Divide-and-Conquer for Enhancing Unlabeled Learning, Stability, and Plasticity in Semi-supervised Continual Learning

Supplementary Material

A. More Implementation Details

A.1. Exemplar Management

We follow the exemplar management strategy of iCaRL [9]. Whenever the new classes are encountered, we adjust the exemplar set. All classes are treated equally, meaning that when k classes have been observed so far and M is the total number of storable samples, $m^t = \lceil M/k \rceil$ samples are allocated for each class at the t-th task. This ensures that the memory budget of M samples is always fully utilized but never exceeded.

Two routines are responsible for sample management: one for selecting samples for new classes and the other for reducing the size of the exemplar sets for previously classes. Algorithm 1 outlines the sample selection process. Exemplars $e_1, \dots, e_m$ are selected and stored iteratively until the target number m is reached. At each iteration, a sample from the current training set is added to the exemplar set. The sample is chosen such that its feature vector brings the average feature vector of the exemplars closest to the average feature vector of the training samples. As a result, the exemplar "set" is effectively a priority-ordered list, where the order of elements matters, and exemplars earlier in the list are more significant. The procedure for removing samples is specified in Algorithm 2, and it is particularly straightforward: to reduce the number of samples from any m' to m, simply discard the samples $e_{m+1}, \dots, e_{m'}$, retaining only the exemplars $e_1, \dots, e_m$.

Algorithm 1: Constructing Exemplar Set

```
Input: Labeled dataset $D_l^{t,(i)} = \{x_{l,(1)}^{t,(i)}, \cdots, x_{l,(n)}^{t,(i)}\}$
       of class $i$, target number of exemplars $m^t$,
       current feature extractor $F^t(\cdot)$.
Output: Exemplar set $E^{t,(i)}$
$\mu_{D_l^{t,(i)}} = \frac{\sum_{x_l^{t,(i)} \in D_l^{t,(i)}} F(x_l^{t,(i)})}{|D_l^{t,(i)}|}$
for $k = 1, \cdots, m^t$ do
    $x_{e,(k)}^{t,(i)} = \underset{x_l^{t,(i)} \in D_l^{t,(i)}}{\text{argmin}} ||\mu_{D_l^{t,(i)}} - \frac{1}{k}(F^t(x_l^{t,(i)}) + \sum_{j=1}^{k-1} F^t(x_{e,(j)}^{t,(i)}))||$
end
$E^{t,(i)} = \{x_{e,(1)}^{t,(i)}, \cdots, x_{e,(m^t)}^{t,(i)}\}$
```

Algorithm 2: Reducing Exemplar Set

```
Input: Target number of exemplars $m^t$, exemplar
       set $E^{t-1,(i)}$ for class $i$
Output: Exemplar set $E^{t,(i)}$ for class $i$
$E^{t,(i)} = \{x_{e,(1)}^{t-1,(i)}, \cdots, x_{e,(m^t)}^{t-1,(i)}\}$
```

A.2. Implementation Details For CUB

For CUB [3], we follow the experimental setup and training pipeline of UaD-CIE [4]. We use a base learning rate of 0.001 during the first task, which is divided by 10 after 80 and 120 epochs (out of a total of 160 epochs). For subsequent tasks, the learning rate is set to 0.0005, with a total of 60 supervised epochs. The training batch size is set to 32, and the testing batch size is set to 50. We use a memory buffer of size 2000, managed in accordance with iCaRL [9]. All loss weights $\lambda_{\texttt{uns}}$, $\lambda_{\texttt{cl}}$, $\lambda_{\texttt{fsr}}$, and $\lambda_{\texttt{cud}}$ are set to 1.0, and temperature parameters β, γ, and ξ are set to 0.1.

