder. But SLCA [174] is a non-prompt-based method that carefully finetunes the pre-trained image encoder with a smaller learning rate than in the classifier head. It models class-wise distributions in the form of a feature

covariance matrix for every class and generates pseudo-training samples (features) of seen classes to align the classifier heads between consecutive tasks. Since SLCA [174] is not a prompt-based method, we report its performance in this section with the same CLIP pre-trained ViT-L/14 [37] as the image encoder in Tab. 5.13. $text^*$ denotes that the learning and selection of text prompts are instructed by vision prototypes in our ProTPS.

Table 5.14: Last accuracy of different methods on CIFAR100 [72] under CDC setting. The models are either trained on CIFAR100 only (C), or continually fine-tuned on CIFAR100 after being trained on ImageNet100 (I2C).

Prompt	Method	C	I2C	FT↑
$v+t$	S-liPrompts [151]	58.9	53.3	-5.6
$visual$	L2P-1 [154]	49.4	45.7	-3.7
	DualPrompt-1 [153]	54.0	49.5	-4.5
	CODA-P-1 [129]	72.3	66.9	-5.4
	HiDe-Prompt-1 [147]	81.0	77.2	-3.8
$text$	CoOp-1 [180] (1000)	67.6	61.1	-6.5
	Continual-CLIP [133]	73.4	73.4	0
	Zero-Shot CLIP [111]	78.3	78.3	0
	AttriCLIP [149]	<u>81.4</u>	<u>82.3</u>	<u>+0.9</u>
$text^*$	**ProTPS** (ours)	**83.3**	**84.4** (+2.1)	**+1.1**

5.3.3.4 More Results on Cross-Datasets Continual Learning

S-liPrompts [151] is a CLIP-based method, that uses both visual prompts and text prompts and derives the classifier via the text encoder. Yet, it is designed for domain incremental learning. By treating each dataset as a domain, we also test S-liPrompts [151] on the Cross-Datasets Continual Learning setting. The results are in Tab. 5.14 and Tab. 5.15. $text^*$ denotes the learning and selection

Table 5.15: Last accuracy of different methods on ImageNet100 + CIFAR100 (I+C) where each model is continually trained on ImageNet100 and CIFAR100 in sequence under CDC setting.

Prompt	Method	I+C
$v+t$	S-liPrompts [151]	39.6
$visual$	L2P-2 [154]	41.6
	DualPrompt-2 [153]	45.8
	CODA-P-2 [129]	62.7
	HiDe-Prompt-2 [147]	75.3
$text$	CoOp-2 [180] (1000)	55.4
	Continual-CLIP [133]	65.9
	Zero-Shot CLIP [111]	68.7
	AttriCLIP [149]	<u>78.3</u>
$text^*$	**ProTPS** (ours)	**85.6** (+7.3)

of text prompts are instructed by vision prototypes in our ProTPS. "$v+t$" denotes both visual and text prompts.

5.3.3.5 SupCon Loss Details

In Tab. 5.9, we compare our proposed "prompt-prototype contrastive loss" and SupCon [68] which is originally proposed to learn good encoders, i.e., representation learning. SupCon [68] loss is defined within a batch of images as:

$$\mathcal{L}_{out}^{sup} = \sum_{i \in I} -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{a \in A(i)} \exp(z_i \cdot z_a / \tau)}, \tag{5.6}$$

where i is the image index in a batch, $P(i)$ is the set of indices of all positives of ith image in this batch distinct from i. $A(i)$ is the set of indices of all images in this batch except i, z_i represents the image feature of ith image, "$\cdot$" denotes the inner product, and τ is a scalar temperature parameter.

We adopt it with the following equation to replace our proposed "prompt-prototype contrastive loss" and report the performance in Tab. 5.9. Given a single image input of class i, there is only one positive pair between our vision prototypes and weight vectors of $classifier_2$ denoted as the yellow grid in the matrix shown in Fig. 5.1, thus the adopted SupCon loss is as follows:

$$\mathcal{L} = -\log \frac{\exp(I_i \cdot T_i/\tau)}{\sum_{j \neq i \text{ or } k \neq i} \exp(I_j \cdot T_k/\tau)}, \tag{5.7}$$

where the numerator is the positive pair and the denominator includes all negative pairs between our vision prototypes and weight vectors of $classifier_2$ in our ProTPS.

5.3.3.6 ProTPS's Upper Bound Accuracy

We further test our ProTPS's upper bound accuracy by using all tasks' training data at the same time, the same as standard supervised training. The performance is reported in Tab. 5.16. We can see that ProTPS can achieve better performance than Linear Probe CLIP which trains a linear classifier based on the CLIP image encoder.

Table 5.16: Upper bound accuracy of 'Linear Probe CLIP' [111] and our ProTPS on CIFAR100 [72] and ImageNet100 [33].

Method	CIFAR100	ImageNet100
Linear Probe CLIP [111]	86.69	95.52
ProTPS (ours)	**87.19**	**95.86**

5.3.3.7 Ablation on ProTPS's prototypes training strategy

The proposed prototype learning strategy is denoted as "refined" in Fig. 5.8. We can see it is critical to help dual classifiers' alignment and achieves the best performance over training from "scratch" or "freezing" them after the proposed initialization method.

