MAY THE FORGETTING BE WITH YOU: ALTERNATE REPLAY FOR LEARNING WITH NOISY LABELS – SUPPLEMENTARY MATERIAL

Anonymous authors
Paper under double-blind review

A DETAILED ALGORITHM

Algorithm 1 describes the overall procedure of our method. For each task, we perform training iteratively on data from the stream only, *i.e. buffer forgetting* (line 9), or on a concatenation of both stream and buffer data, *i.e. buffer learning* (line 6).

At each iteration, we draw a mini-batch from the buffer and update its examples' scores (line 3) to later allow the asymmetric sampling procedure. Every other epoch, we induce buffer forgetting, and is only during these epochs that we fill the buffer with new elements, to take full advantage of the loss gap between noisy and clean samples that these steps induce. We choose as a candidate set to be inserted in the buffer a subset of the current mini-batch that displays low loss values (line 11). From this group, indexes of elements to be actually inserted are picked following reservoir (line 12). If the buffer is full, samples to be replaced are chosen randomly with probability given by their score (line 12). We repeat this procedure for each task, while the model is saved and restored to the previous state for each epoch of buffer learning T_{on} (line 4).

Algorithm 1 Detailed procedure of AER with ABS

```
    Input: stream data $\mathcal{D}_t$, buffer $\mathcal{M}$, training epochs $T$
 1: for epoch in $T$ do
 2:     /* Sampling Scores Update */
 3:     $\mathbf{s}(\mathbf{x}) \leftarrow \{s(x); \forall(x, \tilde{y}) \in \mathcal{M}\}$          ▷ Eq. 4
 4:     if epoch in $T_{on}$ then                                   ▷ Sec. 3.2
 5:         /* Buffer Learning */
 6:         train with $\mathcal{M}$
 7:     else
 8:         /* Buffer Forgetting */
 9:         train on $\mathcal{B} \sim \mathcal{D}_t$
10:         /* Sample Insertion & Replacement */
11:         $\mathcal{R} \leftarrow \{(\mathbf{x}, \tilde{y}) \in \mathcal{D}_t : \mathcal{L}(\mathbf{x}, \tilde{y}) < \alpha\}$   ▷ Sec. 3.3.1
12:         $\mathcal{R} \leftarrow reservoir(\mathcal{R})$
13:         $\mathcal{M}[z \sim p(x)] \leftarrow \mathcal{R}$                      ▷ ABS 5
14:     end if
15: end for
```

A.A DETAILS REGARDING ABS

When describing **ABS** in Sec. 3.3.1, we suggested that the simple symmetric normalization would favour the replacement of samples from either the current or the past task. Indeed, since $\forall \mathbf{x}\ \mathcal{L}(\mathbf{x}) \geq 0$, if we define $z = \sum_{\mathbf{x} \in \mathcal{M}} s(\mathbf{x})$ (symmetric normalization), then it follows that either $\mathcal{L}(\mathbf{x}) \geq \mathcal{L}(\mathbf{x})$ or $\mathcal{L}(\mathbf{x}) \leq \mathcal{L}(\mathbf{x})$ if $z \geq 0$ or $z \leq 0$ respectively. Thus, the symmetric selection favours disproportionally samples from the present or the past.

To avoid this problem, we split the selection into two distinct phases:

1. We initially sample from a binomial distribution ϕ with probability $q = \mathcal{M}\mathcal{M}_{curr}$ to decide if we want to replace a sample from the present or the past;
2. Then, we perform the sample selection considering only either samples from the present or the past, using respectively $p_{curr}(\mathbf{x})$ or $p_{past}(\mathbf{x})$.

This strategy lets us strike a balance between balancing the buffer and ensuring the presence of both *i)* high loss (complex) samples from the past; and *ii)* low loss (clean) samples from the present.

Table A: Final Average Accuracy (FAA) [↑] and standard error across all tasks of different CNL methods on Seq. CIFAR-10 and WebVision with different noise rates. † Additional baselines created by adapting existing loss-based and CL approaches to the multi-epoch scenario.

