In the appendix, we additionally present detailed algorithms of the proposed S-liPrompts, thorough experimental settings of the competing methods and the proposed ones, more ablation experiments, as well as more discussions on related works.

1.1 Algorithm Details

**Inference Pipeline.** Fig.1 illustrates the inference pipeline of the proposed language-image S-Prompts (S-liPrompts). For the inference phase, we suggest the following steps: 1) feeding the tokens of a given test image into the image transformer to obtain the image feature, 2) utilizing K-NN to search for the nearest centroid obtained by K-Means on training data to the given test image, 3) feeding the image token and the nearest centroid associated image prompts into the image transformer, with the centroid related language prompts and class tokens being fed into the text transformer, 4) calculating the language-image prediction probabilities for the final classification. Note that K-NN is conducted on the image feature space of the fixed pre-trained image transformer without using any prompts. **Algorithm Details.** Alg.1 and Alg.2 show the training/testing process details of the proposed S-liPrompts.

1.2 Experimental Details

**Implementation details.** We implement the proposed S-Prompts in PyTorch with NVIDIA RTX 3090 GPUs. We use the same image encoder architecture of ViT-B/16 [1] for S-iPrompts and S-liPrompts across all tasks/domains. The text encoder of S-liPrompts is the same as that of CLIP [23]. In our experiments, the total class number $C$ is 2 for CDDB [12], 50 for CORe50 [18], and 345 for DomainNet [22] respectively. For all the experiments in the main paper, the length of a single image prompt $L_i$ is 10, and the length of one language prompt $L_l$ is 16. The embedding dimension $D_i$ is 768 for both of the used ViT and CLIP. The dimension of word embedding $c_i$ and the embedding dimension of text transformer $D_l$ are both 512. The implementation of K-Means and K-nearest neighbors (KNN) are both based on scikit-learn [21]. The distance measure is the L1 distance.

**Training details.** Our S-Prompts is insensitive to the setting of hyper-parameters and we do not tune much. For all benchmarks, we adopt SGD optimizer with a momentum of 0.9, an initial learning rate

---

This work is done when Yabin Wang visits Singapore Management University  
Corresponding Author

---

Algorithm 1 Training process of the proposed S-liPrompts

1: Initialization: Training data: $\mathbb{D}_s = \{x^s_i, y^s_i\}_{i=1}^{N_s}, s = 1, 2, \ldots, S$; Pre-trained image model: $f$; Pre-trained language model: $g$; Class embeddings: $c_j, j = 1, 2, \ldots, C$; Image Prompt Pool: $P^i_s = \emptyset$; Language Prompt Pool: $P^d_s = \emptyset$; Domain centroids: $M = \emptyset$; Maximum Iteration: $T_{max}$; Batch size: $N_{bs}$; Number of centroids per session: $K$;

