Brain-Inspired Fast- and Slow-Update Prompt Tuning for Few-Shot Class-Incremental Learning

Hang Ran, Xingyu Gao, *Member, IEEE*, Lusi Li, *Member, IEEE*, Weijun Li, *Senior Member, IEEE*, Songsong Tian, Gang Wang, Hailong Shi, and Xin Ning, *Senior Member, IEEE*

***Abstract*— Few-shot class-incremental learning (FSCIL) aims to learn new classes incrementally with a limited number of samples per class. Foundation models combined with prompt tuning showcase robust generalization and zero-shot learning (ZSL) capabilities, endowing them with potential advantages in transfer capabilities for FSCIL. However, existing prompt tuning methods excel in optimizing for stationary datasets, diverging from the inherent sequential nature in the FSCIL paradigm. To address this issue, taking inspiration from the "fast and slow mechanism" of the complementary learning systems (CLSs) in the brain, we present fast- and slow-update prompt tuning FSCIL (FSPT-FSCIL), a brain-inspired prompt tuning method for transferring foundation models to the FSCIL task. We categorize the prompts into two groups: fast-update prompts and slow-update prompts, which are interactively trained through meta-learning. Fast-update prompts aim to learn new knowledge within a limited number of iterations, while slow-update prompts serve as meta-knowledge and aim to strike a balance between rapid learning and avoiding catastrophic forgetting. Through experiments on multiple benchmark tests, we demonstrate the effectiveness and superiority of FSPT-FSCIL. The code is available at https://github.com/qihangran/FSPT-FSCIL.**

***Index Terms*— Brain-inspired, few-shot class-incremental learning (FSCIL), foundation models, meta-learning, prompt tuning.**

Manuscript received 26 November 2023; revised 7 May 2024 and 18 July 2024; accepted 28 August 2024. This work was supported in part by the Science and Technology Innovation (STI) 2030—Major Projects under Grant 2022ZD0208700, in part by the National Natural Science Foundation of China under Grant 62376264 and Grant 62373343, in part by Beijing Natural Science Foundation under Grant L233036, in part by Ningbo Key Research and Development Program under Grant 2023Z231, in part by Zhejiang Province Postdoctoral Research Funding Project under Grant ZJ2023008, and in part by Ningbo Natural Science Foundation under Grant 2023J280. *(Corresponding authors: Xingyu Gao; Lusi Li; Weijun Li; Xin Ning.)*

Hang Ran is with the Institute of Semiconductors, Chinese Academy of Sciences, Beijing 100083, China, and also with the Institute of Microelectronics, Chinese Academy of Sciences, Beijing 100029, China (e-mail: ranhang@semi.ac.cn).

Xingyu Gao and Hailong Shi are with the Institute of Microelectronics, Chinese Academy of Sciences, Beijing 100029, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: gxy9910@gmail.com; shihailong2010@gmail.com).

Lusi Li is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23529 USA (e-mail: lusili@cs.odu.edu).

Weijun Li, Songsong Tian, and Xin Ning are with the Institute of Semiconductors, Chinese Academy of Sciences, Beijing 100083, China (e-mail: wjli@semi.ac.cn; tiansongsong@semi.ac.cn; ningxin@semi.ac.cn).

Gang Wang is with the School of Computing and Data Engineering, NingboTech University, Ningbo 315100, China, and also with the Department of Bioengineering, Imperial College London, SW7 2AZ London, U.K. (e-mail: wanggangnit@nit.zju.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2024.3454237

I. Introduction

In recent years, deep learning (DL) models are capable of surpassing humans in image recognition tasks within closed environments, where the data used for both training and testing are predefined and static [1]. However, real-world applications often operate in open environments where data are not predetermined but increase over time, posing a challenge for traditional DL models that cannot adapt incrementally. To address this issue, the incremental learning (IL) paradigm has emerged in [2], [3], [4], and [5]. In addition to the challenges that arise in open environments, obtaining sufficient training samples is also a significant hurdle for many tasks. Few-shot learning (FSL) [6], [7], [8] aims to address this issue. To simultaneously address the requirements of both IL and FSL, a new learning paradigm called few-shot class-IL (FSCIL) has been formally introduced in [9]. FSCIL organizes the dataset into a base session and multiple incremental sessions. Each IL session comprises a limited number of classes, with only a few samples per class.

FSCIL confronts two major challenges: catastrophic forgetting [10] of previously learned classes and overfitting on the few-shot new classes. Current FSCIL methods [11], [12], [13], [14], [15], [16], [17], [18] primarily draw inspiration from FSL and IL. They use small-scale models (such as ResNet-18) as the backbone. Despite adopting strategies such as self-supervision learning [19] and class augmentation [15], [20] to enhance transferability, the limitations of model size and the lack of initialization with models pretrained on large-scale datasets restrict further improvement in FSCIL performance. From corresponding studies (see [13], [14], [15], [16], [17], [21], and [22]), we observe a discernible trend where the performance of the state-of-the-art (SOTA) model tends to decline rapidly as the number of incremental stages increases. This prompts us to inquire: Can we harness the power of pretrained foundation models to facilitate FSCIL, and if so, what strategies should we use?

The contrastive language-image pretraining (CLIP) model [23], which has been trained on a mammoth dataset of 400 million internet-sourced image–text pairs, has exhibited remarkable robustness in generalization and zero-shot learning (ZSL) capabilities. Given its formidable ZSL proficiency, we are inclined to believe that CLIP harbors significant potential for FSCIL applications. Hence, our subsequent exploration revolves around the effective integration of CLIP into the FSCIL framework. However, there are two pivotal

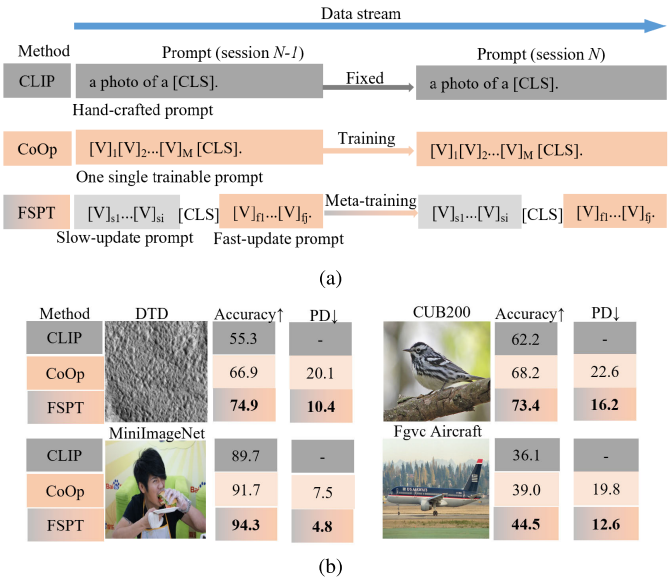


Fig. 1. Prompt engineering (CLIP) versus CoOp versus our fast- and slow-update prompt tuning (FSPT). (a) Visual representation illustrating the differences among the three methods. (b) Average classification ACC and performance dropping (PD) rate on four FSCIL datasets, with CLIP demonstrating ZSL precision.

considerations to address. First, the prompts of CLIP are fixed and difficult to adapt to downstream task data that differ significantly from the pretraining data [23]. Therefore, it is necessary to turn the fixed prompts into learnable prompts, a process known as prompt tuning. To address this issue, Zhou et al. [24] propose a prompt tuning method called context optimization (CoOp) to automatically generate CLIP prompts. However, current prompt tuning methods focus on individual, closed, and nonsequential datasets, which does not align with the inherent sequential nature of the FSCIL paradigm. Under the FSCIL paradigm, changes in learnable prompts during the learning process can lead to rapid forgetting of old classes, resulting in degraded model performance [see Fig. 1(b) for the performance of CoOp]. This is the second and most important challenge in applying CLIP to FSCIL. Therefore, the key issue we aim to tackle is how to design CLIP prompts for FSCIL and how to train them. We resort to biological mechanisms of FSCIL.

As highlighted in [25] and [26], the biological mechanism governing few-shot IL in the human brain is referred to as the "fast and slow mechanism." This mechanism involves the hippocampus and neocortex and plays a crucial role in complementary learning systems (CLSs) [27], [28]. It involves two distinct regulatory systems for the plasticity changes in synaptic connection strength within the brain: fast synaptic plasticity and slow synaptic plasticity. Fast synaptic plasticity allows adaptive adjustments between neurons in a short timeframe, while slow synaptic plasticity entails gradual changes in synaptic connection strength over a longer temporal scale. The interplay between these fast and slow synaptic plasticity mechanisms jointly underpins the brain's learning and memory processes.

Inspired by the aforementioned mechanism, we introduce fast- and slow-update prompt tuning for FSCIL (FSPT-FSCIL), as an effective method for transferring a pretrained foundation model with frozen parameters to the FSCIL task through a novel prompt tuning strategy. Specifically, as shown in Fig. 1(a), we truncate the CLIP prompts into two parts and update them at different rates, with slower updates for the pretruncation text (slow prompts) and faster updates for the posttruncation text (fast prompts), which correspond to slow synaptic plasticity and fast synaptic plasticity, respectively. To train these prompts effectively, we use a meta-learning scheme. Fast-update prompts undergo training within the inner loop of the meta-learning process, enabling the acquisition of new knowledge within a limited number of iterations. Slow-update prompts serve as meta-knowledge and are updated in the outer loop, harmonizing the equilibrium between the need for rapid learning and the risk of catastrophic forgetting.

Our proposed brain-inspired fast and slow prompts are neither completely fixed like CLIP nor fully learnable like CoOp. Instead, they are a combination of two sets of prompts with different update speeds. The benefits of fast and slow prompts in FSCIL can be summarized as follows.

1) It directly uses the streamlined framework of CLIP for inference, without the need for additional modules. This simplicity yields an effective method that scales with the performance of the underlying CLIP model.
2) Through meta-learning with FSCIL as the meta-objective, the slow prompts enable the model to incrementally and rapidly adapt to new classes with few samples without forgetting old knowledge.

Consequently, the rapid updates of the fast prompts, based on the well-learned slow prompts, could guide the model to quickly learn new classes while retaining old knowledge. FSPT-FSCIL undergoes training and testing across various benchmark datasets. When benchmarked against SOTA FSCIL methods, FSPT-FSCIL remarkably surpasses all alternative approaches by a substantial margin. Furthermore, in comparison to current prompt tuning methods, FSPT-FSCIL maintains its superior performance. The main contributions can be summarized as follows.

