Versatile Incremental Learning: Towards Class and Domain-Agnostic Incremental Learning

Min-Yeong Park*, Jae-Ho Lee*, and Gyeong-Moon Park†

Kyung Hee University, Yongin, Republic of Korea
{pmy0792, jh.lee, gmpark}@khu.ac.kr

In this supplementary material, we further validate the effectiveness of the proposed ICON by providing as follows:

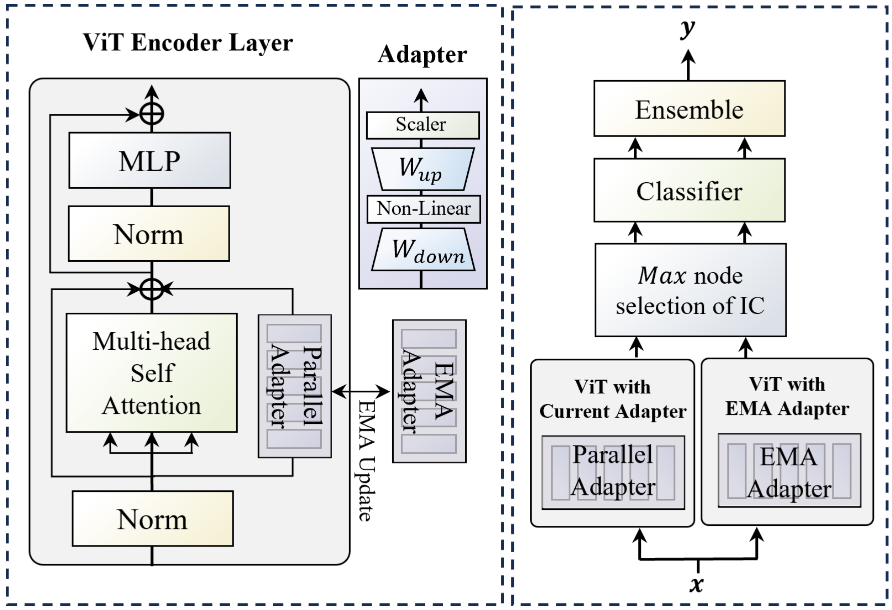
1 Architecture Details

As shown in Figure 1, our architecture has a very simple structure. Based on the typical ViT encoder layer [3], each adapter is added parallel to the attention of the layers included in the adapter locations demonstrated in Table 1. We also introduce trainable scaling parameters that have been proven its effectiveness in [6]. As shown in Figure 1a, we adopt an Exponential Moving Average (EMA) to each adapter to preserve global knowledge, with our proposed CAST. We also apply an adapter ensemble mechanism (see Figure 1b)) that takes the bigger logits from the model with the current adapter or from the model with the EMA adapter, to utilize global knowledge in the inference phase, following our baseline [5].

2 Experimental Details

Datasets. We conducted experiments on three benchmarks, including iDigits [22], CORe50 [14] and DomainNet [17] which are possible to construct an incremental learning scenario that can cause a large shift in distribution (both class and domain change) by clearly distinguishing both classes and domains. We follow [22] to compose a digit recognition incremental scenario, which is composed of four datasets: MNIST [11], SVHN [15], MNIST-M [4] and SynDigit [4]. Each dataset is treated as a different domain. CORe50 [14] is a widely used dataset for domain incremental learning or continual real-world object recognition. It has 50 classes collected from a large variety of views in time, and each

* Equally contributed
† Corresponding author



(a) ViT layer with adapter. (b) Ensemble in inference.

Fig. 1: Detailed architecture visualization of proposed ICON.

class has 11 distinct domains. In an incremental learning setting, the data from 8 domains are used for training and the data from the rest (*i.e.*, unseen) 3 domains as a test set. DomainNet [17] is a very popular dataset for domain incremental learning or domain adaptation. All the data from the 6 distinct domains and it has large 345 classes for classification. Unlike the CORe50 [14], it includes not only real-world data but also data from unreal domains such as painting, clipart and infographic.