A.3. Building USP Based on DER

DER [11] preserves the old network by parameter consolidation. At each incremental step, DER freezes previously learned representations and enhances them by adding new feature extractors, which introduce additional feature dimensions to the old representations. Additionally, DER introduces an auxiliary classifier $A(\cdot)$ to encourage the model to learn diverse and distinguishable features of new concepts. When constructing the USP based on DER, we follow DER's dynamic network expansion during training while replacing $\mathcal{L}_{\texttt{cl}}$ with DER's corresponding training loss while keeping all other loss terms unchanged. Specifically, $\mathcal{L}_{\texttt{cl}}$ is modified as:

$$\mathcal{L}_{\texttt{cl}}(D^t \cup E^t, F^t) = \mathbb{E}_{x_l^t \sim D^t \cup E^t} \left[H(\bar{p}_{x_l^t}^t, \bar{y}_l^t) \right] + \mathcal{L}_S(F^t), \tag{1}$$

where, $\bar{p}_{x_l^t}^t = A^t(F^t(x_l^t))$ represents the prediction output of the auxiliary classifier $A^t(\cdot)$ introduced by DER. $A^t(\cdot)$ is a $(|\mathcal{Y}^t| + 1)$-way classifier that treats all samples in the exemplar set E^t as a single category. $\bar{y}_l^t$ represents the label, where $\bar{y}_l^t = y_l^t$ for $x_l^t \in D^t$ and $\bar{y}_l^t = |\mathcal{Y}^t| + 1$ for $x_l^t \in E^t$. $\mathcal{L}_S(F^t)$ is the regularization loss computed based on the parameters of F^t to prevent excessive model complexity. For detailed calculations, please refer to [11].

A.4. Neural Collapse and Equiangular Tight Frame

Neural collapse refers to the phenomenon occurring at the late stage of training on balanced data (after the training error rate reaches 0). It reveals the geometric structure formed

Table 1. Performance comparisons on a 20-task continual learning benchmark under different data availability settings on ImageNet-100. We report both the original results of NNCSL [6] and the results of our own re-run (denoted as *). In the original paper of NNCSL [6], only the last accuracy is reported, without the average and task-level accuracy.

Labels	**Method**	**Task ID**																				**Avg**
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	
1%	NNCSL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	29.70	-
	NNCSL*	59.50	50.20	39.71	43.50	38.58	34.13	32.88	29.50	30.59	29.84	27.93	30.53	31.09	30.37	30.22	29.70	29.62	29.36	28.79	28.98	34.25
	iCaRL&Fix+USP	**64.80**	50.80	52.93	49.50	44.80	39.67	34.97	34.55	32.49	31.48	29.27	33.13	33.48	34.57	34.05	33.12	32.87	31.29	30.40	28.64	37.84
	DER&Fix+USP	64.40	**55.00**	**53.33**	**51.10**	**47.12**	**43.13**	**41.60**	**41.00**	**38.58**	**37.08**	**35.64**	**38.10**	**37.78**	**36.91**	**36.53**	**34.20**	**33.48**	**33.09**	**33.64**	**32.78**	**40.00**
5%	NNCSL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	51.30	-
	NNCSL*	58.00	55.60	45.43	48.80	27.93	39.53	39.53	39.53	37.59	40.04	39.52	42.13	42.31	43.51	43.16	41.73	39.40	41.69	42.43	43.26	42.56
	iCaRL&Fix+USP	73.60	62.40	68.00	66.00	61.52	56.93	54.80	52.55	51.11	51.84	50.04	52.23	51.85	52.11	52.40	50.85	49.81	49.16	49.05	48.46	54.56
	DER&Fix+USP	**76.00**	**74.80**	**72.00**	**72.00**	**63.68**	**60.20**	**58.63**	**57.10**	**54.93**	**53.12**	**53.20**	**55.20**	**55.17**	**55.63**	**55.89**	**54.70**	**53.58**	**53.53**	**53.37**	**53.62**	**59.32**
25%	NNCSL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**65.60**	-
	NNCSL*	60.00	60.00	51.43	54.30	48.17	43.40	42.12	41.90	44.05	44.44	42.33	44.033	45.53	46.14	45.78	46.24	43.53	41.48	41.67	44.12	46.53
	iCaRL&Fix+USP	78.00	**77.00**	79.73	78.50	71.60	**68.00**	**65.09**	**63.00**	**60.13**	**58.12**	**57.83**	**58.97**	**59.82**	58.17	**59.07**	**55.60**	**55.48**	54.49	53.77	53.78	**63.31**
	DER&Fix+USP	**80.40**	76.60	**79.87**	**79.20**	**71.76**	66.67	64.57	60.80	58.18	56.40	55.89	58.70	57.66	**58.57**	55.39	53.42	51.04	**54.69**	**56.82**	55.54	62.61

