Learning to Prompt for Continual Learning

April 27, 2023

Submitted by

Rohit Kumar(20963)

Avnish Kumar(20964)

Abstract

The current approach to continual learning is to adjust the model's settings to cope with changing data distributions, but this often leads to forgetting what was learned before. To prevent this, most methods use a rehearsal buffer or task identification during testing to retrieve previously learned information. However, this new method proposes a different approach that trains a more efficient memory system without relying on task identification during testing. This new method, called L2P, prompts a pre-trained model to learn tasks sequentially by using small, learnable parameters that are stored in a memory space. The objective is to optimize these prompts to guide the model's predictions and manage both task-specific and task-invariant knowledge while maintaining the model's flexibility. The researchers conducted experiments on popular image classification benchmarks using different challenging continual learning scenarios. L2P consistently outperformed other state-of-the-art methods and even achieved competitive results against rehearsal-based methods without using a rehearsal buffer. Additionally, L2P can be applied to task-agnostic continual learning.

1 Introduction

Ordinary supervised learning trains a model on data that's all the same and doesn't change. Continual learning is different because it trains a single model on data that changes over time, with different tasks presented one after another. However, this type of learning can lead to overfitting

on current data and forgetting of previously learned information, which can hurt performance.

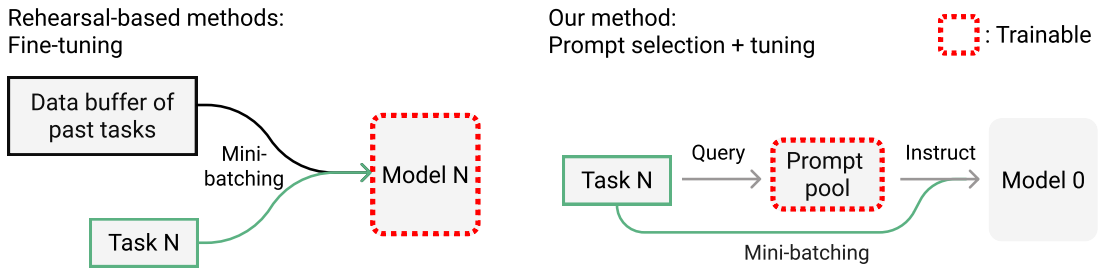

Figure 1: Overview of the L2P framework. Compared with typical methods that adapt entire or partial model weights to tasks sequentially with a rehearsal buffer to avoid forgetting, L2P uses a single backbone model and learns a prompt pool to instruct the model conditionally. Task-specific knowledge is stored inside a prompt pool, thus a rehearsal buffer is no longer mandatory to mitigate forgetting. L2P automatically selects and updates prompts from the pool in an instance-wise fashion, thus task identity is not required at test time. Notably, our largest prompt space is smaller than the size of one 224×224 image.

There is a lot of research on continual learning that focuses on adapting model weights as data distribution changes, while still preserving past knowledge. Many methods achieve good results, but there are still limitations. Some methods use a rehearsal buffer to retrain past examples, but they perform poorly with smaller buffer sizes and are not effective in real-world scenarios where data privacy is important. Other methods assume known task identity at test time to bypass the forgetting issue, but this restricts practical usage. So, simply buffering past data and retraining the model may not be the best approach to retrieve past knowledge. Prior research in continual learning has limitations and raises important questions. The first question is whether the way we currently store past data can be improved to create a more intelligent and concise memory system. The second question is how to select relevant knowledge components for any given sample without knowing its task identity. To answer the first question, researchers have looked to a technique called prompt-based learning, which has been successful in natural language processing. Prompting involves designing text inputs with task-specific information that allows a pre-trained language model to perform task-specific predictions. This technique has the potential to help us store learned knowledge in the continual learning context. However, it is not clear how to use prompting to address the second question directly. One option is to train different prompts for different tasks, but this still requires knowing the task identity at test time. Alternatively, we could use a single shared prompt for all tasks, but this may lead to forgetting previously learned infor-

