

Appendix: Pseudo-Labeling for Class Incremental Learning

Alexis Lechat¹²

alexis.lechat@onera.fr

Stéphane Herbin¹

stephane.herbin@onera.fr

Frédéric Jurie²

frederic.jurie@unicaen.fr

¹ DTIS, ONERA

Université Paris-Saclay

FR-91123 Palaiseau, France

² Normandie Univ

UNICAEN, ENSICAEN, CNRS, GREYC

14000 Caen, France

A Implementation details

A.1 Training details

In this section, we provide all details needed for training the PLCiL. All parameters were selected through cross-validation.

On CIFAR-100, each session has 150 epochs. The labeled mini-batch size B is 32 with $\mu = 7$. The confidence threshold τ is set to 0.8 and λ to 1. On ImageNet-100, training takes 70 epochs per session and uses the following set of parameters: $\{B = 32, \mu = 7, \tau = 0.7, \lambda = 3\}$.

All experiments are trained with SGD. The learning rate is initialized to 0.03 with Nesterov momentum set to 0.9. We use a weight decay of 10^{-4} , a learning rate decay with cosine annealing and warm restart [9] with an initial period $T_0 = 10$ multiplied by a factor $T_{mult} = 2$ after each restart.

The sensitivity of the main hyperparameters μ , τ and λ are detailed in the following Appendix C. In particular, μ controls the ratio unlabeled/labeled data of each batch. Our experiments have shown that a larger μ is generally better at the expense of increased computational costs. We kept μ quite low so our PLCiL could run on limited hardware with computation time similar to the other methods. τ and λ control the pseudo-labeling process and were tuned for each dataset.

A.2 Continual Training of large DNNs

Our experimentation on CIFAR-100 uses a WRN-28-8 backbone architecture (23M trainable parameters), as [6]. Incidentally, self-supervision – or semi-supervision – is used to train large DNNs when the application is data-scarce. Those models, and even larger ones with hundreds of millions of weights, are now prevailing on classification benchmarks.

Despite that, most CL methods are only evaluated on smaller DNNs: *e.g.* ResNet-32 for CIFAR with only 460K parameters, far from state-of-the-art accuracy in batch training. This

Method	ResNet-32		WRN28-8	
	Last (%)	Avg (%)	Last (%)	Avg (%)
GDumb [10]	20.7	35.0	27.8	42.0
iCaRL [10]	47.1	57.9	53.9	63.9
BiC [10]	50.8	62.7	55.9	67.1
WA [10]	52.1	65.6	50.8	64.4
DMC+ [10]	43.9	58.4	50.4	62.8
Ours	46.9	62.1	61.5	74.0

Table 5: Comparison between all tested approaches on CIFAR-100-full with two backbones: ResNet-32 (460K parameters) and WRN28-8 (24M parameters).

questions the quality of the representation that such small models can learn and therefore, their plasticity potential. For reference, the batch accuracy is 80.6% for WRN28-8 and 72.3% for ResNet-32.

We have conducted more experiments on CIFAR-100-full, similar to those of Section 4.1 but with a smaller ResNet-32, to observe the influence of the size of the network. We present performance comparison between these two backbones in Table 5.

With the ResNet-32, our PLCiL falls behind the compared state-of-the-art methods that were designed with this particular backbone. However, our purpose is to see how the methods scale to larger models. Apart from WA, all the methods become more accurate with larger models. The gap stands out with our method with an overall +12.4% obtained when using a larger architecture. Using WRN28-8 instead of ResNet-32 seems less profitable to other methods, and even damages the accuracy of WA. This can be justified by the limited amount of data they can use at each session, making it harder to learn a large number of weights. Our PLCiL addresses this issue and makes larger architectures trainable thanks to the extensive visual diversity submitted to the model through the self-supervised signal.

B Complete ablation results

B.1 Contribution of each loss component

In Table 6, we present the full version of Table 2 from our ablation study (Sec. 4.3). With the accuracy at each session, we can clearly distinguish the difference between l_{kd} and l_{self} . The former focuses on stability with a consistent accuracy across all sessions while the later enhance the learning of new classes during the early steps but is still very prone to forgetting.

These results corroborate the importance of the weight η in the complete loss as discussed in section 3.4. When the proportion of new classes is still high compared to the number of classes already learned (η low), the performance is more dependent on the ability to learn new things, KD should then have a low impact on the training. However, during the later sessions, when the number of classes to retain is very high compared to the novelty (η close to 1), KD becomes crucial against the catastrophic forgetting.

loss	10	20	30	40	50	60	70	80	90	100	Avg acc
l_{sup}	91.7	82.2	74.2	66.0	62.9	58.3	53.4	50.9	46.4	44.1	59.8
$l_{sup} + \lambda \eta l_{kd}$	91.7	47.6	59.1	68.9	71.5	68.7	67.6	66.1	63.5	61.9	63.9
$l_{sup} + \lambda l_{self}$	91.7	85.6	78.7	72.1	67.7	63.6	59.5	56.2	52.5	50.3	65.15
$l_{sup} + \lambda (l_{self} + \eta l_{kd})$	91.7	86.9	84.9	81.1	76.5	73.9	70.8	66.7	63.8	61.5	74.0
$l_{sup} + \lambda (l_{self} + \eta l_{standardkd})$	91.7	84.2	77.8	72.2	66.6	64.0	62.8	57.6	52.9	52.0	65.6

Table 6: CIL on CIFAR-100-full with only specific components of the loss enabled. Accuracy (%) are computed at the end of each session on all the classes learned so far. Average accuracy does not take into account the first session.