by the final layer features and the classifier, which can be defined as a simplex Equiangular Tight Frame (ETF), which refers to a matrix composed of K vectors in $\mathbb{R}^d$, satisfying:

$$E = \sqrt{\frac{K}{K-1}} U(I_K - \frac{1}{K} 1_K 1_K^T), \quad (2)$$

where $E = [e_1, \cdots, e_K]$. $U \in \mathbb{U}^{d \times K}$ allows a rotation and satisfies $U^\top U = I_K$, I_K is the identity matrix, and 1_K is an all-ones vector. All column vectors in E satisfies:

$$e_{k_1}^\top e_{k_2} = \frac{K}{K-1}\delta_{k_1,k_2} - \frac{1}{K-1}, \; \forall k_1, k_2 \in [1, K], \quad (3)$$

where $\delta_{k_1,k_2} = 1$ when $k_1 = k_2$, and 0 otherwise. All vectors have the same L_2-normalization and any pair of two different vectors has the same inner product of $-\frac{1}{K-1}$, which is the minimum possible cosine similarity for K equiangular vectors in $\mathbb{R}^d$.

In our method, we use an simplex equiangular tight frame as the pre-defined class prototype features, with the sample features of each class aligned to it. More details about the neural collapse phenomenon can be found in [12].

B. Additional Experimental Results

Unless otherwise specified, DSGD [5] and USP both adopt iCaRL&FixMatch [5] as the base SSCL learner.

B.1. More SSCL Protocols

B.1.1. NNCSL Protocol

To ensure a comprehensive comparison with recent work, we conduct additional experiments to evaluate our method, USP, against NNCSL [6]. The original NNCSL protocol utilizes a different 20-task setting on ImageNet-100, which is distinct from our primary 10-task setup. To provide a fair comparison, we evaluate USP under NNCSL protocols. The results are presented in Tab. 1. The experiments show that USP consistently outperforms NNCSL across all settings, demonstrating the superior effectiveness and robustness of our approach.

Table 2. Average and last accuracy on 5-task CIFAR10-30 with two more realistic SSCL settings.

Method	**Imbalanced**		**Inconsistent**	
	Avg	**Last**	**Avg**	**Last**
DSGD	62.42	62.96	57.58	59.92
USP	**75.18**	**65.50**	**70.26**	**60.39**

Table 3. Ablation experiments on whether uses low-confidence samples ("LCS") on 5-task CIFAR10-30.

	Avg	**Last**
wo. LCS	68.34	61.01
w. LCS	**81.43**	**73.65**

B.1.2. SSCL with Non-IID Distributions

We consider two more realistic SSCL scenarios: (1) training with a long-tailed class distribution for each task ("*imbalanced*"); (2) training with various data amounts across tasks ("*inconsistent*"). Specifically, we conduct experiments on the 5-task CIFAR10-30. In the imbalanced setting, we set the number of labeled and unlabeled data for each class in each task to $\{30, 150\}$ and $\{600, 3000\}$. In the inconsistent setting, we set the training data sizes for the five tasks to $\{10000 \to 250 \to 125 \to 5000 \to 625\}$. The results are shown in Tab. 2. As can be seen, our method demonstrates stronger robustness, with performance clearly outperforming the previous SOTA SSCL method.

B.2. More Ablation Studies

B.2.1. Utilization of Low-Confidence Unlabeled Data

To present the contribution of DCP, we conduct the following ablation experiments on using the low-confidence unlabeled data: traditional classifier with thresholded pseudo-labeling v.s. our proposed DCP, which is shown in Tab. 3. This comparison demonstrates that reasonably learning from low-confidence samples, rather than simply discarding them to avoid potential errors, can indeed lead to tangible performance improvements.

