Algorithm 1: DualPrompt at training time

Input: Pre-trained transformer-based backbone f, final classification layer f_ϕ, number of tasks T, training set $\{(\boldsymbol{x}_{i,t}, y_{i,t})\}_{i=1}^{n_t}\}_{t=1}^{T}$, G-Prompt $\boldsymbol{g}$, E-Prompt $\mathbf{E} = \{\boldsymbol{e}_t\}_{t=1}^{T}$, task keys $\mathbf{K} = \{\boldsymbol{k}_t\}_{t=1}^{T}$, $\texttt{start}_{\boldsymbol{g}}, \texttt{end}_{\boldsymbol{g}}, \texttt{start}_{\boldsymbol{e}}, \texttt{end}_{\boldsymbol{e}}$, prompting function f_{prompt}, number of training epochs of the t-th task M_t

Initialize: ϕ, $\boldsymbol{g}$, $\mathbf{E}$, $\mathbf{K}$

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

  Select the task-specific E-Prompt $\boldsymbol{e}_t$ and corresponding task key $\boldsymbol{k}_t$

  Generate the prompted architecture $f_{\boldsymbol{g},\boldsymbol{e}_t}$: attach $\boldsymbol{g}$ and $\boldsymbol{e}_t$ to $\texttt{start}_{\boldsymbol{g}}$-th to $\texttt{end}_{\boldsymbol{g}}$-th and $\texttt{start}_{\boldsymbol{e}}$-th to $\texttt{end}_{\boldsymbol{e}}$-th MSA layers respectively, with f_{prompt}.

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

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

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

      Calculate the prompted feature by $f_{\boldsymbol{g},\boldsymbol{e}_t}(\boldsymbol{x})$

      Calculate the per sample loss $\mathcal{L}_x$ via equation 6

    end

    Update ϕ, $\boldsymbol{g}$, $\mathbf{E}$, $\mathbf{K}$ by backpropagation

  end

end

Algorithm 2: DualPrompt at test time

Given components: Pre-trained transformer-based backbone f, trained classification layer f_ϕ, G-Prompt $\boldsymbol{g}$, E-Prompt $\mathbf{E} = \{\boldsymbol{e}_t\}_{t=1}^{T}$, task keys $\mathbf{K} = \{\boldsymbol{k}_t\}_{t=1}^{T}$, $\texttt{start}_{\boldsymbol{g}}, \texttt{end}_{\boldsymbol{g}}, \texttt{start}_{\boldsymbol{e}}, \texttt{end}_{\boldsymbol{e}}$, prompting function f_{prompt}

Input: test example $\boldsymbol{x}$

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

Matching for the index of E-Prompt via $t_{\boldsymbol{x}} = \text{argmin}_t \ \gamma(q(\boldsymbol{x}), \boldsymbol{k}_t)$

Select the task-specific E-Prompt $\boldsymbol{e}_{t_{\boldsymbol{x}}}$

Generate the prompted architecture $f_{\boldsymbol{g},\boldsymbol{e}_{t_{\boldsymbol{x}}}}$: attach $\boldsymbol{g}$ and $\boldsymbol{e}_{t_{\boldsymbol{x}}}$ to $\texttt{start}_{\boldsymbol{g}}$-th to $\texttt{end}_{\boldsymbol{g}}$-th and $\texttt{start}_{\boldsymbol{e}}$-th to $\texttt{end}_{\boldsymbol{e}}$-th MSA layers respectively, with f_{prompt}.

Prediction: $f_{\boldsymbol{g},\boldsymbol{e}_{t_{\boldsymbol{x}}}}(\boldsymbol{x})$

A Algorithms for DualPrompt

The training and test time Algorithms for DualPrompt are illustrated in Algorithm 1 and 2, respectively.