μ	0	1	2	3	7	15	31
Avg acc	59.8	69.2	71.2	72.1	74.0	74.4	75.0
Last acc	44.1	55.5	58.9	59.5	61.5	62.2	63.6

Table 7: Comparison of Last Accuracy and Average Accuracy for different values of μ on CIFAR-100-full.

C Hyperparameters sensitivity

C.1 Ratio labeled-unlabeled data: μ

The hyperparameter μ defines the amount of unlabeled data sampled in each mini-batch, i.e. for each labeled mini-batch size B , our algorithm considers μB unlabeled data. Thus, increasing μ directly increase the visual diversity seen by the model in a self-supervised fashion. However, this also means larger mini-batches for training which can be timely and computationally expensive. In table 7, with B set to 32, we see a consistent increase in performance with larger unlabeled mini-batches. We chose to keep $\mu = 7$ for all our experimentation since it gives satisfactory results and keep the training time and hardware requirement comparable to others CI methods (e.g. $B + \mu B$ gives a total mini-batch size of 256 with $\mu = 7$ and $B = 32$). Higher values of μ only yield minor improvements despite being way more costly.

C.2 Selectivity of the threshold τ and weight of the pseudo-labeling λ

In table 8, we display the results for several combination of the hyperparameters τ and λ . The trends indicate that λ should be kept close to 1, meaning that giving too much weight to the self-supervised part of the loss has a negative impact on the learning of classes.

The PLCiL is less sensitive to the threshold value. For $\lambda \leq 2$, our model reach at least 72.6% average accuracy for all τ tested here. This is due to the fact that our model answer confidently for the majority of the unlabeled data seen, outputting high values that goes beyond most threshold values. This is probably due to the curated nature of our unlabeled data pool (ImageNet) which contains visual information easily transferable to CIFAR (close domains). We did not happen to experiment on it, but we believe that τ could be crucial to filter noisy information when dealing with non-curated unlabeled data or if the domains between the labeled and the unlabeled data were further apart.

τ	λ	Avg acc (%)	Last acc (%)
0.5	1	73.6	61.9
0.5	2	73.5	61.0
0.5	5	71.0	60.9
0.5	7	69.4	58.5
0.5	10	62.2	56.9
0.7	1	73.6	61.7
0.7	2	73.6	61.5
0.7	5	69.6	62.3
0.7	7	69.1	60.8
0.7	10	67.3	57.5
0.8	1	74.7	62.3
0.8	2	74.4	62.4
0.8	5	73.4	62.3
0.8	7	72.7	59.3
0.8	10	67.3	59.3
0.9	1	72.6	59.3
0.9	2	73.6	61.6
0.9	5	73.4	58.5
0.9	7	72.3	59.4
0.9	10	68.5	50.9

Table 8: Evaluation of the PLCiL for different combinations of τ and λ . For this experiment, results are reported on only one permutation of classes instead of the usual 3 runs.

D Data Augmentation

Our approach makes use of data augmentation strategies to leverage the information provided by the unlabeled data. In this section, we study the impact of the two types of augmentation used: weak and strong, and demonstrate that their combination is useful to improve the performance, especially for large scale datasets and scarce annotations.

D.1 List of Augmentations

In this paper, we used the combination of two sets of augmentation: weak α and strong \mathcal{A} .

Weak augmentations consist in random vertical and horizontal translations followed by an horizontal flip occurring with a probability of 0.5. A resizing is applied when needed in order to fit the input requirement of the model.

Strong augmentations are applied according to the CTAugment [10] algorithm. It samples two transformations from the following list of 18: autocontrast, brightness adjustment, color adjustment, contrast adjustment, cutout, histogram equalization, pixel inversion, identity, posterizing, rescaling, rotation, sharpness adjustment, horizontal shear, vertical shear, smoothing, solarizing, horizontal translation, vertical translation. Each selected transformation is applied with a magnitude sampled from a learned range.

D.2 Augmentation Strategies

Pseudo-Labeling allows to regularize the output consistency of the model when using weak and strong augmentation of the same image. This choice of using both weak and strong augmentation is similar to [6] and motivated by the ablation study shown in [4, 5].

