A. A Systematic Analysis of CL Methods

In this section, we present the results of our preliminary analysis to evaluate representative continual learning (CL) strategies in semi-supervised continual learning (SSCL). As shown in Fig. 7, for all the four semi-supervised classifiers, sequential training (ST) on the incremental partially labeled data significantly underperforms joint training (JT) of all the data ever seen, suggesting the existence of catastrophic forgetting. Also, implementation of weight regularization methods on a semi-supervised classifier cannot address this issue in SSCL. While, using a supervised memory buffer (SMB) to replay old labeled data can effectively alleviate catastrophic forgetting, but still remains a performance gap to JT, which is caused by the *catastrophic forgetting of unlabeled data*. However, an additional unsupervised memory buffer (UMB) cannot effectively exploit the incremental unlabeled data to improve SSCL. We extensively evaluate the UMB of various sizes in Fig. 8. Even the large UMB with 10000 images (the size is twice of a typical generator used in ORDisCo) cannot further improve the performance from mitigating the catastrophic forgetting of unlabeled data, while a smaller UMB with 500 or 1000 images results in severe overfitting and thus interferes SSCL. Therefore, a common issue of existing CL strategies is the lack of ability to continually exploit unlabeled data in SSCL.

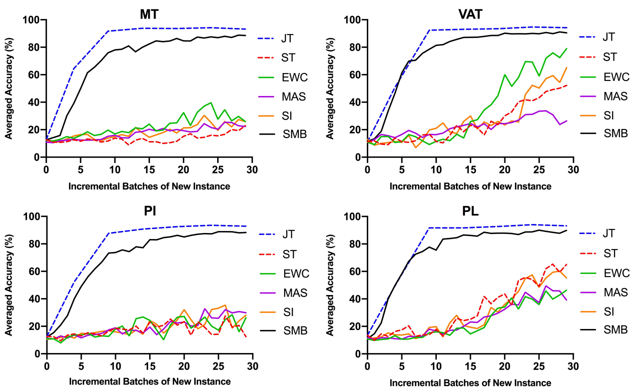


Figure 7. Baselines that combine SSL classifiers and representative CL methods to address SSCL. JT: Joint training of all the training samples ever seen. ST: Sequential training on the incremental data; SMB: ST with a supervised memory buffer to replay the labeled data ever seen.

From the empirical analysis, we observe that using SMB can effectively alleviate catastrophic forgetting in SSCL. Classical memory replay methods for supervised CL [32, 4] use a mean-of-feature strategy to select samples, where the images that are closest to the feature mean of a class are stored in the memory buffer. A commonly-used size of the memory buffer is 20 samples per class, i.e. 200 samples for 10 classes. While, compared with supervised CL, the labeled data are generally insufficient in SSCL to perform selection, e.g., SVHN-1 only provides 300 labeled data in total. Here we adapt the mean-of-feature strategy to select samples from small amounts of labeled data (SMB-mof) in SVHN-3, and compare it with replay of all the labeled data ever seen (SMB), which is similar to unified sampling and is indeed competitive to existing selection methods as analyzed in [5]. We use a fixed size of 200 samples for SMB-mof, so SMB-mof is equal to SMB in the early 6 batches (180 labeled data in total) and then perform selection. The better performance of SMB than SMB-mof, particularly when more labeled data are introduced in the later batches, indicates that selecting representative labeled data via mean-of-feature still suffers from the catastrophic forgetting of labeled data. Since our motivation is to improve continual learning of unlabeled data in SSCL, we replay all the labeled data ever seen in SMB. A potential future work can be designing proper strategy to effectively select labeled data for memory buffer in SSCL.