2: for $s = 1, 2, \ldots, S$ do
3: Initialize image prompt $P^i_s$ for domain $s$
4: Initialize language prompt $P^d_s$ for domain $s$
5: for $\text{iter} = 1, 2, \ldots, T_{max}$ do
6: Extract a mini-batch samples $\{x^s_i, y^s_i\}_{i=1}^{N_{bs}}$ from Training Data $\mathbb{D}_s$
7: Prepend $x^s_i$ with image prompts $P^i_s$ for image encoder by $[x^s_i, P^i_s, x_{cls}]$
8: Prepare input with language prompts $P^d_s$ for text encoder by $t_j = \{P^d_s, c_j\}_{j=1}^C$
9: Generate class representation by $g(t_j)$
10: Compute the prediction probability by $p = \frac{\exp(f([x^s_i, P^i_s, x_{cls}]), g(t_j))}{\sum_{k=1}^C \exp(f([x^s_i, P^i_s, x_{cls}]), g(t_k))}$
11: Calculate Cross-Entropy loss by $\mathcal{L}_B = \mathcal{L}(\sigma(p), y^s_i)$
12: Update $\theta_{P^i_s} = \theta_{P^i_s} - \eta \nabla_{\theta_{P^i_s}} \mathcal{L}_B$
13: Update $\theta_{P^d_s} = \theta_{P^d_s} - \eta \nabla_{\theta_{P^d_s}} \mathcal{L}_B$
14: end for
15: Initialize a temporary memory buffer $B$ to store features
16: for $i = 1, 2, \ldots, N_s$ do
17: Extract a sample $x^s_i$ from training data $\mathbb{D}_s$
18: $B \leftarrow f([x^s_i, x_{cls}])$
19: end for
20: Run K-Means on $B$ to get $K$ domain centroids $\{m^x_i\}_{i=1}^K$
21: $M \leftarrow \{m^x_i\}_{i=1}^K$
22: $P^i_s \leftarrow P^i_s$
23: $P^d_s \leftarrow P^d_s$
24: end for
We use the official implementation of DyTox originally suggests training base-ConViT from scratch. But this case’s performance (about 56%) is much lower than that (86.21%) of training from the pre-trained base-ConViT (78M), which is also higher than that (84.20%) of training from the pre-trained base-ViT (86M), on the CDDB-Hard dataset.

DyTox [2] and L2P [31] are two main baselines in this paper, and we list the parameters below for reference.

We implement EWC [9] and LwF [15] with pre-trained ViT-B/16 using the toolbox [30]. The hyper-parameters remain the same as those in their original paper. DyTox [2] and L2P [31] are two main baselines in this paper, and we list the parameters below for reference.

We use the official implementation of DyTox for its evaluation on CDDB, CORe50 and DomainNet. The original backbone of DyTox is a shallow ConViT [3] (16 layers base-ConViT (78M) is almost the same as the base-ViT (86M) used in other methods including L2P and ours. We use the learning rate of 0.0001 and 50 epochs for CDDB to get the proper performance. For DomainNet and CORe50, the learning rate is 0.001 and the number of epoch is 20. All the other parts of DyTox remain the same as those of the official implements, and the changes of learning rate and the number of epochs are for getting better performance. We evaluate the official L2P code based directly on CDDB and DomainNet by tuning its key parameters, i.e., learning rate, trade-off parameter $\lambda$, prompt pool size, $N$ value for key selection. The learning rate for CDDB is tuned to 0.1, and that for DomainNet is 0.8. We use 30 epoch for CDDB and 20 for DomainNet. For both datasets, the prompt pool size is tuned to 20, and the number of key selection is 5. The trade-off parameter $\lambda$ is adjusted to 0.001.
Table 1: Average accuracies (%) of using random domain selection and our proposed domain selection on CDDB and DomainNet for domain incremental learning. The two domain selection cases are both based on the proposed S-liPrompts.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Random Domain Selection</th>
<th>Proposed Domain Selection</th>
<th>Relative Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDDB</td>
<td>80.12</td>
<td>88.65</td>
<td>8.53</td>
</tr>
<tr>
<td>DomainNet</td>
<td>49.94</td>
<td>67.78</td>
<td>17.84</td>
</tr>
</tbody>
</table>

Table 2: Average accuracies (%) of the originally proposed S-liPrompts (each inference uses one single domain-selected CLIP prediction) and its ensemble version (each inference utilizes the voting strategy on all domain-based CLIP predictions) on the CDDB and DomainNet datasets for domain incremental learning.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Voting for Ensemble</th>
<th>Proposed Domain Selection</th>
<th>Relative Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDDB</td>
<td>65.41</td>
<td>88.65</td>
<td>23.18</td>
</tr>
<tr>
<td>DomainNet</td>
<td>58.85</td>
<td>67.78</td>
<td>8.93</td>
</tr>
</tbody>
</table>