Data Pre-processing. We adopted a very simple data augmentation strategy for all experiments, following L2P [25], DualPrompt [24], LAE [5]. First, input images are randomly resized to 224 × 224 with bilinear interpolation. Unlike the JAX [1] implementation, scale=(0.08, 1.0) and ratio=(3/4, 4/3) were applied with random resized crop as default. Second, cropped images were randomly flipped horizontally. In the inference phase, the images are resized to 256 × 256, and subsequently center cropped to 224 × 224. We used the normalization to the range of [0, 1] as the last step of each augmentation strategy.

Implementation Details. We conducted all the experiments on a single NVIDIA GeForce RTX 3090 GPU. To make fair comparisons, we used standard ImageNet [2] pre-trained ViT-B/16[1] [3] as a backbone of all methods. Furthermore,

[1] storage.googleapis.com/vit_models/imagenet21k/ViT-B_16.npz

for the unification of the implementation library (PyTorch [16]), we used the PyTorch implementation for L2P[2] [25] and DualPrompt[3] [24], which the official code is written in JAX [1]. S-Prompts [23] suggests image S-Prompts (S-iPrompts) and language-image S-Prompts (S-liPrompts). We used S-iPrompts as a comparison model without using text features (using ViT as a backbone, not CLIP [18]), for fair comparison.

Training Details. We trained the model for 5 epochs per task, 3 epochs for training only the classifier and the other 2 epochs for training both classifier and adapters while freezing other parts of pretrained ViT, following our baseline [5]. With the loss function for training, we adopted α to be 1 and β to be different among datasets. We used $\beta = 0.05$ for CORe50 and iDigits and $\beta = 0.01$ for DomainNet. For thresholds in IC, we used γ to be 2.

Hyperparameters. We summarize the hyperparameters for each dataset used in the main experiments (Table 3. of main paper) in Table 1. In Table 1, Warmup epochs means the number of epochs where the model except for the classifier is frozen, and the classifier and adapters are simultaneously trained after Warmup epochs are finished. The number of Clusters indicates the K value of the K-Means algorithm used in the proposed CAST for clustering history shifts. The coefficient of distillation and CAST in the loss function are denoted as α and β. Adapter Location represents the index of the 12 ViT layers to which the adapter is added, and index 0 points to the first layer. We add five adapters sequentially and the hidden dimension size of the feature that is entered into each adapter to be downsampled is written as Adapter Downsample. Furthermore, in our proposed VIL scenario, extreme cases of training all classes sequentially and then re-training classes already learned in other domains (*i.e.*, [CIL, CIL, CIL, ..., DIL, DIL, DIL]) can also occur, which can make the performance difference large. Therefore, we conducted experiments with various sequences (random seeds).

Our ICON does not use a prompt, but most comparison models use it. Since this is accompanied by relatively more hyperparameters than ours, such as prompt pool size, top-K, prompt length, *etc.*, we also provide the hyperparameters of the comparisons that were used in the experiments in Section 6.

Evaluation Metrics. We provide a formal definition of the evaluation metrics that were used in all experiments. Each metric is formally defined as follows:

$$\text{Average Accuracy: } A_T = \frac{1}{T}\sum_{i=1}^{T} a_{T,i}, \tag{1}$$

[2] github.com/JH-LEE-KR/l2p-pytorch
[3] github.com/JH-LEE-KR/dualprompt-pytorch

Table 1: Detailed hyperparameters and experimental configuration of proposed ICON for each dataset.