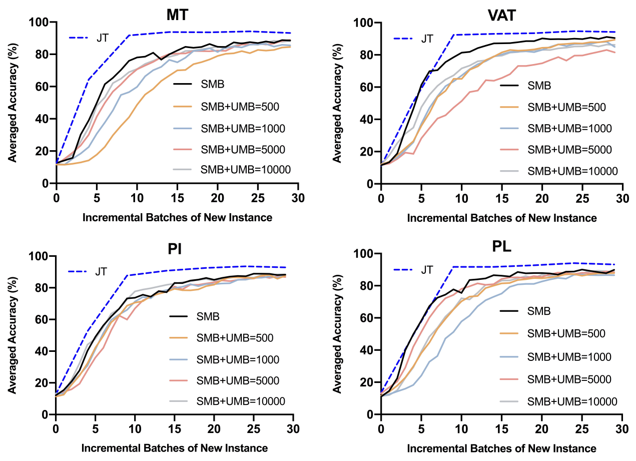


Figure 8. Performance of the baselines that combine SMB with an additional unsupervised buffer (UMB) of various sizes.

Table 6. Comparison of unified sampling (SMB) and mean-of-feature (SMB-mof) to select the supervised memory buffer.

		Batch 15	Batch 20	Batch 25	Batch 30
MT	SMB	79.93	86.40	87.57	88.55
	SMB-mof	79.95	83.76	84.40	85.03
VAT	SMB	87.14	88.55	90.02	90.44
	SMB-mof	86.33	87.20	87.10	88.65
PI	SMB	77.20	86.20	87.30	88.32
	SMB-mof	78.41	83.62	80.87	85.67
PL	SMB	86.54	87.88	88.79	89.88
	SMB-mof	82.24	86.26	85.48	86.72

To visualize the data coverage of UMB and ORDisCo on the training data distribution in Fig. 2, we jointly train a feature embedding layer on SVHN and apply t-SNE projection to visualize the feature embedding of the unlabeled data in

Table 7. Implementations of SSL classifiers on SVHN dataset with 1000 labels. We present the averaged accuracy (%) on the test set of SVHN. We follow similar implementations as [27] and achieve a comparable performance as the published performance. The slight variance of performance might be caused by different random seeds.

	Our Implementation	Published Performance [27]
MT	93.21	94.35
VAT	94.22	94.37
PI	92.89	92.81
PL	93.19	92.38

UMB, the generated data sampled from ORDisCo and all the training data. The size of the UMB is around 10% of all the training data, which is comparable to the storage cost of a typical generator used in ORDisCo. However, the coverage of the generated data is substantially better than the UMB with a similar size.

B. Implementation of SSL Classifiers and CL Methods

For all the semi-supervised classifiers, we jointly train 200,000 iterations with the same semi-supervised split as [27]. To validate our implementations, in Table 7 we summarize the performance of our implementations and the published performance [27] on SVHN with 1000 labels.

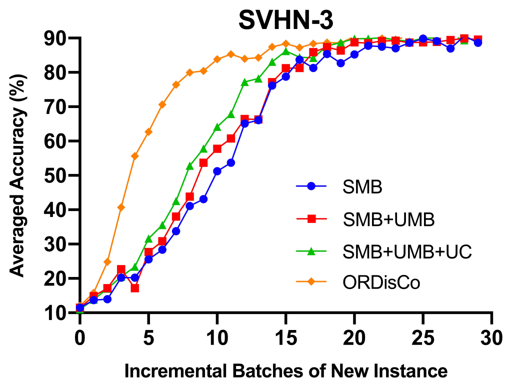
Implementation details of SSL classifiers and CL methods are as follows: For sequential training, we search the number of training epochs among 100, 200, 500 and 1000. 500 training epochs within each batch gives the best average results on the four SSL methods. As for regularization-based SSCL extension (EWC, SI and MAS), we search the best lambda among 0.1, 1, 10 and 100 for EWC and SI, and 10^{-6}, 10^{-4}, 10^{-2} and 1 for MAS, respectively. Since the suitable number of training epochs may change, we do an additional grid search in 250, 500 and 750 for all the three CL methods and choose the number of epochs that achieve the best performance. The results of lambda search are summarized in Table 8. We demonstrate the best accuracy of each trial.