Table 4. Ablation studies on different distillations on 10-task CIFAR100-25.

Method	Avg	Last
logit	53.91	37.97
feature	48.16	33.56
CUD	**54.36**	**38.25**

Table 5. Ablation studies on loss weights of $\mathcal{L}_{\texttt{fsr}}$ on 5-task CIFAR10-30.

$\lambda^{l}_{\texttt{fsr}}$	$\lambda^{u}_{\texttt{fsr}}$	Avg	Last
1.0	0.5	79.52	70.21
1.0	1.0	**81.63**	**73.65**
0.5	1.0	78.38	68.78

B.2.2. More Distillation Methods

We explore the use of existing distillation methods for distilling from unlabeled data, specifically logit distillation and feature distillation. In particular, we apply consistency regularization directly on the logits or features output by the models of the current task and the previous task for unlabeled data. These experiments are compared with our proposed CUD, which are shown in Tab. 4. It is evident that our CUD outperforms both logit and feature distillation.

B.2.3. Hyper-parameters

Confidence Threshold and Feature Dimension. We conduct ablation studies on the confidence threshold τ and the feature dimension d. As Fig. 1 Shown, USP achieves the best performance with appropriately tuned default values. The threshold τ is set following standard practice in semi-supervised learning methods (*e.g.*, FixMatch [10]), and the method demonstrates low sensitivity to variations in d.

Loss Weights. In our paper, the $\mathcal{L}_{\texttt{fsr}}$ sums the labeled and unlabeled parts with the same weight. We further apply different loss weights to labeled and unlabeled data to investigate their impact on the performance of the method. We denote the loss weight for unlabeled data as $\lambda^{u}_{\texttt{fsr}}$ and for labeled data as $\lambda^{l}_{\texttt{fsr}}$, and conduct the corresponding ablation experiments. The experimental results are shown in Tab. 5. The performance is best when the loss weights for labeled and unlabeled data are equal. Increasing or decreasing the relative weight of the unlabeled data leads to a performance drop, indicating that the pseudo-labels obtained through our divide-and-conquer labeling have high quality.

B.2.4. More Backbones and Pre-Training Strategies

In the main text, we follow the experimental setup of DSGD [5] and primarily use ResNet-32 and ResNet-18 without pre-training as the backbones for our method. To further investigate the impact of different backbones and pre-training strategies on the performance of our method, we use iCaRL&Fix as the base SSCL learners and conduct ablation experiments. The experimental results are shown in Tab. 6. We observe that using properly sized networks with appropriate pre-training leads to better USP performance. Simply using larger networks or advanced pre-training without proper adaptation does not guarantee improved SSCL performance (as found in [7]). Making USP more compatible with larger networks and diverse pre-training approaches remains our future work.

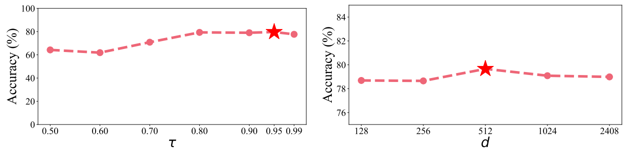

Figure 1. Average accuracy with various confidence thresholds and feature dimensions on 5-task CIFAR10-30.

Table 6. Ablation studies on different backbone architectures on the 5-task CIFAR10-30. Meanwhile, we adopt different pre-training strategies (CLIP [8] and DINO [2]) on ResNet-50 to show the performance potential of our method.

