

## APPENDIX

**Supplementary Metrics:** In addition to the metrics used in the main paper, we also report backwards Backward Transfer (BWT) (Lopez-Paz & Ranzato, 2017) and Forgetting (FTG) (Lee et al., 2019). BWT is a measurement of increase in performance on task  $n$  after training across all tasks  $1 \dots N$ . A higher value is better, indicating that the learner is better at performing task  $n$  after learning the subsequent tasks. A negative value indicates a drop in performance, which is typically expected in class incremental learning. A weakness of this metric is that it measures performance relative to *local* tasks and does not reflect performance on the *global* task of class incremental learning (i.e. the softmax outputs are across only the local per-task categories, not across all of the categories encountered throughout training). FGT is a measurement of decrease in performance on task  $n$  with respect to the *global* task; it is essentially negative backward transfer adopted for class incremental learning. A lower value is better, indicating that the learner has experienced less average performance decrease on task  $n$  throughout training. A weakness of this metric is that it does not account for natural decrease in performance due to the increasingly more difficult global task characteristic in class incremental learning. A key difference between BWT and FGT is that when evaluating task  $n$  performance for BWT, only task  $n$  classes can be returned during inference, whereas for FGT, all tasks classes  $1 \dots n$  can be returned. We include both of these metrics for experiment results during all subsequent sections because while neither is regularly used for class incremental learning, they may be useful to the reader.

$$BWT = \frac{1}{N-1} \sum_{n=1}^{N-1} (A_{N,n} - A_{n,n}) \quad (10)$$

$$FGT = \frac{1}{N-1} \sum_{i=2}^N \sum_{n=1}^{i-1} \frac{|\mathcal{T}_n|}{|\mathcal{T}_{1:i}|} (R_{n,n} - R_{i,n}) \quad (11)$$

where:

$$R_{i,n} = \frac{1}{|\mathcal{D}_n^{test}|} \sum_{(x,y) \in \mathcal{D}_n^{test}} \mathbb{I}(\hat{y}(x, \theta_{i,1:n}) = y) \quad (12)$$

## A DISTILLMATCH ABLATION STUDY

Here, we ablate our method in two experiment scenarios: RandomClass Tasks with Uniform Unlabeled Data Distribution (Table 3a) and ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution (Table 3b).  $\Omega$  curves for both Tables are given in Figure 4. In the former case, we find that the hard distillation loss (7) is the most significant contribution, but the semi-supervised consistency loss (4), class balancing (3), and soft distillation loss (1) add significant performance gains as well. In the later case, we actually find the semi-supervised consistency loss (4) and distillation loss (1) to be the most important, while class balancing (3) and hard distillation loss (7) perform very similarly. This reflects the strength of our method: DM performs well in all of our experiments because it has components which vary in importance depending on the scenario (i.e. coreset size and object-object correlations).

Table 3: Results (%) for Selected Ablation Studies on CIFAR-100 with 20% Labeled Data. Results are reported as an average of 3 runs with mean and standard deviation. Each row represents a part of our method which is removed as part of the study.

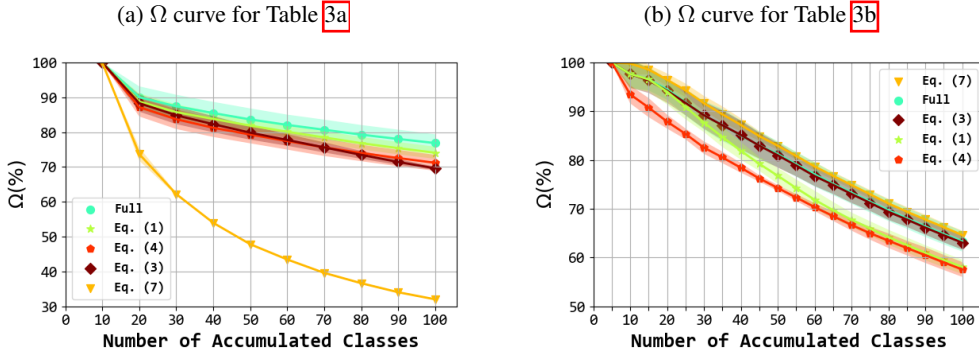
(a) RandomClass Tasks with Uniform Unlabeled Data Distribution, 10 Tasks, no Coreset