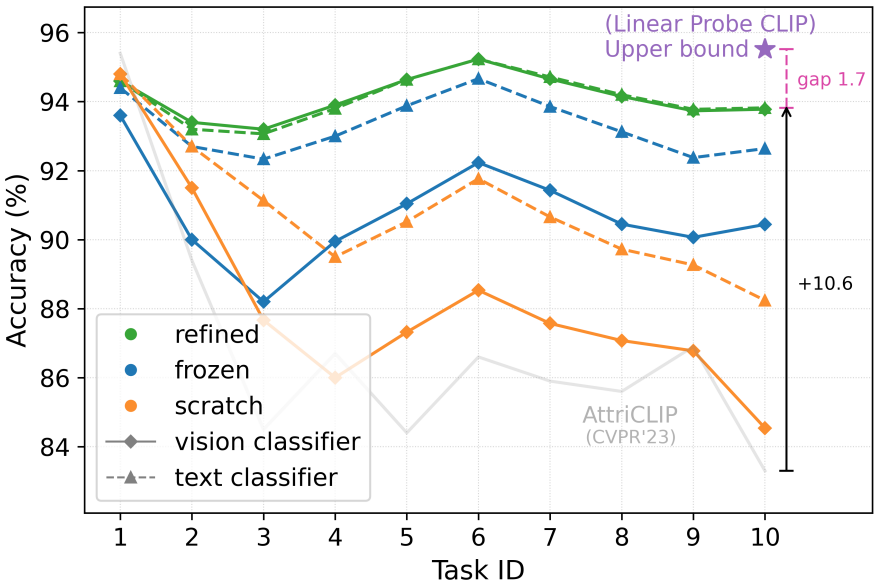


Figure 5.8: Ablation study of ProTPS's on ImageNet100 [33] under CI setting. The "vision classifier" and "text classifier" of ProTPS are well-aligned when using the proposed prototype learning strategy.

Table 5.17: **Full Pilot Study.** Comparison of different CLIP-based methods in CI setting on CIFAR100 [72] with ViT-B/16 [37] (first 3 rows) or ViT-L/14 [37] (second 3 rows) backbone. After learning each task $T-i$, the accuracy on exposed classes from all the trained tasks (i.e., $T-1, 2, ..., t$) is reported. The upper-bound method is 'Linear Probe CLIP' [111] using all tasks' training data at the same time to optimize a linear classifier. The column 'Classifier' denotes the encoder from which the classifier weights are derived.

Method	Classifier	T-1	T-2	T-3	T-4	T-5	T-6	T-7	T-8	T-9	T-10
Continual-CLIP [133]	text	94.7	88.5	76.8	73.6	72.4	72.8	69.6	68.5	68.1	66.7
Zero-Shot CLIP [111]	text	95.6	89.2	77.7	74.6	73.9	74.1	70.6	70.1	69.6	68.3
Prototypes-CLIP	vision	**95.6**	**90.7**	**81.5**	**77.1**	**75.9**	**75.7**	**72.8**	**72.3**	**71.4**	**69.9**
Continual-CLIP [133]	text	95.3	92.3	82.1	79.2	78.5	77.9	75.7	75.2	74.8	73.4
Zero-Shot CLIP [111]	text	95.7	**92.9**	85.2	82.9	82.3	82.1	79.6	79.6	79.2	78.3
Prototypes-CLIP	vision	**97.6**	92.8	**88.1**	**85.3**	**84.3**	**84.3**	**81.9**	**81.9**	**81.2**	**80.0**
Upper-bound (B/16)	-	-	-	-	-	-	-	-	-	-	82.5
Upper-bound (L/14)	-	-	-	-	-	-	-	-	-	-	86.7

5.3.3.8 Full Pilot Study

Inspiration. The full pilot study on CIFAR100 is done with ViT-L/14 [37] and ViT-B/16 [37] backbones in Tab. 5.17. Zero-shot CLIP's 18 prompt templates for CIFAR100 can be found in its official link. On both backbones, we consistently find that applying the basic nearest-prototype classifier to CLIP, denoted as Prototypes-CLIP, achieves slightly better performance than text prompt engineering and ensembling. This pilot study motivates us to use prototypes as guidance for our text prompt selection.

5.3.4 More Text Prompts Visualizations

The "CLIP" column in Fig. 5.6 shows the attention of the original CLIP's prompts on paired images on ImageNet100. The original CLIP's 80 prompt templates for ImageNet can be found in CLIP's official link. The way to create "zero-shot classifier weights" for each class is also introduced in that link. For each class, we use the corresponding "zero-shot classifier weights" for visualization of the CLIP column in Fig. 5.6. More visualizations of text prompts from randomly selected classes are shown in Fig. 5.11. We can see that our learned fine-grained text prompts can attend to region-level image details.

5.4 Summary

We propose ProTPS, which continually learns fine-grained text prompts to mitigate "catastrophic forgetting". Specifically, ProTPS leverages global image-level vision prototypes to guide the selection and learning of semantic features encoded in text prompts. To align ProTPS's prompts and prototypes, we further introduced the "prompt-prototype contrastive loss". Our experiments demonstrate superior performance over existing state-of-the-art methods in CI, CDC, and CDI learning settings. Additionally, we introduced Marine112, a real-world dataset containing 112 fine-grained classes across 6 years and naturally suitable for CDI setting, which brings new challenges to the community. We believe our ProTPS and Marine112 can pave the way for more practical continual learning.

