Knowledge Restore and Transfer for Multi-Label Class-Incremental Learning Supplemental Material

A. Appendix

A.1. Other Related Work

Incremental object detection(IOD) applies incremental learning to object detection specifically. Both KD and ER have been applied to IOD task and implement on different detectors. [18] first uses the KD to the output of Faster R-CNN and subsequent methods [6, 7] add KD terms on the intermediate feature maps and region proposal networks or store a set of exemplars to fine-tune the model. In addition to being applied to CNN detectors, ER and KD have also been applied to the transformer network DETR [11].

Incremental Semantic Segmentation(ISS) methods can be classified into regularization-based and replay-based approaches. The former approaches such as SDR [13] and PLOP [4] propose different KD strategies to regularize a current model in a latent feature space. The second approaches [3, 12] rely on an ER strategy, involving retention of a small set of exemplars or pseudo information for previous categories.

It is evident that the IOD and ISS methods are not directly applicable to MLCIL tasks due to their reliance on specific detection and segmentation frameworks. Therefore, conducting research on MLCIL with only image-level annotations is of great value and significance.

A.2. More Experimental Details

All models are implemented with PyTorch and trained on 2 RTX 3090 GPUS. We resize images to $h \times w = 224 \times 224$ as the input resolution and the size of output feature is $7 \times 7 \times 2048$. The extracted features are fed into the ICA module after linear projection and adding position encodings. We set the dimension $d = 384$ for COCO and $d = 768$ for VOC datasets. For the ICA module, the embedding dimension l is set to 384 for COCO and 768 for VOC datasets, and the number of heads h is set to 8. For the DPL method, the threshold η is initialized as 0.8, and the target value μ is set to 2.9 for COCO and 1.4 for VOC datasets. The sensitive study of hyper-parameter λ in ablation study has summarized that the performance of our methods changes are minimal under different λ and we report the best hyper-parameter values under different protocols. Concretely, the λ is set to 100 for B0 benchmark and 300 for B40(B10) benchmark.

All compared methods, including baselines, CIL methods [9, 17, 16, 14, 2, 1, 5, 19], and MLOIL methods [10, 8], utilize TResNetM pre-trained on ImageNet-21k as the backbone (L2P [20] employs ViT-B/16 pre-trained on ImageNet-21k as the backbone). To adapt SCIL methods for MLCIL tasks, we employ $\mathbf{L}_{ASL}$ as the classification loss instead of cross-entropy loss and rely on the original codebase to implement the method and carefully select hyper-parameters to ensure optimal performance. For the MLOIL methods [10, 8], we directly implement them in MLCIL protocol using their original codebase.

Algorithm 1 Dynamic Pseudo-Label

Require: Session t training set $\mathbf{X}^t$, Initial threshold η;
Require: Session t target value μ^t, Old model Θ^{t-1}.
Ensure: Updated training set $\hat{\mathbf{X}}^t$.

```
Employ model Θ^{t-1} to infer the training set X^t based on initial threshold η to obtain pseudo-label set S^t
Count the number of images M^t in training set X^t
Calculate the the average pseudo labels per image β^t = |S^t| / M^t
while ( |β^t − μ^t| > 1e^{-1})· do
    if β^t > μ^t then
        η^t = η^t + 1e − 2
    else {β^t ≤ μ^t}
        η^t = η^t − 1e − 2
    end if
    Employ model Θ^{t-1} to infer the training set X^t based on η^t to obtain pseudo-label set S^t
    Calculate the the average pseudo labels per image β^t = |S^t| / M^t
end while
Merge the pseudo-label set S^t with the label set Y^t as new ground truth Ŷ^t and obtain the updated training set X̂^t
```

A.3. Discussion of ICA Parameters

The initial parameter count of the ICA module is 2.4M including the Linear Projection and MHSA and MLP components. In the MLCIL task, we add a KT token at the first session and a KR token at each session. Assuming there are a total number of 8 sessions, the amount of extra parame-

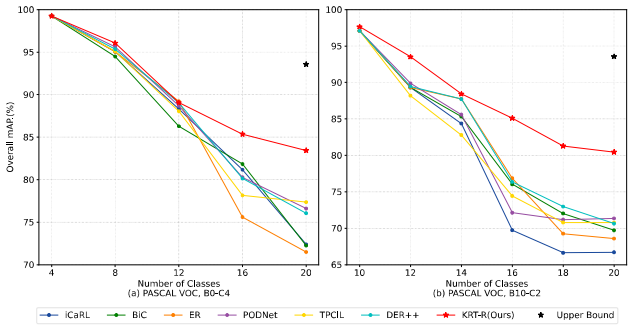

Figure 1: Comparison results (mAP%) on PASCAL VOC.

ters increased by only 9*384=3,456 (0.003M). Compared with the overall parameters (about 30M), extra adding parameters almost could be ignored. Therefore, there is no issue of excessive additional parameters caused by the increment of sessions, instead, our method is suited for long-term incremental learning scenarios.

