ON INCREMENTAL LEARNING WITH LONG SHORT TERM STRATEGY

Anonymous authors
Paper under double-blind review

ABSTRACT

Incremental learning aims at mitigating the forgetting during the sequential learning of deep neural networks. In the process, a procedure (including distillation, replaying, etc.) is usually adopted to help model accumulate knowledge. However, we discover the tuning of such procedure could face the "long short term dilemma" that the optimal procedure of short term learning is not necessarily equivalent to that of long term learning due to their need of different plasticity/stability balances. The existing methods have to take the trade-off to achieve better overall performance along the incremental tasks. In this paper, we propose a novel LongShortTerm strategy that circumvents limitations of widely-used pipeline with single branch and brings model capability in both short and long term into full play. To further control the plasticity/stability balance in LongShortTerm strategy, we discover that for ViT backbone, magnitude of memory augmentation is critical to retention of model and propose Margin-based Data Augmentation to meet different balances in long short term learning. Extensive experiments on two complex CIL benchmarks: ImageNet-100 and ImageNet-1K demonstrate the effectiveness of our LongShortTerm strategy with improvements of 0.59%-3.10% over state-of-the-art solution.

1 INTRODUCTION

Human being is capable of learning from the environment and accumulating knowledge gradually. Given a group of datasets, common artificial intelligence systems could fit each individual dataset well. However, when learning incrementally, models will suffer from *catastrophic forgetting* McCloskey & Cohen (1989); French (1999); Robins (1995) and fail to handle ever-changing environments. To mitigate forgetting, efforts Li & Hoiem (2017); Hou et al. (2019); Castro et al. (2018); Douillard et al. (2020); Simon et al. (2021) have been made to help networks maintain the memory.

However, **on one side**, attempts to preserve preceding knowledge potentially brings adverse effects to accumulating new learnt knowledge more or less. And as the result, the best on accuracy of new and old classes could hardly be achieved simultaneously in single model due to stability-plasticity dilemma Parisi et al. (2019). **On the other side**, accuracy of each task is actually a continuous metric with growing number of incremental steps. In later task, accuracy of old tasks will take more percentage in the overall accuracy. To meet the tendency of the growing weight of preceding performance metric, the training procedure should encourage model to be more retentive with certain sacrifice of newly learnt knowledge. In contrast, model is more likely to be more plastic on current task. Fig. 1 **Upper Left** demonstrates such phenomenon, where the stable "Long Term Learner" and the plastic "Short Term Learner" are trained with different incremental learning methods and yield advanced performance in different stage of training. Based on this, we claim that the optimal procedure on the current task could not necessarily produce the best result in later tasks. They need different balance between learning and reviewing. Or in other words, the optimal solution of short term learning is not equivalent to that of long term learning.

To ameliorate such "Long Short Term Dilemma", existing methods with single branch (visualized in Fig. 1 **Lower**) have to balance the present and future performance. The limitation mainly comes from the fact that each model plays the roles for inference on current task and initializing model on later task and this contradiction is visualized in Fig. 1 **Upper Right**. To further circumvent the trade-off and break limitation of single branch pipeline, we propose the novel **LongShortTerm** strategy

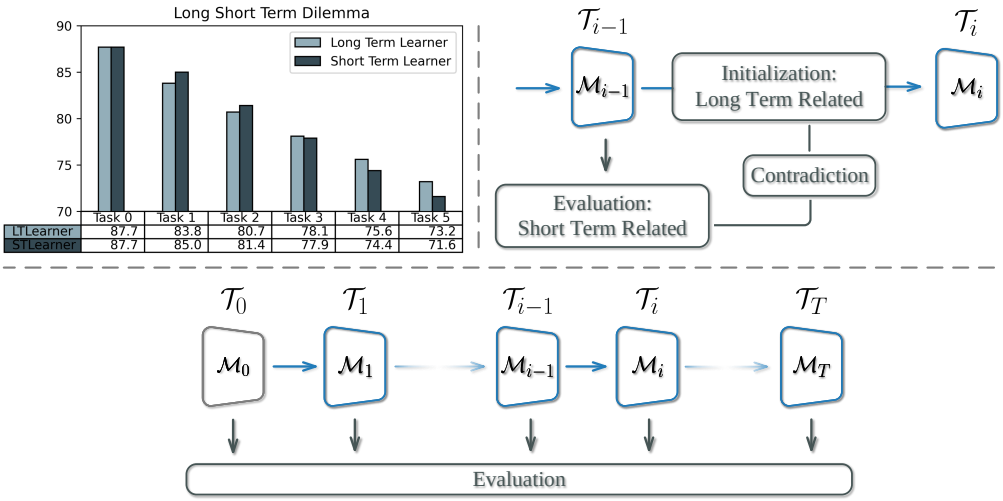

Figure 1: **Upper Left**: Performance bar of "Long Term Learner" and "Short Term Learner" on ImageNet100-B50-C10 task, where the initial base task contains 50 classes and 10 classes for each incremental task. "Long Term Learner" and "Short Term Learner" are trained with different methods. Stable "Long Term Learner" here casts no augmentation on memory and in contrast, plastic "Short Term Learner" casts heavy augmentation on memory. They have different advantage periods in the whole incremental process and its analysis will be clarified in Sec. 3.3. **Upper Right**: Contradiction in single branch pipeline between better long term and short term learning. **Lower**: Pipeline of widely-recognized single branch incremental learning.

to achieve optimal results in both short term and long term learning. In each incremental task, the training process consists of two individual branches. One branch is shortsighted and will greedily tune the model for the best performance in current stage. The other branch will be dedicated to obtain the art results in the future. The shortsighted model takes responsibility for evaluation and the foresighted model acts as the initialization of future tasks.