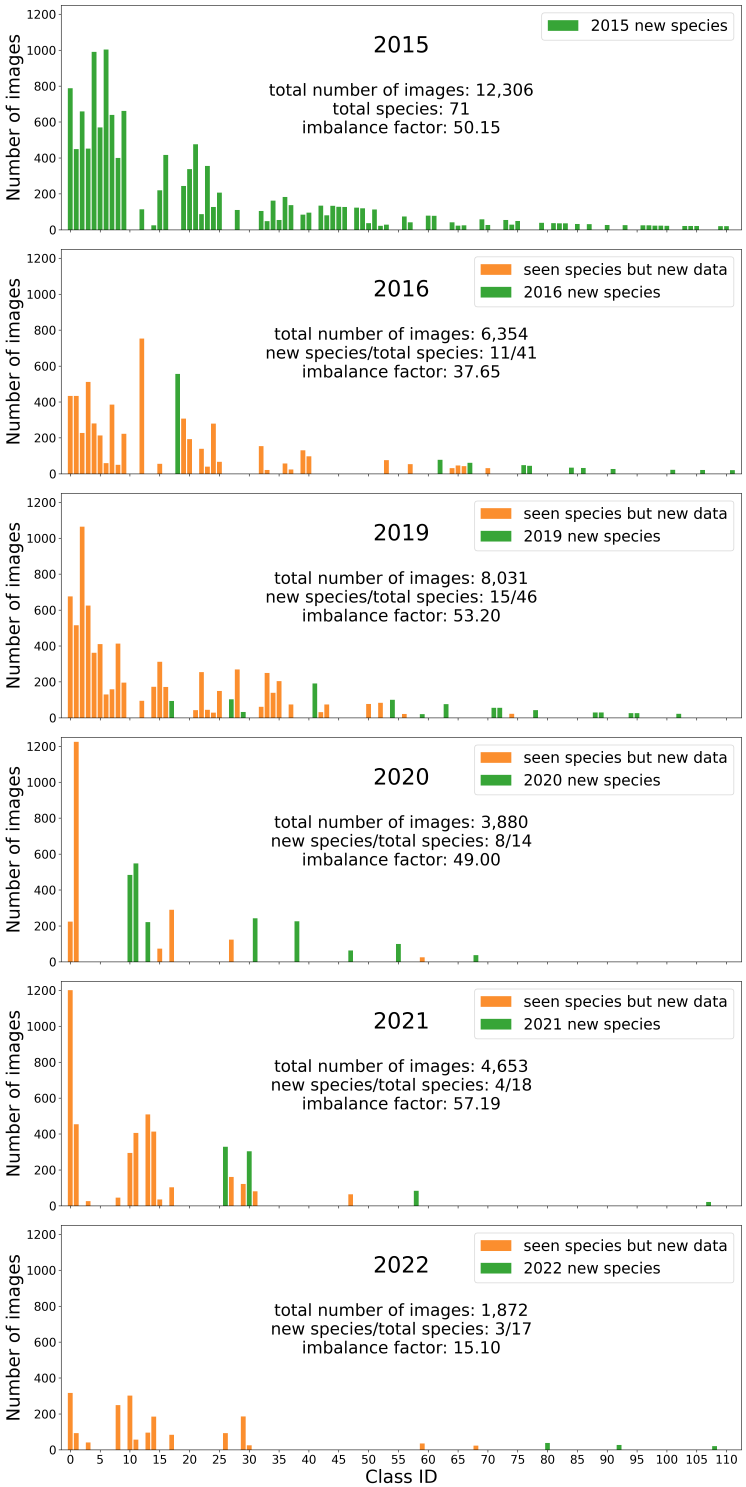


Figure 5.9: **Marine112.** Data distribution in each year.

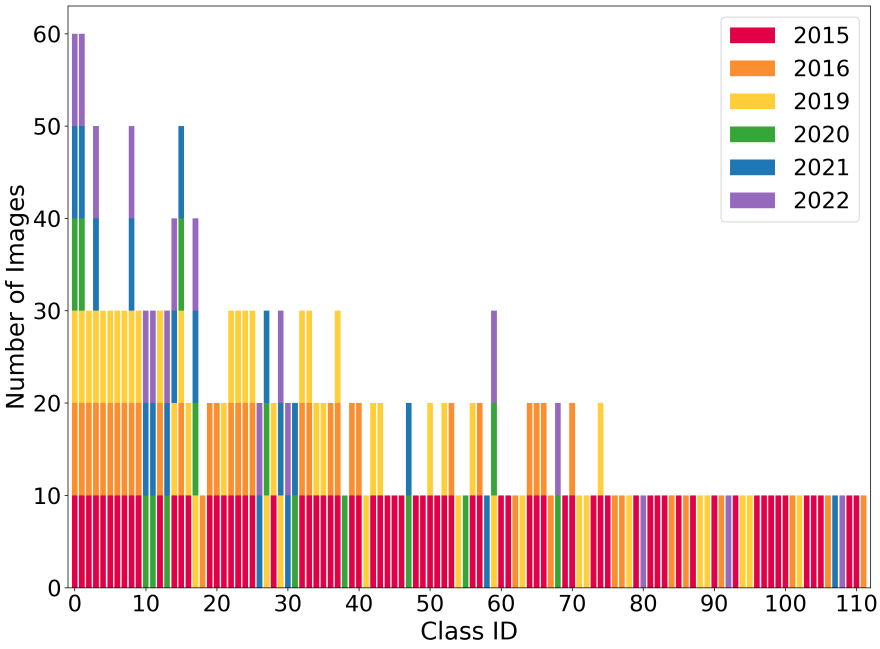


Figure 5.10: **Marine112.** Number of kept test data from each year.