1) We undertake a pioneering study exploring the adaptation of the CLIP model to FSCIL, demonstrating its efficacy and identifying its limitations in this context for the first time.
2) To enhance the generalization of CLIP prompts in FSCIL tasks, we present a simple yet effective brain-inspired fast- and slow-update prompt tuning strategy.
3) Through extensive experimentation, we demonstrate that the proposed method shows scalability and outperforms both SOTA methods that rely on pretrained models and those that do not.

The structure of this article is outlined as follows. In Section II, we give a brief overview of the related work. In Section III, we describe the definition of FSCIL and then present a detailed formulation of the FSPT-FSCIL. In Section IV, the datasets for training and evaluation, implementation details, experimental results, and ablation studies are presented. Finally, the conclusions are summarized in Section V.

II. Related Work

Our approach involves prompt tuning and brain-inspired strategies. In this section, we describe the recent FSCIL methods, prompt tuning methods, and brain-inspired FSCIL methods.

A. FSCIL

The standard paradigm for FSCIL is mainly derived from [9]. To tackle catastrophic forgetting, researchers have proposed methods involving the storage or generation of limited amounts of old-class data, known as replay-based methods [11], [12], [29], [30]. Knowledge distillation is widely used for them. For instance, Zhao et al. [31] propose a class-aware bilateral distillation (CABD) structure that addresses the issue of overfitting and catastrophic forgetting simultaneously. However, replay-based methods may not be viable in scenarios with stringent data privacy requirements.

For nonreplay-based methods, the majority heavily rely on enhancing the transferability of the feature backbone trained on data from the first session. This enhancement is achieved through various techniques such as data augmentation [19], class augmentation [20], [32], [33], increasing the variance of intraclass representations [15], and bolstering feature expression capabilities via scalable networks [34]. In addition, some studies aim to improve classifier performance through statistical classifiers [19] or by increasing interclass margins [13], [14], [15], [35], [36]. Furthermore, some studies [16], [17], [37], [38] integrate meta-learning concepts into FSCIL, enhancing future adaptability to real FSCIL scenarios by creating pseudo FSCIL learning scenarios during the base class phase. Some methods apply parameter regularization. For instance, the F2M [39] overcomes catastrophic forgetting by finding flat minima. The WaRP [40] transforms the original parameter space into a new space, compactly compressing most of the old knowledge into a few crucial parameters, thus providing additional room for the model to effectively learn new classes. The LDC [41] presents a unified framework for both memorizing old class distributions and estimating new class distributions with limited training samples. For a more comprehensive review, refer to the survey paper [42].

The aforementioned methods apply small-scale models (such as ResNet18) as the backbone, limiting the model's expressive capacity. Recently, there has been a gradual shift toward using large-scale pretrained models in FSCIL. Qiu et al. [43] develop a semantic-visual guided transformer (SV-T) to enhance the feature extracting capacity of the pretrained feature backbone on incremental classes. Goswami et al. [44] calibrate higher order statistics for FSCIL with pretrained ViT models. Park et al. [45] introduce PriViLege, based on ViT, effectively tackling the challenges of catastrophic forgetting and overfitting in large-scale models through pretrained knowledge tuning (PKT). These methods are all based on vision transformer and still require the design of strategies such as knowledge distillation to overcome the challenges of FSCIL. The CLIP framework we adopt is simple yet effective and has been proven to have excellent ZSL capabilities, with enormous potential for applications in FSCIL. Our work focuses on the design of text prompts for CLIP.

B. Prompt Tuning for FSCIL

The concept of prompts originated from the field of natural language processing (NLP) [46], [47], aiming to transfer pretrained language models to various downstream tasks. Prompts reformat downstream data into the pretraining data format, using the knowledge acquired from large-scale pretraining, ensuring that the frozen-parameter pretrained model can comprehend and execute specified tasks. Inspired by NLP prompts, many visual-related prompt methods [23], [24], [48], [48], [49], [50] have also been proposed. In the realm of visual tasks, two primary types of prompts emerge: textual prompts and visual prompts. For textual prompts, the CLIP [23] provides prompts similar to "a photo of [cls]." The CoOp [24] and CoCoOp [51] are introduced to automatically learn textual prompts, resulting in improved performance compared with CLIP. In the case of visual prompts such as VP [48] and VPT [50], the exploration mainly involves injecting noise directly onto input images as prompts, exhibiting promising performance on certain datasets. In the field of continual learning, the L2P [48] and DualPrompt [49] represent prompt tuning methods for CIL. Tian et al. [52] propose PL-FSCIL, a prompt tuning approach based on ViT for FSCIL. To the best of our knowledge, there are currently no directly CLIP prompt-based methods for FSCIL. We are pioneering the application of this approach to the FSCIL paradigm.

C. Brain-Inspired FSCIL

In brain-inspired IL methods, CLSs play a crucial role. IL framework FearNet [53] comprises a short-term memory network and a long-term storage network inspired by CLS. Arani et al. [54] propose a dual-memory experience replay method called CLS-ER, which maintains short-term and long-term semantic memories. Sarfraz et al. [55] combine CLS-ER with "synaptic consolidation" by tracking the importance of parameters in the training trajectory and anchoring them to consolidation parameters in semantic memory. Pham et al. [56] focus on how to represent short-term and long-term memories and propose an IL framework called "DualNet."

There is currently limited research on brain-inspired FSCIL. Wang et al. [57] apply a brain-inspired two-step consolidation strategy for FSCIL without forgetting while improving generalization without overfitting. Zhao et al. [58] propose a learning strategy SvF based on the fast and slow learning mechanism to maintain the existing knowledge and quickly learn new knowledge. The core of our approach is also based on CLS and fast–slow learning mechanisms, but the difference lies in that we do not propose additional network modules. Instead, we design prompt structures suitable for FSCIL and prompt tuning methods based on brain-inspired approaches within a multimodal pretraining framework.

III. Problem Formulation

We first introduce the FSCIL learning paradigm and then present the FSPL-FSCIL algorithm.

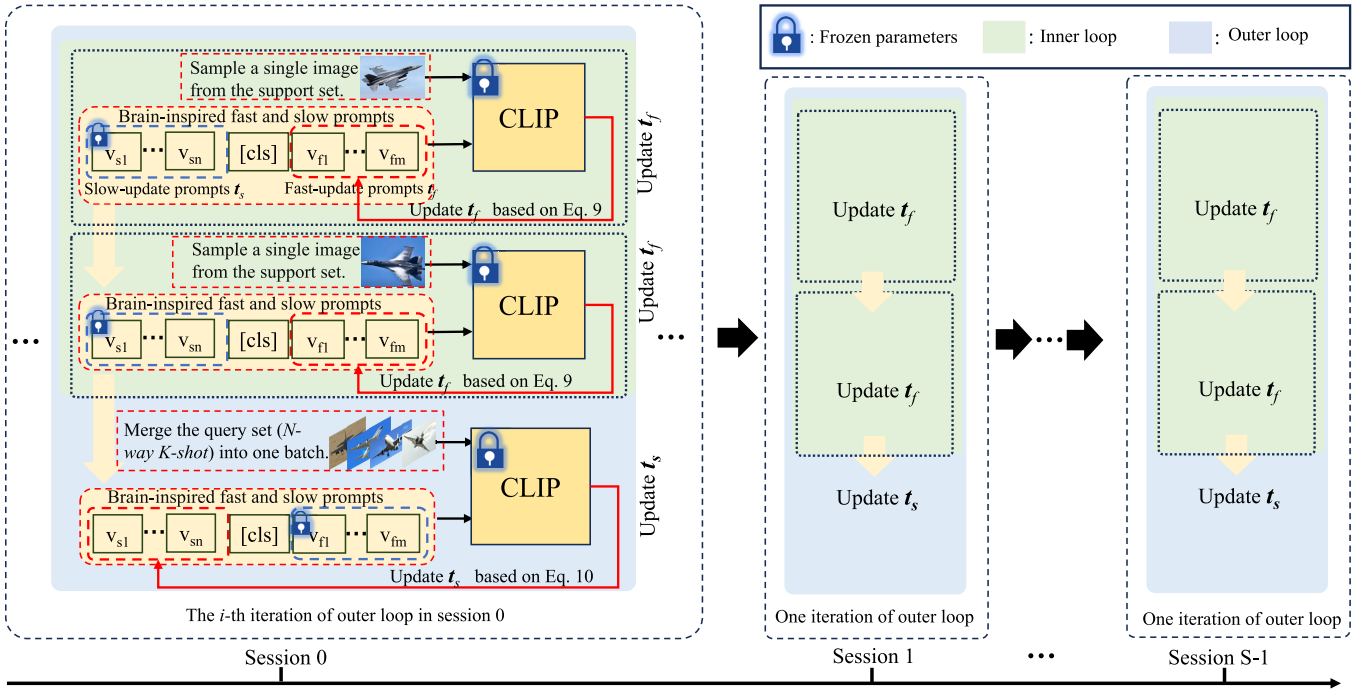


Fig. 2. Illustration of FSPT-FSCIL. This diagram showcases the complete IL process. During the base session, there are N iterations (outer loop) in total. In each iteration (outer loop), the slow prompts are fixed, and the fast prompts are iteratively updated multiple times (i.e., the inner loop operation, with a loop count of 2 in this case) on the support set using the objective function shown in (9). After the inner loop ends, the fast prompts are fixed, and the slow prompts are updated in a single step on the query set using the objective function shown in (10). The outer loop iterates N times in the base session, and each incremental session follows the same learning strategy but only performs one iteration of the outer loop.

A. FSCIL Definition

FSCIL partitions a dataset into a base session with a relatively larger number of classes and data, and incremental sessions, each involving only a few classes with a few samples. These sessions are independently fed to the learning algorithm in a sequential manner. We assume a total of S sessions, with the training dataset for the sth session denoted as $D^{(s)}_{\text{train}} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{|D^{(s)}_{\text{train}}|}$, the testing dataset denoted as $D^{(s)}_{\text{test}} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{|D^{(s)}_{\text{test}}|}$, and the class space denoted as $C^{(s)}$, where the samples for the sth session are drawn from the distribution $P^{(s)}$, where $s = 0, 1, \ldots, S-1$. The learning paradigm for each incremental session with N classes and K samples per class is called N-way K-shot FSCIL. FSCIL requires that the intersection of class spaces between sessions is empty, that is $C^{(s)} \cap C^{(s')} = \emptyset$ for any $s \neq s'$, where $s' = 0, 1, \ldots, S-1$. During training in session s, the model can only access the dataset $D^{(s)}_{\text{train}}$ and cannot access $D^{(s')}_{\text{train}}$ for any $s' < s$. During the testing phase, the model evaluates test samples from all the encountered classes $\{D^{(0)}_{\text{test}}, \ldots, D^{(s)}_{\text{test}}\}$, without knowledge of the session from which the test samples originate. The objective of an FSCIL algorithm is to fit a model $\mathcal{F}(\cdot)$ to minimize the following expected risk:

$$\mathbb{E}_{(\boldsymbol{x}_i, y_i)\sim P^{(0)}\cup\ldots P^{(S-1)}}\left[\mathcal{L}(\mathcal{F}(\boldsymbol{x}_i;\boldsymbol{\theta}), y_i)\right] \tag{1}$$

where $\mathcal{L}(\cdot,\cdot)$ represents the loss function. In practical processes, algorithms approximate the minimization of expected risk by minimizing empirical loss on training data,

Fig. 3. Intuitive interpretation of prompt tuning. CLIP may encounter miss-alignment issues when evaluating on new downstream data. Prompt tuning for text input can address this misalignment problem.

as illustrated below

$$\boldsymbol{\theta}^* = \arg\min_{\theta} \sum_{s=0}^{S-1} \sum_{i=0}^{\left|D^{(s)}_{\text{train}}\right|} \mathcal{L}(\mathcal{F}(\boldsymbol{x}_i;\boldsymbol{\theta}), y_i). \tag{2}$$

However, within the FSCIL paradigm, this straightforward approximation without additional strategies presents two key challenges.