In this study, we try different combinations of augmentation on the incremental CIFAR-100 benchmark. We compare the following settings: the standard weak + strong CTAugment, only weak augmentation, only strong augmentation using CTAugment and strong augmentation using RandAugment for Pseudo-Labeling and CTAugment for prediction. Results are presented in Table 9.

When using only weak augmentation, at each session, the model quickly reaches an accuracy close to 100% on the training data but shows very bad results on the validation set. This behavior suggests an overfitting situation. Strong augmentation with either CTAugment or RandAugment also yields lower performance compared to the original setting. In [6], the authors mentioned the fact that their model trained with only strong augmentations did not converge. Our small ablation study corroborates the findings of [4, 5]: mixing two families of augmentation is essential for consistency regularization and generating labels from weakly augmented data is more consistent for predicting pseudo-labels.

D.3 Impact of Strong Augmentation on Concurrent Methods

In the main experiments, we evaluate each method given the optimization and parameters provided in the original papers or code repository. Each of these methods preprocess their data using random translation and random horizontal flip which is the same as the weak augment used in our approach.

Since strong augmentation is a core component of the PLCiL, we also evaluate here if other methods can benefit from a wider variety of image transformations. In table 10,

Augmentation	Last (%)	Avg (%)
weak + CTAugment	61.5	74.0
weak + weak	2.8	8.4
CTAugment + CTAugment	45.8	62.3
RandAugment + CTAugment	53.8	68.2

Table 9: Class incremental performance for several data argumentation strategies on incremental CIFAR-100-full.

we report the performance of each method on the 3 benchmarks of our study using strong augmentation.

The results show that strong augmentation has mitigated results on the other methods. GDumb and WA are significantly improved on all 3 datasets. BiC and iCaRL on the other hand have some inconsistencies which can lead to a drop in accuracy on some benchmarks. For instance, iCaRL reaches top performance on CIFAR-100-full with strong augmentation, even outclassing our PLCiL in final accuracy, but has the opposite behavior on ImageNet-100-10% where the performance is strongly lowered. DMC+ is the most negatively impacted method with an important degradation of its performance. The model did not converge for both CIFAR-100 experiments.

These experiments suggest that more data augmentation is a simple solution to increase the incremental performance but does not fit some CI methods which have been mostly designed with only weak augmentation. The nature of the dataset is also an important factor. Our PLCiL on the other hand seems to behave consistently with strongly augmented data across the 3 benchmarks evaluated here.

Method	Weak Augment		Strong Augment	
	Last (%)	Avg (%)	Last (%)	Avg (%)
CIFAR-100-full				
GDumb [9]	27.8	42.0	33.9	48.4
iCaRL [5]	53.9	63.9	62.3	69.7
BiC [10]	55.9	67.1	46.9	68.9
WA [9]	50.8	64.4	51.3	65.8
DMC+ [8]	50.4	62.8	No Convergence	
Ours			61.5	74.0
CIFAR-100-20%				
GDumb [9]	28.2	42.2	34.4	50.4
iCaRL [5]	42.7	48.9	47.2	53.9
BiC [10]	43.3	49.8	46.9	60.8
WA [9]	40.5	49.7	52.6	62.1
DMC+ [8]	36.4	42.8	No Convergence	
Ours			59.8	66.5
ImageNet-100-10%				
GDumb [9]	40.6	59.6	49.1	67.0
iCaRL [5]	45.4	57.8	40.1	52.2
BiC [10]	50.7	62.4	38.6	44.9
WA [9]	30.2	54.7	40.0	50.9
DMC+ [8]	56.2	68.1	44.8	63.0
Ours			61.3	73.8

Table 10: Effect of adding strong augmentation in concurrent baselines. The *Weak Augment* column reports the performance from the original implementations and are the results presented in the section 4.1 and 4.2 of our paper. *Strong Augment* lists all new results obtained when applying strong augmentations. We also report the performance of our PLCiL, which uses both, as a reference.

References

- [1] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. ReMixMatch: Semi-Supervised Learning with Distribution Matching and Augmentation Anchoring. In *8th International Conference on Learning Representations, ICLR*, 2020.
- [2] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. AutoAugment: Learning Augmentation Strategies From Data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 113–123, 2019.
- [3] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.
- [4] Ameya Prabhu, Philip Torr, and Puneet Dokania. GDumb: A Simple Approach that Questions Our Progress in Continual Learning. In *The European Conference on Computer Vision (ECCV)*, 2020.
- [5] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental Classifier and Representation Learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [6] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS*, 2020.
- [7] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large Scale Incremental Learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.
- [8] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry P. Heck, Heming Zhang, and C.-C. Jay Kuo. Class-incremental Learning via Deep Model Consolidation. In *IEEE Winter Conference on Applications of Computer Vision, WACV*, 2020.
- [9] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining Discrimination and Fairness in Class Incremental Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2020.