Figure 2: Accuracy curves of the proposed S-liPrompts generalized from five seen CDDB-Hard domains (S1-S5: GauGAN, BigGAN, WildDeepfake, WhichFaceReal, SAN) to three unseen domains (out-of-domain) OOD1-OOD3: FaceForensic++, Glow, StarGAN, which were used by CDDB [12]. The value (S1-S5) along the horizontal direction indicates the S-liPrompts model when trained until the given seen domain data. For example, S3 means that the model was trained along the stream of S1, S2 and S3 sequentially. The curves with different colours indicate the detection performance on the eight seen/unseen domains (S1-S5 and OOD1-OOD3) respectively, and the red one shows the average accuracy.

1.3 More Experiments

Random Domain Selection vs. Proposed Domain Selection. In addition to the comparison on CDDB (Table 4 in the main paper), we further compare the case of performing random domain selection against that of using our proposed domain selection on DomainNet. Table 1 reports the results on these two datasets. It shows the consistently remarkable relative improvements (by more than 8% on CDDB and about 18% on DomainNet) of the proposed domain selection over the random domain selection. This considerable improvement justifies the necessity of the proposed domain selection, which attributes to the significance of the proposed independent prompting paradigm. The out-of-distribution (OOD) experiment in Table 7 (in the main paper) verifies that the random domain selection (in Table 1) has promising results mainly due to the strong generalization of the proposed language-image prompt learning scheme over the CLIP model. Nevertheless, the results can be further improved significantly using the proposed domain selection (see Table 1), showing the clear superiority of the proposed independent prompting paradigm.

Domain Ensemble vs. Proposed Domain Selection. We further study one of the most popular ensemble methods (i.e., voting) on the proposed S-liPrompts. Table 2 summarizes the comparison of the proposed S-liPrompts against the voting-based version in terms of average accuracies on CDDB for deepfake domain incremental learning. The significant relative increases (about 23% on CDDB, 9% on DomainNet) show that the proposed S-liPrompts method favors the proposed domain selection method that chooses the most expertised (i.e., the most related domain-based) prompting model for the inference on each test sample. The excellent performance of the proposed domain-based prompting models verifies the significance of the proposed independent prompting scheme. For the
voting-based inference on each test sample, we first feed all the learned domain-based prompts to the CLIP models one by one resulting in multiple predictions, and we then use the majority voting strategy on the individual results to get the final prediction. The clearly superior performance of the proposed S-liPrompts with domain selection mainly stems from the most expertised model for the given test sample. Except for this model, the rest ones are less expertised for the given test sample, and they might be dominant to corrupt the final prediction when doing the majority voting. In this
Figure 4: Failure examples that are predicted to correct domains but are classified incorrectly using the proposed S-liPrompts on CDDB-Hard. Domain1-Domain5: GauGAN, BigGAN, WildDeepfake, WhichFaceReal, SAN. All images with a green boundary are real, and all images with a red boundary are fake.

case, the voting scheme could lead to clear performance degradation. Moreover, it requests running all the learned CLIP-based prompting models for the voting on each test sample, and thus it is much more time-consuming than the proposed inference scheme.

Test on Unseen Domains. Based on Table 7 in the main paper, Fig additionally shows the detection accuracy curves of the proposed S-liPrompts when applied to three unseen domains (FaceForensic++, Glow and StarGAN). The S-liPrompts models were trained on the five seen domains (GauGAN, BigGAN, WildDeepfake, WhichFaceReal and SAN) from CDDB-Hard sequentially. As studied in
Table 3: Detection accuracies (%) of the proposed S-liPrompts/S-iPrompts and the main competing methods (L2P and DyTox) on the CDDB dataset for deepfake domain incremental learning. **Bold**: best results. **Underline**: second best results.