B Experimental details

For our method, DualPrompt, we use a constant learning rate of 0.005 usng Adam [18] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and a batch size of 128 for all benchmarks. For methods with the ViT architecture, all input images are resized to 224×224 and normalized to $[0, 1]$. Otherwise, we follow their

original implementation in their paper. We set the balancing factor $\lambda = 1$ in equation 6. We train Split CIFAR-100 and 5-datasets for 5 epochs per task and Split ImageNet-R for 50 epochs to ensure models converge properly for each task, thus the issue of forgetting is disentangled from possible underfitting [3]. We further sample 20% of the training set of Split ImageNet-R as a validation set for searching the optimal $\texttt{start}_{\boldsymbol{g}}, \texttt{end}_{\boldsymbol{g}}$, $\texttt{start}_{\boldsymbol{e}}, \texttt{end}_{\boldsymbol{e}}$, and empirically set $\texttt{start}_{\boldsymbol{g}} = 1, \texttt{end}_{\boldsymbol{g}} = 2$, $\texttt{start}_{\boldsymbol{e}} = 3, \texttt{end}_{\boldsymbol{e}} = 5$ for all the setting, since we discover they perform consistently well for all datasets. Following the suggestion of prompt length by [59], we set $L_g = 5$ and $L_e = 20$ for all datasets as well, and we further verify the correctness of this choice in Appendix G. Note that for fair comparison, we set $M = 30, L_p = 20, N = 5$ for L2P, which leads to similar amount of parameters as DualPrompt.

To ensure fair comparison, every aforementioned methods start from the same ImageNet pre-trained ViT-B/16 [9], following the setting in [59]. We carefully re-implement these method and use hyper-parameters by referring to their original source code. Moreover, we make the pre-trained model fully trainable for all methods (except L2P and DualPrompt), as we empirically observe they could not learn as good with a frozen backbone due to limited learning capacity.

C Evaluation metrics

Let $S_{t,\tau}$ be teh evaluation score, *e.g.*, classification accuracy on the τ-th task after training on the t-th task. After the model finishes training on the t-th task, we compute the Average Accuracy (A_t) and Forgetting (F_t) as follows:

$$A_t = \frac{1}{t}\sum_{\tau=1}^{t} S_{t,\tau}$$

$$F_t = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \max_{\tau' \in \{1,\cdots,t-1\}} (S_{\tau',\tau} - S_{t,\tau})$$

Note that Average Accuracy is the overall evaluation metric for continual learning, which includes two aspects: greater learning capacity and less catastrophic forgetting, while Forgetting only serves as a measure of catastrophic forgetting.

D Details of comparing methods

- **Regularization-based methods.** EWC [19] and LwF [27] are representative regularization-based methods that are widely compared.
- **Rehearsal-based methods.** ER [7,13], GDumb [46], BiC [61], DER++ [3] and Co^{2}L [4]. As earlier methods, ER and GDumb achieve very strong performance not only in their own work, but in later literature [34,3] as well. BiC is also a strong method in the class-incremental learning setting. DER++ and Co^{2}L are the latest SOTA methods. We chose a medium and a large buffer size for these rehearsal-based methods, based on recommendation of prior work [3,40,4,59].

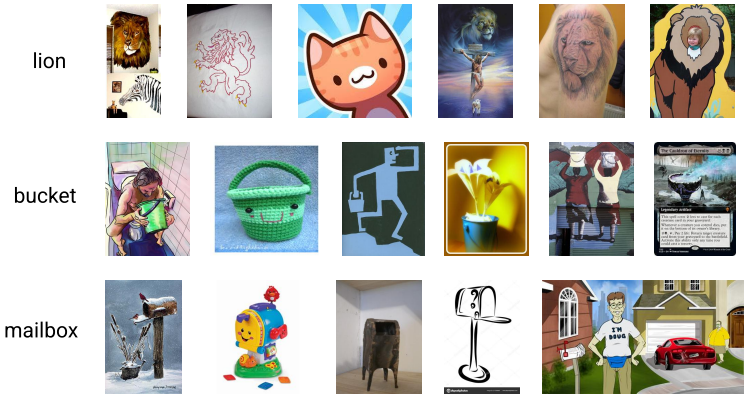

Fig. 5: Representative examples from Split ImageNet-R.