Configuration	iDigits [22]	CORe50 [14]	DomainNet [17]
Optimizer	Adam [9]	Adam [9]	Adam [9]
Base LR	0.0028125	0.0028125	0.0028125
Optimizer Betas	β_1, β_2 = 0.9, 0.999	β_1, β_2 = 0.9, 0.999	β_1, β_2 = 0.9, 0.999
Batch Size	24	24	24
EMA Decay	0.9999	0.9999	0.9999
Total Epochs	5	5	5
Warmup Epochs	3	3	3
Number of Clusters	2	3	3
α	1	1	1
β	0.05	0.05	0.01
Adapter Location	[0, 1, 2, 3, 4]	[0, 1, 2, 3, 4]	[0, 1, 2, 3, 4]
Adapter Downsample	5	5	5

where T is the total number of tasks seen until the current task, and $a_{n,i}$ is the test accuracy on task i after training the n^{th} task.

$$\text{Forgetting: } F_T = \frac{1}{T-1} \sum_{i=1}^{T-1} f_{T,i}, \tag{2}$$

where $f_{j,i}$ is a measure of forgetting on task i after training task j. $f_{j,i}$ is defined as the difference between the best accuracy achieved on task i in the past and the final accuracy of task i evaluated after training task j :

$$f_{j,i} = \max_{k \in \{1,\cdots,j-1\}} (a_{k,i} - a_{j,i}), \tag{3}$$

To validate the performance of the proposed ICON on all of the scenarios, we also took an average of Avg. Acc of all scenarios. It can be formulated as follows:

$$\text{Average: } A_{Avg} = \frac{1}{|S|} \sum_{s=1}^{S} A_s, \tag{4}$$

where $|S|$ is the total number of scenarios, and A_s is the final average accuracy of scenario s. In this paper, we used $|S|$=3, for CIL, DIL, and VIL.

3 Additional Experiments and Analysis

Robustness to Various Number of Tasks. We conducted additional experiments to validate the robustness of our method on various numbers of task sequences as shown in Table 2 with the results of standard configuration included in the main paper. The number of the entire task sequence for the VIL setting is dependent on the number of domains in it and the number of classes in a single task. For DomainNet, we conducted experiments in different task configurations

Table 2: Average accuracy on various number of tasks.

Method	iDigits [22]		CORe50 [14]			DomainNet [17]		
	8 Tasks	20 Tasks	16 Tasks	40 Tasks	80 Tasks	18 Tasks	30 Tasks	138 Tasks
LAE [5]	79.67	59.34	86.97	77.11	72.05	47.16	49.01	49.22
ICON (Ours)	**81.59**	**75.11**	**88.04**	**83.18**	**76.34**	**48.95**	**53.37**	**50.84**

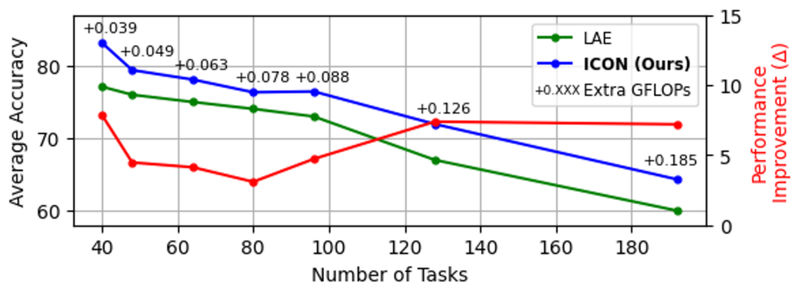


Fig. 2: Average accuracy on a higher number of tasks in CORe50. Extra GFLOPs is obtained in comparison with our baseline LAE.

so that the number of entire tasks becomes 18 and 138. Since the entire class existing in DomainNet is 345, the sequence of tasks become 18 by comprising a single task with 115 classes, and 138 by comprising a single task with only 23 classes. In the same way, we configured the number of tasks for CORe50 to be 16 and 80. 16 tasks for CORe50 are in the case of a single task with 25 classes, and 80 tasks are in the case of a single task with 5 classes. For iDigits, by having 5 classes in a single task, we can configure it to have 8 tasks entirely. As it can be seen in Table 2, in iDigits and CORe50, as the number of tasks increases, the average accuracy degrades since catastrophic forgetting becomes severe. Also, ICON achieved noticeable performance improvements compared to our baseline LAE [5] for all results on various task numbers, showing that the effectiveness of ICON is robust for various number of tasks. To thoroughly show the results on a more diverse number of tasks in CORe50, we visualize average accuracies in Figure 2. As shown in the figure, the performance improvement of ICON compared to the baseline LAE (see red line) is consistent with the number of tasks, verifying that our proposed ICON is robust with regard to the number of tasks.