For the sake of tuning the balance between learning and reviewing, we delve into data augmentation and discover that magnitude of memory augmentation acts as the key factor for incremental learning of Vision Transformer Dosovitskiy et al. (2021). Empirically, lighter augmentation allows learners to be more retentive and more friendly to later tasks, heavier augmentation enables learners to be more plastic for the current tasks. Thus, we combine the discovery and LongShortTerm strategy to propose a novel incremental pipeline with dual branches and break the "Long Short Term Dilemma".

In summary, our contribution are in four aspects: **i)** We discover "Long Short Term Dilemma" that the best solutions for short term and long term learning may not overlap due to the different optimal balance between learning and reviewing. **ii)** To tackle the dilemma, we break the limitation of pipeline with single branch and propose the LongShortTerm strategy to split the training process into two branches with their long term and short term targets. **iii)** We discover that magnitude of memory augmentation poses significant effects to the balance between plasticity and retention of incremental model and further affect the trade-off between long term and short term learning. Together with LongShortTerm strategy, we boost the short term performance and keep the long term endurance simultaneously. **iv)** We conduct experiments on incremental benchmarks including ImageNet-100 and ImageNet-1K. The performance gain over both previous CNN and ViT state-of-the-art methods is 0.59%-3.10%.

2 Related Work

2.1 Incremental Learning

Incremental learning Thrun (1998) (IL) is proposed to tackle the catastrophic forgetting French (1999); Robins (1995) and allow neural networks to accumulate knowledge from a sequence of

tasks. To alleviate forgetting, methods based on parameters regularization Kirkpatrick et al. (2017); Aljundi et al. (2018) constrained parameters that were important to preceding tasks. Parameter-isolation based methods Mallya et al. (2018); Kanakis et al. (2020); Abati et al. (2020); Yan et al. (2021) expanded the capacity of neural networks with extra or dedicated parameters to mitigate interference between new and existing knowledge. Among all IL algorithms, the mainstream are based on distillation. LwF Li & Hoiem (2017) first regarded the pre-trained model as a teacher to transfer knowledge. Dhar *et al.* Dhar et al. (2019) proposed to penalize the change of attention map. Moreover, in order to distill more knowledge from preceding data, iCaRL Rebuffi et al. (2017) and its improved variants Hou et al. (2018); Castro et al. (2018) introduced rehearsal mechanism, where few (meta-)data are pre-stored and replayed to review knowledge in later steps. Following up on this, Wu *et al.* Wu et al. (2019) and Hou *et al.* Hou et al. (2019) discovered and reduce the bias in classifiers. PODNet Douillard et al. (2020) transfered more spatial information from intermediate attentions. GeoDL Simon et al. (2021) constrained the geodesic flow in lower dimensions. TOPIC Tao et al. (2020b) and TPCIL Tao et al. (2020a) dedicated to preserve topology of pre-stored memory during the feature drifting.

2.2 Vision Transformer

Transformer Vaswani et al. (2017) was first proposed to process natural language processing (NLP) tasks. It introduced self-attention architecture to model the relationship between words in sentences. In order to bring such mechanism into computer vision, Dosovitskiy *et al.* introduced ViT Dosovitskiy et al. (2021) by splitting the image and taking patches as tokens. Later, DeiT Touvron et al. (2021a), DeepViT Touvron et al. (2021b), CrossViT Chen et al. (2021), SwinTransformer Liu et al. (2021) and a line of other transformer variants are proposed, bringing attractive improvements on self attention. In addition to classification, transformer also empowers many other vision tasks, such as object detection Zhu et al. (2020), semantic segmentation Zheng et al. (2021), self supervision He et al. (2022), object ReID He et al. (2021) and image compression Qian et al. (2022). ViTIL Yu et al. (2021) and DyTox Douillard et al. (2022) first replaced CNN with ViT as the backbone in incremental learning and demonstrated the promising capabilities in addressing catastrophic forgetting.

3 Proposed Method

3.1 Background

Class incremental learning aims to classify growing number of classes after learning sequentially from an initial task $\{\mathcal{T}_0 : \mathcal{D}_0\}$ and T incremental tasks $\{\mathcal{T}_1 : \mathcal{D}_1, \mathcal{T}_2 : \mathcal{D}_2, ..., \mathcal{T}_T : \mathcal{D}_T\}$, where $\mathcal{D}_i$ is dataset of task $\mathcal{T}_i$. The label set $\mathcal{Y}_i$ of $\mathcal{D}_i$ is exclusive each other, *i.e.*, $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset, i \neq j$. During the learning process of $\mathcal{T}_i$, the model $\mathcal{M}_i$ is trained on the basis of $\mathcal{M}_{i-1}$ to **i).** incrementally predict data from $\mathcal{D}_i$ and **ii).** remain its capacity on all previously seen classes $\mathcal{Y}_{0:i-1}$ with memory $\mathcal{D}^M_{0:i-1} = \cup_{k=0}^{i-1}\mathcal{D}^M_k$, where superscript M denotes **M**emory and $\mathcal{D}^M_k$ is a subset sampled from $\mathcal{D}_k$ for replaying. After the training, the expert model $\mathcal{M}_i$ and rebuilt memory $\mathcal{D}^M_{0:i} = \mathcal{D}^M_{0:i-1} \cup \mathcal{D}^M_i$ will be left for the next incremental step. The challenge here mainly comes from how to accumulate knowledge with incremental data $\mathcal{D}_i$ and very limited memory $\mathcal{D}^M_{0:i-1}$.