Benchmark	**Seq. CIFAR-10**			**Seq. WebVision**
		symm		*instance*
Noise rate	20	40	60	N/A
Joint	79.65± 0.94	73.12± 1.61	60.53± 2.36	54.80± 2.66
Finetune	18.83± 0.23	18.01± 0.23	15.99± 0.58	15.96± 0.59
Reservoir	50.53± 2.37	33.64± 1.53	22.92± 1.64	27.10± 2.12
+ *CoTeaching*	50.11± 4.35	34.89± 1.67	22.98± 1.34	27.80± 0.85
+ *DivideMix*	55.69± 6.77	38.87± 2.16	26.13± 1.37	29.93± 2.34
GDumb	35.45± 2.44	27.76± 2.08	19.41± 1.93	25.00± 3.14
+ *CoTeaching*	36.94± 1.09	31.26± 1.15	19.75± 2.79	25.20± 1.56
+ *DivideMix*	38.60± 0.86	32.25± 9.92	21.06± 4.97	28.00± 2.15
PuriDivER	30.96± 0.50	27.23± 3.66	24.31± 1.59	29.10± 0.99
PuriDivER.ME	55.49± 0.50	49.44± 2.83	41.74± 3.16	36.40± 1.64
DividERMix	57.07± 1.12	45.65± 3.29	32.19± 5.30	36.20± 1.91
OURs	60.82± 2.49	59.47± 2.42	45.07± 1.66	34.20± 2.49
w. buffer fit.	69.12± 0.71	64.81± 0.65	50.04± 1.36	36.84± 2.89
w. consolidation	68.82± 0.97	67.14± 1.06	**54.59**± 0.18	38.87± 1.89
w. PuriDivER cons.	**71.09**± 0.88	**67.62**± 0.94	54.42± 1.02	**39.11**± 1.11
OURS - no ACE	59.26± 1.41	54.91± 1.76	35.85± 2.51	31.73± 0.94
w. buffer fit.	65.56± 0.95	61.82± 1.21	44.01± 3.55	34.00± 1.56

B EXPERIMENTS

To evaluate our proposal we build upon the open-source codebase provided by Mammoth (Buzzega et al. (2020); Caccia et al. (2022); Boschini et al. (2022)), a CL framework based on `PyTorch`.

DATASETS

We empirically validate our method on four different classification benchmarks as mentioned in the main paper. For experiments on CIFAR (Krizhevsky et al. (2009)) and NTU-60 (Shahroudy et al. (2016)), we corrupt the labels of the datasets at hand to obtain different noise configurations, which we then keep fixed for each of the experiments for fairness of results comparison across multiple methods.

In the process of injecting symmetric noise, we replace the ground-truth label with probability $r \in [0, 1]$ determined by the designated noise rate. The asymmetric or *class-dependent* noise setting is an approximation of real-world corruption patterns, which alters labels within the same superclass. For example, in the CIFAR-100 dataset, each image comes with a "fine" label (specific class) and a "coarse" label (superclass). Here, label transitions are parameterized by r such that the wrong class and true class have probability r and $1 - r$, respectively. This results in sample ambiguity occurring only between similar classes, as it would in a realistic scenario.

In each experiment, samples from the main dataset are split into disjoint sets based on their class and organized into tasks, following the ClassIL setup. We obtain the following versions of the datasets.
Seq. CIFAR-10 The original dataset contains $50,000$ train and $10,000$ test low-resolution color images in 10 different classes. During training the model encounters 2 classes per task, namely ("airplane", "car"), ("bird", "cat"), ("deer", "dog"), ("frog", "horse"), ("ship", "truck").
Seq. CIFAR-100 This original dataset is like the CIFAR-10, except it has 100 classes with 600 images each. Images are grouped into 20 superclasses, thus each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs). Following this

Table B: Final Average Accuracy (FAA) [↑] and standard error across all tasks of different CNL methods on Seq. CIFAR-100 with different noise rates. † Additional baselines created by adapting existing loss-based and CL approaches to the multi-epoch scenario.