Result of Joint Training. To demonstrate the gap between the proposed method and joint training results, we measured the joint training results as shown in Table 3. While the performances of our proposed ICON still have gaps compared to those of joint training, it is noteworthy that our ICON significantly

Table 3: Comparisons of joint training, the existing SOTA (LAE), and the proposed ICON in the VIL scenario.

Method	Dataset		
	iDigits [22]	CORe50 [14]	DomainNet [17]
LAE [5]	59.34±0.95	77.11±1.37	49.01±1.18
ICON (Ours)	75.11±2.39	83.18±1.21	53.37±0.47
Joint Training	**87.18±0.13**	**91.66±0.25**	**76.91±0.61**

Table 4: Comparison of computational cost on ViT-B/16.

Method	L2P [25]	DualPrompt [24]	CODA-P [19]	S-Prompts [23]	LAE [5]	**ICON (Ours)**
GFLOPs	116.27	105.89	140.12	75.32	**71.34**	<u>71.52</u>

Table 5: Experiments on different backbones in CORe50.

Dataset	ViT-S/16 [3]		ViT-L/16 [3]	
	LAE [5]	**ICON (Ours)**	LAE [5]	**ICON (Ours)**
iDigits [22]	53.40	**55.34**	69.68	**72.42**
DomainNet [17]	41.80	**45.10**	51.80	**54.09**
GFLOPs	37.09	37.27	247.81	247.99

narrowed the gaps by effectively dealing with the challenges in VIL, especially in iDigits.

Computational Complexity and Scalability. We analyze computational complexity in Table 4, which demonstrates that our method demands fewer or equivalent resources compared to the baselines, while achieving the best performances. In addition, to investigate the scalability, we conducted experiments on various sizes of backbones and extreme numbers of tasks. As shown in Table 5, the proposed ICON performed the best in all sizes of backbones with equivalent costs compared to the baseline LAE. Moreover, even when extended to extreme numbers of tasks, the proposed ICON only required negligible extra costs as indicated in Figure 2. Therefore, our method is applicable in dynamic and large-scale scenarios without issues in terms of computational complexity and scalability.

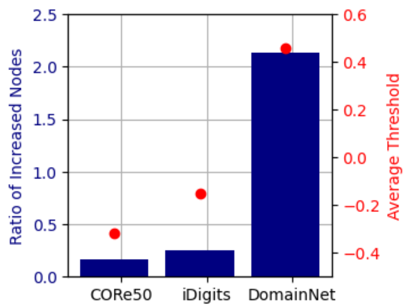


Fig. 3: Average threshold and the number of increased nodes.

Table 6: Ablation of Dynamic Threshold (DT).

DT	iDigits [22]	CORe50 [14]	DomainNet [17]
	74.79	82.71	52.99
✓	**75.11**	**83.18**	**53.37**

Analysis on Incremental Classifier. We further investigated our proposed Incremental Classifier, IC, regarding the average thresholds which are the criteria for increasing nodes for each label, and the number of increased nodes in Figure 3. In Figure 3, the ratio of increased nodes, (*i.e.* the number of increased nodes relative to the total number of classes in the entire tasks), was the most numerous in DomainNet, indicating that it is composed of images with huge domain gap among domains. In CORe50 and iDigits which have relatively smaller domain gaps, the nodes were increased less since existing nodes in the classifier can accommodate knowledge of multiple domains when a new task arrives. Also, the average thresholds of classes in the entire task are demonstrated in Figure 3. Since domain differences are big in the order of DomainNet, iDigits, and CORe50, the average thresholds follow the same order. The higher the threshold, the more the model increases the nodes because the number of classes that do not exceed the threshold increase.