1) The model parameters obtained through minimizing empirical risk in session s are unable to adapt to tasks in session s' for any $s' < s$, leading to catastrophic forgetting.
2) The few-shot samples in the incremental sessions are challenging to describe the overall distribution $P^{(s)}$, resulting in severe overfitting.

B. FSPL-FSCIL Architecture

1) CLIP: The proposed FSCIL architecture is illustrated in Fig. 2. At the backbone level, we use the pretrained visual-language multimodal model CLIP, which has been

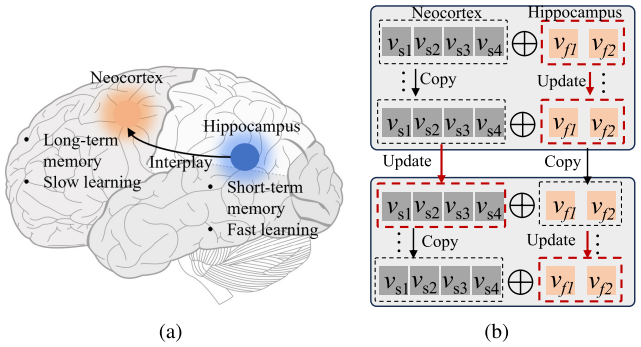


Fig. 4. (a) CLS and (b) fast- and slow-update prompts inspired by the CLS.

pretrained on the WebImageText (WIT) dataset. CLIP consists of an image encoder $f(\cdot)$ and a text encoder $g(\cdot)$. The image encoder can take the form of a Vision Transformer (ViT), and the text encoder is built on top of a transformer and aims to generate text representations from natural language.

Specifically, given a sequence of words (tokens/prompts), such as "a photo of a cat," CLIP first converts each token into a lower cased byte pair encoding (BPE) representation [59], which is essentially a unique numeric ID. The vocabulary size in CLIP is 49 152. Each text sequence is enclosed with [SOS] and [EOS] tokens and capped at a fixed length of 77 tokens. After that, these IDs are mapped to 512-D word embedding vectors, which are then passed to the transformer. Finally, the features at the [EOS] token position are layer normalized and further processed by a linear projection layer. During ZSL inference, CLIP first maps an image $\boldsymbol{x}$ and prompts $\boldsymbol{p}$ through their respective frozen encoders into a shared feature space. It calculates the probabilities of various labels corresponding to the input image using the following formula:

$$p(y=i|\boldsymbol{x}) = \frac{\exp\big(\cos(g(\boldsymbol{p}_i), f(\boldsymbol{x})/\tau)\big)}{\sum_{j=1}^{K}\exp\big(\cos(g(\boldsymbol{p}_j), f(\boldsymbol{x})/\tau)\big)} \tag{3}$$

where τ is a temperature parameter learned by CLIP, K is the number of classes, and p_i is the text prompts with token of the ith class, which could have the form of "a photo of a class$_i$." The classification result is determined by selecting the index associated with the maximum value.

CLIP's manual prompt design often fails to generalize across diverse datasets, causing misalignment when directly applied to new datasets, as seen in Fig. 3. To solve this, we transform the prompts into a learnable format. Given that the existing prompt tuning methods are primarily designed for closed tasks and struggle to handle continuous input, we introduce a brain-inspired approach.

2) Brain-Inspired Fast- and Slow-Update Prompts: The CLS is a theory of human learning and memory in the brain. As shown in Fig. 4(a), it suggests that the hippocampus is capable of rapidly acquiring knowledge, which is then consolidated and deepened in memory through slower learning processes in the neocortical system. It is one of the mechanisms in the human brain that enables efficient continuous learning.

Drawing from the concept of fast and slow learning mechanisms, we devise two distinct sets of prompts for CLIP: fast-update prompts and slow-update prompts. As illustrated in Fig. 4(b), we first initialize the sequence of words (tokens/prompts) in CLIP with N "X" characters, forming a prompt $\boldsymbol{p}$ denoted as "X X X X ... X." Subsequently, we partition $\boldsymbol{p}$ into two segments, $\boldsymbol{t}_s$ and $\boldsymbol{t}_f$. Specifically, the first M tokens of $\boldsymbol{p}$ serve as slow prompts, while the last N tokens serve as fast prompts.

During prompt tuning process, $\boldsymbol{t}_f$ is updated at a relatively higher frequency, hence termed as fast-update prompts, which model the synaptic updates in the hippocampus. Following tokenization and embedding operations, the representation of the fast prompts is as follows:

$$\boldsymbol{t}_f = [\boldsymbol{v}]_{f1}, [\boldsymbol{v}]_{f2}, \ldots, [\boldsymbol{v}]_{fm} \tag{4}$$

where each $\boldsymbol{v}$ is an embedding vector, with the same dimensionality as the word embedding of CLIP. m is a hyperparameter that refers to the number of prompt tokens to be quickly updated. Simultaneously, $\boldsymbol{t}_s$ is updated at a relatively lower frequency, termed as slow-update prompts, which model the synaptic updates in the neocortex. After tokenization and embedding operations, the representation of the slow prompts is as follows:

$$\boldsymbol{t}_s = [\boldsymbol{v}]_{s1}, [\boldsymbol{v}]_{s2}, \ldots, [\boldsymbol{v}]_{sn} \tag{5}$$

where n is a hyperparameter that refers to the number of prompt tokens to be slowly updated. We design $\boldsymbol{t}_s$ as the function of the new cortex, responsible for storing long-term memory, while $\boldsymbol{t}_f$ assumes the role of the hippocampus, enabling rapid learning of new samples. Finally, we concatenate the fast and slow prompts together and insert the class token into them to obtain the final text prompt p as follows:

$$\boldsymbol{p}_i = [\boldsymbol{v}]_{s1}, [\boldsymbol{v}]_{s2}, \ldots, [\boldsymbol{v}]_{sn}, [\text{cls}_i], [\boldsymbol{v}]_{f1}, [\boldsymbol{v}]_{f2}, \ldots, [\boldsymbol{v}]_{fm} \tag{6}$$

where $[\text{cls}_i]$ refers to the word embedding vector corresponding to the name of class i. Prompts $\boldsymbol{t}_s$ and $\boldsymbol{t}_f$ are the parameters to be learned in FSPT-FSCIL, and the ideal objective function is expressed as

$$\boldsymbol{p}^* = \arg\min_{t_f, t_s} \sum_{t=0}^{S-1} \sum_{i=0}^{\left|D_{\text{train}}^{(t)}\right|} \mathcal{L}\big(f(\boldsymbol{x}_i), g\big([\boldsymbol{t}_s, \text{cls}_i, \boldsymbol{t}_f]\big)\big). \tag{7}$$

This function aims to find the fast prompts and slow prompts that minimize the overall loss during session 0 to S-1, where the loss function $\mathcal{L}(\cdot,\cdot)$ is defined as

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{p}_i) = -\log\left(\frac{\exp\big(\cos(g(\boldsymbol{p}_i), f(\boldsymbol{x})/\tau)\big)}{\sum_{j=1}^{K}\exp\big(\cos(g(\boldsymbol{p}_j), f(\boldsymbol{x})/\tau)\big)}\right) \tag{8}$$

where τ is a temperature parameter learned by CLIP, K is the number of classes, p_i is the text prompts with token of the ith class, and p_j is the text prompts with token of the jth class. Once the prompts are determined, we can calculate the probabilities of various labels corresponding to the input image using (3).

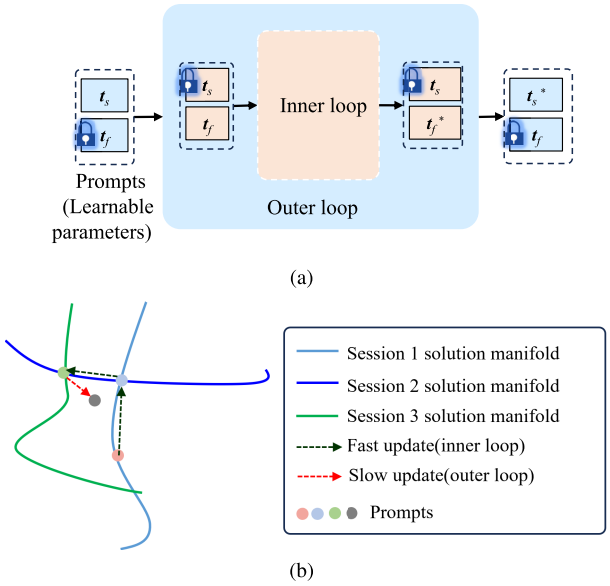


Fig. 5. (a) Meta-learning for prompts and (b) toy example of fast- and slow-update prompt-tuning. FSPT-FSCIL rapidly learns prompts for new classes within the inner loop through meta-learning, and then, in the outer loop, it learns from the slow prompts acting as meta-knowledge to balance knowledge between new and old classes.

C. Meta-Learning for FSPT-FSCIL

As shown in (7), $\boldsymbol{t}_f$ and $\boldsymbol{t}_s$ are the parameters to be learned in the proposed framework, and the objective of this section is to develop a training algorithm to implement (7). A common method for such objective functions is to alternately perform two convex optimization steps: the first is to optimize $\boldsymbol{t}_s$ while fixing $\boldsymbol{t}_f$, and the second is to optimize $\boldsymbol{t}_f$ while fixing $\boldsymbol{t}_s$. These two steps are repeated alternately until convergence. As shown in Fig. 5(a), our plan is to update the prompts through optimization-based meta-learning, with the primary motivations encompassing the following two aspects.