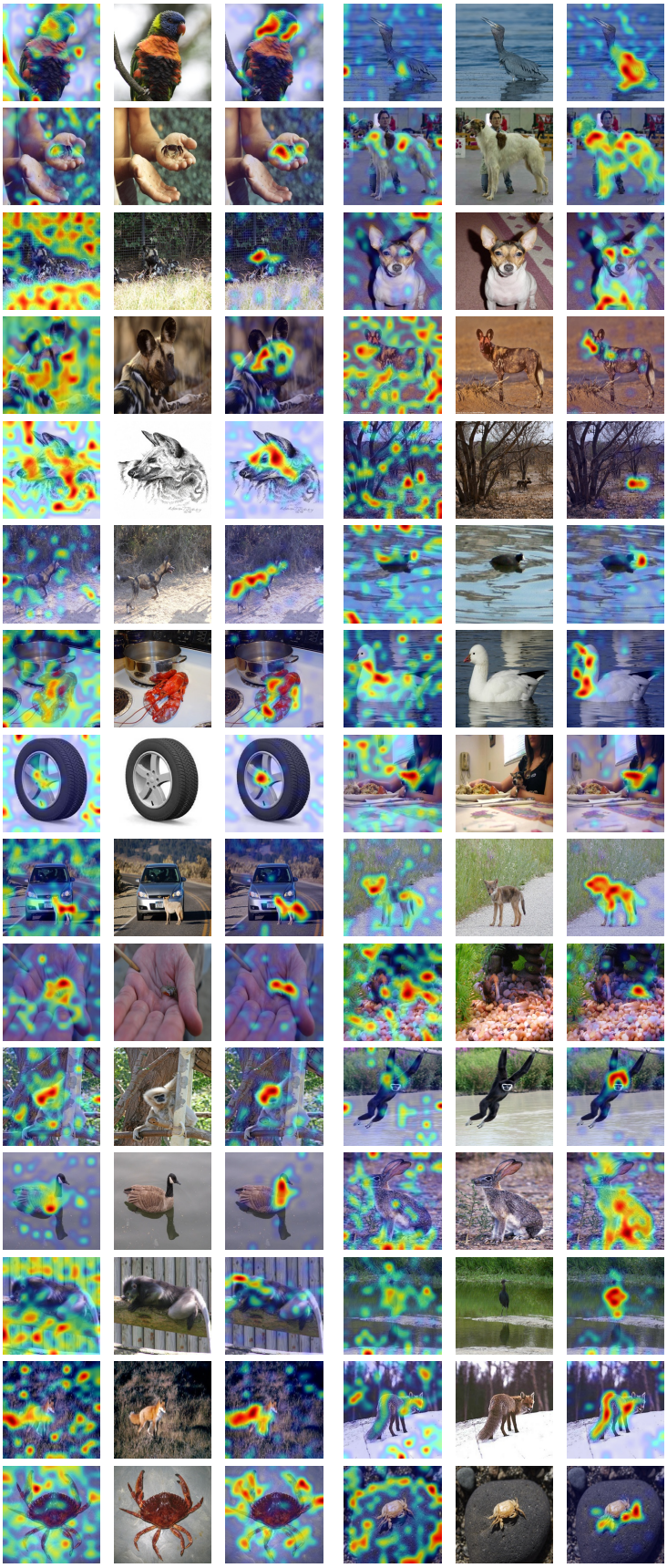


Figure 5.11: Text prompts visualized via Grad-CAM [124] on randomly selected classes of ImageNet100. The 1st and 4th columns are from CLIP. The 2nd and 5th columns are original images. The 3rd and 6th columns are from our ProTPS method.

Chapter 6

CONCLUSIONS

In this dissertation, we have explored the challenges and opportunities in continual learning of object classification in the real world. We have proposed three novel frameworks, i.e., HCIL, ESA, and ProTPS, to address the catastrophic forgetting problem under different real-world scenarios. These frameworks represent significant advancements toward achieving continual learning of object classification in real-world scenarios.

The proposed HCIL is a hierarchical class incremental learning model, which can jointly perform the hierarchical classification task and class incremental learning without catastrophically forgetting previously trained classes. The hierarchical classifier allows coarse-level prediction to be the final output if the fine-level confidence score is too low and achieves higher accuracy on tail-class species when training data follows a long-tail (imbalanced) distribution than a flat classifier.

The proposed ESA is an expert-and-samples-aware incremental learning framework for continual learning under long-tailed distribution, with the multi-expert idea and a dynamic weighting technique to deal with the exacerbated forgetting introduced by the long-tail distribution. ESA gathers 'opinions' from different light-weight experts before making the final prediction. The dynamic weighting in ESA is designed for the CIL-LT, where (1) different tasks can share some classes but with new data samples, and (2) the training data of each task follows a long-tail distribution with the imbalance factor in a wide range.

The proposed ProTPS is a multimodal transformer-based framework that continually learns fine-grained text prompts so that the textual features of newly arrived classes do not overlap with those of trained classes, thereby mitigating catastrophic forgetting. ProTPS leverages global image-level vision prototypes to guide the selection and learning of semantic features encoded in text prompts. To align ProTPS's prompts and prototypes, we further introduced the "prompt-prototype

contrastive loss".

From the dataset perspective, we collect a new hierarchical dataset for continual learning, three under-studied CIL-LT datasets, and a real-world marine species dataset, named Marine112, to bring new challenges to the continual learning community. Marine112 is a fine-grained dataset with long-tail distribution and is naturally suited for the class and domain incremental (CDI) learning setting.

The proposed and collected datasets present novel real-world challenges to the continual learning community, and the proposed frameworks represent significant advancements toward achieving continual learning of object classification in real-world scenarios as demonstrated by extensive experiments and ablation studies. Future work could include exploring the potential of self-supervised and unsupervised learning techniques for continual learning, and investigating the possibility of applying similar ideas to other tasks, e.g. object detection, under continual learning scenarios.