mation. To this end, we propose a new continual learning method called Learning to Prompt for Continual Learning (L2P), which is orthogonal to popular rehearsal-based methods and applicable to practical continual learning scenarios without known task identity or boundaries. Figure 1 gives an overview of our method in contrast to typical continual learning methods. L2P leverages the representative features from pre-trained models; however, instead of tuning the parameters during the continual learning process, L2P keeps the pre-trained model untouched, and instead learns a set of prompts that dynamically instruct models to solve corresponding tasks. Specifically, the prompts are structured in a key-value shared memory space called the prompt pool, and we design a query mechanism to dynamically lookup a subset of task-relevant prompts based on the instance wise input features. The prompt pool, which is optimized jointly with the supervised loss, ensures that shared prompts encode shared knowledge for knowledge transfer, and unshared prompts encode task-specific knowledge that help maintain model plasticity. Our design explicitly decouples shared and task-specific knowledge, thus largely reducing the interference between task-specific knowledge during optimization, leading to minimal catastrophic forgetting without the necessity of a rehearsal buffer. The instance wise query mechanism removes the necessity of knowing the task identity or boundaries, enabling the most challenging, yet under-investigated task-agnostic continual learning. The selected prompts are then prepended to the input embeddings (Figure 2), which implicitly add task-relevant instruction to pre-trained models, so that the model recalls the most relevant features to conduct corresponding tasks.

In summary, this work makes the following contributions:

1. We propose L2P, a novel continual learning framework based on prompts for continual learning, providing a new mechanism to tackle continual learning challenges through learning a prompt pool memory space, which are served as parameterized “instructions" for pre-trained models to learn tasks sequentially. The method is applicable to handle the most challenging task-agnostic continual learning.
2. We conduct comprehensive experiments to demonstrate the effectiveness of L2P on multiple continual learning benchmarks, including class- and domain-incremental, and task-agnostic settings. The proposed L2P outperforms previous state-of-the-art methods consistently on all benchmarks. Surprisingly, even when a rehearsal buffer is *not* used, L2P still achieves competitive results against rehearsal-based methods, which is ideal in real-world scenarios when rehearsal buffer is prohibited.
3. To the best of our knowledge, we are the first to introduce the idea of prompting in the field of continual learning. We expect that our method provides a different perspective for solving frontier challenges in continual learning.

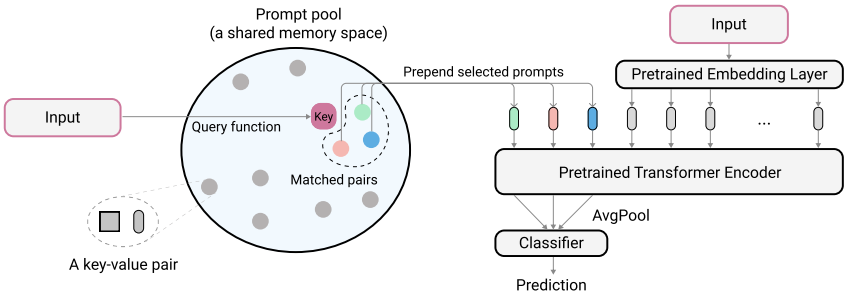

Figure 2: Illustration of L2P at test time. We follow the same procedure at training time: First, L2P selects a subset of prompts from a key-value paired *prompt pool* based on our proposed instance-wise query mechanism. Then, L2P prepends the selected prompts to the input tokens. Finally, L2P feeds the extended tokens to the model, and optimize the prompt pool through the loss defined in equation 5. The objective is learning to select and update prompts to instruct the prediction of the pre-trained backbone model.

2 Prerequisites

2.1 Continual learning protocols

Continual learning is usually defined as training machine learning models on non-stationary data from sequential tasks. We define a sequence of tasks $\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_T\}$, where the t-th task $\mathcal{D}_t = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{n_t}$ contains tuples of the input sample $\boldsymbol{x}_i^t \in \mathcal{X}$ and its corresponding label $y_i^t \in \mathcal{Y}$. The goal is to train a single model $f_\theta : \mathcal{X} \to \mathcal{Y}$ parameterized by θ, such that it predicts the label $y = f_\theta(\boldsymbol{x}) \in \mathcal{Y}$ given an unseen test sample $\boldsymbol{x}$ from arbitrary tasks. Data from the previous tasks may not be seen anymore when training future tasks.

