Continual Learning Based on OOD Detection and Task Masking - Appendix

1. Average Incremental Accuracy

In the paper, we reported the accuracy after all tasks have been learned. Here we give the *average incremental accuracy*. Let A_k be the average accuracy over all tasks seen so far right after the task k is learned. The average incremental accuracy is defined as $\mathcal{A} = \sum_{k=1}^{t} A_k/t$, where t is the last task. It measures the performance of a method throughout the learning process. Tab. 1 shows the average incremental accuracy for the TIL and CIL settings. Fig. 1 and Fig. 2 plot the TIL and CIL accuracy A_k at each task k for every dataset, respectively. We can clearly see that our proposed method CLOM and CLOM(-c) outperform all others except for MNIST-5T, for which a few systems have the same results.

2. Network Parameter Sizes

We use AlexNet-like architecture [3] for MNIST and ResNet-18 [2] for CIFAR10. For CIFAR100 and Tiny-ImageNet, we use the same ResNet-18 structure used for CIFAR10, but we double the number of channels of each convolution in order to learn more tasks.

We use the same backbone architecture for CLOM and baselines, except for OWM and HyperNet, where we use the same architecture as in their original papers. OWM uses an Alexnet-like structure for all datasets. OWN has difficulty to work with ResNet-18 because it is not obvious how to deal with batch normalization in OWM. HyperNet uses a fully-connected network for MNIST and ResNet-32 for other datasets. We found it very hard to change HyperNet because the network initialization requires some arguments which were not explained in the paper. In Tab. 2, we report the network parameter sizes after the final task in each experiment has been trained.

Due to hard attention embeddings and task specific heads, CLOM requires task specific parameters for each task. For MNIST, CIFAR10, CIFAR100-10T, CIFAR100-20T, Tiny-ImageNet-5T, and Tiny-ImageNet-10T, we add task specific parameters of size 7.7K, 17.6K, 68.0K, 47.5K, 191.0K, and 109.0K, respectively, after each task. The contrastive learning also introduces task specific parameters from the projection function g. However, this can be discarded during deployment as it is not necessary for inference or testing.

3. Details about Augmentations

We follow [1, 6] for the choice of data augmentations. We first apply *horizontal flip*, *color change* (*color jitter* and *grayscale*), and *Inception crop* [5], and then four *rotations* (0°, 90°, 180°, and 270°). The details about each augmentation are the following.

Horizontal flip: we flip an image horizontally with 50% of probability; **color jitter**: we add a noise to an image to change the brightness, contrast, and saturation of the image with 80% of probability; **grayscale**: we change an image to grayscale with 20% of probability; **Inception crop**: we uniformly choose a resize factor from 0.08 to 1.0 for each image, and crop an area of the image and resize it to the original image size; **rotation**: we rotate an image by 0°, 90°, 180°, and 270°. In Fig. 3, we give an example of each augmentation by using an image from Tiny-ImageNet [4].

4. Hyper-parameters

Here we report the hyper-parameters that we could not include in the main paper due to space limitations. We use the values chosen by [1, 6] to save time for hyper-parameter search. We first train the feature extractor h and projection function g for 700 epochs, fine-tune the classifier f for 100 epochs. The s for the pseudo step function in Eq. 7 of the main paper is set to 700. The temperature τ in contrastive loss is 0.07 and the resize factor for Inception crop ranges from 0.08 to 1.0.

For other hyper-parameters in CLOM, we use 10% of training data as the validation data and select the set of hyper-parameters that gives the highest CIL accuracy on the validation set. We train the output calibration parameters $(\boldsymbol{\sigma}, \boldsymbol{\mu})$ for 160 iterations with learning rate 0.01 and batch size 32. The following are experiment specific hyper-parameters found with hyper-parameter search.

- For MNIST-5T, batch size = 256, the hard attention regularization hyper-parameters are $\lambda_1 = 0.25$, and $\lambda_2 = \cdots = \lambda_5 = 0.1$.
- For CIFAR10-5T, batch size = 128, the hard attention regularization hyper-parameters are $\lambda_1 = 1.0$, and $\lambda_2 = \cdots = \lambda_5 = 0.75$.
- For CIFAR100-10T, batch size = 128, the hard attention regularization hyper-parameters are $\lambda_1 = 1.5$, and $\lambda_2 = \cdots = \lambda_{10} = 1.0$.
- For CIFAR100-20T, batch size = 128, the hard attention regularization hyper-parameters are $\lambda_1 = 3.5$, and $\lambda_2 = \cdots = \lambda_{20} = 2.5$.
- For Tiny-ImageNet-5T, batch size = 128, the hard attention regularization hyper-parameters are $\lambda_1 = \cdots = \lambda_5 = 0.75$.
- For Tiny-ImageNet-10T, batch size = 128, the hard attention regularization hyper-parameters are $\lambda_1 = 1.0$, and $\lambda_2 = \cdots = \lambda_{10} = 0.75$.