<table>
<thead>
<tr>
<th></th>
<th>GauGAN</th>
<th>BigGAN</th>
<th>WildDeepfake</th>
<th>WhichFaceReal</th>
<th>SAN</th>
<th>Average</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-liPrompts</td>
<td>99.30</td>
<td>96.75</td>
<td>82.06</td>
<td>96.25</td>
<td>68.89</td>
<td><em>88.65</em></td>
<td>68.89</td>
<td><em>99.30</em></td>
</tr>
<tr>
<td>S-iPrompts</td>
<td>90.30</td>
<td>81.88</td>
<td>72.76</td>
<td>84.25</td>
<td>43.30</td>
<td>74.50</td>
<td>43.30</td>
<td>90.30</td>
</tr>
<tr>
<td>L2P</td>
<td>80.73</td>
<td>62.60</td>
<td>58.98</td>
<td>57.48</td>
<td>46.59</td>
<td>61.28</td>
<td>46.59</td>
<td>80.73</td>
</tr>
<tr>
<td>DyTox</td>
<td>48.91</td>
<td>50.00</td>
<td>59.19</td>
<td>50.00</td>
<td>51.62</td>
<td>48.91</td>
<td>51.62</td>
<td>59.19</td>
</tr>
</tbody>
</table>

Table 4: Precisions (%) of the proposed S-liPrompts/S-iPrompts and the main competing methods (L2P and DyTox) on the used CDDB dataset for deepfake domain incremental learning. Here precision is the ratio \( tp/(tp + fp) \) where \( tp \) is the number of true positives and \( fp \) is the number of false positives. **Bold**: best results. **Underline**: second best results.

<table>
<thead>
<tr>
<th></th>
<th>GauGAN</th>
<th>BigGAN</th>
<th>WildDeepfake</th>
<th>WhichFaceReal</th>
<th>SAN</th>
<th>Average</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-liPrompts</td>
<td>99.80</td>
<td>98.45</td>
<td>81.97</td>
<td>96.54</td>
<td>75.76</td>
<td><em>90.50</em></td>
<td>75.76</td>
<td><em>99.80</em></td>
</tr>
<tr>
<td>S-iPrompts</td>
<td>90.95</td>
<td>83.64</td>
<td>73.21</td>
<td>84.08</td>
<td>43.75</td>
<td>75.13</td>
<td>43.75</td>
<td>90.95</td>
</tr>
<tr>
<td>L2P</td>
<td>83.10</td>
<td>77.29</td>
<td>65.39</td>
<td>57.79</td>
<td>66.67</td>
<td>70.05</td>
<td>57.79</td>
<td>83.10</td>
</tr>
<tr>
<td>DyTox</td>
<td>52.99</td>
<td>50.00</td>
<td>85.86</td>
<td>50.00</td>
<td>50.00</td>
<td>57.77</td>
<td>50.00</td>
<td>85.86</td>
</tr>
</tbody>
</table>

the main paper, the results in Fig.2 demonstrate that the proposed S-liPrompts owns an excellent generalization ability on the unseen domains.

**Successes and Failures.** Fig.3 lists some success cases which are predicted to incorrect domains but are classified correctly in the end using S-liPrompts on CDDB-Hard. It is interesting to see that most of them are predicted to a semantically close domain. For example, as shown in the first two rows in Fig.3 some faces from Domain 1, 2, 4, 5 are predicted to either Domain 3 (WildDeepfaces) or Domain 4 (WhichFaceReal), where all the images are of faces as shown by the exemplars of Domain 3 and Domain 4 in row 4 and row 5 of Fig.3. In this case, they are very likely to be classified correctly, due to the excellent learning power of S-liPrompts on Domain 3 (WildDeepfaces) and Domain 4 (WhichFaceReal). The same observation can be derived on those images at row 3 in Fig.3. As their architecture details are very similar to those in the exemplars of Domain 5 (SAN), they are assigned to Domain 5 that still leads to the right classification. Fig.4 shows some classical failure cases. As we can see in Table 6 (in the main paper), the challenging cases are mostly on the domains of WildDeepfake and SAN. For the SAN cases, we can see that some irregular-looking real buildings (e.g., wired structures) are misclassified as fake, while those fake buildings with realistic-looking structures are classified as real, as displayed in Fig.4. For the WildDeepfake domain, there exist a large number of real faces that are often of blurry or occluded or low contrast or non-frontal. These real faces are generally hard for real/fake detection. On the other hand, the fake faces are often of the same quality. This highly increases the challenge for the deepfake detection.