Benchmark	**Seq. CIFAR-100**				
	symm			*asymm*	
Noise rate	20	40	60	20	40
Joint	54.77± 0.61	38.46± 0.92	23.36± 1.09	56.70± 0.57	42.61± 0.92
Finetune	08.65± 0.13	07.55± 0.14	06.15± 0.17	07.78± 0.14	05.73± 0.09
Reservoir	25.14± 0.28	14.64± 0.23	8.92± 0.23	29.42± 0.39	18.91± 0.86
+ *CoTeaching*	25.79± 0.61	14.46± 0.49	8.92± 0.30	32.18± 2.55	20.76± 2.44
+ *DivideMix*	33.31± 0.27	22.91± 0.43	13.58± 1.02	36.98± 0.78	26.10± 1.10
GDumb	16.96± 0.61	11.31± 0.45	7.62± 0.28	17.25± 0.28	11.75± 0.06
+ *CoTeaching*	17.02± 0.50	13.17± 0.31	8.17± 0.99	17.07± 0.54	12.05± 0.62
+ *DivideMix*	19.26± 0.97	15.67± 0.97	10.51± 0.32	18.80± 1.55	13.29± 0.29
PuriDivER	27.53± 0.53	24.36± 0.40	17.81± 0.43	25.46± 1.44	18.84± 0.64
PuriDivER.ME	41.25± 0.63	37.61± 0.85	27.18± 0.76	41.65± 0.49	30.22± 0.74
DividERMix	29.21± 0.31	22.41± 0.51	14.21± 1.07	29.38± 0.43	21.23± 1.14
OURs	44.34± 0.48	38.64± 0.57	26.34± 0.85	41.24± 0.40	29.26± 0.91
w. buffer fit.	**47.58**± 0.50	41.58± 0.63	30.13± 0.98	42.85± 0.37	31.49± 0.38
w. consolidation	46.11± 1.46	40.27± 0.40	**34.81**± 1.63	**43.67**± 0.73	**32.64**± 0.48
w. PuriDivER cons.	45.62± 2.01	**42.08**± 1.72	31.95± 0.41	42.63± 0.99	30.47± 0.57
OURs - no ACE	34.88± 1.95	29.51± 0.98	22.02± 0.91	33.16± 1.97	25.07± 1.56
w. buffer fit.	45.69± 1.61	39.83± 30.83	30.57± 0.74	43.36± 0.93	32.61± 1.02

categorization, we organize classes in 10 tasks, each containing 5 classes from the same superclass.
Seq. WebVision The original dataset (Li et al. (2017)) contains over 2.4 million training images belonging to 1,000 categories, where the number of images per category varies a lot (from 300 to more than 10,000). It has a validation set of 50,000 images (50 images per category). According to Li et al. (2017) the estimated noise rate lies between 20% and 34%. Similarly to other CNL methods (Bang et al. (2022); Karim et al. (2022); Kim et al. (2021)) we took from WebVision only 10 classes from the ones containing the largest number of images each. We obtain a training set of 34,286 images, which we resize to 84 × 84.
Seq. NTU-60 It comprises 60 action classes with 56,880 video samples, including 3D skeletal data (25 body joints per frame), all captured simultaneously using three Kinect V2 cameras. We here split the dataset into 6 tasks of 10 classes each.

Notice that since some labels are incorrect, real class distribution for each task might vary. Details on the noisy labels injected on Seq. CIFAR-10/100, Seq. NTU-60 and the classes used for Seq. WebVision are released with the code.

DETAILS ON THE EXPERIMENTAL SETTINGS

Architecture We use ResNet (He et al. (2016)) family as a backbone for all the methods involved in our evaluation. ResNet18 is used for CIFAR-10/100 and ResNet34 is used for WebVision, as in (Bang et al. (2022)). All the experiments do not feature pretraining.

Augmentation We apply random crops and horizontal flips to both stream and buffer examples, for each dataset at hand. For the implementations of PuriDivER, we use AutoAugment (Cubuk et al. (2019)) as in the original paper (Bang et al. (2022)).

Training We deliberately hold batch size out of the hyperparameter space and keep it fixed to 32 for both stream and buffer examples. For each task, we train for 50 and 30 epochs for CIFAR-10/100 and WebVision, respectively.

Table C: Final Average Accuracy (FAA) [↑] and standard error across all tasks of different CNL methods on Seq. NTU-60 with different noise rates. † Additional baselines created by adapting existing loss-based and CL approaches to the multi-epoch scenario.