A.4. The Algorithms of DPL Module

As mentioned in our main paper, the algorithm of DPL is presented in Algorithm 1.

A.5. More Comparison Results and Detailed

In order to demonstrate that KRT can achieve MLCIL tasks without any pre-trained, effectively learning new classes, we train KRT from scratch on a completely No pre-trained model and compare with FT approach. The results in Table 1 show that KRT exhibits significant improvements even **without any pre-trained** in the model.

Pre-trained Model	FT (Baseline)		KRT (Ours)		Upper-bound
	Last Acc	**Avg Acc**	**Last Acc**	**Avg Acc**	
No	9.60	21.37	**48.38**	**52.09**	62.29

Table 1: Results (mAP%) on MS-COCO under the B40-C10.

Moreover, Figure 1 presents a comparison of VOC curves on B0C4 and B10C2 benchmarks. Table 2 and Table 3 provide detailed per-session performance of different methods on COCO B0C20 and B0C10 benchmarks, respectively. Similarly, Table 4 and Table 5 illustrate the per-session performance of various methods on COCO B40C10 and B40C5 benchmarks. Additionally, Table 6 and Table 7 display detailed per-session performance of different methods on the VOC B0C4 and B10C2 benchmarks.

A.6. More Visualization Results

We provide more visualization results of cross-attention maps examples of the KR and KT tokens in Figure 2

References

[1] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and et al. Dark experience for general continual learning: a strong, simple baseline. *NIPS*, 33:15920–15930, 2020.

[2] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, pages 233–248, 2018.

[3] Sungmin Cha, YoungJoon Yoo, and et al. Ssul: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *NIPS*, 34:10919–10930, 2021.

[4] Arthur Douillard, Yifu Chen, and et al. Plop: Learning without forgetting for continual semantic segmentation. In *CVPR*, Jun 2021.

[5] Arthur Douillard, Matthieu Cord, and et al. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102. Springer, 2020.

[6] Tao Feng, Mang Wang, and et al. Overcoming catastrophic forgetting in incremental object detection via elastic response distillation. In *CVPR*, pages 9427–9436, 2022.

[7] K J Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *CVPR*, Jun 2021.

[8] Chris Dongjoo Kim and et al. Imbalanced continual learning with partitioning reservoir sampling. In *ECCV*, 2020.

[9] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[10] Yan-Shuo Liang and Wu-Jun Li. Optimizing class distribution in memory for multi-label online continual learning. *arXiv preprint arXiv:2209.11469*, 2022.

[11] Yaoyao Liu, Bernt Schiele, Andrea Vedaldi, and Christian Rupprecht. Continual detection transformer for incremental object detection. In *CVPR*, pages 23799–23808, 2023.

[12] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. *ICCV*, Jan 2021.

[13] Umberto Michieli and Pietro Zanuttigh. Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In *CVPR*, Jun 2021.

[14] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017.

[15] Tal Ridnik, Emanuel Ben-Baruch, and et al. Asymmetric loss for multi-label classification. In *ICCV*, pages 82–91, 2021.

[16] Matthew Riemer, Ignacio Cases, and et al. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

[17] Jonathan Schwarz, Wojciech Czarnecki, and et al. Progress & compress: A scalable framework for continual learning. In *ICML*, pages 4528–4537. PMLR, 2018.

[18] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *ICCV*, pages 3400–3409, 2017.

[19] Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *ECCV*, pages 254–270. Springer, 2020.

[20] Zifeng Wang, Zizhao Zhang, and et al. Learning to prompt for continual learning. In *CVPR*, pages 139–149, 2022.

[21] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019.

Method	Buffer Size	sessions				Average	Last mAP%
		1	2	3	4	mAP%	impro.
FT [15]	0	86.50	54.6	26.53	27.88	51.38	47.37
iCaRL [14]	20/class	86.50	70.72	66.56	52.41	69.05	22.84
BIC [21]		86.50	75.16	68.08	51.51	70.31	23.74
ER [16]		86.50	69.48	60.74	58.04	68.69	17.21
TPCIL [19]		86.50	75.13	67.81	63.41	73.21	11.84
PODNet [5]		86.50	75.71	70.52	66.96	74.92	8.29
DER++ [1]		86.50	75.84	73.28	68.17	75.95	7.08
KRT-R(Ours)		**86.63**	**78.84**	**77.29**	**75.25**	**79.47**	-