Metrics: After the *i*-th incremental step, the model will be tested on all seen learnt task $\mathcal{T}_{0:i}$ and yield the corresponding accuracy $\mathcal{A}_{0:i}$ as written below:

$$\begin{aligned} \mathcal{A}_{0:i} =& \alpha\mathcal{A}_{0:i-1} + \beta\mathcal{A}_i = \alpha\mathcal{A}_{\text{Old}} + \beta\mathcal{A}_{\text{New}}, \\ \alpha =& \frac{|\mathcal{Y}_{0:i-1}|}{|\mathcal{Y}_{0:i}|},\ \beta = \frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{0:i}|}, \end{aligned} \tag{1}$$

where $\mathcal{A}_{\text{Old}}$ and $\mathcal{A}_{\text{New}}$ denote the average accuracy on memory classes and new learnt classes. And it is critical to balance $\mathcal{A}_{\text{Old}}$ and $\mathcal{A}_{\text{New}}$ with their weights for better overall accuracy. To evaluate the whole learning process, all accuracy over incremental steps is averaged and reported as **Inc Acc** $= \frac{1}{T+1}\sum_{i=0}^{T}\mathcal{A}_{0:i}$.

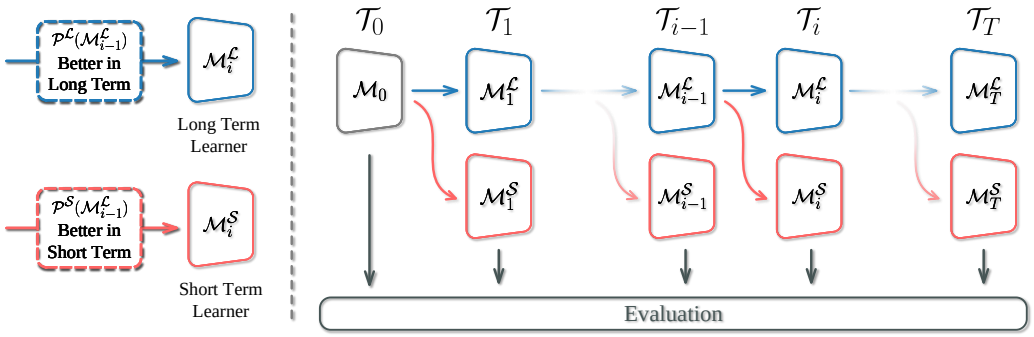

Figure 2: **Pipeline of LongShortTerm Strategy.** During the whole incremental journey, the long term learners are trained iteratively to approach a better $\mathcal{M}_T^{\mathcal{L}}$. Simultaneously, in order to achieve the best performance in each intermediate stage $\mathcal{T}_i$, another branch is built to greedily train a short term learner $\mathcal{M}_i^{\mathcal{S}}$ on the basis of latest long term learner $\mathcal{M}_{i-1}^{\mathcal{T}}$ in each incremental step. The life of each short term learner lasts only one step and will not affect the optimal long term learning.

3.2 LongShortTerm Strategy

In the i-th incremental step, given an incremental task sequence $\{\mathcal{T}_i : \mathcal{D}_i, \mathcal{T}_{i+1} : \mathcal{D}_{i+1}, \cdots, \mathcal{T}_T : \mathcal{D}_T\}$, our goal is to train a sequence of models to perform well along the whole learning procedure. To achieve this goal, on $\mathcal{T}_i$, the well-recognized pipeline trains the model $\mathcal{M}_i$ with procedure $\mathcal{P}(\mathcal{M}_{i-1})$ and takes it as initialization for next task.

However, we discover that tuning of procedure $\mathcal{P}$ could be contradictory between achieving better performance on current task and acting as a good initialization for future tasks. On one hand, supposing the procedure is greedy, it is capable to produce the best $\mathcal{M}_i$ on current task $\mathcal{T}_i$. Because $\mathcal{T}_i$ is a task of short term, we name such procedure $\mathcal{P}^{\mathcal{S}}$ and the model $\mathcal{M}_i^{\mathcal{S}}$. On the other hand, supposing the procedure is visionary, it is competent to obtain better $\mathcal{M}_I$ on a certain later task $\mathcal{T}_I$. Because $\mathcal{T}_I$ is a task of long term, we name such procedure $\mathcal{P}^{\mathcal{L}}$ and corresponding model $\mathcal{M}_I^{\mathcal{L}}$. Due to the different training procedure, $\mathcal{M}_i^{\mathcal{L}}$, which is the prefix model of $\mathcal{M}_I^{\mathcal{L}}$, will not necessarily overlap with $\mathcal{M}_i^{\mathcal{S}}$ and perform no better than $\mathcal{M}_i^{\mathcal{S}}$ on $\mathcal{T}_i$.

With regard to how to induce such divergence, in this paper, we take the increasing nature of memory classes for an easy example. In an uniform incremental setting, with the increasing number of tasks, memory classes takes more percentage of all seen classes. This indicates the greater α in Equ. 1. As the result, the accuracy of memory classes tends to be more and more important in later tasks. And this leads to a more stable training procedure for long term learning and relatively more plastic procedure for short term learning. The detailed comparison is illustrated in Fig. 1 **Upper Left**.