– **Prompt-based method.** L2P [59]is the current state-of-the-art prompt-based method, we configure DualPrompt to have the similar amount of additional parameters as L2P for fair comparison.
– **Architecture-based methods.** All these methods are based on ResNet-18, as recommended in the original work. We either directly taken reported results in their original work or strictly follow the original implementation and hyper-parameter settings of these methods to reproduce the results. Some other well-known methods [17,11,45] are not compared here because they either have been outperformed by our compared methods or only verified on simpler task-incremental setting.

E Split ImageNet-R: large intra-class diveristy

Figure 5 shows some representative examples of three different classes from Split ImageNet-R. We can observe that although the images of the same row share the same label, they actually differ a lot. This observation accords well with the result in Figure 1 and Table 1 that rehearsal-based methods require a large buffer size to perform well due to large intra-class diversity in Split ImageNet-R.

F Searching for multi-layered prompts

Based on the strategy in 4.2, we first search for the multi-layered E-Prompt on the validation set of Split ImageNet-R. Due to the large search space, we made a simplified assumption that the MSA layers to attach prompts should be contiguous. Moreover, since $\texttt{start}_e = \texttt{end}_e = 5$ is the best option in the single layer case, it is natural to include the 5-th layer when searching for multi-layered E-Prompt. Nevertheless, we also include several cases when the 5-th layer is not included for completeness.

Table 5: Searching results for multi-layered E-Prompt.

i_g	j_g	Avg. Acc
5	5	65.55
3	4	65.76
4	5	66.59
5	6	66.12
3	**5**	**67.12**
4	6	66.41
5	7	64.53
1	12	67.09

Table 6: Searching results for multi-layered G-Prompt.

i_e	j_e	Avg. Acc
2	2	67.70
1	**2**	**68.46**
1	3	67.79
1	5	67.73
6	8	65.10
1	12	63.13

We discover that $\texttt{start}_{\boldsymbol{e}} = 3, \texttt{end}_{\boldsymbol{e}} = 5$ yields the best performance in terms of average accuracy. Note that when we attach E-Prompt to every MSA layer ($\texttt{start}_{\boldsymbol{e}} = 1, \texttt{end}_{\boldsymbol{e}} = 12$), it actually leads to comparable accuracy. However, we still choose $\texttt{start}_{\boldsymbol{e}} = 3, \texttt{end}_{\boldsymbol{e}} = 5$ since it has less additional parameters.

We then fix $\texttt{start}_{\boldsymbol{e}} = 3, \texttt{end}_{\boldsymbol{e}} = 5$, and search for multi-layered G-Prompt, given that we have $\texttt{start}_{\boldsymbol{g}} = \texttt{end}_{\boldsymbol{g}} = 2$ yields the best performance as a single-layered E-Prompt. We conduct similar searching process, with a preference of including the 2nd layer. The searching results are shown in Table 6.

We discover that $\texttt{start}_{\boldsymbol{g}} = 1, \texttt{end}_{\boldsymbol{g}} = 2$ leads to best average accuracy. Moreover, simply share all MSA layers results in overall negative effect on the accuracy.

Although our search strategy is not exhaustive, we find the combination of $\texttt{start}_{\boldsymbol{g}} = 1, \texttt{end}_{\boldsymbol{g}} = 2, \texttt{start}_{\boldsymbol{e}} = 3, \texttt{end}_{\boldsymbol{e}} = 5$ works quite well for all benchmark datasets.

G Searching for prompt length

Following the suggestion by [59], we use set the base length of prompts as 5, and perform grid search on the lengths of G-Prompt and E-Prompt from $\{5, 10, 20, 40\} \times \{5, 10, 20, 40\}$, based on the optimal positions obtained in the previous step.