BIBLIOGRAPHY

[1] 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)*, pages 139–154, 2018. 2.1.1

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

[3] FirstName Alpher. Frobnication. *IEEE TPAMI*, 12(1):234–778, 2002.

[4] FirstName Alpher and FirstName Fotheringham-Smythe. Frobnication revisited. *Journal of Foo*, 13(1):234–778, 2003.

[5] FirstName Alpher, FirstName Fotheringham-Smythe, and FirstName Gamow. Can a machine frobnicate? *Journal of Foo*, 14(1):234–778, 2004.

[6] FirstName Alpher and FirstName Gamow. Can a computer frobnicate? In *CVPR*, pages 234–778, 2005.

[7] Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6897–6907, 2022.

[8] Anonymous. The frobnicatable foo filter, 2024. ECCV submission ID 00324, supplied as supplemental material `00324.pdf`.

[9] Anonymous. Frobnication tutorial, 2024. Supplied as supplemental material `tr.pdf`.

[10] Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. *arXiv preprint arXiv:2201.12604*, 2022. 1.3, 4.1, 4.2.3, 4.2.4, 4.3

[11] Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors. *Computer Vision – ECCV 2022*. Springer, 2022.

[12] 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*, pages 8218–8227, 2021. 1.3, 4.1, 4.2.2, 4.2.3, 4.2, 4.3

[13] Mathieu Bouville. Crime and punishment in scientific research, 2008.

[14] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018. 1.3

[15] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 4.1, 4.2.3

[16] Jiarui Cai, Yizhou Wang, and Jenq-Neng Hwang. Ace: Ally complementary experts for solving long-tailed recognition in one-shot. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 112–121, 2021. 1.3, 2.2, 4.1.2

[17] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019. 2.2

[18] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018. 1.1, 2.1.1, 2.5.1

[19] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9233–9242, 2020. 2.1.1

[20] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018. 2.1.1, 4.1, 4.2.3, 4.3

[21] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 2.1.1

[22] 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. 1.1, 2.1.1

[23] Meng-Che Chuang, Jenq-Neng Hwang, and Craig S Rose. Aggregated segmentation of fish from conveyor belt videos. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1807–1811. IEEE, 2013. 1.2

[24] Meng-Che Chuang, Jenq-Neng Hwang, Kresimir Williams, and Richard Towler. Tracking live fish from low-contrast and low-frame-rate stereo videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(1):167–179, 2014. 1.2

[25] William J. Clancey. *Transfer of Rule-Based Expertise through a Tutorial Dialogue*. Ph.D. diss., Dept. of Computer Science, Stanford Univ., Stanford, Calif., 1979.

[26] William J. Clancey. Communication, Simulation, and Intelligent Agents: Implications of Personal Intelligent Machines for Medical Education. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83)*, pages 556–560, Menlo Park, Calif, 1983. IJCAI Organization.

[27] William J. Clancey. Classification Problem Solving. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, pages 45–54, Menlo Park, Calif., 1984. AAAI Press.

[28] William J. Clancey. The Engineering of Qualitative Models. Forthcoming, 2021.

[29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[30] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019. 4.2.3

[31] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019. 2.2

[32] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang. Dynamic head: Unifying object detection heads with attentions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7373–7382, 2021.

[33] 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*, pages 248–255. Ieee, 2009. (document), 1.1, 2.3, 5.2.1, 5.2, 5.3, 5.4, 5.5, 5.4, 5.10, 5.3.1, 5.13, 5.16, 5.8

[34] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012. 4.2

[35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2.3

[36] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5138–5146, 2019. 2.1.1

[37] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. (document), 2.3, 5.1, 5.2.1, 5.3.3.1, 5.3.3.3, 5.17, 5.3.3.8

[38] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Tackling catastrophic forgetting and background shift in continual semantic segmentation. *arXiv preprint arXiv:2106.15287*, 2021. 2.1.1

[39] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European Conference on Computer Vision*, pages 86–102. Springer, 2020. 1.1, 2.1.1, 2.5.4

[40] Arthur Douillard and Timothée Lesort. Continuum: Simple management of complex continual learning scenarios. *arXiv preprint arXiv:2102.06253*, 2021.

[41] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. *arXiv preprint arXiv:2111.11326*, 2021. 1.1, 2.1.1, 2.1.2, 2.5.1, 4.1.1

[42] Robert Engelmore and Anthony Morgan, editors. *Blackboard Systems*. Addison-Wesley, Reading, Mass., 1986.

[43] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017. 1.1, 2.1.2, 2.5.1

[44] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[45] Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. *arXiv preprint arXiv:1903.04476*, 2019. 1.1, 2.1.2, 2.5.1

[46] Shilpi Gupta, Nissreen Abu-Ghannam, Roberto Massini, Yaseen Mottiar, Illimar Altosaar, Micha Peleg, Mark D Normand, and Maria G Corradini. Trends in application of imaging technologies to inspection of fish and. 1.2

[47] Diane Warner Hasling, William J. Clancey, and Glenn Rennels. Strategic explanations for a diagnostic consultation system. *International Journal of Man-Machine Studies*, 20(1):3–19, 1984.

[48] Diane Warner Hasling, William J. Clancey, Glenn R. Rennels, and Thomas Test. Strategic Explanations in Consultation—Duplicate. *The International Journal of Man-Machine Studies*, 20(1):3–19, 1983.