Ablation of Dynamic Threshold. The impact of using dynamic threshold (DT) in IC is demonstrated in Table 6. The comparison was conducted in the setting in which the threshold value is set to a constant of 0.5. The strategy of setting threshold dynamically was effective in leveraging IC, by successfully deciding the optimal threshold value on the basis of domain differences calculated using accuracies per class.

Incremental Classifier Compared to Standard CIL Strategy. The mechanism of Incremental Classifier (IC) is quite different from a standard strategy in CIL research. The existing strategy in CIL only increases output nodes corresponding to new classes naively when new tasks arrive. However, our proposed IC can increase output nodes of both old and new classes to prevent semantic drift of the classifier while learning diverse domains, with our novel strategies that decide when to expand each node and handle the node selection issue for each class. In this way, IC can tackle the challenge successfully while learning various

Table 7: Additional experiments on CNN.

Dataset	Fine-tuning		EWC [10]		LwF [13]		LAE [5]		**ICON (Ours)**	
	Avg. Acc↑	Forgetting↓	Avg. Acc↑	Forgetting↓	Avg. Acc↑	Forgetting↓	Avg. Acc↑	Forgetting↓	Avg. Acc↑	Forgetting↓
iDigits [22]	23.22±0.24	25.13±0.94	29.91±0.64	16.63±0.98	28.64±0.43	17.94±0.51	30.43±0.36	15.62±1.12	**37.34±0.73**	**7.55±0.47**
CORe50 [14]	13.49±0.69	27.61±0.08	14.14±1.34	23.98±0.36	23.29±0.81	12.22±0.61	37.36±0.49	6.34±0.55	**45.67±0.66**	**5.92±0.41**
DomainNet [17]	11.34±0.37	20.91±0.42	13.74±0.73	13.83±0.89	14.83±1.53	13.16±0.36	17.68±0.67	12.46±0.69	**23.73±0.55**	**7.32±0.28**

Table 8: Average accuracy on various number of shifts per task.

# shifts per task	iDigits [22]	CORe50 [14]	DomainNet [17]
1	**75.11**	**83.18**	53.37
2	71.77	82.66	53.23
4	69.24	82.70	**53.89**
6	71.82	82.28	53.66
8	69.06	82.51	53.68
10	68.45	81.70	53.61

domains associated with a single class, with dynamic thresholding and knowledge distillation to mitigate catastrophic forgetting effectively. Consequently, IC successfully tackles forgetting in the classifier caused by various domains while the existing strategy in CIL cannot.

Architecture Generalizability. Existing prompt-based methods are not flexible enough to be combined with architectures other than the Transformer family. On the other hand, our proposed ICON uses a bottleneck adapter that is sufficiently applicable to CNN and others. Therefore, we applied ICON to CNN structure without pre-trained weights other than ViT and demonstrated its performance with Fine-tuning, EWC [10], LwF [13] and LAE [5] similarly applicable in the CNN structure. We used ResNet-152 (60M) [7], which has a similar number of parameters as ViT-B/16 (86M). The bottleneck adapter was implemented with 1×1 convolution layers for up and down projection and inserted in the shallower 23 of 50 convolution blocks parallelly. As shown in Table 7, our proposed ICON also achieved significantly better performance compared to other IL methods based on CNN architecture, emphasizing the broader applicability of ICON beyond being limited to ViT.

Multiple Shifts per Task. We further explored using shifts more than one that are saved in the shift pool for each task, as shown in Table 8. For iDigits, when used more than one shift for a task, the performance drops showing that having too many shifts in the shift pool can cause too strict regularization in CAST, and CORe50 has similar results. For DomainNet, the performance was robust against the number of shifts saved per task.