1) *Meta-Learning With Inner and Outer Loops:* The outer loop consists of several inner loops, and this structure is well-suited for simulating a mechanism for fast and slow learning. Within the inner and outer loops, we intend to update fast and slow prompts separately, replaying inner loop samples to the outer loop to achieve memory consolidation.
2) *Learn to Learn:* Meta-learning is perceived as the process of learning how to learn. Through meta-learning, our aim is to acquire meta-knowledge with the capability of generalization, allowing proficient performance in future real-world incremental categories.

Meta-learning consists of two phases: meta-training and meta-testing. We transform the training of base classes into the meta-training phase and the IL phase into the meta-testing phase. We serve the slow-update prompts as the meta-knowledge. Ideally, as shown in Fig. 5(b), meta-knowledge should possess the following attributes: under the influence of meta-knowledge, the combination of slow and fast prompts forms a prompts solution. During IL (meta-testing), it should quickly reach the solution manifold for new classes while remaining close to the solution manifold for old classes. In other words, we aim to acquire the above meta-knowledge during meta-training on base classes to achieve nonforgettable rapid learning during the incremental phase. The meta-learning for fast and slow prompts aims to achieve a better balance between rapid adaptation and nonforgetting, which is the key issue in IL.

1) Meta-Training: We train the prompts based on the meta-learning algorithm, as illustrated in Algorithm 1. This algorithm comprises both inner loop learning and outer loop learning. One update for slow prompts in the outer loop corresponds to multiple updates for fast prompts in the inner loop. Specifically, we first organize the base class data $D_{\text{train}}^{(0)}$ into a task set $\{T\}$ in the N-way K-shot format. We then randomly select a task T_i from $\{T\}$ for inner loop training. Task T_i contains a total of $N \times K$ samples, and we perform online parameter iterations for each sample, resulting in $N \times K$ steps in the inner loop. In practice, we use N-way one-shot for inner loop updates, but it is also possible to randomly sample k samples from T_i and fix the number of steps to k. For each updating step in the inner loop, we calculate gradients for the fast prompts $\boldsymbol{t}_f$ using only one sample and update it with a relatively large learning rate l_i. During the inner loop, we keep the slow-update prompts fixed, thus transforming the optimization objective for the inner loop into

$$\boldsymbol{t}_f^* = \arg\min_{t_f} \mathcal{L}\big(f(\boldsymbol{x}_i), g\big([\boldsymbol{t}_s{}^*, \text{cls}_i, \boldsymbol{t}_f]\big)\big). \tag{9}$$

After completing the inner loop, we sample another task T_j from $\{T\}$, and then feed the entire T_j into the model. We calculate gradients for the slow prompts $\boldsymbol{t}_s$ using (8) and update it with a lower learning rate l_o. During the outer loop, we have the option to either fix the fast prompts or leave them unfixed. To simplify the computation of second-order gradients, we choose to fix the fast prompts and only update the slow prompts. Therefore, the objective function for the outer loop becomes

$$\boldsymbol{t}_s^* = \arg\min_{t_s} \sum_{i=0}^{|T_j|} \mathcal{L}\big(f(\boldsymbol{x}_i), g\big([\boldsymbol{t}_s, \text{cls}_i, \boldsymbol{t}_f{}^*]\big)\big). \tag{10}$$

2) Meta-Testing: After meta-training on the base classes, meta-knowledge is endowed with the ability to guide the movement of prompts within the prompt space during the IL process. This enables them to rapidly converge toward the solution space of the few-shot new classes while mitigating their deviation from the solution space of the old classes, as depicted in Fig. 5.

Finally, we treat the IL as a meta-testing and, after each learning process, use the current prompts $\boldsymbol{p}^{(s)}$ to evaluate test samples for all previously encountered classes. As shown in Algorithm 2, In each incremental session, we use the same inner and outer loop settings as in the base session for training. We perform a single outer update for each incremental session due to the limited number of classes and samples. We record the classification accuracy (ACC) achieved at each session and compare it with other methods.

Algorithm 1 Meta-Training for Prompts

Input: Training dataset $D_{train}^{(0)}$; Fixed CLIP encoders $f(\cdot)$ and $g(\cdot)$
Output: Fast prompts $\boldsymbol{t}_f^*$ and slow prompts $\boldsymbol{t}_s^*$

Randomly initialize fast prompts $\boldsymbol{t}_f$
Create a task set $\{T\}$ for $D_{train}^{(0)}$ as an *N-way K-shot* task collection
Randomly initialize $\boldsymbol{t}_s$
$\boldsymbol{t}_s^* = \boldsymbol{t}_s$
while not converged **do** ▷ Outer loop
  $T_i \leftarrow$Sample a task from $\{T\}$
  for $j = 1, 2, \ldots, k$ **do** ▷ Inner loop
    $(\boldsymbol{x}_j, cls_j) = T_i[j]$
    $\boldsymbol{p}_j = [\boldsymbol{t}_s{}^*, cls_j, \boldsymbol{t}_f{}^{(j-1)}]$
    $\boldsymbol{t}_f^{(j)} \leftarrow$ One step updating for $\boldsymbol{t}_s$ using (9)
  end for
  $\boldsymbol{t}_f^* = \boldsymbol{t}_f^{(k)}$
  $\boldsymbol{t}_s = \boldsymbol{t}_s^*$
  $T_j \leftarrow$Sample a task from $\{T\}$
  $X_q = T_j[:, 0], CLS_q = T_j[:, 1]$
  $\boldsymbol{p} = [\boldsymbol{t}_s, CLS_q, \boldsymbol{t}_f^*]$
  $\boldsymbol{t}_s^* \leftarrow$ Update $\boldsymbol{t}_s$ using (10)
end while

Algorithm 2 Meta-Testing

Input: Training dataset $\{D_{train}^{(0)}, D_{train}^{(1)}, \ldots, D_{train}^{(S-1)}\}$; Testing dataset $\{D_{test}^{(0)}, D_{test}^{(1)}, \ldots, D_{test}^{(S-1)}\}$; Fixed CLIP encoders $f(\cdot)$ and $g(\cdot)$
Output: Final fast prompts $\boldsymbol{t}_f$; Final slow prompts $\boldsymbol{t}_s$; Classification results $\{R^{(0)}, R^{(1)}, \ldots, R^{(S-1)}\}$

for $s \leftarrow 1, \ldots S-1$ **do**
  Create a task set $\{T\}$ for $D_{train}^{(s)}$ as an *N-way K-shot* task collection
  Updating $\boldsymbol{t}_f$ and $\boldsymbol{t}_s$ based on Algorithm 1
  $p(y|D_{test}^{(s)}[:, 0]) \leftarrow$Calculate the prediction results using (3)
  $R^{(s)} \leftarrow$Compute the classification ACC using $p(y|D_{test}^{(s)}[:, 0])$ and $D_{test}^{(s)}[:, 1]$
end for

IV. Experiments

We first describe the benchmark datasets used for FSPT-FSCIL, the baselines for comparison, and the implementation details. Then, FSPT-FSCIL is compared with the baselines. Finally, we conduct ablation studies.

A. Datasets

Traditional FSCIL benchmark datasets encompass three main ones: CIFAR100 [60], miniImageNet [61], and CUB200 [62]. We will continue our evaluation on these datasets while using a standardized task split. In addition, leveraging a visual foundational model pretrained on a large-scale upstream dataset, we introduce two distinct datasets FGVC Aircrafer [63] and DTD [64], which do not overlap with the pretraining data of CLIP. The task splits are as follows.

1) *CIFAR-100 and miniImageNet:* The two datasets contain a total of 100 classes, and 60 classes are allocated for base class phase training. Each incremental phase adopts a five-way five-shot configuration, yielding eight incremental phases.
2) *CUB-200:* This dataset consists of 200 classes, with 100 classes designated for base class phase training. Incremental phases are structured with a ten-way five-shot setup, leading to ten incremental phases.
3) *FGVC Aircraft:* It is a dataset for aircraft model recognition, covering 100 classes, and 60 of them are used as base classes. Incremental phases implement a five-way five-shot configuration across eight incremental phases.
4) *Describable Textures (DTD):* It is a dataset for texture-type recognition, comprising 47 classes, and 32 serve as base classes. Incremental phases are arranged in a three-way five-shot setup, resulting in five incremental phases.

B. Baselines

Our comparative analysis encompasses a range of methodologies including SOTA FSCIL methods, CLIP framework-based approaches, and prompting-based continuous methods. This diverse set of baselines will facilitate a comprehensive evaluation of the performance of our novel approach.

1) SOTA FSCIL Methods: We consider the current SOTA methods in the field of FSCIL as our primary baselines for comparison. To further enhance the fairness of the comparison, we introduce two additional baseline models. First, given the significant role and widespread application of prototypic network [65] in FSCIL, we leverage CLIP's visual encoder to extract features and construct prototypes, using them to test the categories at each stage. Second, we use CLIP's visual encoder as the backbone of the current SOTA FACT method, retraining it to obtain an enhanced version of FACT that is bolstered by CLIP's pretrained visual model, serving as another baseline, denoted as FACT w/CLIP.

2) CLIP Framework-Based Methods: To validate the core contribution of FSPT-FSCIL, we introduce two additional baselines that leverage pretraining data and foundation model for comparison. We select two CLIP framework-based classification methods for additional comparison. First, we establish ZSL using CLIP as a baseline, denoted as CLIP-ZSL, with the aim of establishing the lower bound of the CLIP model's performance on FSCIL tasks. In addition, we apply the prompt tuning-based method CoOp [24] to the FSCIL task as another baseline, denoted as CoOp-FSCIL.

3) Prompting-Based CIL Methods: To the best of our knowledge, we are pioneers in prompting-based FSCIL methods. We broaden the scope of our comparative analysis and include the prompting-based CIL method L2P [48] as a baseline for additional comparison.

C. Implementation Details

To ensure a fair comparison among different methods, we use backbone architectures with similar GFLOPs.

Specifically, we use the Vision Transformer [66] architecture, ViT-L/14, as the backbone for all methods. During the base class training phase, we conduct 10 000 outer loop steps for cifar100 and miniImageNet. For cub200, FGVCAircraft, and DTD, we execute 5000 outer loop steps. Each outer loop step comprises five inner loop steps. In other words, we train the slow-update prompt once after every five iterations of fast-update prompts. We provide a comprehensive analysis of the inner loop steps in Section IV-E4. During each incremental session (i.e., meta-testing phase), we carry out one outer loop step. Throughout the entire meta-learning process, both inner and outer loop learning rates remain constant. The outer loop learning rate is set at 1e-4, and the inner loop learning rate is set at 0.03. For CoOp, L2P, CLIP, and the enhanced version of FACT, we construct evaluation models using their publicly available code.