**Feature Spaces of Competing Methods.** Fig.5 uses t-SNE to visualize the resulting feature spaces of pre-trained ViT, CLIP, the two most competing methods DyTox [2] and L2P [31] as well as our S-iPrompts and S-liPrompts on CDDB-Hard. As we discussed in the main paper, dependent learning across domains is very likely to mix up the subspaces of old/new domains. The t-SNE results of L2P and DyTox can verify this to some extent. In contrast, our proposed S-Prompts (especially S-liPrompts) is capable of producing much more separable subspaces. This demonstrates that our independent learning could reach a more favorable feature space where subspaces of different domains are pushed far away from each other. This property could lead to better performances that were reported in the main paper, showing the clear superiority of the proposed independent prompting paradigm over the commonly-respected dependent learning principle.

**Metrics to Measure Separation Degree.** We used t-SNE as it is a popular choice for visualization. Though the significant superiority of our proposed S-liPrompts is shown clearly via t-SNE, it is mainly in terms of domain separation rather than class separation. This phenomenon is mainly from the fact that t-SNE visualization is limited to the 2-dim projection of the original high-dimensional data. Therefore, apart from using t-SNE visualization, we additionally use quantitative metrics to measure the degree of class separation domain by domain. Accordingly, we (re-)collected the classification/detection accuracies in Tables 1 & 2 (main paper), which reflects the average domain-wise accuracies and thus can be a favorable metric for this purpose. Following the related paper [12], we additionally introduce a precision based metric to measure the domain-wise class separation
Figure 5: t-SNE visualization on the resulting feature spaces of pre-trained ViT, proposed S-iPrompts, DyTox [2], L2P [31], pre-trained CLIP, and proposed S-liPrompts on the CDDB-Hard dataset.

Table 5: Memory overheads of the proposed S-iPrompts (ViT-based), S-liPrompts (CLIP-based), the ViT-based prompting methods (DyTox and L2P), and ViT-based non-prompting ones (Others without expansion) on the CDDB dataset. In the setup, there are 5 sessions, each of which includes 2 classes (real and deepfake classes). The average increase corresponds to the parameter increase per session.

<table>
<thead>
<tr>
<th></th>
<th>DyTox</th>
<th>L2P</th>
<th>Proposed S-iPrompts</th>
<th>Proposed S-liPrompts</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Model</td>
<td>86M</td>
<td>86M</td>
<td>86M</td>
<td>201M</td>
<td>86M</td>
</tr>
<tr>
<td>Total Increase</td>
<td>7.09M</td>
<td>92.16K</td>
<td>0.26M</td>
<td>0.40M</td>
<td>0</td>
</tr>
<tr>
<td>Average Increase</td>
<td>1.42M</td>
<td>18.43K</td>
<td>52.22K</td>
<td>80.89K</td>
<td>0</td>
</tr>
<tr>
<td>Relative Increase (Increase/Base)</td>
<td>1.65%</td>
<td>0.02%</td>
<td>0.05%</td>
<td>0.03%</td>
<td>NA</td>
</tr>
</tbody>
</table>

The results in Table 3 and Table 4 demonstrate the consistent superiority of the proposed S-iPrompts/S-liPrompts methods in terms of the class separation using such various metrics.