In pipeline with single branch, since the current model $\mathcal{M}_i$ will be applied on evaluation and to initialize the model in following-up long term learning, it will be trapped in the trade-off between long term target and short term target, which is a "Long Short Term Dilemma", as demonstrated in Fig. 1 **Upper Right**. To resolve this dilemma, we propose LongShortTerm strategy to avoid the trade-off by assigning targets to two individual learning branches and achieve the best in both short term and long term. In practice, given the latest model $\mathcal{M}_{i-1}$, the incremental model $\mathcal{M}_i$ will initialize its parameters with $\mathcal{M}_{i-1}$ and be trained incrementally twice in long term and short term branches as illustrated in Fig. 2:

Greedy Short Term Branch: $\mathcal{P}^{\mathcal{S}}$ encourages the model to focus only on all seen classes and perform well as much as possible. The short term model $\mathcal{M}_i^{\mathcal{S}}$ will be used for evaluation.

Visionary Long Term Branch: $\mathcal{P}^{\mathcal{L}}$ encourages the model to act as the prefix of a better model in later steps. The long term model $\mathcal{M}_i^{\mathcal{L}}$ will serve as initialization of possible future incremental steps.

Theoretically, the strategy could maintain the performance in both long short terms with two branches, taking different balance between learning and reviewing.

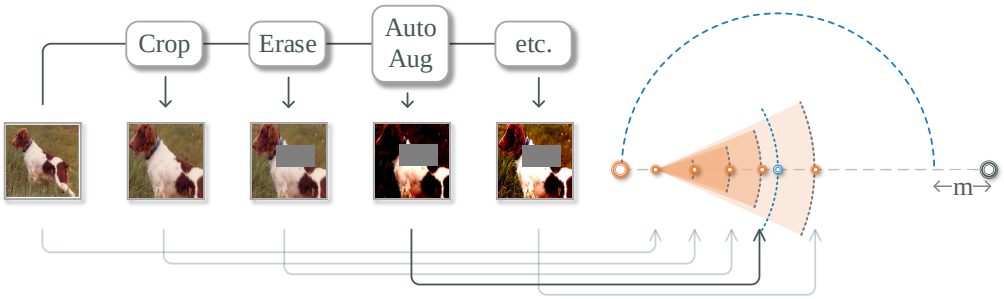

Figure 3: **Pipeline of Margin-based Data Augmentation (MDA).** The raw image is augmented with a pool of augmenters. MDA adopts augmentation as strong as possible until the augmented image hits the boundary, which is close enough to the other centroid with the margin m.

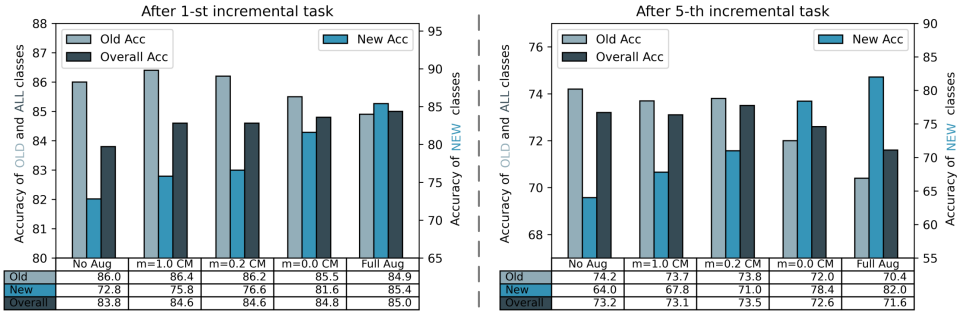

Figure 4: Performance bar of models with different augmentation magnitude after 1-st (**Left**) and 5-th (**Right**) incremental task on ImageNet100-B50-C10 task, where the initial base task contains 50 classes and 10 classes for each incremental task.

3.3 Pitfall of Memory Augmentation

In Sec. 3.2, we claim that the model could benefit from our LongShortTerm strategy by controlling the balance between plasticity and stability. In this section, we will further introduce our discovery of memory augmentation for ViT and adopt it as the key factor in tuning such balance.

Margin-based Data Augmentation. As visualized in Fig. 1 (**Upper Left**), heavier data augmentation helps to achieve better accuracy in the earlier steps over light augmentation. But such advantage will not last until the end. In order to dive deeper into the issue on magnitude of augmentations, we propose to control magnitude of multiple data augmentation with margin. As demonstrated in Fig. 3, given an input image (x_i, y_i) and a pool of augmenters $A = \{a_1, a_2, \cdots\}$, the image x_i is processed by augmenters sequentially and produces $x^A = \{x_i^0\} \cup \{x_i^k = a_k(x_i^{k-1})\}$, where $x_i^0 = x_i$. To ensure the diversity of x^A, we shuffle the augmenters before augmentation. We measure the augmentation magnitude of x_i^k with the similarity between its class centroid w_{y_i}. And the similarity margin with any other centroids will be used to constrain the magnitude. The formulation is written below:

$$\begin{aligned} a(x_i; m) &= \arg\min_{x_i^k \in x^A} \cos(\mathcal{M}(x_i^k), w_{y_i}) \\ s.t.\ \cos(\mathcal{M}(x_i^k), w_{y_i}) &\geq \cos(\mathcal{M}(x_i^k), w_j) + m,\ \forall j \neq y_i, \end{aligned} \tag{2}$$

where $\mathcal{M}(x_i^k)$ denotes the feature of image x_i^k extracted by model $\mathcal{M}$ and we take the corresponding weight vector in classifier as the class centroid. With **M**argin-based **D**ata **A**ugmentation (MDA), the magnitude of augmentation could be tuned with a margin m and a smaller m will lead to heavier data augmentation. Note that augmentation pool A excludes mixing methods, *i.e.,* CutMix Yun et al. (2019) and MixUp Zhang et al. (2018), and more details could be found in Sec. 4.4.