Ablation	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
$\ell_{pl}$ - eq. (7)	$7.7 \pm 0.5$	$32.0 \pm 0.2$	$-5.8 \pm 1.9$	$56.6 \pm 1.9$
$w(k)$ - eq. (3)	$30.2 \pm 1.9$	$69.6 \pm 0.5$	$-4.8 \pm 0.2$	$10.5 \pm 0.5$
$\ell_{ul}$ - eq. (4)	$33.3 \pm 0.9$	$71.2 \pm 2.3$	$-0.7 \pm 0.3$	$7.7 \pm 0.2$
$\ell_{dst}$ - eq. (1)	$35.2 \pm 1.1$	$74.1 \pm 1.7$	$-4.8 \pm 0.4$	$8.0 \pm 0.9$
Full Method	$37.5 \pm 0.7$	$76.9 \pm 2.5$	$-1.0 \pm 1.0$	$6.5 \pm 0.5$

(b) ParentClass Tasks with PositiveSuperclass Unlabeled Distribution, 20 Tasks, 400 image coreset

Ablation	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
$\ell_{pl}$ - eq. (7)	$19.3 \pm 1.1$	$64.6 \pm 0.9$	$-17.9 \pm 0.3$	$28.8 \pm 1.0$
$w(k)$ - eq. (3)	$19.4 \pm 0.6$	$63.1 \pm 1.4$	$-17.4 \pm 0.4$	$27.2 \pm 0.7$
$\ell_{ul}$ - eq. (4)	$17.1 \pm 0.7$	$57.6 \pm 1.5$	$-14.0 \pm 0.1$	$21.8 \pm 0.6$
$\ell_{dst}$ - eq. (1)	$17.7 \pm 0.8$	$58.1 \pm 1.5$	$-15.9 \pm 0.9$	$22.7 \pm 1.0$
Full Method	$19.7 \pm 0.8$	$63.3 \pm 2.1$	$-18.2 \pm 0.7$	$24.9 \pm 0.6$

Figure 4:  $\Omega$  curves showing task number  $t$  on the x-axis and  $A_{t,1:t}$  on the y-axis.



## B ADDITIONAL EXPERIMENT DETAILS

We used a batch size of 64 for labeled training data and 128 for unlabeled training data. As done in (Lee et al., 2019), we train over 200 epochs per task with a tuned learning rate decaying by 0.1 after 120, 160, and 180 epochs. When a coreset is present, we include finetuning of the final layer in our model using only the coreset and class balancing, as introduced in GD (Lee et al., 2019). If finetuning, the model is trained over the first 180 epochs in the same manner, but after 180 epochs the learning rate is reset to 10% of the initial learning rate and is trained for 20 additional epochs with decays by 0.1 after 10, 15 epochs. We use stochastic gradient descent with 0.9 momentum and 0.0005 L2 weight decay.

As also done in (Lee et al., 2019), we hold  $\lambda_{dst}$  to a constant value, 1, and include a small temperature scaling, 2, for the softmax activations used in eq. (1). All results are averaged over 3 repeats and generated with a common deep learning architecture (WRN-28-2) (He et al., 2016). Results were generated using a combination of Titan X and 2080 Ti GPUs. Although we did not record specific run-times here as they are machine specific, we find our method to have a similar run-time to GD.