Memory Consumption Comparison. We further report the memory overheads of the proposed S-iPrompts/S-liPrompts methods (ViT/CLIP-based), the ViT-based prompting methods (DyTox and L2P) as well as those ViT-based non-prompting methods (without expansion). As the non-prompting methods do not expand their architectures for the increasing tasks, their model parameters keep the same for the learning process. Therefore, the main comparison is on the prompting based methods.
Table 6: Accuracies (%) of the exemplar-based DyTox using two different backbones (i.e., base-ViT [1] and base-ConViT [3]) on the three benchmarks.

<table>
<thead>
<tr>
<th>Backbones Used by Competing Methods</th>
<th>DyTox using Pre-trained base-ViT</th>
<th>DyTox using Pre-trained base-ConViT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDDB</td>
<td>84.20</td>
<td>86.21</td>
</tr>
<tr>
<td>CORe50</td>
<td>80.11</td>
<td>79.21</td>
</tr>
<tr>
<td>DomainNet</td>
<td>60.83</td>
<td>62.94</td>
</tr>
</tbody>
</table>

Table 7: Accuracies (%) of the proposed S-iPrompts and the main competing exemplar-free methods (L2P and DyTox) on the three used datasets. Note that CDDB is for continual binary-class classification, and CORe50 and DomainNet are both for continual multi-class classification. DyTox generally fails for the two multi-class classification tasks (CORe50 and DomainNet), as it requests a distillation on exemplars for the more challenging balance problem among multiple classes from new/old domains.

<table>
<thead>
<tr>
<th>Method</th>
<th>Prompting Scheme</th>
<th>Corresponding Method</th>
<th>CDDB</th>
<th>CORe50</th>
<th>DomainNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>DyTox</td>
<td></td>
<td></td>
<td>61.28</td>
<td>78.33</td>
<td>40.15</td>
</tr>
<tr>
<td>L2P</td>
<td></td>
<td></td>
<td>51.27</td>
<td>Fails</td>
<td>50.62</td>
</tr>
<tr>
<td>Proposed S-iPrompts</td>
<td>independent image prompting + independent language prompting</td>
<td>CLIP+ S-iPrompts</td>
<td>72.29</td>
<td>83.13</td>
<td>50.62</td>
</tr>
<tr>
<td>Relative Improvement</td>
<td></td>
<td></td>
<td>13.23</td>
<td>4.80</td>
<td>10.47</td>
</tr>
</tbody>
</table>

including ours. As shown in Table 5, the proposed S-iPrompts increases the model parameters for the domain-specific prompts that are of 52.22K (0.05% relative increase) for each domain (session), and the proposed S-liPrompts’s increase is of 80.89K (0.03% relative increase) per session. By comparison, DyTox needs to add one task-attention block over the ViT-based model and dynamically expands the domain-specific tokens (prompts) every session. L2P is like ours directly using ViT-based model without adding any neural network blocks, and it assigns a certain memory to save the pool of used prompts once for the initialization. From the results, we can see that DyTox’s increase is the most, and that of L2P is the least. The relative increase ratio of the proposed S-liPrompts/S-iPrompts is very close to that of L2P, while obtaining clearly better performances than the other competing methods.

Backbones Used by Competing Methods. As presented in the main paper, to compare fairly, we used the same backbone (i.e., base-ViT [1]) for all the real competitors (except for DyTox that used a more advanced ViT model, i.e., ConViT [3]) as well as the proposed methods (S-iPrompts, S-liPrompts) for all the experiments (Table 1, 2, 3). We reported the results of DyTox with base-ConViT [3] rather than base-ViT [1] in the main paper, since the performance (86.21%) of the former model is better than that (84.20%) of the latter model on CDDB. Hence, we follow the suggestion of the original DyTox paper to use a more advanced ViT model (ConViT) for DyTox in the main paper. Besides, DyTox with base-ViT performs very badly on CDDB with the average accuracy being 48.89%, which is even worse than that of the random guess on CDDB. We further evaluate DyTox with base-ViT on the other two datasets (CORe50 and DomainNet), and we find that the ViT-based exemplar-free DyTox still fails on these two datasets as discovered on the ConViT-based DyTox in the main paper. This is mainly because DyTox requests a distillation on exemplars, which are more demanded to balance the multiple classes from new/old domains on CORe50 and DomainNet, for a promising prompt learning. Therefore, we report the results of the better DyTox (DyTox with base-ConViT) in the main paper, and we here merely report its results using exemplars on the three benchmark datasets. The results of Table 7 show that DyTox with base-ConViT generally outperforms (or at least comparable with) DyTox using base-ViT on the three used datasets.

