A. Potential negative societal impact

L2P is a strong continual learning method and has great potential to be applied in various fields. However, there are some ways it could be misused. Our method takes a well-pretrained model as a backbone, thus any bias and fairness issues [38] in the original model may be carried over during the continual learning process. We encourage any users to thoroughly check the pretrained model to mitigate any bias and fairness issues. Moreover, the method could be deployed in safety-critical applications, such as autonomous driving systems [15], which may present potential security issues in terms of adversarial attacks [33]. We would recommend testing the robustness of our method in future work and design corresponding defense techniques to deal with potential security concerns.

B. Limitations

Although our method is demonstrated on vision models, it does not make any assumption of modalities. We leave exploration on other modalities as future work. Additionally, L2P assumes there are pre-trained sequence-based models. While they have become common assets and future directions in advanced communities, how to generalize our framework to other vision architectures (e.g. ConvNet) could be an appealing research direction.

How to achieve continual learning that can satisfy the real-world requirements is an important direction that remains challenging. For example, the task-agnostic setting is known as the most challenging setting and is very close to real-world scenarios. Although our method takes a step further towards this goal, however, the current commonly used Gaussian scheduled CIFAR-100 is synthetic and still far from realistic. Thus, we think it also requires more complex benchmarks to evaluate the ability of task-agnostic continual learning methods and push forward the advances of this real-world challenge.

C. Dataset details and licensing information

Split CIFAR-100 (class-incremental). This dataset splits the original CIFAR-100 [22] into 10 tasks, 10 disjoint classes per task. Since the tasks are from a single original dataset, they share some similarities and some classes could be from the same superclass. Although CIFAR-100 is a simple image classification dataset, it remains quite challenging for continual learning studies, especially in the class-incremental setting [34].

5-datasets (class-incremental). We also use a challenging dataset proposed in [12]. This dataset consists of five image classification datasets: CIFAR-10, MNIST [24], Fashion-MNIST [62], SVHN [40], and notMNIST [2]. Although each dataset alone is not hard, the sequential training of them is fairly challenging even with ImageNet pre-trained models, since models are susceptible to forgetting when the tasks are diverse [39].

CORe50 (domain-incremental). This is a widely used dataset specifically designed for continual object recognition [30]. It is a collection of 50 objects collected in 11 distinct domains, where 8 of them (120,000 samples) are used for training, and the rest are considered as a single test set (45,000). Methods are trained on each domain sequentially.

Gaussian scheduled CIFAR-100 (task-agnostic). The distribution of data shifts gradually throughout the learning process [52], the probability that a class is present in a batch follows a Gaussian distribution centered with intervals. There is no explicit task boundaries between batches, thus requiring methods to be able to implicitly adapt to non-stationary data distribution without utilizing any task-specific information during both training and inference.

- CIFAR-10 and CIFAR-100 [22], Fashion-MNIST [62] are licensed under the MIT license.
- MNIST [24] is licensed under the Creative Commons Attribution-Share Alike 3.0 license.
- CORe50 [30] is under the Creative Commons Attribution 4.0 International license.
- The licensing information is not available for SVHN [40], notMNIST [2].

D. Algorithm details

To better illustrate our proposed method, we present a whole picture of the training procedure in Algorithm 1. Note that for prediction, we simply replace loss calculation to label prediction. Optionally, we can replace the top-N keys lookup by equation 4, when task boundary prior is known.

Algorithm 1: Learning to Prompt for Continual Learning (L2P)

Input: Pre-trained input embedding layer f_e, pre-trained self-attention layers f_r, final classification layer g_ϕ, number of tasks T, training set $\{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{n_t}\}_{t=1}^{T}$, prompt pool $\mathbf{P} = \{P_j\}_{j=1}^{M}$, prompt keys $\mathbf{K} = \{K_j\}_{j=1}^{M}$, number of training epochs of the t-th task E_t, learning rate η, balancing parameter λ

Initialize: $g_\phi,\ \mathbf{P},\ \mathbf{K}$

for $t = 1, \cdots, T$ **do**

  for $e = 1, \cdots, E_t$ **do**

    Draw a mini-batch $B = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{l}$

    Initialize the sets of chosen keys and prompts for current batch: $\mathbf{K}_B = \{\}, \mathbf{P}_B = \{\}$.

    for $(\boldsymbol{x}, y)$ *in* B **do**

      Calculate query feature $q(\boldsymbol{x})$

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

      Select top-N prompts associated with the keys in $\mathbf{P}_{\boldsymbol{x}}$

      Calculate the input embedding sequence $\boldsymbol{x}_e = f_e(\boldsymbol{x})$

      Prepending $\boldsymbol{x}_e$ with corresponding top-N prompts by $\boldsymbol{x}_p = [P_{s_1}; \cdots; P_{s_N}; \boldsymbol{x}_e]$ (equation 2)

      Calculate per sample loss $\mathcal{L}_{\boldsymbol{x}} = \mathcal{L}(g_\phi(f_r^{\text{avg}}(\boldsymbol{x}_p)), y) + \lambda \sum_{\mathbf{K}_{\boldsymbol{x}}} \gamma\left(q(\boldsymbol{x}), \boldsymbol{k}_{s_i}\right)$ (equation 5)

      Update sets of chosen keys and prompts: $\mathbf{K}_B = \mathbf{K}_B \cup \mathbf{K}_{\boldsymbol{x}}, \mathbf{P}_B = \mathbf{P}_B \cup \mathbf{P}_{\boldsymbol{x}}$

    end

    Calculate per batch loss $\mathcal{L}_B$ by accumulating $\mathcal{L}_{\boldsymbol{x}}$

    for (K, P) *in* $\text{zip}(\mathbf{K}_B, \mathbf{P}_B)$ **do**

      Update K by $K \leftarrow K - \eta \nabla_K \mathcal{L}_B$

      Update P by $P \leftarrow P - \eta \nabla_P \mathcal{L}_B$

    end

    Update ϕ by $\phi \leftarrow \phi - \eta \nabla_\phi \mathcal{L}_B$

  end

end