Analysis. For fair comparison, in Fig. 4 **Left**, all models are initialized with the same pre-trained model on $\mathcal{T}_0$. After the short term learning on $\mathcal{T}_1$, model with heavier memory augmentation yields worse $\mathcal{A}_{\text{Old}}$, better $\mathcal{A}_{\text{New}}$ and better $\mathcal{A}_{0:1}$. When it comes to the long term learning ($\mathcal{T}_5$ here in Fig. 4 **Right**), training procedure with heavier augmentation still suffers from more forgetting and achieves better $\mathcal{A}_{\text{New}}$. However, since memory takes more percentage of all learnt knowledge in later steps, α in Equ. 1 is an increasing metric when calculating the overall performance. Thus, despite better $\mathcal{A}_{\text{New}}$, heavier augmentation fails to maintain its advantage to the end of the marathon incremental journey.

To conclude, magnitude of memory augmentation could affect the trade-off between learning new knowledge and reviewing memory. It will further pose impacts to the endurance of model, *i.e.,* long term or short term, due to the increasing importance of reviewing memory.

3.4 LongShortTerm with Proper Augmentation

As aforementioned, LongShortTerm strategy in Sec. 3.2 could allow model to bring its capability in both short and long term into full play with different plasticity/stability balance. And with MDA in Sec. 3.3, such balance could be finely tuned. Combining them, the overall pipeline is concluded in Fig. 2 and Alg. 1, training loss in each procedure is summarized below:

Training Losses. To simplify the training procedure of each step, the overall loss only contains cross entropy loss $\mathcal{L}_{ce}$ and distillation loss $\mathcal{L}_{dis}$:

$$\begin{aligned} \mathcal{L}_{ce}(x, y) &= \sum_{i=1}^{|\mathcal{Y}|} -y_i \log p_i, \\ \mathcal{L}_{dis}(x) &= 1 - \cos(\mathcal{M}_{i-1}(x), \mathcal{M}_i(x)), \\ \mathcal{L}(x, y) &= \mathcal{L}_{ce}(x, y) + \lambda \mathcal{L}_{dis}(x), \end{aligned} \tag{3}$$

where λ is the weight of distillation loss, $\mathcal{M}_i(x)$ and $\mathcal{M}_{i-1}(x)$ are the class tokens output by the last attention layers in $\mathcal{M}_i$ and $\mathcal{M}_{i-1}$.

Algorithm 1 LongShortTerm strategy (i-th task)

Input: Incremental data $\mathcal{D}_i$ and memory $\mathcal{D}_{0:i-1}^{M}$.
Input: Previous long short term models $\mathcal{M}_{i-1}^{\mathcal{L}}$ and $\mathcal{M}_{i-1}^{\mathcal{S}}$.
Output: Long short term models $\mathcal{M}_{i}^{\mathcal{L}}$ and $\mathcal{M}_{i}^{\mathcal{S}}$.
Output: Memory $\mathcal{D}_{0:i}^{M}$.

Inference on $\mathcal{T}_{0:i-1}$ with $\mathcal{M}_{i-1}^{\mathcal{S}}$.
Training on $\mathcal{T}_i$:
1: Initialize the long short term models $\mathcal{M}_{i}^{\mathcal{L}}$ and $\mathcal{M}_{i}^{\mathcal{S}}$ with $\mathcal{M}_{i-1}^{\mathcal{L}}$.
2: Train the long term model $\mathcal{M}_{i}^{\mathcal{L}} = \mathcal{P}^{\mathcal{L}}(\mathcal{M}_{i-1}^{\mathcal{L}}; m^{\mathcal{L}})$ with margin $m^{\mathcal{L}}$.
3: Train the short term model $\mathcal{M}_{i}^{\mathcal{S}} = \mathcal{P}^{\mathcal{S}}(\mathcal{M}_{i-1}^{\mathcal{L}}; m^{\mathcal{S}})$ with margin $m^{\mathcal{S}}$, where $m^{\mathcal{S}} < m^{\mathcal{L}}$.
4: Rehearse $\mathcal{D}_i$ with $\mathcal{M}_{i}^{\mathcal{L}}$ and update memory $\mathcal{D}_{0:i}^{M} = \mathcal{D}_{0:i-1}^{M} \cup \mathcal{D}_{i}^{M}$.
Inference on $\mathcal{T}_{0:i}$ with $\mathcal{M}_{i}^{\mathcal{S}}$.

4 Experiments

4.1 Datasets and Protocols

Following prior works Douillard et al. (2020); Hou et al. (2019); Rebuffi et al. (2017), we conduct experiments on two datasets: 1). **ImageNet-1K** Deng et al. (2009) is a dataset of large scale. It consists of 1,000 classes, with more than 1.2 million images in total. 2). **ImageNet-100** Deng et al. (2009) is a subset of ImageNet, which has randomly sampled 100 classes. The base task $\mathcal{T}_0$ starts an incremental scenario with M classes, followed by several incremental tasks with N classes for each task. We denote such scenario as B[M]-C[N]. The scenario is evaluated with the metrics mentioned in Sec 3.1.