[49] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pages 466–483. Springer, 2020. 1.1, 2.1.1

[50] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 3.1.2

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2.4.1, 3.4.2

[52] Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International conference on machine learning*, pages 4182–4192. PMLR, 2020. 2.5.5

[53] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 1.1

[54] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 2.5.5

[55] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 1.1, 2.1.1, 2.5.1, 2.5.4

[56] Zhiyuan Hu, Yunsheng Li, Jiancheng Lyu, Dashan Gao, and Nuno Vasconcelos. Dense network expansion for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11858–11867, 2023.

[57] Phoenix X Huang, Bastiaan J Boom, and Robert B Fisher. Hierarchical classification with reject option for live fish recognition. *Machine Vision and Applications*, 26(1):89–102, 2015. 2.4.2

[58] Tsung-Wei Huang, Jenq-Neng Hwang, Suzanne Romain, and Farron Wallace. Live tracking of rail-based fish catching on wild sea surface. In *2016 ICPR 2nd Workshop on Computer Vision for Analysis of Underwater Imagery (CVAUI)*, pages 25–30. IEEE, 2016. 1.2

[59] Tsung-Wei Huang, Jenq-Neng Hwang, and Craig S Rose. Chute based automated fish length measurement and water drop detection. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1906–1910. IEEE, 2016. 1.2

[60] 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. *Advances in Neural Information Processing Systems*, 32, 2019. 1.1, 2.1.2, 2.5.1

[61] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *European Conference on Computer Vision*, pages 699–715. Springer, 2020. 1.1, 2.1.1

[62] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002. 2.2

[63] Ying Jin, Yinpeng Chen, Lijuan Wang, Jianfeng Wang, Pei Yu, Lin Liang, Jenq-Neng Hwang, and Zicheng Liu. Decoupling object detection from human-object interaction recognition. *arXiv preprint arXiv:2112.06392*, 2021. 2.3

[64] C.D. Jones, A.B. Smith, and E.F. Roberts. Article title. In *Proceedings Title*, volume II, pages 803–806. IEEE, 2003.

[65] Dahuin Jung, Dongyoon Han, Jihwan Bang, and Hwanjun Song. Generating instance-level prompts for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11847–11857, 2023. 1.4

[66] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[67] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*, 2017. 2.1.1

[68] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020. 2.5.5, 5.1.3, 5.2.5, 5.9, 5.3.3.5

[69] 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*, 114(13):3521–3526, 2017. 1.1, 1.3, 1.4, 2.1.1, 4.1, 4.2.3, 4.3, 5.11

[70] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE, 2017. 2.4.2

[71] Kamran Kowsari, Rasoul Sali, Lubaina Ehsan, William Adorno, Asad Ali, Sean Moore, Beatrice Amadi, Paul Kelly, Sana Syed, and Donald Brown. Hmic: Hierarchical medical image classification, a deep learning approach. *Information*, 11(6):318, 2020. 2.4.2

[72] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. (document), 4.2, 5.1, 5.1.1, 5.2, 5.4, 5.5, 5.5, 5.9, 5.7, 5.13, 5.14, 5.16, 5.17

[73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2.4.1

[74] Jogendra Nath Kundu, Rahul Mysore Venkatesh, Naveen Venkat, Ambareesh Revanur, and R Venkatesh Babu. Class-incremental domain adaptation. In *European Conference on Computer Vision*, pages 53–69. Springer, 2020.

[75] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning*, pages 5905–5914. PMLR, 2021.

[76] FirstName LastName. The frobnicatable foo filter, 2014. Face and Gesture submission ID 324. Supplied as supplemental material `fg324.pdf`.

[77] FirstName LastName. Frobnication tutorial, 2014. Supplied as supplemental material `tr.pdf`.

[78] Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019. 2.1.1

[79] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. *arXiv preprint arXiv:2112.03857*, 2021. 2.3

[80] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934. PMLR, 2019. 1.1, 2.1.2, 2.5.1

[81] Yujie Li, Xin Yang, Hao Wang, Xiangkun Wang, and Tianrui Li. Learning to prompt knowledge transfer for open-world continual learning. *arXiv preprint arXiv:2312.14990*, 2023. 2.5.3

[82] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1.4, 2.1.1

[83] Zhuoyun Li, Changhong Zhong, Sijia Liu, Ruixuan Wang, and Wei-Shi Zheng. Preserving earlier knowledge in continual learning with the help of all previous feature extractors. *arXiv preprint arXiv:2104.13614*, 2021. 1.1

[84] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2.2

[85] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[86] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. 5.11

[87] Wenzhuo Liu, Xinjian Wu, Fei Zhu, Mingming Yu, Chuang Wang, and Cheng-Lin Liu. Class incremental learning with self-supervised pre-training and prototype learning. *arXiv preprint arXiv:2308.02346*, 2023.

[88] Xialei Liu, Yu-Song Hu, Xu-Sheng Cao, Andrew D Bagdanov, Ke Li, and Ming-Ming Cheng. Long-tailed class incremental learning. In *European Conference on Computer Vision*, pages 495–512. Springer, 2022. 1.3

[89] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2.3, 3.4.2

[90] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2.3

[91] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2537–2546, 2019. 1.4, 5.2.1, 5.2.4, 5.3.2

[92] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on robot learning*, pages 17–26. PMLR, 2017. 5.11

[93] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017. 2.1.1, 5.2.1, 5.11

[94] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022. 5.11

[95] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[96] Mark D McDonnell, Dong Gong, Amin Parveneh, Ehsan Abbasnejad, and Anton van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. *arXiv preprint arXiv:2307.02251*, 2023.