Table 2: Comparison results on MS-COCO dataset with B0-C20 benchmark

Method	Buffer Size	sessions								Average	Last mAP%
		1	2	3	4	5	6	7	8	mAP%	impro.
FT [15]	0	92.51	59.68	41.23	31.86	22.03	23.07	19.27	16.93	38.33	53.24
iCaRL [14]	20/class	92.51	73.62	67.38	60.20	52.19	43.63	44.25	43.84	59.72	26.33
BIC [21]		92.51	79.53	73.97	64.16	57.17	49.79	50.63	50.95	64.96	19.22
ER [16]		92.51	75.14	61.34	56.83	48.55	51.44	49.28	47.19	60.28	22.98
TPCIL [19]		92.51	77.86	69.12	67.34	62.85	63.25	62.12	60.57	69.45	9.60
PODNet [5]		92.51	79.95	73.45	68.22	63.17	62.98	60.36	58.82	69.93	11.35
DER++ [1]		92.51	79.23	76.27	70.68	68.88	67.12	64.16	63.11	72.74	7.06
KRT-R(Ours)		**92.25**	**81.30**	**77.26**	**74.69**	**73.22**	**72.80**	**70.61**	**70.17**	**76.54**	-

Table 3: Comparison results on MS-COCO dataset with B0-C10 benchmark.

Method	Buffer Size	sessions					Average	Last mAP%
		1	2	3	4	5	mAP%	impro.
FT [15]	0	82.88	28.55	28.06	18.78	16.99	35.05	58.19
iCaRL [14]	20/class	82.41	67.87	63.40	58.27	55.74	65.61	19.44
BIC [21]		82.41	71.51	61.76	55.52	55.91	65.55	19.27
ER [16]		82.41	70.79	66.58	63.34	61.59	68.94	13.59
TPCIL [19]		82.41	72.62	71.65	68.62	66.54	72.37	8.64
PODNet [5]		82.41	71.40	70.76	65.90	64.22	70.96	10.96
DER++ [1]		82.41	77.07	73.92	68.11	66.31	73.56	8.87
KRT-R(Ours)		**82.37**	**79.54**	**78.27**	**75.95**	**75.18**	**78.26**	-

Table 4: Comparison results on MS-COCO dataset with B40-C10 benchmark.

Method	Buffer Size	sessions									Average	Last mAP%
		1	2	3	4	5	6	7	8	9	mAP%	impro.
FT [15]	0	82.41	45.74	16.43	17.59	13.6	12.51	10.99	10.44	10.66	24.49	61.77
iCaRL [14]	20/class	82.41	74.1	62.8	60.4	61.6	56.4	54.9	54.8	53.9	62.37	18.53
BIC [21]		82.41	75.50	61.35	56.13	57.37	52.94	52.17	51.97	51.72	60.17	20.71
ER [16]		82.41	74.24	66.58	62.92	63.88	61.03	60.30	59.96	58.12	65.49	14.31
TPCIL [19]		82.41	74.91	70.21	69.04	68.35	66.25	66.01	64.79	64.24	69.58	8.19
PODNet [5]		82.41	76.18	68.21	66.31	66.45	61.45	61.5	59.98	58.89	66.82	13.54
DER++ [1]		82.41	78.2	74.15	70.09	67.03	61.9	61.91	62.97	62.14	68.98	10.29
KRT-R(Ours)		**82.37**	**80.63**	**78.15**	**76.46**	**76.43**	**74.34**	**73.62**	**72.82**	**72.43**	**76.36**	-

Table 5: Comparison results on MS-COCO dataset with B40-C5 benchmark.

Method	Buffer Size	sessions					Average	Last mAP%
		1	2	3	4	5	mAP%	impro.
FT [15]	0	99.25	94.16	85.51	68.67	62.88	82.09	20.55
iCaRL [14]	2/class	99.25	95.03	88.31	81.16	72.38	87.23	11.05
BIC [21]		99.25	94.48	86.29	81.83	72.24	86.82	11.19
ER [16]		99.25	95.05	89.22	75.6	71.49	86.12	11.94
TPCIL [19]		99.25	95.1	88.04	78.15	77.35	87.58	6.08
PODNet [5]		99.25	95.64	88.71	80.28	76.60	88.09	6.83
DER++ [1]		99.25	95.35	89.02	80.13	76.05	87.96	7.38
KRT-R(Ours)		**99.77**	**96.06**	**89.06**	**85.43**	**83.43**	**90.73**	-

Table 6: Comparison results on PASCAL VOC dataset with B0-C4 benchmark.