Representation Spaces of Competing Methods. As we mentioned in the main paper, the model faces intra-class domain confusion and inter-domain con-

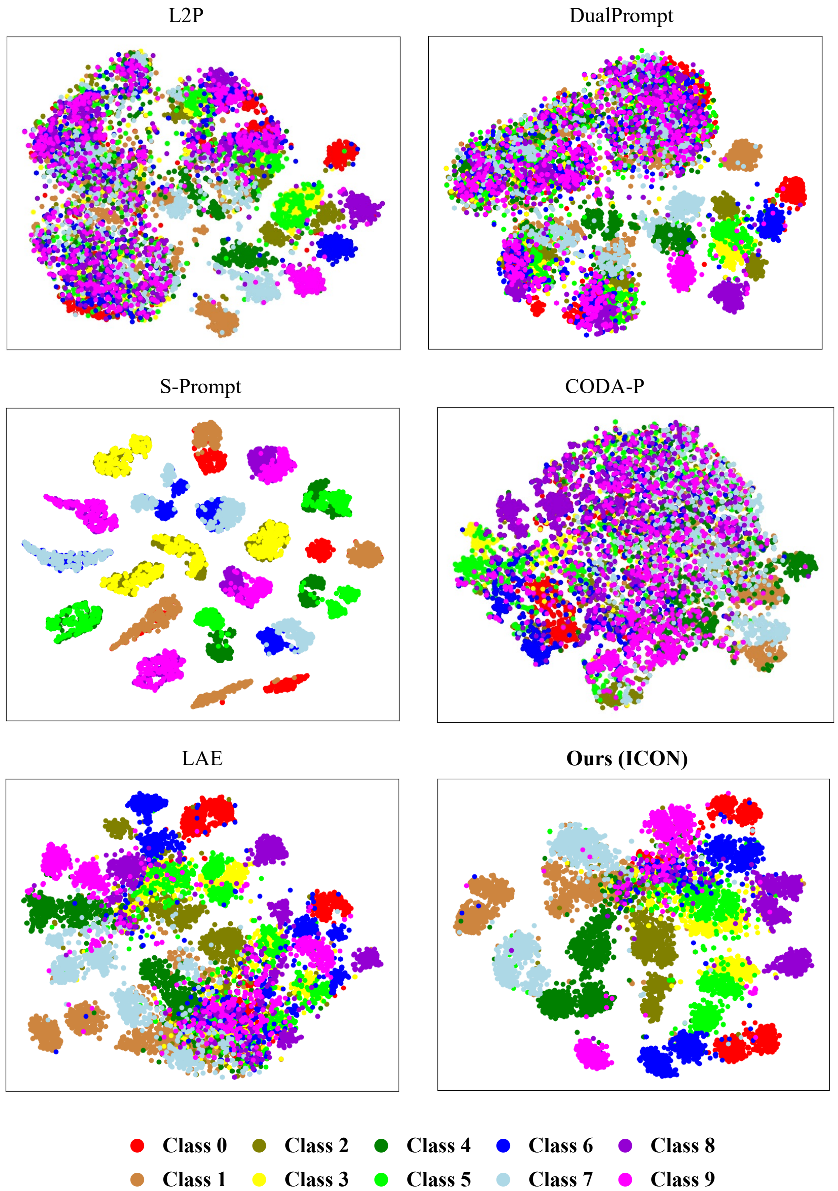


Fig. 4: t-SNE visualization on the resulting feature spaces of L2P [25], DualPrompt [24], S-Prompts [23], CODA-P [19], LAE [5] and proposed ICON on the iDigits dataset in VIL scenario.

fusion in the proposed VIL scenario, and these confusions are very likely to appear in a mixed state in feature representation space. Therefore, we visual-

Table 9: Average accuracy on various locations of adapter.

Location	iDigits [22]	CORe50 [14]	DomainNet [17]
First 5 (**Ours**)	**75.11**	**83.18**	**53.37**
Last 5	52.94	77.79	50.31
All	62.57	83.10	55.14

ized the feature representation spaces of competing methods in the VIL scenario using t-SNE (Figure 4). As shown in Figure 4, in the case of prompt-based methods, such as L2P [25] and DualPrompt [24], only the most recently learned classes are biased and separated. However, unlike other prompt-based methods, CODA-P [19] shows a lot of mixtures, which we expect to be the result of the weighted sum of prompt using attention. S-Prompts [23] strongly rely on task-specific prompt and selection mechanisms, therefore, representations of classes learning in the same task are overlapped. LAE [5] using an adapter, shows relatively more separation compared to the aforementioned methods, but there are still many mixed representations. In contrast, our proposed ICON produces much more separated class-specific representation subspaces. This demonstrates that ICON has well-accumulated knowledge without interfering with previously learned knowledge away from intra-class domain confusions and inter-domain confusions.