## C HYPERPARAMETER SWEEPS

We tuned hyperparameters using a grid search. We did this for two scenarios: (i) RandomClass Tasks with Uniform Unlabeled Data Distribution and (ii) ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution. The former is applied for all experimental scenarios which do not include a coreset, and the latter is applied for all scenarios which do include a coreset. We chose this division as we found the coreset size to greatly affect the other hyperparameters. DR and E2E use hyperparameters chosen for GD (as done in (Lee et al., 2019)), while Base uses hyperparameters from DM. The hyperparameters were tuned using k-fold cross validation with three folds of the training data on only half of the tasks. We do not tune hyperparameters on the full task set because tuning hyperparameters with hold out data from all tasks may violate the principal of continual learning that states each task is visited only once (van de Ven & Tolia, 2019). The results reported outside of this section are on the CIFAR-100 testing split (defined in the dataset).

Table 4: Hyperparameters, chosen with grid search

Coreset		Yes		No	
Hyperparameter	Range	DM	GD	DM	GD
Learning Rate	5e-3, 1e-2, 5e-2, 1e-1, 5e-1	1e-1	1e-1	1e-1	5e-3
Weight FixMatch Loss	0.1, 0.5, 1, 5	1.0	-	1.0	-
TPR	0.01, 0.05, 0.1, 0.2, 0.5, 0.8, 0.95	0.05	-	0.5	-
$\epsilon$ (Fix Match)	0.7, 0.85, 0.9, 0.95	0.9	-	0.9	-

## D FULL RESULTS

We provide additional detail to the results from Section 5.3 by reporting (i) the original results with additional metrics and standard deviations (Tables 5 and 6) and (ii)  $\Omega$  curves for each experiment in Figure 5.

Table 5: Full results (%) on CIFAR-100 with 20% Labeled Data. Results are reported as an average of 3 runs with standard deviation. The results from these tables do not include a coreset (and use the same set of hyperparameters, as described in Appendix C)

(a) RandomClass Tasks with Uniform Unlabeled Data Distribution, 5 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$15.6 \pm 0.9$	$52.5 \pm 2.5$	$-25.7 \pm 26.2$	$43.8 \pm 2.3$
E2E	$12.5 \pm 0.9$	$46.1 \pm 0.9$	$1.4 \pm 0.6$	$42.5 \pm 1.2$
DR	$16.0 \pm 0.9$	$53.7 \pm 0.7$	$0.3 \pm 0.7$	$41.6 \pm 1.5$
GD	$32.1 \pm 0.2$	$69.9 \pm 0.9$	$0.5 \pm 0.8$	$5.0 \pm 0.3$
DM	$44.8 \pm 1.4$	$84.4 \pm 3.0$	$2.5 \pm 0.1$	$1.2 \pm 0.1$

(b) RandomClass Tasks with Uniform Unlabeled Data Distribution, 10 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$8.2 \pm 0.1$	$34.7 \pm 0.8$	$-32.2 \pm 24.6$	$56.2 \pm 2.0$
E2E	$7.5 \pm 0.5$	$32.3 \pm 0.6$	$-0.5 \pm 0.4$	$56.0 \pm 1.8$
DR	$8.3 \pm 0.3$	$36.4 \pm 0.2$	$-1.9 \pm 0.3$	$57.4 \pm 1.3$
GD	$21.4 \pm 0.6$	$60.0 \pm 1.9$	$-14.6 \pm 0.1$	$18.4 \pm 1.5$
DM	$37.5 \pm 0.7$	$76.9 \pm 2.5$	$-1.0 \pm 1.0$	$6.5 \pm 0.5$

(c) RandomClass Tasks with Uniform Unlabeled Data Distribution, 20 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$4.3 \pm 0.4$	$22.0 \pm 0.8$	$-41.6 \pm 13.8$	$69.4 \pm 0.5$
E2E	$4.0 \pm 0.3$	$21.1 \pm 0.6$	$-4.1 \pm 0.8$	$67.7 \pm 1.4$
DR	$4.3 \pm 0.4$	$22.4 \pm 0.7$	$-7.1 \pm 0.2$	$70.6 \pm 1.2$
GD	$13.4 \pm 1.9$	$42.7 \pm 1.1$	$-29.2 \pm 3.5$	$37.4 \pm 0.8$
DM	$21.1 \pm 1.0$	$60.8 \pm 0.8$	$-8.8 \pm 0.7$	$17.3 \pm 1.7$