Depending on the task transition environment, continual learning can be categorized into multiple settings with slightly different challenges. The common task-, class-, and domain-incremental setting assume task data $\mathcal{D}_t$ arrives in sequence $t = \{1, ..., T\}$ in a discrete manner. Different from class-incremental, task-incremental learning assumes task identity is known at test time and are often regarded as the simplest setting. Different from task- and class-incremental settings where each task has different classes, domain-incremental learning maintains the same set of classes for every task and only changes the distribution of $\boldsymbol{x}$ by task. In the more challenging task-agnostic setting, task data in $\mathcal{D}$ changes smoothly, and the task identity t is unknown. Our paper tackles the more challenging class-incremental and domain-incremental, and further explores the task-agnostic settings.

2.2 Prompt-based learning and baselines

Prompt-based learning is an emerging technique in NLP. In contrast to traditional supervised fine-tuning, this type of methods design task-specific prompt functions to instruct pre-trained models perform corresponding tasks conditionally . One of recent techniques, Prompt Tuning (PT) , proposes to simply condition frozen T5-like language models to perform down-stream NLP tasks by learning prompt parameters that are prepended to the input tokens to instruct the model prediction. Without loss of generality, here we introduce the definition of PT using the image modality transformer-based sequence models . The definition is easy to generalize to other modalities and sequence-based models.

Given an input of 2D image $\boldsymbol{x} \in \mathbb{R}^{H \times W \times C}$ and a pre-trained vision transformer (ViT) $f = f_r \circ f_e$ (excluding the classification head), where f_e is the input embedding layer, and f_r represents a stack of self-attention layers . Images are reshaped to a sequence of flattened 2D patches $\boldsymbol{x}_p \in \mathbb{R}^{L \times (S^2 \cdot C)}$, where L is the token length, *i.e.*, the number of patches, S is the patch size and C is the original number of channels. To simplify notation, we assume the first token in $\boldsymbol{x}_p$ is the `[class]` token as part of the pre-trained model . The pre-trained embedding layer $f_e : \mathbb{R}^{L \times (S^2 \cdot C)} \rightarrow \mathbb{R}^{L \times D}$ projects the patched image to the embedding feature $\boldsymbol{x}_e = f_e(x) \in \mathbb{R}^{L \times D}$, where D is the embedding dimension. When solving multiple downstreaming tasks, we keep the large-scale pre-trained backbone frozen to maintain its generality following PT. The direct application of PT is to prepend learnable parameters $P_e \in \mathbb{R}^{L_p \times D}$, called a prompt, to the embedding feature $\boldsymbol{x}_p = [P_e; \boldsymbol{x}_e]$, and feed the extended sequences to the model function $f_r(\boldsymbol{x}_p)$ for performing classification tasks. Different tasks have independent prompts and share one copy of the large model.

Compared with ordinary fine-tuning, literature shows that prompt-based learning results in a sequence-based model having higher capacity to learn features Despite its successes in transfer learning to train individual prompts for each task, prompting can not be directly applied to continual learning scenarios where test-time task identity is unknown.

3 Learning to Prompt L2P

3.1 From prompt to prompt pool

The motivations of introducing prompt pool are threefold. First, the task identity at test time is unknown so training task-independent prompts is not feasible. Second, even if the task-independent prompt can be known at test time, it prevents possible knowledge sharing between similar tasks Third, while the naive way of learning a single shared prompt for all tasks enables knowledge sharing, it still causes severe forgetting issue. Ideally one would learn a model that is able to share

knowledge when tasks are similar, while maintaining knowledge independent otherwise. Thus, we propose using a *prompt pool* to store encoded knowledge, which can be flexibly grouped as an input to the model. The prompt pool is defined as

$$\mathbf{P} = \{P_1, P_2, \cdots, P_M\}, \quad M = \text{total \# of prompts}, \tag{1}$$

where $P_j \in \mathbb{R}^{L_p \times D}$ is a single prompt with token length L_p and the same embedding size D as $\boldsymbol{x}_e$. Following the notations in Section 2.2, we let $\boldsymbol{x}$ and $\boldsymbol{x}_e = f_e(\boldsymbol{x})$ be the input and its corresponding embedding feature, respectively. Note that we omit the task index t of $\boldsymbol{x}$ in our notation as our method is general enough to the task-agnostic setting. Denoting $\{s_i\}_{i=1}^N$ as a subset of N indices from $[1, M]$, we can then adapt the input embedding as follows:

$$\boldsymbol{x}_p = [P_{s_1}; \cdots; P_{s_N}; \boldsymbol{x}_e], \quad 1 \leq N \leq M, \tag{2}$$

where ; represents concatenation along the token length dimension. Prompts are free to compose, so they can jointly encode knowledge (e.g. visual features or task information) for the model to process. Ideally, we want to achieve a more fine-grained knowledge sharing scheme via prompt combinations at the instance-wise level: similar inputs tend to share more common prompts, and vice versa.