C. SSCL of New Instance on SVHN-3 and CIFAR10-13

The performance of SSCL of new instance on SVHN-3 and CIFAR10-13 is summarized in Fig. 9, with ablation study in Table 9. ORDisCo achieves a much better performance than other baselines, particularly on fewer incremental batches of partially labeled data. The improved performance on smaller amounts of data indicates that ORDisCo can *quickly learn a usable model* from incremental data, without waiting for more data to be collected. The ablation study also shows that the online semi-supervised generative replay substantially improves SSCL, while the regularization of discrimination consistency is more effective when the partially labeled data are limited, i.e. fewer incremental batches (Table 9) and labels (Table 3).

Table 8. Hyperparameter search of weight regularization methods in SSL classifiers to address SSCL on SVHN-3. We present the best accuracy (%) during incremental learning of 30 partially labeled batches.

	lambda	MT	PI	PL	VAT
EWC	0.1	21.23	29.61	16.84	66.97
	1	45.76	28.15	33.50	61.51
	10	23.54	28.25	18.56	78.96
	100	35.80	28.45	18.28	29.61
SI	0.1	40.36	35.45	60.28	20.38
	1	32.22	27.56	47.64	65.13
	10	30.04	31.36	45.92	40.65
	100	16.21	27.37	64.64	19.54
MAS	10^{-6}	36.95	32.90	49.58	33.70
	10^{-4}	23.52	21.41	20.20	15.87
	10^{-2}	20.46	17.78	15.84	13.51
	1	12.66	12.15	11.47	11.75



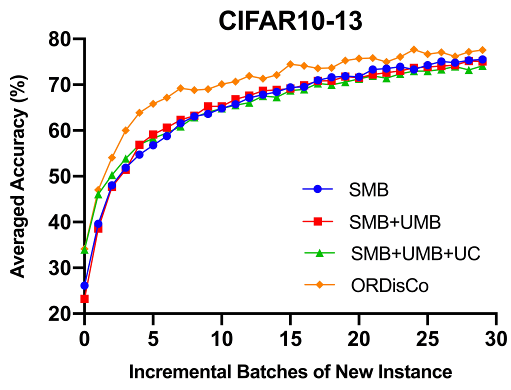


Figure 9. Averaged accuracy (%) of classification on CIFAR10. UMB: An unsupervised memory buffer of a similar size as our generator; UC: Using the unified classifier to exploit UMB.

Next, we consider to use a much larger UMB. The orig-

Table 9. Ablation study on SVHN-3 and CIFAR10-13. The online semi-supervised generative replay (Replay) substantially improves SSCL, while the regularization of discrimination consistency (Reg) is more effective on fewer incremental batches (15 Batch).

Method	SVHN-3 15 Batch	SVHN-3 30 Batch	CIFAR10-13 15 Batch	CIFAR10-13 30 Batch
SMB	76.16	88.60	64.23	74.22
ORDisCo (+Replay, -Reg)	85.44	89.03	71.70	**78.59**
ORDisCo (+Replay, +Reg)	**87.44**	**90.75**	**74.45**	77.56

inal size of UMB (around 5000 images) is comparable to the one of a typical generator used in ORDisCo. While, the larger UMB (around 20000 images) is 4 times larger than the original one and also much larger than all the memory cost of both the generator and the discriminator to train ORDisCo. As shown in Table 10, to much enlarge the UMB with or without the unified classifier (UC) [8] can only slightly improve SSCL but substantially underperforms ORDisCo.

Table 10. SMB with larger UMB. To use a larger UMB (4×UMB) with or without the unified classifier (UC) only slightly improves SSCL of new instance, but significantly underperforms ORDisCo.