We do not search hyper-parameter λ_t for each task $t \geq 2$. However, we found that larger λ_1 than λ_t, $t > 1$, results in

Method	MNIST-5T		CIFAR10-5T		CIFAR100-10T		CIFAR100-20T		T-ImageNet-5T		T-ImageNet-10T	
	TIL	CIL	TIL	CIL	TIL	CIL	TIL	CIL	TIL	CIL	TIL	CIL
CIL Systems												
OWM	**99.9**	**98.5**	87.5	67.9	62.9	41.9	66.8	37.3	26.4	18.7	31.3	17.6
MUC	**99.9**	87.2	95.2	67.7	80.3	50.5	77.8	32.7	61.1	48.1	56.5	34.8
PASS	**99.9**	92.0	88.0	63.6	77.3	52.9	78.4	38.0	55.1	39.9	52.2	30.1
LwF.R	**99.9**	92.8	96.6	70.7	87.6	65.4	90.9	62.8	61.7	48.0	61.3	40.9
iCaRL	**99.9**	98.0	96.4	74.7	86.9	68.4	88.9	64.5	60.9	50.7	60.0	44.1
Mnemonics	**99.9**	98.3	96.4	75.2	86.4	67.7	88.9	64.5	61.0	50.7	60.4	44.5
BiC	**99.9**	95.3	93.9	74.9	88.9	68.7	91.5	61.9	52.7	36.7	57.4	35.5
DER++	**99.9**	98.3	94.4	79.3	86.0	67.6	85.7	56.6	62.6	49.7	66.2	46.8
TIL Systems												
HAT	**99.9**	90.8	96.7	73.0	84.3	55.6	85.5	41.8	61.4	48.0	63.1	40.2
HyperNet	99.8	71.5	95.0	63.5	77.0	44.4	82.1	33.8	23.5	13.8	28.8	12.2
SupSup	99.7	81.8	97.0	73.6	90.5	58.6	91.6	51.4	63.4	51.0	67.1	45.9
CLOM(-c)	**99.9**	97.0	**98.7**	**91.9**	**92.3**	75.4	**94.3**	70.1	**68.5**	57.0	**72.0**	56.1
CLOM	**99.9**	98.3	**98.7**	**91.9**	**92.3**	**75.9**	**94.3**	**71.0**	**68.5**	**58.6**	**72.0**	**56.5**

Table 1. Average incremental accuracy. Numbers in bold are the best results in each column.

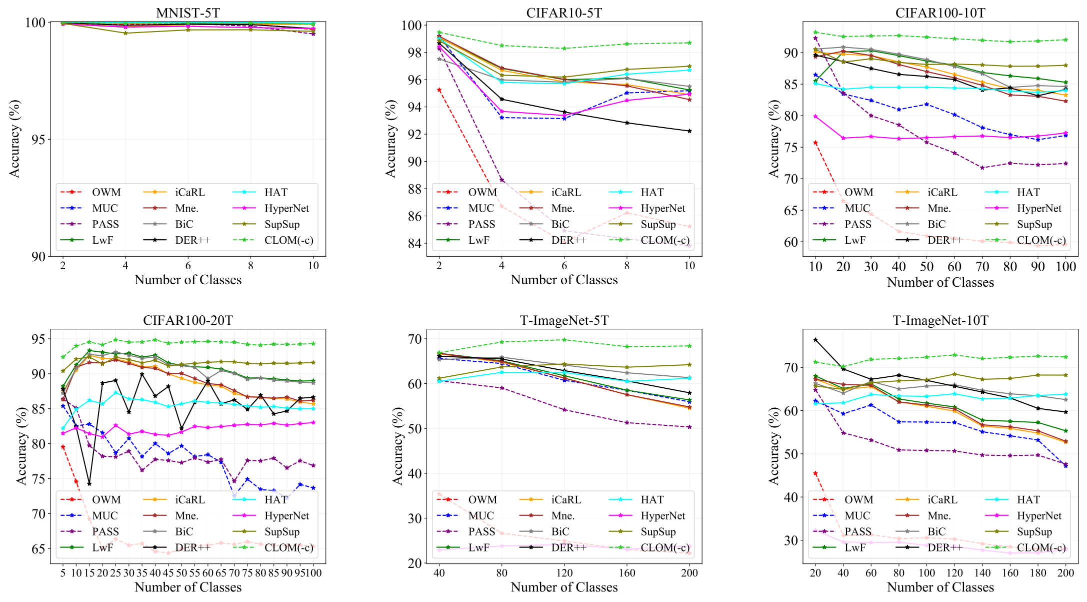

Figure 1. TIL performance over number of classes. The dashed lines indicate the methods that do not save any samples from previous tasks. The calibrated version, CLOM, is omitted as its TIL accuracy is the same as CLOM(-c). Best viewed in color.

better accuracy. This is because the hard attention regularizer $\mathcal{L}_r$ gives lower penalty in the earlier tasks than later tasks by definition. We encourage greater sparsity in task 1 by larger λ_1 for similar penalty values across tasks.

For the baselines, we use the best hyper-parameters reported in their original papers or in their code. If some hyper-parameters are unknown, *e.g.*, the baseline did not use a particular dataset, we search for the hyper-parameters as we do for CLOM.