3.2 Instance-wise prompt query

We design a key-value pair based query strategy to dynamically select suitable prompts for different inputs (see Figure 2). This key-valued memory query mechanism shares some design principles with methods in other fields, such as Differentiable Neural Computer [14] and VQ-VAE [**?**], which have external memory to maintain, and employ them for a different purpose.

We associate each prompt as value to a learnable key: $\{(\boldsymbol{k}_1, P_1), (\boldsymbol{k}_2, P_2), \cdots, (\boldsymbol{k}_M, P_M)\}$, where $\boldsymbol{k}_i \in \mathbb{R}^{D_k}$. And we denote the set of all keys by $\mathbf{K} = \{\boldsymbol{k}_i\}_{i=1}^M$. Ideally, we would like to let the input instance itself decide which prompts to choose through query-key matching. To this end, we introduce a query function $q : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{D_k}$ that encodes input $\boldsymbol{x}$ to the same dimension as the key. Moreover, q should be a deterministic function with respect to different tasks and has no learnable parameters. We directly use the whole pre-trained model as a frozen feature extractor to get the query features: $q(\boldsymbol{x}) = f(\boldsymbol{x})[0, :]$ (we use the feature vector corresponding to `[class]`). Other feature extractors like ConvNet are feasible as well.

Denote $\gamma : \mathbb{R}^{D_k} \times \mathbb{R}^{D_k} \to \mathbb{R}$ as a function to score the match between the query and prompt key (we find cosine distance works well). Given an input $\boldsymbol{x}$, we use $q(\boldsymbol{x})$ to lookup the top-N keys by simply solving the objective:

$$\mathbf{K}_{\boldsymbol{x}} = \underset{\{s_i\}_{i=1}^N \subseteq [1,M]}{\operatorname{argmin}} \sum_{i=1}^{N} \gamma\left(q(\boldsymbol{x}), \boldsymbol{k}_{s_i}\right), \tag{3}$$

where $\mathbf{K}_{\boldsymbol{x}}$ represents the a subset of top-N keys selected specifically for $\boldsymbol{x}$ from $\mathbf{K}$. Note that the design of this key-value strategy decouples the query mechanism learning and prompt learning processes, which has been experimentally shown to be critical (see Section **??**). Furthermore, querying prompts is done in an instance-wise fashion, which makes the whole framework *task-agnostic*, meaning that the method works without needing clear task boundaries during training, nor task identity at test time.

Optionally diversifying prompt-selection. Although our method does not need task boundary information, in real-world scenarios and experimental datasets, it is quite common that the task transition is discrete and so task boundaries are known at train time. We find that adding such a prior into our framework can help the model learn better task-specific prompts, especially when tasks have high diversity. To this end, we propose a simple extension to add task boundary prior, which is optional for L2P.

During training of task t, we maintain a prompt frequency table $H_t = [h_1, h_2, \cdots, h_M]$, where each entry represents the normalized frequency of prompt P_i being selected up until task $t-1$. To encourage the query mechanism to select diverse prompts, we modify equation 3 to

$$\mathbf{K}_{\boldsymbol{x}} = \underset{\{s_i\}_{i=1}^{N} \subseteq [1,M]}{\operatorname{argmin}} \sum_{i=1}^{N} \gamma\left(q(\boldsymbol{x}), \boldsymbol{k}_{s_i}\right) \cdot h_{s_i}, \tag{4}$$

where h_{s_i} penalizes the frequently-used prompts being selected to encourage diversified selection. Equation 4 is only applicable during training; at test time, equation 3 is used.