Regarding CLIP-ZSL, classification is performed directly using the CLIP ZSL framework [23]. For the CIL approach L2P [48], we follow the implementation details described in the corresponding paper, using the Adam optimizer with a fixed learning rate of 0.03. We retrain and evaluate its performance on FSCIL, conducting 50 epochs of training during the first session and five epochs per task within each incremental session. In addition, we validate the FSCIL performance variations across different epochs in Section IV-D3. As for CoOp-FSCIL, we set the token size to 16, identical to the total size of the fast and slow prompts in our method, initializing each token as "X." Training is conducted using SGD with a fixed learning rate of 0.002, training for 50 epochs during the first session and five epochs per task in each incremental session. Similarly, we verify the FSCIL performance across different epochs in Section IV-D3. Finally, for FACT w/ CLIP, we replace backbone of FACT [20] with CLIP's ViT-L/14 and keep it fixed. We then train the additional MLP module during the first session using a learning rate of 0.002 and the SGD algorithm and perform inference during the incremental session.

All the experiments are conducted on a dedicated NVIDIA GTX 3090 GPU. We report the Top-1 ACC of the model on all the seen class test samples after training at each session. Upon completing the final session testing, we obtain the average Top-1 ACC across all the sessions (Mean), as well as the difference in ACC between the first and last sessions, referred to as the PD rate.

D. Comparison With Baseline Methods

1) Effectiveness of the CLIP Framework in FSCIL: To validate the effectiveness of CLIP's concise framework in FSCIL, we compare CLIP framework-based methods with two baseline models. The first is a prototypic network with a pretrained CLIP visual encoder, which performs well in FSCIL. The second is the SOTA FSCIL method FACT [20] enhanced with the same CLIP visual encoder.

Fig. 6 summarizes the results. From the average performance displayed in the top-left corner, we observe that FSPT-FSCIL is a robust FSCIL learner, outperforming both prototypic networks (with a CLIP encoder) and FACT (with a CLIP encoder). It is worth noting that the ZSL ACC on the CIFAR-100 dataset surpasses the final ACC of FACT. This underscores the effectiveness of the CLIP framework in FSCIL tasks. Specifically, FSPT-FSCIL achieves leading results on three datasets, ties on another, and falls behind on the CUB200 dataset. As for why it did not secure the lead on the FGVCAircraft dataset for fine-grained aircraft model classification and lagged behind on the CUB200 dataset for fine-grained bird species classification, we surmise that these shortcomings are attributed to the inherent limitations of the pretrained CLIP text encoder, which struggles to extract effective semantic features for aircraft model names (e.g., "707-320," "727-200") and bird species names (e.g., "Eared Grebe" and "Horned Grebe"), making it challenging to align with image features. Consequently, context learned through prompt tuning may lack meaningful semantic information or even contain noise, making them difficult to generalize to incremental sessions requiring semantic fine distinctions. Thus, the conjecture is that in such datasets, the CLIP framework may not be as effective as some strategies directly based on visual features such as prototypic network. Nonetheless, our approach exhibits substantial improvements compared with the original CLIP and CoOp methods.

2) Effectiveness of the Fast- and Slow-Update Prompts: As shown in Fig. 6, FSPT-FSCIL exhibits clear advantages among all the methods based on the CLIP framework including CoOp and zero-shot CLIP. As expected, prompt tuning methods (CoOp and FSPT-FSCIL) outperform hand-crafted prompts, with CLIP ZSL serving as the lower bound for CLIP's performance in FSCIL. We also observe that prompts learned by CoOp lead to rapid forgetting in incremental sessions, resulting in a sharp decline in performance, while FSPT-FSCIL generalizes far beyond CoOp on all the datasets. This demonstrates the effectiveness of the brain-inspired fast–slow prompt structure.

3) Effectiveness of Meta-Learning: We have established the remarkable superiority of FSPT-FSCIL within the realm of CLIP framework-based methodologies. Herein, we delve deeper into assessing the merit of meta-learning for fast–slow prompts, contrasting it with conventional training methods. Specifically, we train our model using both the traditional CoOp approach and the meta-learning method, administering both for five epochs on incremental few-shot classes in each session. Postepoch, we rigorously test the model's performance to evaluate its effectiveness. The results on DTD dataset are shown in Fig. 7. We observe that meta-learning attains favorable outcomes with only a single epoch of training and sustains stability throughout multiple epochs, thus evading overfitting. Conversely, the FSCIL performance using CoOp training demonstrates considerable fluctuations across epochs and is prone to catastrophic forgetting. The experimental results reinforce the effectiveness of meta-learning in training fast–slow prompts.

4) Comparison With SOTA Methods: We first evaluate FSPT-FSCIL on miniImageNet and CIFAR100. The experimental results are summarized in Tables I and II. Our proposed FSPT-FSCIL approach significantly outperforms the top-performing traditional method, M2SD, by achieving an improvement of **25.7**% in ACC and **20.8**% in PD on

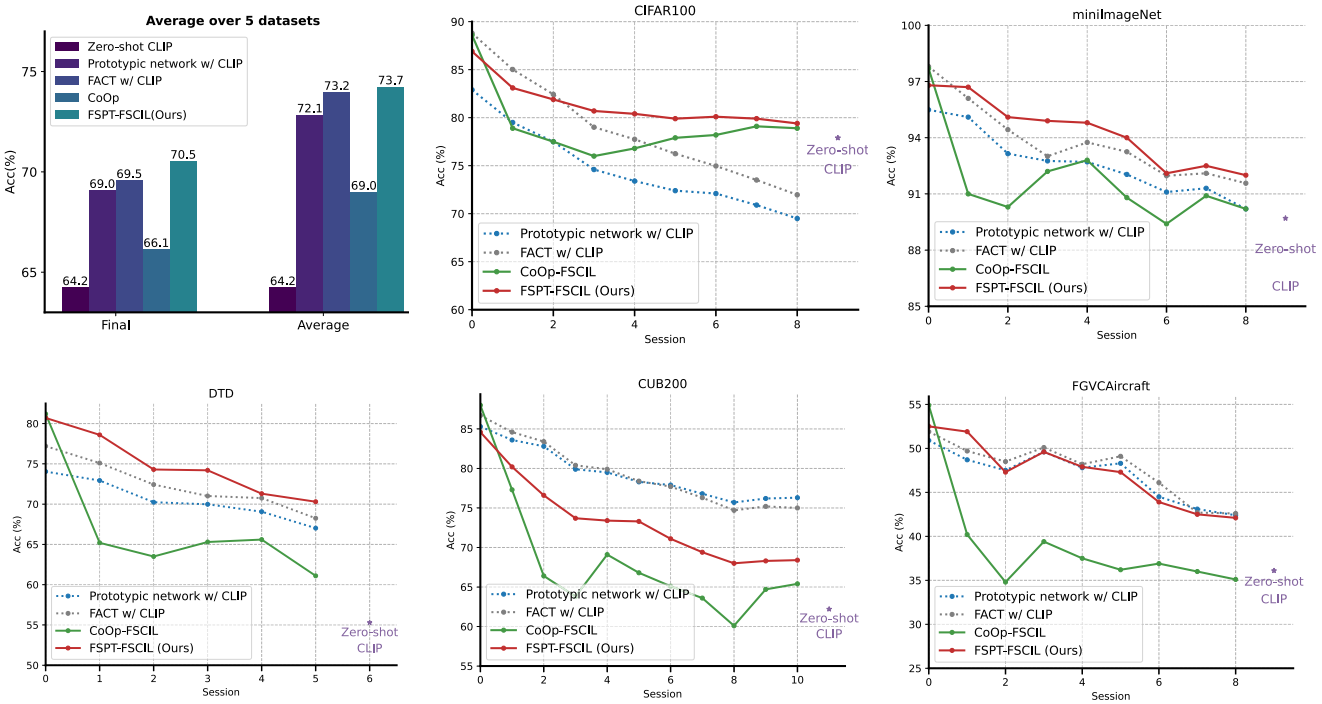


Fig. 6. Results of FSCIL on the five datasets. Overall, under the same backbone conditions, FSPT-FSCIL (red solid lines) successfully transfers CLIP's concise framework to FSCIL tasks and achieves a certain improvement compared with classical prototypic networks and the SOTA method, FACT (dashed lines). Furthermore, FSPT-FSCIL significantly enhances CLIP's ZSL (stars) and the performance of CoOp (green solid lines) in FSCIL. "Final" refers to the classification ACC for all seen classes after learning in the final session. "Average" refers to the average ACC across all sessions.

TABLE I

EVALUATION ON MINIIMAGENET. THE TOP ROWS LIST NON-PRETRAINED METHODS, WHILE THE BOTTOM ROWS LIST METHODS BASED ON PRETRAINED FOUNDATION MODELS. † INDICATES MODELS PRETRAINED ON WIT. ⋆ INDICATES MODELS PRETRAINED ON IMAGENET-21K. THE RESULTS OF L2P, FACT W/ CLIP, AND COOP-FSCIL ARE OBTAINED BY RUNNING THE PUBLICLY AVAILABLE CODE IN THE FSCIL SETTING. CLIP-ZSL IS REPRODUCED UNDER ZSL SETTING WITH THE VANILLA CLIP MODEL