[97] J. Mei, Jenq-Neng Hwang, S. Romain, Craig S. Rose, Braden Moore, and Kelsey Magrane. Video-based hierarchical species classification for longline fishing monitoring. In *ICPR Workshops*, 2020. 3.2, 3.3, 3.4.2

[98] Jie Mei and Jenq-Neng Hwang. Esa: Expert-and-samples-aware incremental learning under longtail distribution. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5740–5744. IEEE, 2024. (document)

[99] Jie Mei, Jenq-Neng Hwang, Suzanne Romain, Craig Rose, Braden Moore, and Kelsey Magrane. Absolute 3d pose estimation and length measurement of severely deformed fish from monocular videos in longline fishing. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2175–2179. IEEE, 2021. 5.3.2

[100] Jie Mei, Jenq-Neng Hwang, Suzanne Romain, Craig Rose, Braden Moore, and Kelsey Magrane. Video-based hierarchical species classification for longline fishing monitoring. In *International Conference on Pattern Recognition*, pages 422–433. Springer, 2021. (document), 3.3

[101] Jie Mei, Suzanne Romain, Craig Rose, Kelsey Magrane, and Jenq-Neng Hwang. Hcil: Hierarchical class incremental learning for longline fishing visual monitoring, 2022. (document), 1.1, 1.3, 1.4

[102] Jie Mei, Jingxi Yu, Suzanne Romain, Craig Rose, Kelsey Magrane, Graeme LeeSon, and Jenq-Neng Hwang. Unsupervised severely deformed mesh reconstruction (dmr) from a single-view image, 2022. 5.3.2

[103] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II 12*, pages 488–501. Springer, 2012. 2.5.4

[104] Fei Mi, Lingjing Kong, Tao Lin, Kaicheng Yu, and Boi Faltings. Generalized class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 240–241, 2020. 1.3

[105] NASA. Pluto: The 'other' red planet. https://www.nasa.gov/nh/pluto-the-other-red-planet, 2015. Accessed: 2018-12-06.

[106] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2.5.5

[107] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[108] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

[109] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. arxiv. *arXiv preprint arXiv:1710.06924*, 2017.

[110] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 524–540. Springer, 2020. 4.1, 4.2.3, 4.3

[111] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. (document), 1.3, 1.4, 2.5.2, 2.5.5, 5.1, 5.1.1, 5.1.2, 5.1.3, 5.2.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.13, 5.14, 5.15, 5.16, 5.17

[112] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, Ling Shao, and Ming-Hsuan Yang. An adaptive random path selection approach for incremental learning. *arXiv preprint arXiv:1906.01120*, 2019.

[113] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13588–13597, 2020.

[114] 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*, pages 2001–2010, 2017. 1.1, 1.3, 1.4, 2.1.1, 2.5.1, 2.5.4, 4.1, 4.2.3, 4.3

[115] James Rice. Poligon: A System for Parallel Problem Solving. Technical Report KSL-86-19, Dept. of Computer Science, Stanford Univ., 1986.

[116] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses, 2021. 2.3

[117] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 4.2.3, 4.2

[118] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995. 1.1, 2.1.1, 2.5.1

[119] Arthur L. Robinson. New ways to make microcircuits smaller. *Science*, 208(4447):1019–1022, 1980.

[120] Arthur L. Robinson. New Ways to Make Microcircuits Smaller—Duplicate Entry. *Science*, 208:1019–1026, 1980.

[121] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[122] 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. 2.1.2

[123] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

[124] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. (document), 5.6, 5.11

[125] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018. 1.1, 2.5.1

[126] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017. 2.1.1

[127] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015. 2.5.1

[128] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2.4.1

[129] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages

11909–11919, 2023. 2.5.3, 5.2.1, 5.2, 5.3, 5.5, 5.6, 5.7, 5.11, 5.3.3.1, 5.3.3.2, 5.13, 5.14, 5.15

[130] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2.4.1

[131] Yu-Ming Tang, Yi-Xing Peng, and Wei-Shi Zheng. When prompt-based incremental learning does not meet strong pretraining. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1706–1716, 2023. 2.5.3, 5.13, 5.3.3.3

[132] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12183–12192, 2020.

[133] Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv:2210.03114*, 2022. 1.4, 2.5.2, 5.1, 5.1.1, 5.1.2, 5.2.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.3.3.1, 5.13, 5.14, 5.15, 5.17

[134] Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022.

[135] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998. 2.1.1

[136] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020. 2.5.5

[137] Marco Toldo and Mete Ozay. Bring evanescent representations to life in lifelong class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16732–16741, 2022. 2.5.4

[138] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 2.3

[139] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. (document), 5.2.5, 5.4

[140] Cristina Vasconcelos, Vighnesh Birodkar, and Vincent Dumoulin. Proper reuse of image classification features improves object detection. 2022. 2.3

[141] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5.3.3.1

[142] Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. A primer on kernel methods. *Kernel methods in computational biology*, 47:35–70, 2004.

[143] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[144] Fu-Yun Wang, Da-Wei Zhou, Liu Liu, Han-Jia Ye, Yatao Bian, De-Chuan Zhan, and Peilin Zhao. Beef: Bi-compatible class-incremental learning via energy-based expansion and fusion. In *The Eleventh International Conference on Learning Representations*, 2022. 2.5.4

[145] Gaoang Wang, Jenq-Neng Hwang, Craig Rose, and Farron Wallace. Uncertainty-based active learning via sparse modeling for image classification. *IEEE Transactions on Image Processing*, 28(1):316–329, 2018.