3.3 Optimization objective for L2P

At every training step, after selecting N prompts following the aforementioned query strategy, the adapted embedding feature $\boldsymbol{x}_p$ is fed into the rest of the pre-trained model f_r and the final classifier g_ϕ parametrized by ϕ. Overall, we seek to minimize the end-to-end training loss function:

$$\begin{aligned} \min_{\mathbf{P},\mathbf{K},\phi} \quad & \mathcal{L}(g_\phi(f_r^{\text{avg}}(\boldsymbol{x}_p)), y) + \lambda \sum_{\mathbf{K}_{\boldsymbol{x}}} \gamma\left(q(\boldsymbol{x}), \boldsymbol{k}_{s_i}\right), \\ & s.t., \quad \mathbf{K}_{\boldsymbol{x}} \text{ is obtained with equation 3}, \end{aligned} \tag{5}$$

where $f_r^{\text{avg}} = \text{AvgPool}(f_r(\boldsymbol{x}_p)[0 : NL_p, :])$, i.e., the output hidden vectors corresponding to the $N \cdot L_p$ prompt locations are averaged before the classification head. The first term is the softmax cross-entropy loss, the second term is a surrogate loss to pull selected keys closer to corresponding query features. λ is a scalar to weight the loss.

Evaluation of L2P for continual learning:

Average Accuracy:

Accuracy on Cifar:

Accuracy on Imagenet-R:

Dataset	Acc@1	Acc@5	Loss	Forgetting	Backward
Cifar100_l2p	83.5200	97.3500	0.6504	6.7333	-6.6000
Imagenet-R	58.8765	77.9441	2.0681	8.2327	-8.2327
Cifar100 (modified)	81.2327	95.5000	0.6612	7.2575	-6.9753

	Acc@1	Acc@5	Loss
Task 1	84.600	98.000	0.653
Task 2	81.500	96.700	0.753
Task 3	83.000	97.200	0.657
Task 4	83.900	96.400	0.659
Task 5	88.000	97.500	0.563
Task 6	77.200	97.600	0.810
Task 7	81.400	96.100	0.726
Task 8	81.200	96.700	0.697
Task 9	87.800	98.500	0.469
Task 10	86.600	98.800	0.517

Comparision Between L2P and Dual Prompt:

3.4 Limitations

- Single prompt pool > Doesn't replicate human brain functioning. > Inevitable interference between task-specific and task-invariant knowledge.
- Still needs rehearsal data to outperform rehearsal-based methods. Attach prompt only at the input.
- It does not acknowledge which specific task it is prone to forgetting.

3.5 Our Contributions

L2p was not evaluated in Imagenet – R dataset, So, we evaluated and the performance was not satisfactory.

We tried individual prompt but the performance still lagging.

We further implemented Dual prompt technique which is further extension of L2p and compare the results on Imagenet-R and cifar100.

We also tried L2p on TinyImagenet but due to server issue training could not complete

	Acc@1	Acc@5	Loss
Task 1	55.538	74.615	2.132
Task 2	62.267	77.333	2.023
Task 3	52.693	80.796	2.078
Task 4	49.826	74.653	2.272
Task 5	57.263	76.257	2.146
Task 6	57.168	79.032	2.063
Task 7	54.578	75.045	2.315
Task 8	59.326	74.831	2.206
Task 9	75.120	86.762	1.501
Task 10	64.986	80.115	1.944

L2p

Dataset	Acc@1	Acc@5	Loss	Forgetting	Backward
Cifar100_l2p	83.5200	97.3500	0.6504	6.7333	-6.6000
Imagenet-R	58.8765	77.9441	2.0681	8.2327	-8.2327

Dual Prompt

Dataset	Acc@1	Acc@5	Loss	Forgetting	Backward
Cifar100_l2p	85.4700	97.8200	0.5615	5.3889	-5.3997
Imagenet-R	60.2027	77.9334	2.4525	2.5463	-1.6067

3.6 Novel Idea

Instead of adding only two prompt pool adding three can be beneficial imitating brain functioning suggested by triple memory network. We can use Active Learning in which model itself chooses to train on the tasks it is forgetting.

References

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

[2] Yaroslav Bulatov. notmnist dataset, 2011.

[3] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *NeurIPS*, 2020.

[4] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *ICCV*, 2021.

[5] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence.

In *ECCV*, pages 532–547, 2018.

[6] Arslan Chaudhry, Albert Gordo, Puneet Kumar Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. *arXiv preprint arXiv:2002.08165*, 2(7), 2020.

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

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

[9] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *TPAMI*, 2021.

[10] 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.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021.

[12] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *ECCV*, 2020.

[13] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

[14] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016. 7

[15] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020.

[16] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 2020.

[17] Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *ICRA*, 2019.