Method	Backbone	ACC(%) ↑									Mean ↑	PD↓
		0	1	2	3	4	5	6	7	8		
TOPIC [9]	ResNet18	61.3	50.1	45.2	41.2	37.5	35.5	32.2	29.5	24.4	39.4	36.9
Self-promoted [38]	ResNet18	64.5	63.8	59.5	55.5	52.5	49.6	46.7	43.8	41.9	52.8	22.6
CEC [16]	ResNet18	72.0	66.8	63.0	59.4	56.7	53.7	51.2	49.2	47.6	57.8	24.4
Replay-based [29]	ResNet18	71.8	67.1	63.2	59.8	57.0	54.0	51.6	49.5	48.2	58.0	23.6
MetaFSCIL [17]	ResNet18	72.0	67.9	63.8	60.3	57.6	55.1	52.9	50.8	49.2	58.9	22.8
F2M [39]	ResNet18	72.1	67.5	63.2	59.7	56.7	53.8	51.1	49.2	47.8	57.9	24.3
CLOM [15]	ResNet18	73.1	68.1	64.2	60.4	57.4	54.3	51.5	49.4	48.0	58.5	25.1
FACT [20]	ResNet18	72.6	69.6	66.4	62.8	60.6	57.3	54.3	52.2	50.5	60.7	22.1
C-FSCIL [13]	ResNet12	76.4	71.1	66.5	63.3	60.4	57.5	54.8	53.1	51.4	61.6	25.3
ALICE [35]	ResNet18	80.6	70.6	67.4	64.5	62.5	60.0	57.8	56.8	55.7	64.0	24.9
NC-FSCIL [14]	ResNet12	84.0	76.1	72.0	67.8	66.4	64.0	61.5	59.5	58.3	67.8	25.7
M2SD [33]	ResNet18	82.1	79.9	75.4	71.3	68.3	64.3	61.1	58.6	56.5	68.6	25.6
L2P⋆ [48]	ViT-L/16	96.5	94.3	91.7	90.8	89.7	87.5	84.4	84.3	83.5	89.2	13.0
LIMIT+SV-Swin-T⋆ [43]	Swin-T	90.6	89.2	86.8	85.4	84.8	83.4	81.9	81.9	81.7	85.1	8.9
CLIP-ZSL† [23]	ViT-L/14	-	-	-	-	-	-	-	-	-	89.7	-
CoOp-FSCIL† [24]	ViT-L/14	97.7	91.0	90.3	92.2	92.8	90.8	89.4	90.9	90.2	91.7	7.5
FACT w/ CLIP† [20]	ViT-L/14	97.8	96.1	93.4	92.0	92.8	92.3	92.0	92.1	91.1	93.3	6.7
FSPT-FSCIL (Ours)†	ViT-L/14	96.8	**96.7**	**95.1**	**94.9**	**94.8**	**94.0**	**92.1**	**92.5**	**92.0**	**94.3**	**4.8**

miniImageNet. Similarly, for CIFAR100, we observed an improvement of **13.9**% in Acc and **18.9**% in PD. Furthermore, when we benchmarked FSPT-FSCIL against methods that have undergone pretraining on large-scale upstream data, our approach still emerged as the winner in both the metrics. Specifically, FSPT-FSCIL surpassed FACT with ViT-L/14 on miniImageNet. On CIFAR100, FSPT-FSCIL demonstrated an enhancement of **2.0**% in Acc and **1.8**% in PD compared with CoOp-FSCIL. These findings validate the effectiveness and robustness of our proposed FSPT-FSCIL approach. It is noteworthy that the comparison to FACT underscores that the effectiveness of our method is not merely attributed to the expansion of the backbone network's scale. Rather, it is primarily derived from the innovative fast–slow prompt structure and the meta-learning strategy that we introduce in this article.

TABLE II

EVALUATION ON CIFAR100. THE TOP ROWS LIST NON-PRETRAINED METHODS, WHILE THE BOTTOM ROWS LIST METHODS BASED ON PRETRAINED FOUNDATION MODELS. † INDICATES MODELS PRETRAINED ON WIT. ⋆ INDICATES MODELS PRETRAINED ON IMAGENET-21K

Method	Backbone	ACC(%) ↑									Mean ↑	PD↓
		0	1	2	3	4	5	6	7	8		
TOPIC [9]	ResNet18	64.1	55.9	47.1	45.2	40.1	36.4	34.0	31.6	29.4	42.6	34.7
Self-promoted [38]	ResNet18	64.1	65.9	61.4	57.3	53.7	50.8	48.6	45.7	43.3	54.5	20.8
CEC [16]	ResNet20	73.1	68.9	65.3	61.2	58.1	55.6	53.2	51.3	49.1	59.5	24.0
FACT [20]	ResNet18	74.6	72.1	67.6	63.5	61.4	58.4	56.3	54.2	52.1	62.1	22.5
CLOM [15]	ResNet18	74.2	69.8	66.2	62.4	59.3	56.5	54.4	52.2	50.3	61.0	23.9
F2M [39]	ResNet18	64.7	62.1	59.0	55.6	52.6	50.0	48.1	46.3	44.7	53.7	20.0
C-FSCIL [13]	ResNet12	77.5	72.4	67.5	63.3	59.8	57.0	54.4	52.5	50.5	61.6	27.0
ALICE [35]	ResNet18	79.0	70.5	67.0	63.4	61.2	59.2	58.1	56.3	54.1	63.2	24.9
MetaFSCIL [17]	ResNet18	74.5	70.1	66.8	62.8	59.5	56.5	54.4	52.6	50.0	60.8	24.5
NC-FSCIL [14]	ResNet12	82.5	76.8	73.3	69.7	66.2	62.9	61.0	59.0	56.1	67.5	26.4
L2P⋆ [48]	Vit-L/16	89.9	86.0	81.8	80.3	80.0	74.6	73.2	72.6	65.0	78.2	24.9
LIMIT+SV-Swin-T⋆ [43]	Swin-T	86.8	82.8	80.4	77.2	76.1	74.0	72.9	71.7	69.8	76.8	17.0
CLIP-ZSL† [23]	Vit-L/14	-	-	-	-	-	-	-	-	-	77.9	-
CoOp-FSCIL† [24]	Vit-L/14	88.6	78.9	77.5	76.0	76.8	78.3	79.2	79.8	79.3	79.4	9.3
FACT w/ CLIP† [20]	Vit-L/14	87.8	84.0	81.4	78.0	77.8	76.3	75.0	72.5	71.9	78.3	15.9
FSPT-FSCIL (Ours)†	Vit-L/14	86.9	83.1	**81.9**	**80.7**	**80.4**	**79.9**	**80.1**	**79.9**	**79.4**	**81.4**	**7.5**

TABLE III

EVALUATION OF PROMPTING-BASED METHODS ON EACH DATASET

Method	MiniImageNet(%)		CUB200(%)		CIFAR100(%)		FGVCAircraft(%)		DTD(%)	
	Mean↑	PD↓	Mean↑	PD↓	Mean↑	PD↓	Mean↑	PD↓	Mean↑	PD↓
L2P [48]	89.2	13.0	72.3	17.4	78.2	24.9	26.3	31.6	60.8	29.7
FACT w/ CLIP [20]	93.3	6.7	78.7	10.7	78.3	15.9	44.5	13.1	72.3	10.5
LIMIT+SV-Swin-T [43]	85.1	8.9	**78.7**	**8.0**	76.8	17.0	-	-	-	-
CLIP-ZSL [23]	89.7	-	62.2	-	77.9	-	36.1	-	55.3	-
CoOp-FSCIL [24]	91.7	7.5	68.2	22.6	79.4	9.3	39.0	19.8	66.9	20.1
FSPT-FSCIL (Ours)	**94.3**	**4.8**	73.4	16.2	**81.4**	**7.5**	**44.5**	**12.6**	**74.9**	**10.4**

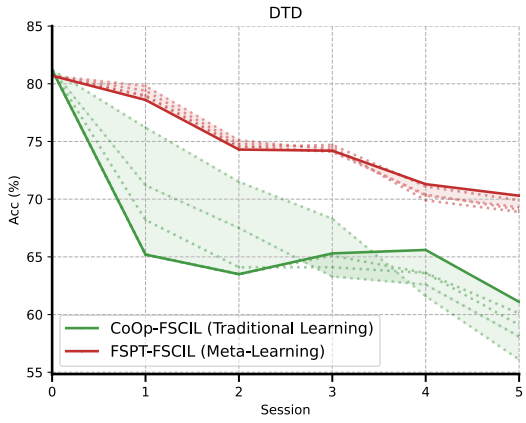


Fig. 7. Impact of different iteration counts on FSCIL performance. Train for five epochs in each incremental session and evaluate the model obtained after each epoch. The dashed lines represent the models after epochs 1–4, while the solid line represents the model after the fifth epoch.

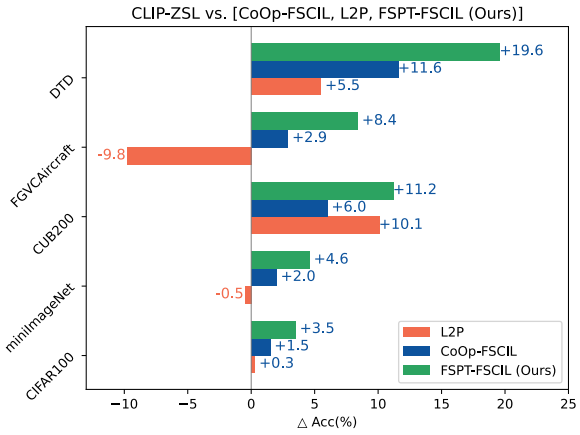
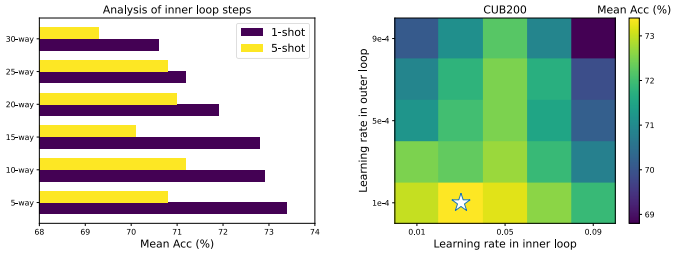


Fig. 8. FSCIL performance gains or losses of prompting-based methods compared with ZSL with CLIP.

5) A Deeper Comparison on More Datasets: We adapt CUB200, DTD, and FGVCAircraft into FSCIL paradigms to further validate the advantages of our method compared with methods with pretrained models. The experimental results are shown in Table III. Furthermore, in Fig. 8, we display the performance gains or losses of prompting-based methods compared with ZSL with CLIP. For DTD and FGVCAircraft, our method achieves improvements compared with all other methods, including prompt tuning methods and SOTA FSCIL methods with pretrained models. The results across multiple datasets further demonstrate the effectiveness of our approach in FSCIL.

6) Limitations: From Fig. 6 and Table III, we observe that FSPT-FSCIL lags behind FACT and SV-T [43] methods on the CUB200 dataset, which is attributed to a limitation of the CLIP framework. Specifically, the current CLIP text encoder struggles to capture textual semantic features that require fine-grained classification, and it poses a significant challenge to learn semantically meaningful context tokens during prompt tuning, thereby hampering prompt generalization.



(a) (b)

Fig. 9. Hyperparameter analysis for meta-learning. (a) Inner loop steps. (b) Learning rate.

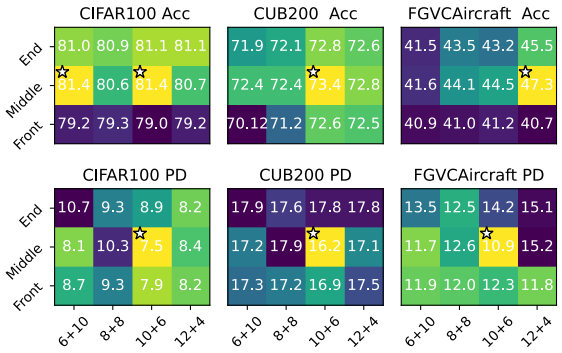


Fig. 10. Hyperparameter analysis for prompts. The horizontal axis of the each heatmap represents "slow prompt token size" + "fast prompt token size." The vertical axis represents the position of the class token.