(d) ParentClass Tasks with Uniform Unlabeled Data Distribution, 20 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$3.5 \pm 0.1$	$18.5 \pm 0.5$	$-33.5 \pm 6.0$	$54.3 \pm 0.8$
E2E	$3.2 \pm 0.2$	$18.1 \pm 0.6$	$-14.6 \pm 3.5$	$53.0 \pm 0.1$
DR	$3.7 \pm 0.1$	$19.4 \pm 0.6$	$-17.6 \pm 1.3$	$56.6 \pm 0.1$
GD	$10.5 \pm 0.2$	$37.4 \pm 1.8$	$-25.1 \pm 0.1$	$29.1 \pm 0.8$
DM	$20.8 \pm 0.8$	$57.8 \pm 1.4$	$-10.8 \pm 0.8$	$14.8 \pm 0.3$

Table 6: Full results (%) on CIFAR-100 with 20% Labeled Data. Results are reported as an average of 3 runs with standard deviation. The results from these tables are with a 400 image coreset (and use the same set of hyperparameters, as described in Appendix C)

(a) ParentClass Tasks with Uniform Unlabeled Data Distribution, 20 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$14.6 \pm 1.4$	$53.4 \pm 2.4$	$-14.7 \pm 6.4$	$29.8 \pm 0.6$
E2E	$19.5 \pm 0.9$	$59.3 \pm 1.7$	$-14.5 \pm 0.2$	$23.1 \pm 0.5$
DR	$20.1 \pm 0.8$	$57.8 \pm 1.5$	$-15.2 \pm 0.4$	$31.9 \pm 3.3$
GD	$21.4 \pm 0.9$	$57.7 \pm 1.8$	$-12.5 \pm 0.4$	$8.0 \pm 1.7$
DM	$24.4 \pm 0.4$	$67.5 \pm 1.3$	$-15.1 \pm 1.3$	$21.9 \pm 1.5$