4 Additional Ablation Results

Results on Various Locations of Adapter. We conducted experiments of ICON with different locations of adapters (Table 9). For all datasets, inserting adapters to the first 5 layers of the backbone was the best, and the performances significantly dropped when used them in the last 5 layers of the backbone. The result indicates that adapting to new task is optimal in the early layers, and using them in the later layers prevents flexible adjustment of representations to each new task.

Hyperparameter Sensitivity. In this section, we demonstrated the result of our experiments with different hyperparameter values, α, β, and γ which are used in the loss function. The hyperparameter α adjusts the impact of knowledge distillation from the previous classifier which is involved in our proposed Incremental Classifier.

As shown in Table 10, using small α prevents fully leveraging the benefit of knowledge distillation, while the excessive effect of knowledge distillation learning the current task. We set $\alpha = 1.00$ for all datasets as default. We also explored the impact of CAST loss via the coefficient β that controls the power of regularization of the direction of the current task in Table 11. For iDigits and CORe50, the performance was the best when $\beta = 0.05$, while the performance of DomainNet was the best when $\beta = 0.01$, indicating relatively the weak intensity

Table 10: Average accuracy on various α.

α	iDigits [22]	CORe50 [14]	DomainNet [17]
0.25	74.73	81.30	52.45
0.50	74.90	82.51	53.21
1.00	**75.11**	83.18	**53.37**
1.50	74.97	**83.23**	53.06
2.00	74.88	82.89	52.80

Table 11: Average accuracy on various β.

β	iDigits [22]	CORe50 [14]	DomainNet [17]
0.01	74.51	79,85	**53.37**
0.02	74.78	80.55	53.21
0.03	74.22	82.15	53.17
0.04	73.95	83.08	53.19
0.05	**75.11**	83.18	53.27
0.06	74.69	83.21	53.07
0.07	74.95	**83.50**	53.03
0.08	73.89	83.02	52.75
0.09	74.06	83.24	52.90
0.10	74.32	83.11	52.66

Table 12: Average accuracy on various γ.

γ	iDigits [22]	CORe50 [14]	DomainNet [17]
0.5	72.00	80.74	**53.53**
1.0	73.12	81.23	53.41
1.5	74.51	82.70	53.22
2.0	**75.11**	**83.18**	53.37
2.5	75.08	82.52	53.12
3.0	74.99	82.74	52.99

of regularization allows learning more difficult tasks (DomainNet). Lastly, the impact of the scaling factor γ used when deciding the threshold in IC is demonstrated in Table 12. Using too small γ can cause overwriting of the classifier by not increasing nodes in the classifier dynamically even when the domain difference is huge. Meanwhile, using too big γ can degrade the performance since the model can be confused while selecting which logit to use for a single label at inference time, after increasing nodes too easily even when not necessary.

5 Limitations and Future Work

Despite achieving noticeable performance and successfully resolving the problem from the absence of prior knowledge of the following tasks, our work has a few limitations as well. We conducted experiments on widely used three benchmarks, that can be configured VIL scenario. Then we tried to experiment with additional datasets, but other widely used datasets have the following problems when constructing the VIL scenario. ImageNet-R [8] and VLCS [20] have severe imbalance problems, and thus in some classes, it becomes a few-shot (about 1 to 5) learning task when constructing a VIL scenario. Moreover, PACS [12] and OfficeHome [21] have an insufficient number of classes (both have 7 classes) to strictly evaluate incremental scenarios and have imbalance problems too. Thus, the VIL scenario requires more complex and well-organized large benchmarks to evaluate the effectiveness of VIL methods and encourage the advances of this real-world challenge.