TABLE IV

ABLATION STUDIES FOR PROMPT STRUCTURE AND LEARNING STRATEGY ON THE DTD DATASET. "B" REFERS TO BASELINE

Method	Prompt structure		Learning strategy		Mean	PD
	Single	Fast-slow	Meta	Traiditioal		
B1	✓			✓	66.9	20.1
B2	✓		✓		72.8	14.3
B3		✓	✓		74.9	10.4

This phenomenon leads to CLIP framework-based methods not achieving optimal performance on the CUB200 and FGVCAircraft datasets, falling below solutions based on visual features. This limitation can be mitigated using stronger multimodal models. Furthermore, despite slightly lower performance compared with the FACT and SV-T methods, our approach, leveraging the CLIP framework, maintains simplicity and still demonstrates significant advantages over CLIP itself and CoOp.

E. Ablation Study

The primary technical contributions of FSPT-FSCIL involve the utilization of visual foundational models within FSCIL and the implementation of meta-learning-based fast and slow prompting training strategies. In this section, we will analyze the relevant parameters of meta-learning and prompts and assess the impact of different foundational models on the performance of FSCIL.

1) Contribution of Prompts and Meta-Learning: To validate the significance of our contributions in the form of fast–slow prompts and meta-learning, we conduct ablation experiments on the DTD dataset. The results are presented in Table IV.

TABLE V

ABLATION STUDIES FOR GENERALIZATION OF FAST AND SLOW PROMPTS META-LEARNED FROM THE BASE SESSION ON THE DTD DATASET. THE SYMBOL "✓" INDICATES THE USE OF ONLY THE CORRESPONDING PROMPTS FOR INFERENCE DURING THE INCREMENTAL STAGE, WHILE DISCARDING THE OTHER PROMPTS

Slow prompts	Fast prompts	Mean↑	PD↓
	✓	33.9	55.4
✓		60.5	31.9
✓	✓	65.7	26.4

In baseline 2, the term "meta-learning single prompt" refers to a scenario where the inner and outer loops update the same prompt. By comparing baseline 1 and baseline 2, we observe a notable enhancement in FSCIL performance when prompts undergo meta-learning, surpassing traditional training methods. In addition, when comparing baseline 2 with baseline 3, it becomes evident that the incorporation of the fast–slow prompt structure leads to a further boost in overall performance.

2) Analysis for Fast- and Slow-Update Prompts: To further explore the differences in generalization performance and respective roles of fast and slow prompts obtained through meta-learning in base class learning, we specifically evaluated the system's performance when relying solely on each prompt. Specifically, during the incremental stage, we did not conduct further training but directly used the prompts obtained from base class learning for inference. When using only the fast prompts, we set each token of the slow prompts to a placeholder "X"; conversely, when using only the slow prompts, the tokens of the fast prompts are also set to "X" accordingly. From the experimental results in Table V, it is evident that relying solely on fast prompts results in significantly insufficient performance. This is actually understandable, as the fast prompts are designed to rapidly adapt to new categories, rather than to resist catastrophic forgetting. On the other hand, the slow prompts demonstrated their indispensable importance in terms of generalization performance, further confirming their effectiveness as an effective mechanism for preventing forgetting.

3) Hyperparameter Analysis for Meta-Learning: The differential update speed of prompts in the inner and outer loops is a key hyperparameter in meta-learning, which involves analyzing two specific parameters: the number of inner loop update steps corresponding to a single outer loop iteration, and the learning rates for both inner and outer loops. For the former, we set up multiple sets of different numbers of inner loop update steps. Specifically, we organize the dataset into N-way K-shot format. Since each update in the inner loop uses only one sample, the number of iterations in the inner loop in this format is $N * K$. As shown in Fig. 9(a), we select multiple sets of N and K to obtain different numbers of inner loop update steps. Each mode is trained with the same parameters on CUB200 dataset, with the outer loop learning rate fixed at 1e-4 and the inner loop learning rate set at 0.03. The experimental results, as shown in Fig. 9(a), indicate that in this scenario, our method performs better in the one-shot

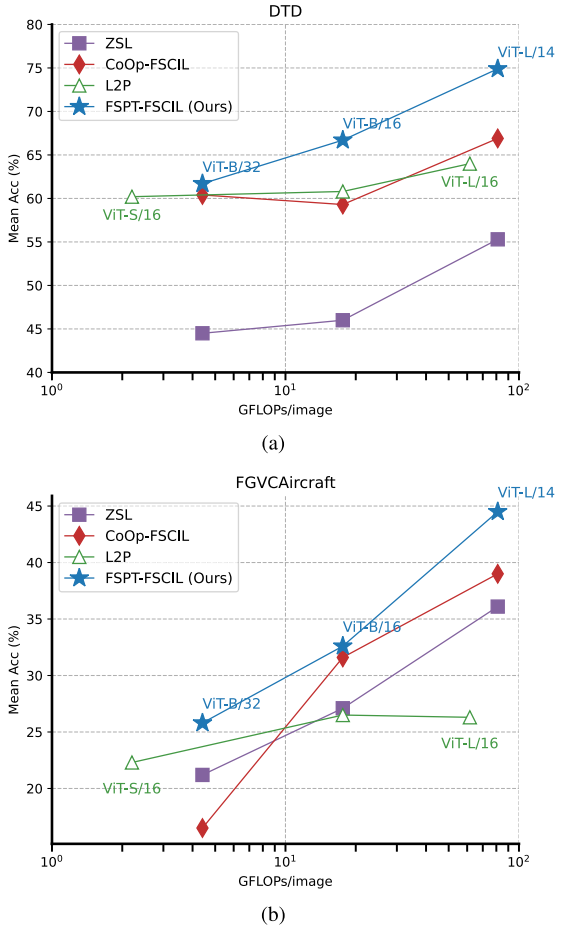


Fig. 11. Analysis for different foundation backbones. The experiments are conducted on (a) DTD and (b) FGVCAircraft datasets.

inner loop tasks, particularly achieving the best results in the five-way one-shot setting. In the case of five-way one-shot, the inner loop will use five samples to update the fast prompts five times, with each sample representing a different class. This allows for quick learning from one class to another. Therefore, we decided to set the inner loop steps to 5.

Next, we conducted an analysis of the learning rates. According to the OML [67] method, we fixed the outer loop learning rate in the range of 0.0001–0.0009, and the inner loop learning rate in the range of 0.01–0.09. After extensive combination experiments on CUB200, the results we obtained are shown in Fig. 9(b). In these experiments, the model achieved the best performance when the outer loop learning rate was set to 0.0001 and the inner loop learning rate was set to 0.03. Finally, based on the results above, we determined the parameter settings for meta-learning.

4) Hyperparameter Analysis for Prompts: In the analysis for prompts, we primarily explore the impact of prompt token length and the location of the class token. To ensure a fair comparison with CoOp [24], we fix the total number of prompt tokens at 16. Our main focus is on investigating the effect of different combinations of fast and slow prompts, specifically dividing them into four groups. The slow prompt tokens are set at 6, 8, 10, and 12, corresponding to fast prompt tokens at 10, 8, 6, and 4. In addition, we explore the impact of inserting the class token at the front, end, or in the middle of the combined slow and fast prompts. We conducted validation on three datasets, and the experimental results are shown in Fig. 10. Although the conclusions obtained on multiple datasets are not consistent, overall, the combination of 10 slow prompt tokens and six fast prompt tokens, with the class token inserted between them, yield relatively better results. We ultimately select this parameter combination for comparison with various baselines. Furthermore, slow prompts serve as meta-knowledge, playing a role in balancing rapid learning and resistance to forgetting. It is natural for them to occupy more tokens.

5) Scalable Analysis: The main technical contribution of this article lies in the application of a meta-learning-based prompt tuning technique to adapt pretrained foundation models to FSCIL tasks. We chose VIT-L/14 as the primary backbone model and achieved SOTA performance across multiple datasets. To investigate the impact of different scales of foundation models on performance, we conduct experiments with various backbone models. Specifically, for CLIP-based methods, we analyze three backbone models: VIT-B/32, VIT-B/16, and VIT-L/14. For L2P [48], we experiment with three backbone models: VIT-S/16, VIT-B/16, and VIT-L/16. FSCIL experiments are conducted on the FGVCAircraft and DTD datasets, and the results are presented in Fig. 11, with the number of giga floating-point operations per image (GFLOPs/image) inference serving as a quantifiable measure of model scale.

The experimental results reveal three phenomena or conclusions. First, among models of similar scales, FSPT-FSCIL demonstrates stronger performance in FSCIL compared with the other three prompting-based methods. In other words, to achieve the same ACC, our method requires a smaller-scale pretrained model. Second, when applying the L2P method to FSCIL tasks, the improvement in backbone models does not significantly impact the final performance. Finally, as the capabilities of the foundation models improve, FSPT-FSCIL exhibits substantial performance enhancements in FSCIL tasks. This indicates that our approach can be extended to larger scale visual foundation models to achieve superior performance.

V. Conclusion

Addressing the challenges in current FSCIL research, which included the lack of utilization of pretrained foundational models and the fact that these foundational models were primarily designed for independent downstream tasks rather than the few-shot and streaming data context of FSCIL, we introduced a brain-inspired fast and slow prompt-tuning strategy. This strategy successfully applied a vision-language foundational model to FSCIL tasks. We used an optimization-based meta-learning approach for cross-training the fast and slow prompts. Our method was validated on five FSCIL benchmark datasets and demonstrated superior performance in terms of average classification ACC and forgetfulness. Notably, when compared with methods that did not use pretrained models, our approach naturally achieved significant advantages. Furthermore, when compared with methods using

foundational models or prompt-tuning, our strategy continued to exhibit superiority, underscoring the effectiveness of our brain-inspired prompt-tuning strategy. In addition, we demonstrated that the performance of our method improved with the enhancement of foundational model capabilities, making it adaptable to future, more powerful foundational models to enhance FSCIL performance.

The proposed method is simple yet effective, falling under the text prompt-tuning strategy within the CLIP framework. However, since text prompts or tokens struggle to represent the semantics of fine-grained classification, there is room for improvement at the fine-grained classification level. In the future, based on the fast and slow prompts introduced in this article, we can incorporate visual prompts, such as VP [48] and VPT [50], for multimodal prompt-tuning. In addition, the prompt-tuning in our method only modifies input tokens. The concept of fast and slow prompts can be extended to other forms of prompt tuning, such as LoRA [68], Adapter-Tuning [69], and Prefix-Tuning [70], to further tackle the challenges of fine-grained FSCIL tasks.

REFERENCES

[1] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[2] D. Zhou, Y. Yang, and D. Zhan, "Learning to classify with incremental new class," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2429–2443, Jun. 2022.