Benchmark	**Seq. NTU-60**	
noise rate	20	40
Joint	68.26± 0.69	63.02± 0.88
SGD	14.30± 0.51	12.48± 1.07
Reservoir	35.16± 2.16	16.21± 0.27
+ *CoTeaching*	44.43± 0.78	32.03± 1.86
+ *DivideMix*	40.92± 0.97	32.07± 1.73
GDumb	11.34± 0.21	7.34± 0.68
+ *CoTeaching*	13.81± 2.04	9.18± 0.51
+ *DivideMix*	15.96± 1.16	6.59± 1.11
PuriDivER	39.33± 1.59	38.86± 0.79
PuriDivER.ME†	43.10± 1.11	38.07± 1.06
DividERMix†	32.61± 1.69	20.23± 0.63
OURs	46.69± 1.08	44.56± 1.04
w. buffer fit.	**49.59**± 1.34	**48.18**± 0.63
w. consolidation	48.73± 1.20	45.19± 0.05

Buffer consolidation with MixMatch At the end of each task, we finetune the model on the buffer examples only, for 255 epochs. During this stage, we use SGD with Warm Restart (SGDR) through Cosine Annealing and a batch size of 64. For the purpose of label co-refinement, we set the number of different augmentations η of Eq. 6 to perform on the samples in the *uncertain* set to 3.

ADDITIONAL RESULTS

We repeat each of the experiments five times. We report in Tab. A, Tab. B, and Tab. C the Final Average Accuracy for all the experiments as in Tab. 1 and Tab. 2 of the main paper, here with standard error values.

We also provide the final forgetting measure in Eq. 1 for all methods of the main comparison in Tab. D.

$$FF \triangleq \frac{1}{T-1}\sum_{j=0}^{T-2} f_j, \text{s.t.} f_j = \max_{t\in 0,\dots,T-2} a_j^t - a_j^{T-1} \tag{1}$$

where a_j^t is the accuracy of the model on the j^{th} task after training on t tasks. These additional results depict a **lower** degree of **forgetting** of our proposal w.r.t. the baselines.

When paired with 1 and 2 of the manuscript, such evidence shows higher overall effectiveness in learning from a noisy source of data, allowing more stable convergence on the current task and lower losses due to forgetting.

C COMPUTATIONAL COSTS

We perform all the experiments on a Tesla V100-SXM2-16GB GPU. In Tab. G we report the computational costs of different methods in our setting, in terms of runtime and consumed memory.

Not only our method achieves superior performance, as shown in Tab. 1, but also exhibits a lower overall training time.

D MODEL ABLATION

We analyse the **contribution of each component** of the proposed approach to the final accuracy. To achieve this, we merge the base rehearsal method used in our research, *i.e.* ER-ACE, with our indi-

Table D: Final Forgetting (FF) [↓] of CNL methods on our selection of bencharks. † Additional baselines created by adapting existing loss-based and CL approaches to the multi-epoch scenario.

Benchmark	**CIFAR-10**			**CIFAR-100**					**WebVision**	**NTU-60**	
	symm			*symm*			*asymm*		*instance*	*symm*	
Noise rate	20	40	60	20	40	60	20	40	N/A	20	40
Joint	0	0	0	0	30	0	0	0	0	0	0
Finetune	41.79	37.60	27.41	81.52	71.51	57.96	73.91	54.71	73.00	85.09	73.46
Reservoir	43.31	52.59	47.49	55.88	55.79	44.90	43.48	35.22	33.75	54.53	61.33
+ *CoTeaching*	40.93	54.75	48.09	55.20	54.95	36.07	54.77	26.03	28.75	34.30	29.03
+ *DivideMix*	12.09	14.12	23.39	22.33	26.73	20.94	23.45	16.73	18.92	18.45	18.70
PuriDivER	44.15	43.53	36.74	20.52	18.21	14.77	22.51	17.26	45.12	41.29	34.25
PuriDivER.ME†	42.36	44.75	41.10	24.34	25.06	26.83	25.40	21.82	22.42	25.76	18.41
DividERMix†	38.67	48.11	59.76	24.93	22.76	20.14	25.93	18.67	14.25	6.89	21.83
OURs	15.20	16.41	20.97	22.89	21.26	22.13	21.19	16.90	18.20	12.94	14.05
w. buffer fit.	18.88	20.28	23.52	22.22	20.76	22.12	22.83	18.79	21.55	26.35	14.25
w. consolidation	11.81	15.50	15.62	19.03	11.67	12.02	20.15	9.28	13.25	8.54	0.29

vidual contributions, one by one, and conduct training on Seq. CIFAR-100 with a 60% symmetric noise setting. As can be seen from the results in Tab. E, each additional feature produces an increase in performance.