Methods	ImageNet-100			ImageNet-1K		
	B50-C10	B50-C5	B10-C10	B500-C100	B500-C50	B100-C100
CNN-based						
ResNet18 Joint	81.18			69.65		
iCaRL Rebuffi et al. (2017)	65.04	59.53	61.60	51.36	46.72	55.95
BiC Wu et al. (2019)	70.07	64.96	64.35	62.65	58.72	57.60
UCIR Hou et al. (2019)	70.84	68.32	55.33	64.34	61.28	57.14
Mnemonics Liu et al. (2020)	72.58	71.37	-	64.54	63.01	-
PODNet Douillard et al. (2020)	75.54	74.33	-	66.95	64.13	-
TPCIL Tao et al. (2020a)	76.27	74.81	-	64.89	62.88	-
DDE Hu et al. (2021)	76.71	75.41	-	66.42	64.71	-
ViT-based						
ViT-ti Joint	85.62			76.59		
ViTIL Yu et al. (2021)	79.43	76.92	69.68	69.20	-	65.13
Ours	**80.82**	**78.61**	**70.27**	**72.30**	**68.85**	**67.08**

Table 1: Comparison with other methods on ImageNet-100 (B50-C10, B50-C5 and B10-C10 settings) and ImageNet-1K (B500-C100, B500-C50 and B100-C100 settings). We compare with CNN-based methods (**Upper** part) and ViT-based methods (**Lower** part). And "ResNet18/ViT-ti Joint" denote the results of jointly trained models, which are trained with all samples of all classes.

4.2 Implementation Details

To reduce the model parameters, ViT-tiny Dosovitskiy et al. (2021) (5M paras.) is adopted as the backbone network across all tasks. For ImageNet-100 and ImageNet-1K, all images are resized to 224×224. The pre-training on initial classes takes 800 epochs with all augmentations adopted. For each incremental steps, training will last 200 epochs. All training process is launched with learning rate of 5e-4 and batch-size of 512. Following ViTIL Yu et al. (2021), we replace PatchEmbedding with ConvStem Xiao et al. (2021) and train the classifier with larger learning rate. Fine-tune trick in Hou et al. (2019); Douillard et al. (2020); Hu et al. (2021) is also employed. When fine-tuning with balanced data, data augmentation is all applied. But note that whether using CutMix Yun et al. (2019)/MixUp Zhang et al. (2018) in fine-tuning depends on whether CutMix/MixUp is adopted in training stage, *e.g.* in fine-tuning stage of "m=1.0 CM" setting, CutMix/MixUp is on and in fine-tuning stage of "No Aug" setting, CutMix/MixUp is off. About hyper-parameters margin $m^{\mathcal{L}}$ and $m^{\mathcal{S}}$ for long term and short term branches will be discussed in Sec. 4.4.

4.3 Comparison with Other Methods

We report our performances in Tab. 1. On ImageNet-100, our method achieves performance of 80.82%, 78.61% and 70.27% on all typical settings. On B50-C10, B50-C5 settings, the initial task have half of total classes and initial model have more general embedding capability. Our method outperforms CNN-based SOTA DDE Hu et al. (2021) with margins of 4.11% and 3.2% and boosts ViT-based method ViTIL Yu et al. (2021) by 1.39% and 1.69%. On B10-C10 setting, with much less knowledge in initial task, our method still yields consistent improvements over CNN-based SOTA BiC Wu et al. (2019) and ViTIL with 5.92% and 0.59%. This demonstrates that our advantage could cover all scenarios and not limited on some specific settings, since settings starts with more initial classes, *e.g.* B50, lays more emphasis on anti-forgetting property of methods and in contrast, settings starts with less initial classes, *e.g.* B10, underlines learning property of methods. On ImageNet-1K, our method obtains performance of 72.30%, 68.85% on all B500 settings, surpassing the CNN-based best models PODNet Douillard et al. (2020) and DDE Hu et al. (2021) 5.35+% and 4.14+%. Similarly, 3.1% performance gain over ViTIL could also be observed. On B100-C100 setting, the advantage of 9.48% and 1.95% over BiC and ViTIL validates the effectiveness of our method.

4.4 Ablation Studies and Parameter Analysis

Effectiveness of Tuning Memory Augmentation. To demonstrate the impact of memory augmentation, we tuned the margin in MDA in Equ. 2. As a reminder, greater m will filter out more augmentation and smaller m will allow more augmentation cast on images. Specifically, when $m = 1$,