[3] Y. Liu, X. Hong, X. Tao, S. Dong, J. Shi, and Y. Gong, "Model behavior preserving for class-incremental learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7529–7540, Oct. 2023.

[4] H. Zhao, H. Wang, Y. Fu, F. Wu, and X. Li, "Memory-efficient class-incremental learning for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5966–5977, Oct. 2022.

[5] E. Belouadah, A. Popescu, and I. Kanellos, "A comprehensive study of class incremental learning algorithms for visual tasks," *Neural Netw.*, vol. 135, pp. 38–54, Mar. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608020304202

[6] N. Lai, M. Kan, C. Han, X. Song, and S. Shan, "Learning to learn adaptive classifier-predictor for few-shot learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3458–3470, Aug. 2021.

[7] P. Tian, W. Li, and Y. Gao, "Consistent meta-regularization for better meta-knowledge in few-shot learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7277–7288, Dec. 2022.

[8] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surveys*, vol. 53, no. 3, pp. 1–34, Jun. 2020, doi: 10.1145/3386252.

[9] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12183–12192.

[10] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, 2017.

[11] S. Dong, X. Hong, X. Tao, X. Chang, X. Wei, and Y. Gong, "Few-shot class-incremental learning via relation knowledge distillation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 2, pp. 1255–1263.

[12] A. Cheraghian, S. Rahman, P. Fang, S. K. Roy, L. Petersson, and M. Harandi, "Semantic-aware knowledge distillation for few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2534–2543.

[13] M. Hersche, G. Karunaratne, G. Cherubini, L. Benini, A. Sebastian, and A. Rahimi, "Constrained few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 9057–9067.

[14] Y. Yang, H. Yuan, X. Li, Z. Lin, P. Torr, and D. Tao, "Neural collapse inspired feature-classifier alignment for few-shot class-incremental learning," in *Proc. 11th Int. Conf. Learn. Represent.*, 2023, pp. 1–18. [Online]. Available: https://openreview.net/forum?id=y5W8tpojhtJ

[15] Y. Zou, S. Zhang, Y. Li, and R. Li, "Margin-based few-shot class-incremental learning with class-level overfitting mitigation," in *Proc. Adv. Neural Inf. Process. Syst.*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022, pp. 27267–27279. [Online]. Available: https://openreview.net/forum?id=hyc27bDixNR

[16] C. Zhang, N. Song, G. Lin, Y. Zheng, P. Pan, and Y. Xu, "Few-shot incremental learning with continually evolved classifiers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12455–12464.

[17] Z. Chi, L. Gu, H. Liu, Y. Wang, Y. Yu, and J. Tang, "MetaFSCIL: A meta-learning approach for few-shot class incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 14166–14175.

[18] S. Palit, B. Banerjee, and S. Chaudhuri, "Prototypical quadruplet for few-shot class incremental learning," *Proc. Comput. Sci.*, vol. 222, pp. 25–34, Jan. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050923009067

[19] J. Kalla and S. Biswas, "S3C: Self-supervised stochastic classifiers for few-shot class-incremental learning," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 432–448.

[20] D.-W. Zhou, F.-Y. Wang, H.-J. Ye, L. Ma, S. Pu, and D.-C. Zhan, "Forward compatible few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 9046–9056.

[21] X. Xu, S. Niu, Z. Wang, W. Guo, L. Jing, and H. Yang, "Multi-feature space similarity supplement for few-shot class incremental learning," *Knowl.-Based Syst.*, vol. 265, Apr. 2023, Art. no. 110394. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705123001442

[22] P. Mazumder and P. Singh, "Mitigate forgetting in few-shot class-incremental learning using different image views," *Neural Netw.*, vol. 165, pp. 999–1009, Aug. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608023003544

[23] A. Radford et al., "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learn.*, vol. 139, 2021, pp. 8748–8763.

[24] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *Int. J. Comput. Vis.*, vol. 130, no. 9, pp. 2337–2348, Sep. 2022.

[25] A. Roxin and S. Fusi, "Efficient partitioning of memory systems and its importance for memory consolidation," *PLoS Comput. Biol.*, vol. 9, no. 7, Jul. 2013, Art. no. e1003146.

[26] S. Fusi, "Computational models of long term plasticity and memory," 2017, *arXiv:1706.04946*.

[27] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly, "Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory," *Psychol. Rev.*, vol. 102, no. 3, p. 419, 1995.

[28] D. Kumaran, D. Hassabis, and J. L. McClelland, "What learning systems do intelligent agents need? Complementary learning systems theory updated," *Trends Cognit. Sci.*, vol. 20, no. 7, pp. 512–534, Jul. 2016.

[29] H. Liu et al., "Few-shot class-incremental learning via entropy-regularized data-free replay," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 146–162.

[30] Y. Cui et al., "Uncertainty-guided semi-supervised few-shot class-incremental learning with knowledge distillation," *IEEE Trans. Multimedia*, vol. 25, pp. 6422–6435, 2022.

[31] L. Zhao et al., "Few-shot class-incremental learning via class-aware bilateral distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 11838–11847.

[32] Z. Song, Y. Zhao, Y. Shi, P. Peng, L. Yuan, and Y. Tian, "Learning with fantasy: Semantic-aware virtual contrastive constraint for few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 24183–24192.

[33] J. Lin, Z. Wu, W. Lin, J. Huang, and R. Luo, "M2SD: Multiple mixing self-distillation for few-shot class-incremental learning," in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 4, pp. 3422–3431.

[34] B. Yang et al., "Dynamic support network for few-shot class incremental learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2945–2951, Mar. 2023.

[35] C. Peng, K. Zhao, T. Wang, M. Li, and B. C. Lovell, "Few-shot class-incremental learning from an open-set perspective," in *Proc. Eur. Conf. Comput. Vis.*, Tel Aviv, Israel. Cham, Switzerland: Springer, Oct. 2022, pp. 382–397.

[36] H. Ran, W. Li, L. Li, S. Tian, X. Ning, and P. Tiwari, "Learning optimal inter-class margin adaptively for few-shot class-incremental learning via neural collapse-based meta-learning," *Inf. Process. Manage.*, vol. 61, no. 3, May 2024, Art. no. 103664.

[37] D.-W. Zhou, H.-J. Ye, L. Ma, D. Xie, S. Pu, and D.-C. Zhan, "Few-shot class-incremental learning by sampling multi-phase tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 11, pp. 12816–12831, Nov. 2023.

[38] K. Zhu, Y. Cao, W. Zhai, J. Cheng, and Z.-J. Zha, "Self-promoted prototype refinement for few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6801–6810.

[39] G. Shi, J. Chen, W. Zhang, L.-M. Zhan, and X.-M. Wu, "Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima," in *Proc. 35th Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 6747–6761.

[40] D.-Y. Kim, D.-J. Han, J. Seo, and J. Moon, "Warping the space: Weight space rotation for class-incremental few-shot learning," in *Proc. 11th Int. Conf. Learn. Represent.*, 2022, pp. 1–19.

[41] B. Liu, B. Yang, L. Xie, R. Wang, Q. Tian, and Q. Ye, "Learnable distribution calibration for few-shot class-incremental learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 10, pp. 12699–12706, Mar. 2023.

[42] S. Tian, L. Li, W. Li, H. Ran, X. Ning, and P. Tiwari, "A survey on few-shot class-incremental learning," *Neural Netw.*, vol. 169, pp. 307–324, Jan. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608023006019

[43] W. Qiu, S. Fu, J. Zhang, C. Lei, and Q. Peng, "Semantic-visual guided transformer for few-shot class-incremental learning," in *Proc. IEEE Int. Conf. Multimedia Expo*, Mar. 2023, pp. 2885–2890.

[44] D. Goswami, B. Twardowski, and J. van de Weijer, "Calibrating higher-order statistics for few-shot class-incremental learning with pre-trained vision transformers," 2024, *arXiv:2404.06622*.

[45] K.-H. Park, K. Song, and G.-M. Park, "Pre-trained vision and language transformers are few-shot incremental learners," 2024, *arXiv:2404.02117*.

[46] P. Liu, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, 2023.

[47] T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," 2020, *arXiv:2012.15723*.

[48] Z. Wang et al., "Learning to prompt for continual learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 139–149.

[49] Z. Wang et al., "DualPrompt: Complementary prompting for rehearsal-free continual learning," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 631–648.

[50] M. Jia et al., "Visual prompt tuning," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Oct. 2022, pp. 709–727.

[51] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Conditional prompt learning for vision-language models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 16816–16825.

[52] S. Tian, L. Li, W. Li, H. Ran, L. Li, and X. Ning, "PL-FSCIL: Harnessing the power of prompts for few-shot class-incremental learning," 2024, *arXiv:2401.14807*.

[53] R. Kemker and C. Kanan, "FearNet: Brain-inspired model for incremental learning," 2017, *arXiv:1711.10563*.

[54] E. Arani, F. Sarfraz, and B. Zonooz, "Learning fast, learning slow: A general continual learning method based on complementary learning system," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–22. [Online]. Available: https://openreview.net/forum?id=uxxFrDwrE7Y

[55] F. Sarfraz, E. Arani, and B. Zonooz, "Synergy between synaptic consolidation and experience replay for general continual learning," in *Proc. Conf. Lifelong Learn. Agents*, 2022, pp. 920–936.

[56] Q. Pham, C. Liu, and S. Hoi, "DualNet: Continual learning, fast and slow," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 16131–16144.

[57] L. Wang, Q. Li, Y. Zhong, and J. Zhu, "Few-shot continual learning: A brain-inspired approach," 2021, *arXiv:2104.09034*.

[58] H. Zhao, Y. Fu, M. Kang, Q. Tian, F. Wu, and X. Li, "MgSvF: Multi-grained slow versus fast framework for few-shot class-incremental learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 3, pp. 1576–1588, Mar. 2024.

[59] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," 2015, *arXiv:1508.07909*.

[60] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Master's thesis, Univ. Toronto, Toronto, ON, Canada, 2009.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[62] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. (2011). *The Caltech-UCSD Birds-200–2011 Dataset*. [Online]. Available: http://www.vision.caltech.edu/visipedia/CUB-200.html

[63] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," 2013, *arXiv:1306.5151*.

[64] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 3606–3613.

[65] J. Wang and Y. Zhai, "Prototypical Siamese networks for few-shot learning," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 178–181.

[66] A. Dosovitskiy, "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–21. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[67] K. Javed and M. White, "Meta-learning representations for continual learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[68] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," 2021, *arXiv:2106.09685*.

[69] N. Houlsby et al., "Parameter-efficient transfer learning for NLP," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2790–2799.

[70] X. Lisa Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," 2021, *arXiv:2101.00190*.