Method	Buffer Size	sessions						Average	Last mAP%
		1	2	3	4	5	6	mAP%	impro.
FT [15]	0	97.09	86.97	82.49	61.65	49.54	43.01	70.12	37.45
iCaRL [14]	2/class	97.09	89.32	84.37	69.74	66.63	66.70	78.98	13.76
BIC [21]		97.09	89.94	85.33	76.03	72.04	69.71	81.69	10.75
ER [16]		97.09	89.33	87.74	76.86	69.26	68.58	81.48	11.88
TPCIL [19]		97.09	88.19	82.79	74.44	70.69	70.77	80.66	9.69
PODNet [5]		97.09	89.89	85.59	72.13	71.18	71.35	81.21	9.11
DER++ [1]		97.09	89.44	87.72	76.36	72.97	70.64	82.37	9.82
KRT-R(Ours)		**97.64**	**93.52**	**88.42**	**85.10**	**81.28**	**80.46**	**87.73**	-

Table 7: Comparison results on PASCAL VOC dataset with B10-C2 benchmark.

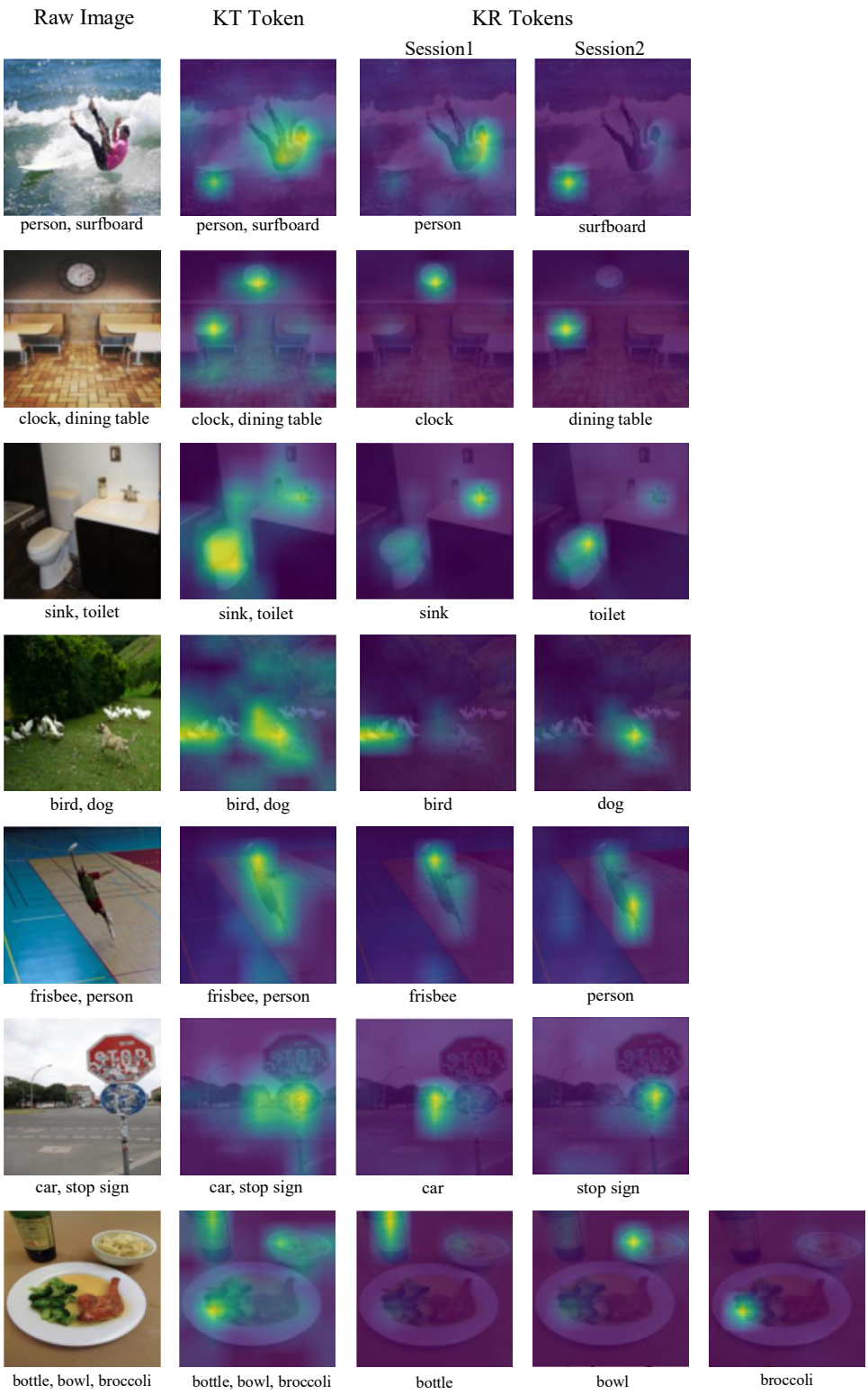

Figure 2: Visualization of ICA module.