Term	Aug Settings	ImageNet-100										Inc Acc
		Task 0	Task 1		Task 2		Task 3		Task 4		Task 5	
		Overall	Overall		Overall		Overall		Overall		Overall	
			Old	New	Old	New	Old	New	Old	New	Old	New
LongTerm	Full Aug: m=-1.0 w/ CM	87.7	85.0		81.4		77.9		74.4		71.6	79.7
			84.9	85.4	80.9	84.2	76.9	85.4	73.3	82.8	70.4	82.0
LongTerm	m=0.0 w/ CM	87.7	84.8		81.6		79.0		75.7		72.6	80.2
			85.5	81.6	81.7	81.0	78.5	82.8	75.5	77.8	72.0	78.4
LongTerm	m=0.2 w/ CM	87.7	84.6		81.4		79.1		75.4		73.5	80.3
			86.2	76.6	82.5	74.8	79.5	76.6	76.1	69.8	73.8	71.0
LongTerm	m=1.0 w/ CM	87.7	84.6		81.9		78.6		75.2		73.1	80.2
			86.4	75.8	83.0	75.4	79.4	73.4	76.1	67.8	73.7	67.8
LongTerm	No Aug: m=1.0 w/o CM	87.7	83.8		80.7		78.1		75.6		73.2	79.9
			86.0	72.8	82.5	70.0	79.6	67.6	76.8	66.0	74.2	64.0
ShortTerm	m=1.0 w/ CM	87.7	85.0		81.5		79.1		76.5		74.4	80.7
			84.9	85.4	83.2	70.8	79.8	73.8	77.5	68.4	75.0	68.6
ShortTerm	m=0.2 w/ CM	87.7	85.0		81.5		79.5		76.5		74.1	80.7
			84.9	85.4	83.0	73.0	80.1	75.2	77.3	69.8	74.8	68.2
ShortTerm	m=0.0 w/ CM	87.7	85.0		81.5		79.1		76.5		74.4	80.9
			84.9	85.4	82.7	78.4	79.5	79.4	76.5	74.2	74.2	76.8
ShortTerm	Full Aug: m=-1.0 w/ CM	87.7	85.0		81.9		79.6		76.3		74.2	80.9
			84.9	85.4	82.2	80.0	79.2	82.4	75.9	79.6	73.4	81.0

Table 2: Ablation studies and parameter analysis. In **Upper** part (starting with "Long Term"), we discuss the impact of augmentation magnitude of memory in long term branch. In **Lower** part (starting with "Short Term"), we discuss that in short term branch. For fair comparison, all short term branches here derive from the same long term branch with "No Aug: m=1.0 w/o CM". It is noteworthy that "CM" denotes CutMix/MixUp of memory.

no operations in augmentation pool are applied and when $m = -1$, all operations are casts on the input image sequentially. Results are listed in Table. 2 **Upper** part. Comparison between " m = -1.0/0.0/0.2/1.0 w/ CM " and " m=1.0 w/o CM " indicates that model could benefits from heavier augmentation with more capability to learn from the new task, especially "Full Aug" achieves the best 85.0% overall acc on $\mathcal{T}_1$. However, "Full Aug" fails to perform well with the worst old acc of 70.4% and overall acc of 71.6% on $\mathcal{T}_5$. On contrast, learners with lighter augmentation gradually gain their advantage on later tasks, *i.e.,* $\mathcal{T}_3, \mathcal{T}_4$ and $\mathcal{T}_5$ with higher yet more important old acc and achieve 73.0+% overall acc, 1.5+% higher than "Full Aug". Especially, when CutMix/MixUp is disabled, " m=1.0 w/o CM " produces the best 74.2% old acc on $\mathcal{T}_5$. In general, improvements of 0.6% Inc Acc is observed between "m=0.2 w/ CM" and "Full Aug". This suggests that proper magnitude of memory augmentation should be set with our MDA to balance the forgetting and learning for better Inc Acc even in a single branch pipeline.

Effectiveness of Long Short Term Strategy. To investigate the contribution and property of LongShortTerm strategy, we select "No Aug: m=1.0 w/o CM" as the fixed long term branch and let it fork to build short term branches with different augmentation settings, which are shown in Table. 2 **Lower** part. To draw a clear parallel with single branch long term pipeline, we take "ShortTerm-Full Aug" as examples. On all incremental tasks, in despite of certain loss of old acc, "ShortTerm-Full Aug" still outperforms "LongTerm-No Aug" in a consistent manner due the huge advantage of new acc, especially on $\mathcal{T}_5$ (17.0% over 64.0%). When compared with "LongTerm-Full Aug", since "ShortTerm-Full Aug" derives from "LongTerm-No Aug", forgetting is significantly mitigated and old acc is boosted by 1.3% in all tasks. Similar phenomena could be observed in other settings and make all long short term models surpass all single branch long term models by 0.4+% Inc Acc. In a nutshell, models trained with our long short term strategy tend to inherit higher old acc from long term branch and act well in new acc with greedy training. This demonstrates the limitation of single branch pipeline and validates the effectiveness of LongShortTerm strategy.

4.5 More about Memory Augmentation

Typically, heavy memory augmentation pushes domain of each class away from each other and induces larger domain gap between incremental domain and memory domain. The large domain gap reduces the rick of mis-classification and boost the overall accuracy, since which, heavy augmentation gains its short-term advantage.

However, the cost of producing large domain gap is more feature drifting of memory. In Fig. 5, we take the initial class "$y = 0$" as example and plot t-SNEs of its memory, training samples (note that