Backbone	ResNet20		ResNet32		ResNet50		CLIP		DINO	
	Avg	Last	Avg	Last	Avg	Last	Avg	Last	Avg	Last
DSGD	72.63	69.43	77.33	**76.41**	73.81	65.01	72.43	72.02	77.29	70.41
USP	**80.00**	**69.59**	**79.66**	70.43	**75.17**	**67.24**	**80.88**	**74.08**	**78.86**	**71.35**

B.3. Discussions on Memory Buffer Size

By default, we follow the setup of iCaRL [9] and use a buffer size of 5120 to store a portion of the labeled data from each task as the exemplar set. To further investigate the impact of buffer size, we conduct additional ablation experiments, with the results presented in Tab. 7. As shown, a buffer size of 5120, which is the typical choice for most replay-based methods [1, 6, 9], achieves the best performance. Using a fixed-size exemplar buffer is a standard practice in continual learning [5, 6, 9], as it reflects realistic memory constraints and enables fair comparisons with existing SSCL methods. While labeled data are indeed scarce in SSCL, the memory budget may still be insufficient to retain all labeled samples—particularly in settings with long task sequences (*i.e.*, task ID → ∞) or high supervision levels (*e.g.*, CIFAR100-125 or ImageNet100-100, where the number of labeled samples reaches 12.5K and 10K, respectively, far exceeding the our default memory buffer size of 5120). In such scenarios, USP adopts an iCaRL-style exemplar buffer to strike a balance between memory efficiency and model performance.

Although USP is designed under the realistic assumption of limited memory, our three key components—FSR, DCP, and CUD—are orthogonal to buffer size and remain effective even under larger or unlimited memory settings. Notably, DCP and CUD can also effectively leverage the unlabeled sample pool to address distribution shifts across tasks.

Table 7. Ablation studies on memory buffer size of exemplar set E^t on 5-task CIFAR10-30.

Buffer Size	CIFAR10-30		CIFAR10-150	
	Avg	Last	Avg	Last
250	71.66	59.93	79.25	66.76
500	73.21	61.75	80.71	72.48
5120	**79.66**	**70.43**	**84.78**	**78.21**

Table 8. Comparisons with CL-based baselines (combine FixMatch [10] to exploit unlabeled data) using a larger buffer size 20K, which is enough to retain all labeled samples.

Method	CIFAR100-125		ImageNet100-100	
	Avg	Last	Avg	Last
iCaRL&Fix (20K)	62.07	46.56	40.40	26.91
+ USP (20K)	**68.65**	**55.17**	**56.91**	**51.73**
DER&Fix (20K)	68.75	54.83	62.02	53.46
+ USP (20K)	**70.60**	**61.33**	**62.17**	**58.34**

To further verify the performance of USP under idealized conditions where the buffer is sufficiently large to retain all labeled samples, we conduct additional experiments on CIFAR100-125 and ImageNet100-100 with a buffer size of 20K. As shown in Tab. 8, USP continues to achieve strong performance in this setting, demonstrating the robustness and generality of our approach.

References

[1] Matteo Boschini, Pietro Buzzega, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Continual semi-supervised learning through contrastive interpolation consistency. *Pattern Recognition Letters*, 162:9–14, 2022. 3

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE/CVF International Conference on Computer Vision*, 2021. 3

[3] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2018. 1

[4] Yawen Cui, Wanxia Deng, Haoyu Chen, and Li Liu. Uncertainty-aware distillation for semi-supervised few-shot class-incremental learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 1

[5] Yan Fan, Yu Wang, Pengfei Zhu, and Qinghua Hu. Dynamic sub-graph distillation for robust semi-supervised continual learning. In *AAAI Conference on Artificial Intelligence*, 2024. 2, 3

[6] Zhiqi Kang, Enrico Fini, Moin Nabi, Elisa Ricci, and Karteek Alahari. A soft nearest-neighbor framework for continual semi-supervised learning. In *IEEE/CVF International Conference on Computer Vision*, 2023. 2, 3

[7] Kuan-Ying Lee, Yuanyi Zhong, and Yu-Xiong Wang. Do pre-trained models benefit equally in continual learning? In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023. 3

[8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 3

[9] 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. 1, 3

[10] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 2020. 3, 4

[11] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1

[12] Yibo Yang, Haobo Yuan, Xiangtai Li, Zhouchen Lin, Philip Torr, and Dacheng Tao. Neural collapse inspired feature-classifier alignment for few-shot class-incremental learning. In *International Conference on Learning Representations*, 2022. 2