As shown in Figure 6, we indeed verify $L_g = 5, L_e = 20$ is the optimal choice on the validation set of Split ImageNet-R. We also empirically observe this configuration works well on other datasets, thus we use this combination of prompt lengths for other datasets in experiments showed in the main text.

H Additional results on 5-datasets

For completeness, we also demonstrate the effectiveness of our method on 5-datasets [11], which is a collection of five diverse image classification datasets, CIFAR-10 [21], MNIST [23], Fashion-MNIST [62], SVHN [42], and notMNIST [2].

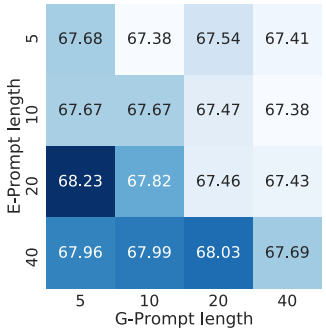

Fig. 6: Grid search on prompt length.

Table 7: Additional results on 5-datasets [11].

Method	**Buffer size**	**5-datasets** Avg. Acc (↑)	Forgetting (↓)
ER	250	80.32±0.55	15.69±0.89
BiC		78.74±1.41	21.15±1.00
DER++		80.81±0.07	14.38±0.35
Co^2L		82.25±1.17	17.52±1.35
ER	500	84.26±0.84	12.85±0.62
BiC		85.53±2.06	10.27±1.32
DER++		84.88±0.57	10.46±1.02
Co^2L		86.05±1.03	12.28±1.44
FT-seq	0	20.12±0.42	94.63±0.68
EWC		50.93±0.09	34.94±0.07
LwF		47.91±0.33	38.01±0.28
L2P		81.14±0.93	4.64±0.52
DualPrompt		**88.08±0.36**	**2.21±0.69**
Upper-bound	-	93.93±0.18	-

Despite the simplicity of each task in 5-datasets, the benchmark mimics the real-world setting where task diversity is large, thus contributing to a more comprehensive evaluation of CL methods. Since each task in 5-dataset is relatively easier than that of Split CIFAR-100 and ImageNet-R, rehearsal-based methods generally require smaller buffer size to perform well. However, a buffer size of 500 is already considered large for 5-datasets [11,40]. Although DualPrompt still consistently outperforms competing methods, we argue that in real world continual learning scenarios, it is rare that tasks are too diverse: for example, we don't expect a digit classifier to continually learn to classify animals. Thus, we only show the performance on 5-datasets as a proof-of-concept that our method works well when task diversity is large.

Table 8: Comparison between DualPrompt with query strategy introduced in Section 4.1, and with the perfect match (known test time task identity) on Split ImageNet-R.

Method	**Matching Acc**	**Avg Acc (↑)**	**Forgetting (↓)**
DualPrompt-Query	55.8	68.13	4.68
DualPrompt-Perfect Match	100	71.97	3.95

I Relationship between query accuracy and performance

To demonstrate the relationship between query accuracy and performance, we compare DualPrompt with the query strategy introduced in 4.1 to DualPrompt with known test time task identity to select E-Prompt. The result is shown in Table 8. Interestingly, although the matching accuracy is not that great, DualPrompt is quite robust to it and still achieves an accuracy very close to perfect match. We contribute this robustness to mismatching to the design of our method. First, the task-invariant instruction captured in G-Prompt still remains useful for prediction even if E-Prompt is noisy. Second, the query strategy is based on the input feature, thus implicitly taking into account task similarity. Even when mismatching happens, our method tends to choose the E-Prompt from one of the most similar tasks. We also note that their is still forgetting even if we use ground truth task identity to select the corresponding task-specific E-Prompt. This part of forgetting results from the bias in the final softmax classification head [34,46,67], a common issue in class-incremental learning that could be mitigated in parallel.