In a similar and more comprehensive manner, for an in-depth analysis of the effects of the utilized loss function, namely the asymmetric cross-entropy (ACE) loss, we conducted various experiments using plain cross-entropy (CE), *i.e.* **OURs - no ACE** in Tab. A and Tab. B. The results indicate that the contribution of the asymmetric loss is significant, aligning with both Caccia et al. (2022) and our initial expectations.

Table E: Ablation study for AER and ABS; FAA [↑] on Seq. CIFAR-100; 60 % symmetric noise.

Benchmark	**Seq. CIFAR-100**			
ER-ACE	α	AER	ABS	**FAA**
✓				11.65
✓	✓			19.97
✓	✓	✓		24.19
✓	✓	✓	✓	**26.34**
✓	✓		✓	21.68

Table F: Final Average Accuracy of PuriDivER trained on CIFAR-10 with 40% noise rate

Benchmark		**PuriDivER**		
	noise rate	*20%*	*40%*	*60%*
Online	data augmentation	51.09	48.13	38.81
	no data augmentation	56.74	50.49	43.55
Offline	data augmentation	55.49	52.54	46.40

To explore the potential contribution of **PuriDivER's consolidation** strategy to our approach's effectiveness, we integrate it and present the outcomes in Tab. A and Tab. B.

The results obtained are in line with the ones obtained with our consolidation strategy, thus supporting the notion that with our technique the buffer is sufficiently clean and diverse.

E COMPARISON OF PURIDIVER ONLINE VS OFFLINE

PuriDivER in its original form comprises a task setup of online (single epoch) continual learning from a blurry data stream. While changing this blurry setting to our ClassIL one is a minor change, it is more difficult to tailor its online setup to our offline one without hurting its performances, as outlined in Sec. 4.4. In Tab. F we report results for the online ClassIL setting both with and without data augmentation. For the offline setting (50 epochs), we always perform data augmentation for fairness of comparison with all the other evaluated methods.

Table G: Comparison of Computational Cost [↓] of different methods when trained on CIFAR-10 with 40% noise.

Computational Cost	**PuriDivER**	**PuriDivER.ME**	**OURs**
Total Time (*hours*)	5h15m	2h50m	**1h30m**
Epoch Time (*seconds*)	76.90s	**15.69s**	18.56s
Task Time (*minutes*)	73m50s	19m39s	**16m33s**
Memory Used (*GB*)	7.77	7.50	**6.75**

As can be seen from Tab. F, we successfully adapt the method to our multi-epoch and ClassIL incremental setting improving performance in most cases.

F DETAILS ON SPR AND CNLL

In the main paper we provide a comparison between our proposal and SPR and CNLL. Nevertheless, as these methods were originally designed for the single-epoch setting, we had to design specific adjustments to make them viable for our scenario.

SPR initially stores samples in a *delayed buffer* – then splits into clean and noisy sets, with the former stored in a separate long-term buffer – and then optimizes the model for approximately 7,000 training iterations through a self-supervised (SSL) objective. This implies that SPR involves approximately $448\times$ iterations than standard training[1], making it unfeasible for our scenario due to time constraints. Indeed, while on CIFAR-10 our method takes around 16 minutes to complete 1 task (Tab. G), SPR would require over 119 hours. We thus opt to distribute the training iterations of SPR across 25 epochs (see Tab. 4). Finally, as SPR employs two distinct memory buffers, we set the buffer size to 1000 for a fair comparison.

CNLL uses variable-length buffers to store confident clean and noisy samples, which implies a CL setting with unrestricted memory across tasks. To ensure fairness in comparison, we adhere to the well-established memory-budgeted CL (Chaudhry et al. (2019); van de Ven et al. (2022); Buzzega et al. (2020)) setting. Thus, for CNLL we allocate a total memory budget of 2,500 exemplars across all 5 buffers specified by the original method.

As we move from the single to multi-epoch setting, we find a reduced effectiveness of the regularization of CNLL; such a result is in line with our hypothesis of Sec. 3.2: as more epochs are allowed to learn the current task, sample selection based on the small-loss criterion fails to distinguish clean and noisy samples. Moreover, we find that such an outcome is maintained even in an unrestricted setting, where the memory budget is not a concern.

Finally, the performance gap w.r.t. our proposal is even more pronounced for SPR‡, where our method attains significantly higher accuracy in considerably less time; indeed, our reduced version of SPR requires around $109\times$ more time than our proposal, in line with our estimation.