[146] Gaoang Wang, Jenq-Neng Hwang, Kresimir Williams, and George Cutter. Closed-loop tracking-by-detection for rov-based multiple fish tracking. In *2016 ICPR 2nd Workshop on Computer Vision for Analysis of Underwater Imagery (CVAUI)*, pages 7–12. IEEE, 2016. 1.2

[147] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *Advances in Neural Information Processing Systems*, 2023. 2.5.3, 5.2.1, 5.2, 5.3, 5.5, 5.6, 5.7, 5.3.3.1, 5.13, 5.14, 5.15

[148] Runqi Wang, Yuxiang Bao, Baochang Zhang, Jianzhuang Liu, Wentao Zhu, and Guodong Guo. Anti-retroactive interference for lifelong learning. In *European Conference on Computer Vision*, pages 163–178. Springer, 2022.

[149] Runqi Wang, Xiaoyue Duan, Guoliang Kang, Jianzhuang Liu, Shaohui Lin, Songcen Xu, Jinhu Lü, and Baochang Zhang. Attriclip: A non-incremental learner for incremental knowledge learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3654–3663, 2023. 1.4, 2.5.2, 5.1.2, 5.2.1, 5.2.2, 5.2.2, 5.2, 5.3, 5.4, 5.2.3, 5.2.3, 5.5, 5.6, 5.7, 5.3.1, 5.3.3.1, 5.13, 5.14, 5.15

[150] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X Yu. Long-tailed recognition by routing diverse distribution-aware experts. *arXiv preprint arXiv:2010.01809*, 2020. 2.2

[151] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, 35:5682–5695, 2022. 1.4, 5.1.2, 5.3.3.1, 5.14, 5.3.3.4, 5.15

[152] Zifeng Wang, Tong Jian, Kaushik Chowdhury, Yanzhi Wang, Jennifer Dy, and Stratis Ioannidis. Learn-prune-share for lifelong learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 641–650. IEEE, 2020.

[153] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022. 2.5.3, 5.2.1, 5.2, 5.3, 5.5, 5.6, 5.3.3.1, 5.3.3.2, 5.3.3.2, 5.13, 5.14, 5.15

[154] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022. 1.4, 2.5.3, 5.2.1, 5.2, 5.3, 5.5, 5.6, 5.3.3.1, 5.13, 5.14, 5.15

[155] Andrew R Webb, Keith D Copsey, and Gavin Cawley. *Statistical pattern recognition*, volume 2. Wiley Online Library, 2011. 2.5.4

[156] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128, 2009. (document), 3.5, 4.2

[157] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020. 2.1.2

[158] Darren J White, C Svellingen, and Norval JC Strachan. Automated measurement of species and length of fish by computer vision. *Fisheries Research*, 80(2-3):203–210, 2006. 1.2

[159] Kresimir Williams, Nathan Lauffenburger, Meng-Che Chuang, Jenq-Neng Hwang, and Rick Towler. Automated measurements of fish within a trawl using stereo images from a camera-trawl device (camtrawl). *Methods in Oceanography*, 17:138–152, 2016. 1.2

[160] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020.

[161] Tz-Ying Wu, Pedro Morgado, Pei Wang, Chih-Hui Ho, and Nuno Vasconcelos. Solving long-tailed recognition with deep realistic taxonomic classifier. *arXiv preprint arXiv:2007.09898*, 2020. 2.4.2

[162] Tz-Ying Wu, Gurumurthy Swaminathan, Zhizhong Li, Avinash Ravichandran, Nuno Vasconcelos, Rahul Bhotika, and Stefano Soatto. Class-incremental learning with strong pretrained models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9601–9610, 2022. 2.5.1

[163] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 1.1, 1.3, 1.4, 4.1, 4.2.3, 4.3

[164] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018. 2.5.5

[165] Liuyu Xiang, Guiguang Ding, and Jungong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 247–263. Springer, 2020. 2.2

[166] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021. 1.1, 2.1.2, 2.5.4

[167] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: Hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2.4.2

[168] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, Qing Yang, and Cheng-Lin Liu. Convolutional prototype network for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2358–2370, 2020.

[169] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017. 1.1, 2.5.1

[170] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 4.2.3

[171] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. 2.1.1

[172] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12203–12213, 2020.

[173] Chi Zhang, Nan Song, Guosheng Lin, Yun Zheng, Pan Pan, and Yinghui Xu. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12455–12464, 2021.

[174] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. *arXiv preprint arXiv:2303.05118*, 2023. 5.13, 5.3.3.3

[175] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[176] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *arXiv preprint arXiv:2110.04596*, 2021.

[177] Yongshun Zhang, Xiu-Shen Wei, Boyan Zhou, and Jianxin Wu. Bag of tricks for long-tailed visual recognition with deep convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3447–3455, 2021.

[178] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217, 2020. 2.1.2

[179] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9719–9728, 2020. 2.2

[180] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. (document), 1.3, 1.4, 2.5.2, 5.1.2, 5.2.1, 5.2.2, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.3.3.1, 5.13, 5.14, 5.15

[181] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5871–5880, 2021. 2.5.4

[182] Hongguang Zhu, Yunchao Wei, Xiaodan Liang, Chunjie Zhang, and Yao Zhao. Ctp: Towards vision-language continual pretraining via compatible momentum contrast and topology preservation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22257–22267, 2023.

[183] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9296–9305, 2022. 2.5.4

[184] Xinqi Zhu and Michael Bain. B-cnn: branch convolutional neural network for hierarchical classification. *arXiv preprint arXiv:1709.09890*, 2017. 2.4.2, 3.1.2, 3.2.3

[185] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015. 2.3

[186] Boaz Zion. The use of computer vision technologies in aquaculture–a review. *Computers and electronics in agriculture*, 88:125–132, 2012. 1.2