Table 8: Accuracies (%) of ablating components of the proposed S-liPrompts on the CDDB dataset. Note that we use the official implements of CLIP to do Zero-shot classification on three benchmarks without any change. Here we use the text templates of ImageNet as their implementation for CLIP (Zero-shot). LIP: Language-Image Prompt, LP: Language Prompt.

<table>
<thead>
<tr>
<th>Method</th>
<th>Prompting Scheme</th>
<th>Corresponding Method</th>
<th>CDDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLIP (Zero-shot)</td>
<td>handcrafted language prompts</td>
<td>Vanilla CLIP</td>
<td>49.52</td>
</tr>
<tr>
<td>S-liPrompts</td>
<td>dependent image prompts tuning + dependent language prompts tuning</td>
<td>CLIP + L2P</td>
<td>64.90</td>
</tr>
<tr>
<td>S-liPrompts</td>
<td>independent image prompting + independent language prompting</td>
<td>CLIP+ S-liPrompts</td>
<td>88.65</td>
</tr>
<tr>
<td>S-liPrompts</td>
<td>independent image prompting + independent language prompting</td>
<td>CLIP+ S-liPrompts</td>
<td>88.65</td>
</tr>
</tbody>
</table>

Table 9: Accuracies (%) of ablating components of the proposed S-liPrompts on the CORe50 and DomainNet datasets. LIP: Language-Image Prompt.
Contribution of Independent Image-end Prompting. Table 7 reports the results that are copied from Table 1, 2, 3 of the main paper. The considerable relative improvement (13% on CDDB, 5% on CORe50, 10% on DomainNet) of the proposed S-iPrompts (independent prompting) over the two main competitors L2P and DyTox (dependent prompting) justifies the significant superiority of the contribution of independent image prompting. We conduct this comparison fairly in the scenario of exemplar-free domain incremental learning (DIL), which is the main aim of our paper, i.e., better data security, privacy and less memory consumption for DIL. In the context of exemplar-free DIL, DyTox collapses on CORe50 and DomainNet, because it requires a distillation on selected exemplars for a better balance among multiple classes from new/old domains on these two datasets, which seems essential for a promising dependent prompting [2]. In addition, we also compared the proposed S-iPrompts against those state-of-the-art exemplar-based methods for a reference. We are delighted to find that S-iPrompts (exemplar-free) outperforms the exemplar-based DyTox clearly (about 4% increase) and the exemplar-based L2P (about 2% improvement) on CORe50, although it is outperformed by the exemplar-based DyTox on CDDB and DomainNet using a large number of exemplars (e.g., 17,250 exemplars on DomainNet). The cost of using large exemplar data is generally not favorable in the real-world scenarios.

Contribution of Independent Language-Image Prompting. Table 8 presents the results that are mainly from Table 4 in the main paper, with additional notes in the second and third columns for further clarification. As shown in Table 8, performing CLIP or CLIP+S-liPrompts directly cannot achieve improved performance over S-liPrompts using ViT as a backbone. In contrast, applying the proposed language-image prompting scheme on the CLIP model makes remarkable improvement. From the comparison between S-liPrompts (final, \( K = 5 \)) and S-liPrompts (fixed Language Prompt weights), we can find the contribution of language-image prompting brings an significant improvement of about 16% over the proposed S-liPrompts that is based on the contribution of independent image-end prompting. By comparing S-liPrompts (fixed Image & Language Prompt Length) against S-liPrompts (fixed Language Prompt weights), we can see the considerable superiority (about 8% relative improvement) of S-liPrompts (independent image-end prompting) over L2P-like learning paradigm (dependent image-end prompting), which verifies the significant contribution of independent image-end prompting again. Additionally, the results in Table 1, 2, 3 of the main paper show that our best model S-liPrompts supasses the best of the state-of-the-art exemplar-free methods significantly (30% relative improvement on average) for the three standard DIL benchmark datasets, and even outperforms them clearly (an average increase of 6%) when they use exemplars.