G HYPERPARAMETERS

We choose to use different buffer sizes relying on the dataset length. For experiments conducted on CIFAR-10 and CIFAR-100, the buffer size is set to 500 and 2000, respectively. For experiments on WebVision, the buffer size is 1000 as in (Bang et al. (2022)). Finally, we set the buffer size to 500 for experiments on NTU.

We select the other hyperparameters by performing a grid search and using the Final Average Accuracy (FAA) as the selection criterion for the best parameters. Here, we report the best values for each model, categorized by dataset and noise type.

---

[1] assuming a buffer size of 500, batch size of 32, and 10,000 samples

CIFAR-10

Noise type: *sym* – 20%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.1; λ_u: 0.01
- **DividERMix**: lr: 0.1
- **ER**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.1
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.01
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.03

Noise type: *sym* – 40%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.1; λ_u: 0.01
- **DividERMix**: lr: 0.1
- **ER**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.03
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.03
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.03

Noise type: *sym* – 60%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.1; λ_u: 0.01
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.1
- **ER + CoTeaching**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.03
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.03
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.01

CIFAR-100

Noise type: *sym* – 20%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.05; λ_u: 0.01
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.01
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.05

Noise type: *sym* – 40%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.1; λ_u: 0.1
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.05

Noise type: *sym* – 60%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.1; λ_u: 0.1
- **DividERMix**: lr: 0.1
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.05

Noise type: *asym* – 20%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.05; λ_u: 0.005
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.01
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.1

Noise type: *asym* – 40%

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.1; $lr_{\text{consolidation}}$: 0.05; λ_u: 0.1
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.01
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.1

WEBVISION

Noise type: *unknown*

- **Joint**: lr: 0.03
- **SGD**: lr: 0.03
- **OURs**: lr: 0.03
- **OURs + buffer fit**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.03; $lr_{\text{consolidation}}$: 0.01; λ_u: 0.05
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.1
- **ER + CoTeaching**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.1
- **ER + DivideMix**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.001; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.1
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.05
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.05

NTU RGB-D

Noise type: *sym* – 20%

- **Joint**: lr: 0.1
- **SGD**: lr: 0.1
- **OURs**: lr: 0.1
- **OURs + buffer fit**: lr: 0.3; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.1; λ_r: 0.01
- **DividERMix**: lr: 0.03
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.3; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.03
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.1
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.3

Noise type: *sym* – 40%

- **Joint**: lr: 0.1
- **SGD**: lr: 0.03
- **OURs**: lr: 0.1
- **OURs + buffer fit**: lr: 0.3; $lr_{\text{buffer fit.}}$: 0.05
- **OURs + consolidation**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.1; λ_r: 0.01
- **DividERMix**: lr: 0.1
- **ER**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05
- **ER + CoTeaching**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **ER + DivideMix**: lr: 0.1; $lr_{\text{buffer fit.}}$: 0.05
- **PuriDivER**: lr: 0.3; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **PuriDivER.ME**: lr: 0.03; $lr_{\text{buffer fit.}}$: 0.05; α: 0.1
- **GDumb**: $lr_{\text{buffer fit.}}$: 0.1
- **GDumb + CoTeaching**: $lr_{\text{buffer fit.}}$: 0.1
- **GDumb + DivideMix**: $lr_{\text{buffer fit.}}$: 0.03

REFERENCES

Jihwan Bang, Hyunseo Koh, Seulki Park, Hwanjun Song, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on a contaminated data stream with blurry task boundaries. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022.

Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In *Advances in Neural Information Processing Systems*, 2020.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New Insights on Reducing Abrupt Representation Change in Online Continual Learning. In *International Conference on Learning Representations Workshop*, 2022.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. In *International Conference on Machine Learning Workshop*, 2019.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 113–123, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016.

Nazmul Karim, Umar Khalid, Ashkan Esmaeili, and Nazanin Rahnavard. Cnll: A semi-supervised approach for continual noisy label learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022.

Chris Dongjoo Kim, Jinseo Jeong, Sangwoo Moon, and Gunhee Kim. Continual learning on noisy data streams via self-purified replay. In *IEEE International Conference on Computer Vision*, 2021.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, Jesse Berent, Abhinav Gupta, Rahul Sukthankar, and Luc Van Gool. Webvision challenge: Visual learning and understanding with web data. *arXiv preprint arXiv:1705.05640*, 2017.

Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016.

Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 2022.