Method	SVHN-1 15 Batch	SVHN-1 30 Batch	CIFAR10-5 15 Batch	CIFAR10-5 30 Batch
SMB	38.08	73.32	59.61	67.18
SMB+UMB	30.26	59.06	52.90	61.88
SMB+UMB+UC	33.99	65.23	53.61	61.03
SMB+4×UMB	30.74	62.96	57.94	65.45
SMB+4×UMB+UC	31.50	66.93	58.45	66.26
ORDisCo	**55.07**	**85.52**	**66.58**	**73.13**

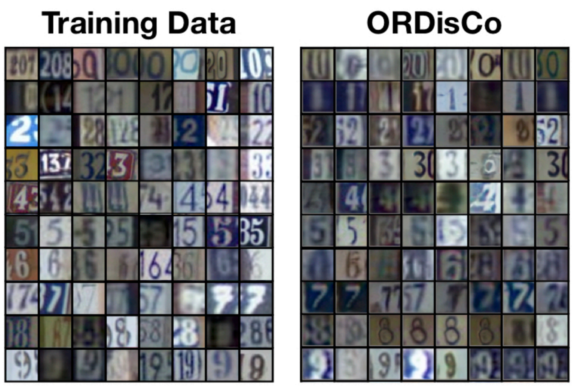
Method	SVHN-3 15 Batch	SVHN-3 30 Batch	CIFAR10-13 15 Batch	CIFAR10-13 30 Batch
SMB	76.16	88.60	64.23	74.22
SMB+UMB	77.14	89.54	68.89	75.06
SMB+UMB+UC	83.11	89.36	67.26	74.12
SMB+4×UMB	79.72	90.20	69.42	75.96
SMB+4×UMB+UC	80.55	89.05	69.03	75.56
ORDisCo	**87.44**	**90.75**	**71.35**	**77.56**

D. Conditional Samples

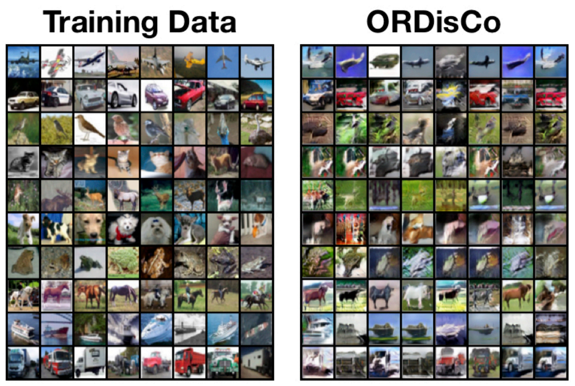
Here we show the conditional samples of ORDisCo learned from incremental partially labeled data in Fig. 10.

E. Complexity Analysis of Generative Replay

We propose an online semi-supervised generative replay strategy in ORDisCo, which is a time- and storage-efficient way to exploit the incremental data. We provide a complexity analysis in Table 1 and Fig. 11, of two commonly-used strategies that sample and replay generated data in an offline manner, and our online strategy. Generative replay applies generative models to continually recover the learned



(a) SVHN-3



(b) CIFAR10-13

Figure 10. Conditional generation in SSCL. We show the conditional samples of ORDisCo after incremental learning of 10 batches on SVHN-3 (a) and CIFAR10-13 (b).

data distribution to overcome catastrophic forgetting. Many existing work replay generated data in an offline manner, which can be conceptually separated as two strategies: (1) All the generators learned on each task or batch are saved, and replay conditional samples to a classifier for inference [28]; and (2) After training on each task or batch, the generator is saved to replay conditional samples with the next task or batch to learn a new generator. Then the saved generator is updated by the new one [35, 43].