In addition, we evaluate the ablation on CORe50 and DomainNet. Table 9 shows the results. The observation on these two datasets is consistent with that on CDDB. In particular, for DomainNet, the considerable improvement of the final S-liPrompts model demonstrates its clear superiority over the ablated model. In comparison, the improvement on CORe50 is clear but relatively smaller for the CLIP case, while the increase is remarkable when comparing our ViT-based method (S-iPrompts) against ViT-based L2P on CORe50. This might be because CORe50’s domain gaps are not as large as those of CDDB and DomainNet (the CORe50 domains are captured by the same Kinect sensor under different backgrounds and lighting [18]). On CORe50, we find most of the evaluated methods have much less forgetting compared to those on CDDB and DomainNet due to the smaller domain gaps of CORe50. In this case, the more powerful language-image prompting scheme might share beneficial knowledge from those similar domains, which does not harm the performance a lot. In contrast, the relative improvements are very remarkable on CDDB (Table 8) and DomainNet (Table 9), showing the significant superiority of the proposed prompting method in the cases with large domain gaps.

1.4 More Discussions on Related Works

As presented in the main paper, there exist some expansion-based continual learning methods like [28, 11, 8, 25, 7, 5, 14]. They either dynamically expand the network architectures or reshape their internal structures task by task. This results in the following three issues: 1) Though they generally separate the structure and parameter learning significantly, they still request effective sharing of parameters among tasks. The sharing-driven network tuning inevitably results in a non-trivial trade-off (or so-called tug-of-war) between old task learning and new task learning [24, 10]. 2) They

generally require that the task indexes should be provided for the inference. 3) They often result in a significantly growing number of model parameters for a large number of tasks.

To overcome the second issue of the expansion-based methods, [27] [16] suggest learning a single classifier on a concatenation of all resulting feature maps from different subsets of parameters, which connects all the tasks/domains. Nevertheless, it suffers from the dramatically increased memory overhead issue when handling a long task sequence, which thus requires a complex post-processing for pruning. To overcome this large memory consumption (i.e., the third issue mentioned above), [2] suggests expanding prompts instead over a transformer network. The prompt expansion leads to a small amount of parameter increase. Yet, as discussed in the main paper, like many of traditional methods, the dependent learning across tasks always goes for a tug-of-war (i.e., the first issue mentioned above), where it can only achieve a trade-off between transferring and forgetting, and thus it is almost impossible to achieve the best for every task/domain. By comparison, our paper breaks the commonly-respected learning principle and instead establishes a new independent prompting paradigm to play a win-win game, where it can achieve the best for each domain. In addition, our proposed S-Prompts can address another two issues well with a rough domain identifier and a negligible parameter increase through the proposed prompting paradigm.

There have been also emerging some other transformer-based incremental learning methods like [29] [8] [13], and other incremental prompting methods like [17] [26]. The transformer-based methods without prompting demand much more tuning on the whole network than the prompting methods like [31] and ours. Also, they keep applying traditional dependent learning paradigm to play the tug-of-war game. The prompting method [17] suggests learning episodic memory prompts to explicitly carry previously learned knowledge to subsequent tasks for class-incremental event detection. However, it still follows the conventional learning paradigm to perform experience replay and knowledge distillation for the trade-off between transfer and interference.

References