We obtain the results by running the following codes

- OWM: https://github.com/beijixiong3510/OWM
- MUC: https://github.com/liuyudut/MUC
- PASS: https://github.com/Impression2805/CVPR21_PASS
- LwF.R: https://github.com/yaoyao-liu/class-incremental-learning

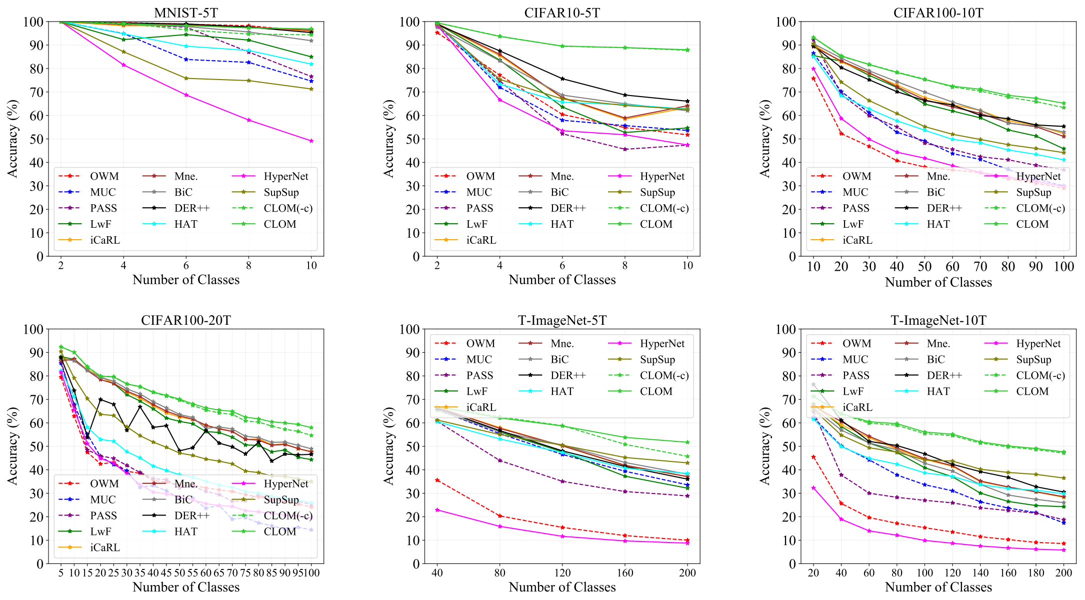

Figure 2. CIL performance over number of classes. The dashed lines indicate the methods that do not save any samples from previous tasks. Best viewed in color.

Method	MNIST-5T	CIFAR10-5T	CIFAR100-10T	CIFAR100-20T	T-ImageNet-5T	T-ImageNet-10T
OWM	5.27	5.27	5.36	5.36	5.46	5.46
MUC-LwF	1.06	11.19	45.06	45.06	45.47	45.47
PASS	1.03	11.17	44.76	44.76	44.86	44.86
LwF.R	1.03	11.17	44.76	44.76	44.86	44.86
iCaRL	1.03	11.17	44.76	44.76	44.86	44.86
Mnemonics	1.03	11.17	44.76	44.76	44.86	44.86
BiC	1.03	11.17	44.76	44.76	44.86	44.86
DER++	1.03	11.17	44.76	44.76	44.86	44.86
HAT	1.04	11.23	45.01	45.28	44.97	45.11
HyperNet	0.48	0.47	0.47	0.47	0.48	0.48
SupSup	0.58	11.16	44.64	44.64	44.67	44.65
CLOM	1.07	11.25	45.31	45.58	45.59	45.72

Table 2. Number of network parameters (million) after the final task has been learned.

- iCaRL: https://github.com/yaoyao-liu/class-incremental-learning
- Mnemonics: https://github.com/yaoyao-liu/class-incremental-learning
- BiC: https://github.com/sairin1202/BIC
- DER++: https://github.com/aimagelab/mammoth
- HAT: https://github.com/joansj/hat
- HyperNet: https://github.com/chrhenning/hypercl
- SupSup: https://github.com/RAIVNLab/supsup

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[4] Y. Le and X. Yang. Tiny imagenet visual recognition challenge, 2015. 1

[5] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Van-

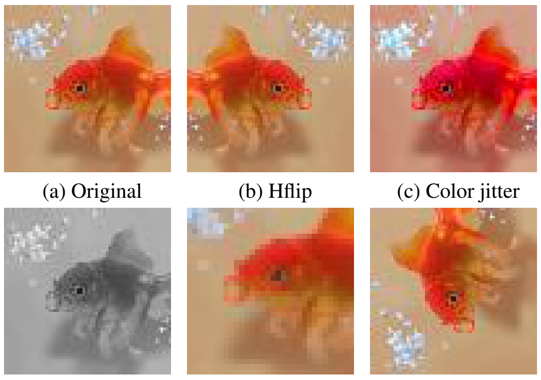

Figure 3. An original image and its view after each augmentation. Hflip and Crop refer to horizontal flip and Inception crop, respectively.

houcke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1

[6] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020. 1