As shown in Fig. 11, Strategy 1 has to store the learned data distribution of each task or batch as a dedicated generator and replays them from each generator for inference. Thus, for SSCL of B incremental batches, the storage and time complexities of Strategy 1 are O(B) and O(B), respective. Strategy 2 continually updates one generator after learning each task or batch. While, to overcome catastrophic forgetting in the generative model, Strategy 2 has to sample sufficient generated data from the old generator after learning each task for replay, even though the inference for classification is not required at that time. So the storage and time complexities of Strategy 2 are O(1) and O(B^2). If inference is required after learning each task or batch, the

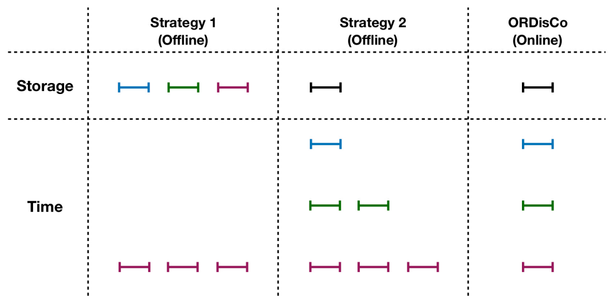


Figure 11. Storage and time complexity analysis of two offline strategies and our online strategy. We use 3 batches as an example to illustrate the storage and time complexity of the three strategies. The blue, green, and purple lines represent the storage and time costs of the first, the second and the third batch, respectively. While the black line represents the model is continually updated.

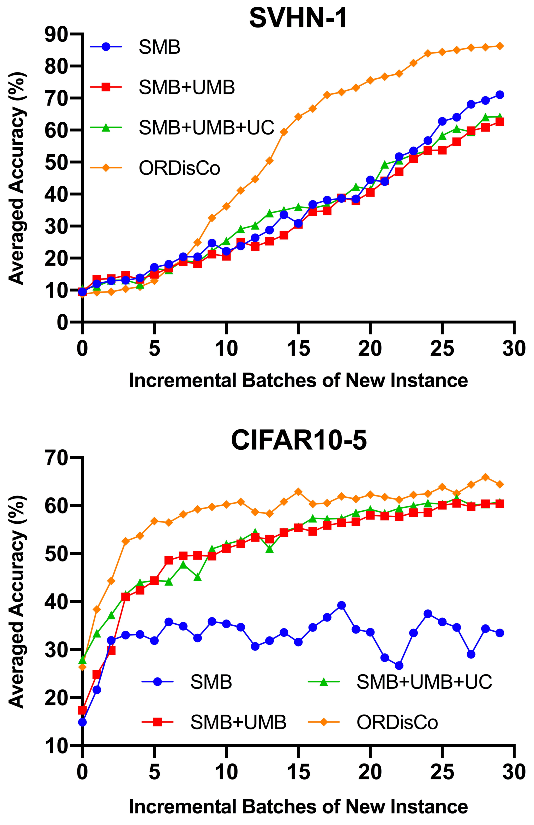


Figure 12. Selecting SMB of a fixed size through the mean-of-feature approach. SMB: A supervised memory buffer of a fixed size, selected by the mean-of-feature approach; UMB: An unsupervised memory buffer of a similar size as our generator; UC: Using the unified classifier to exploit UMB.

time complexity of Strategy 1 will be the same as Strategy 2, i.e. O(B^2). In contrast, the online replay strategy in ORDisCo continually updates one generator and replays a constant amount of generated data during training, where the storage and time complexities are O(1) and O(B). Note that although our online replay strategy has the same storage complexity as Strategy 2, the extra generator used in Strategy 2 results in more storage costs than ours.

F. Selecting SMB of a Fixed Size

Here we consider selecting SMB of a fixed size through a widely-used mean-of-feature approach [32, 4, 5] for ORDisCo and all the baselines. We keep SMB of the size 200 images, i.e. 20 images per class. As shown in Fig. 12, the smaller and fixed SMB results in more severe overfitting in SSCL, while ORDisCo substantially outperforms other baselines due to more effectively exploiting the incremental unlabeled data.

G. Hyperparameters of ORDisCo

α and λ are the two hyperparameters used in ORDisCo. Following [15], we keep $\alpha = 0.5$ to balance the discriminator predictions of fake data-label pairs from generator and classifier. While, λ is the hyperparameter to address catastrophic forgetting of unlabeled data in SSCL. We make a grid search of λ among 0.1, 0.01 and 0.001 based on training error, and keep $\lambda = 0.001$.