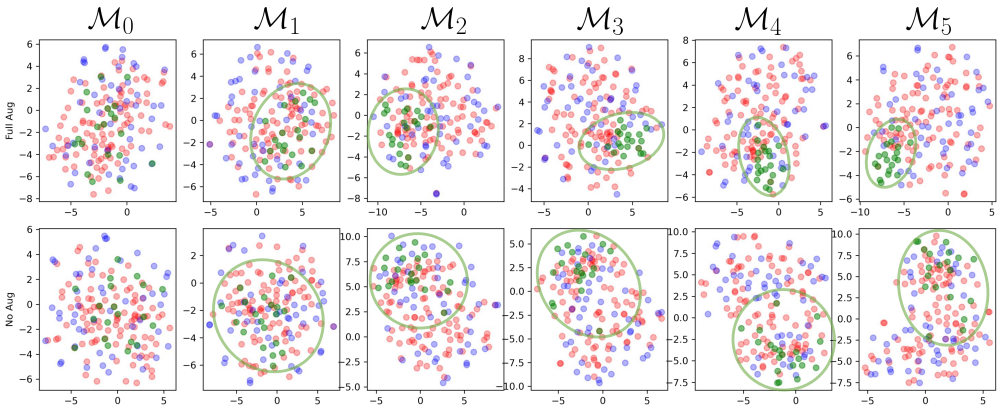

Figure 5: Visualization of initial class $y = 0$ in ImageNet-100 B50-C10 setting. We plot the t-SNE Van der Maaten & Hinton (2008) of 6 incremental steps in 6 columns. T-SNEs embedded by models trained with Full Aug and No Aug are shown in first and second row. Memory, training samples and validation samples are denoted by green dots, red dots and blue dots. We additionally highlight the domain of memory with green circles. Best viewed in colors.

MMD	Train vs Val	Train vs Memo	Val vs Memo
No Aug	0.163	0.490	0.805
m=1.0 w/ CM	0.149	0.518	0.810
m=0.2 w/ CM	0.156	0.668	1.002
m=0.0 w/ CM	0.154	0.726	1.058
Full Aug	0.154	0.736	1.072

Table 3: MMDs between **Memo**ry, **Train**ing samples and **Val**idation samples. MMD is calculated between features of two domains. We average and report MMDs of all initial classes $\mathcal{Y}_0$ with the ultimate model $\mathcal{M}_T$ in ImageNet-100 B50-C10 setting. Note that greater MMD means larger domain gap.

the training samples here are sampled from the original training set and exclusive with memory) and validation samples. At the very beginning in first column, domain of memory completely overlaps with that of val samples. After the first incremental task, "Full Aug" suffers from more overfit on the limited memory than "No Aug". The domain of memory drifts away from that of val (blue dots) and training samples (red dots). If we take val samples and training samples as real distribution of class, the domain of memory detaches from its real distribution gradually. In Tab. 3, domain of training and val samples are always in a good match, but more augmentation will induce larger domain gap in both "Train vs Memo" and "Val vs Memo". This verifies that stronger augmentation will lead to much more serious "detach problem". Ansince the memory will be used to estimate the distribution of the corresponding class. The huge domain gap misleads the classifier and makes it fail to play the role of class centroid. As the result, the accuracy of preceding classes drops dramatically and hurt the overall performance in the later steps.

5 Conclusion

In this paper, we discover the "Long Short Term Dilemma" that the optimal solution of short term learning could not necessarily equivalent to that of long term learning. And basic pipeline with one branch could either select one of them or take the trade-off. To break such limitation, we proposed LongShortTerm strategy to assign goals of long term and short term learning to two branches and achieve the attractive on both sides. In addition, although how to generally induce better long term models still needs more investigation, we discover that magnitude of memory augmentation could determine the model's endurance that whether model is capable to performer well in the long term or the short term. With proposed MDA mechanism, we precisely tuned the memory augmentation. And together with LongShortTerm strategy, we boosted model performance over single branch pipeline in almost all individual tasks in ImageNet-100 and ImageNet-1K incremental benchmarks.

REFERENCES

Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *CVPR*, pp. 3931–3940, 2020.

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

Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, pp. 233–248, 2018.

Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, pp. 357–366, 2021.

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*, pp. 248–255. Ieee, 2009.

Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *CVPR*, pp. 5138–5146, 2019.

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. *ICLR*, 2021.

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pp. 86–102. Springer, 2020.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *CVPR*, pp. 9285–9295, 2022.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pp. 16000–16009, 2022.

Shuting He, Hao Luo, Pichao Wang, Fan Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. In *ICCV*, pp. 15013–15022, 2021.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *ECCV*, pp. 437–452, 2018.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pp. 831–839, 2019.

Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. In *CVPR*, pp. 3957–3966, 2021.

Menelaos Kanakis, David Bruggemann, Suman Saha, Stamatios Georgoulis, Anton Obukhov, and Luc Van Gool. Reparameterizing convolutions for incremental multi-task learning without task interference. In *ECCV*, pp. 689–707. Springer, 2020.

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. *PNAS*, 114(13):3521–3526, 2017.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017.

Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 12245–12254, 2020.

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 *ICCV*, pp. 10012–10022, 2021.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *ECCV*, pp. 67–82, 2018.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Yichen Qian, Ming Lin, Xiuyu Sun, Zhiyu Tan, and Rong Jin. Entroformer: A transformer-based entropy model for learned image compression. *ICLR*, 2022.

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

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2): 123–146, 1995.

Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. In *CVPR*, pp. 1591–1600, 2021.

Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *ECCV*, pp. 254–270. Springer, 2020a.

Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *CVPR*, pp. 12183–12192, 2020b.

Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pp. 10347–10357. PMLR, 2021a.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, pp. 32–42, 2021b.

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

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.

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

Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *NIPS*, 34:30392–30400, 2021.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, pp. 3014–3023, 2021.

Pei Yu, Yinpeng Chen, Ying Jin, and Zicheng Liu. Improving vision transformers for incremental learning. *arXiv preprint arXiv:2112.06103*, 2021.

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

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018.

Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, pp. 6881–6890, 2021.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *ICLR*, 2020.