Furthermore, in IC, there can be other algorithms to replace max pooling which are more effective based on further analysis even though they work well currently. In the same way, deciding the number of clusters in CAST as the sequential tasks increase can be addressed in the future work. Also, while only a single domain and a single group of classes increase in the current VIL setting, having more than a single domain and a single group of classes can be included in the future work as well.

6 Hyperparameters of Comparison Models.

1. L2P [25]
   - iDigits
     - Prompt pool size: 20
   - CORe50
     - Prompt pool size: 40
   - DomainNet
     - Prompt pool size: 30
   - Common
     - Prompt top-K: 5
     - Prompt length: 5
     - Batch size: 16

2. S-Prompts [23]
   - iDigits
     - Number of clusters: 4
   - CORe50
     - Number of clusters: 8
   - DomainNet
     - Number of clusters: 6
   - Common
     - Prompt length: 10
     - Batch size: 128

3. DualPrompt [24]
   - iDigits
     - E-Prompt pool size: 20
   - CORe50

  - E-Prompt pool size: 40
- DomainNet
  - E-Prompt pool size: 30
- Common
  - G-Prompt layer index: [0, 1]
  - G-Prompt length: 5
  - E-Prompt layer index: [2, 3, 4]
  - E-Prompt length: 5
  - E-Prompt top-K: 1
  - Batch size: 24

4. CODA-P [19]
   - iDigits
     - E-Prompt pool size: 20
   - CORe50
     - E-Prompt pool size: 40
   - DomainNet
     - E-Prompt pool size: 30
   - Common
     - G-Prompt length: 0
     - E-Prompt length: 8
     - Batch size: 128

5. LAE [5]
   - Common
     - Adapter location: [0, 1, 2, 3, 4] of Multi-head Self-Attention layer
     - Adapter downsample: 5
     - EMA decay: 0.9999
     - Batch size: 24

References

1. Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs (2018), `http://github.com/google/jax`
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 248–255. Ieee (2009)
3. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: Proceedings of the International Conference on Learning Representations (2020)
4. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: Proceedings of the International Conference on Machine Learning. pp. 1180–1189. PMLR (2015)

5. Gao, Q., Zhao, C., Sun, Y., Xi, T., Zhang, G., Ghanem, B., Zhang, J.: A unified continual learning framework with general parameter-efficient tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11483–11493 (2023)
6. He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., Neubig, G.: Towards a unified view of parameter-efficient transfer learning. In: Proceedings of the International Conference on Learning Representations (2022)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
8. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8340–8349 (2021)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations (2015)
10. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
12. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5542–5550 (2017)
13. Li, Z., Hoiem, D.: Learning without forgetting. In: Proceedings of the European Conference on Computer Vision. pp. 614–629. Springer (2016)
14. Lomonaco, V., Maltoni, D.: Core50: a new dataset and benchmark for continuous object recognition. In: Conference on Robot Learning. pp. 17–26. PMLR (2017)
15. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning (2011)
16. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Proceedings of the Advances in Neural Information Processing Systems **32** (2019)
17. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1406–1415 (2019)
18. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of the International Conference on Machine Learning. pp. 8748–8763. PMLR (2021)
19. Smith, J.S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., Kira, Z.: Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11909–11919 (2023)
20. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2011)

21. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5018–5027 (2017)
22. Volpi, R., Larlus, D., Rogez, G.: Continual adaptation of visual representations via domain randomization and meta-learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4443–4453 (2021)
23. Wang, Y., Huang, Z., Hong, X.: S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (eds.) Proceedings of the Advances in Neural Information Processing Systems (2022)
24. Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.Y., Ren, X., Su, G., Perot, V., Dy, J., et al.: Dualprompt: Complementary prompting for rehearsal-free continual learning. In: Proceedings of the European Conference on Computer Vision. pp. 631–648. Springer (2022)
25. Wang, Z., Zhang, Z., Lee, C.Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., Pfister, T.: Learning to prompt for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 139–149 (2022)