(b) ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution, 20 Tasks

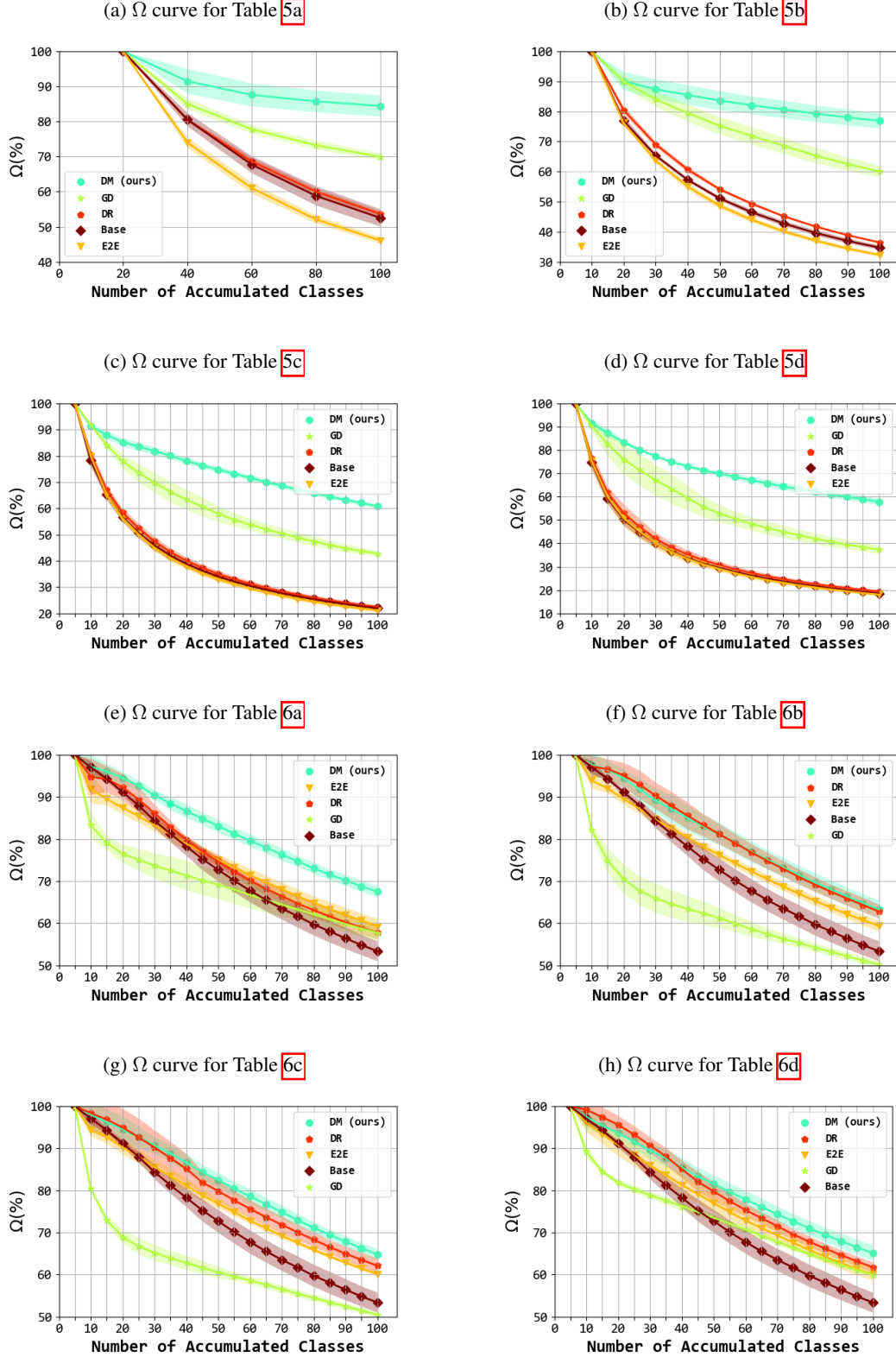
Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$14.6 \pm 1.4$	$53.4 \pm 2.4$	$-14.7 \pm 6.4$	$29.8 \pm 0.6$
E2E	$18.9 \pm 1.2$	$59.4 \pm 1.3$	$-16.6 \pm 1.0$	$22.2 \pm 0.3$
DR	$18.8 \pm 1.0$	$62.8 \pm 1.7$	$-17.6 \pm 0.7$	$27.5 \pm 0.3$
GD	$17.9 \pm 0.8$	$50.2 \pm 0.8$	$-10.6 \pm 0.8$	$-2.1 \pm 2.0$
DM	$19.7 \pm 0.8$	$63.3 \pm 2.1$	$-18.2 \pm 0.7$	$24.9 \pm 0.6$

(c) ParentClass Tasks with NegativeSuperclass Unlabeled Data Distribution, 20 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$14.6 \pm 1.4$	$53.4 \pm 2.4$	$-14.7 \pm 6.4$	$29.8 \pm 0.6$
E2E	$19.9 \pm 1.2$	$60.1 \pm 0.5$	$-16.1 \pm 1.0$	$22.5 \pm 0.4$
DR	$20.1 \pm 1.9$	$62.1 \pm 1.8$	$-16.8 \pm 0.2$	$28.7 \pm 1.0$
GD	$18.1 \pm 0.6$	$50.5 \pm 0.7$	$-10.9 \pm 1.2$	$-1.7 \pm 1.6$
DM	$20.7 \pm 1.5$	$64.8 \pm 1.3$	$-17.4 \pm 0.7$	$24.7 \pm 1.3$

(d) ParentClass Tasks with Random Unlabeled Data Distribution, 20 Tasks

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
Base	$14.6 \pm 1.4$	$53.4 \pm 2.4$	$-14.7 \pm 6.4$	$29.8 \pm 0.6$
E2E	$19.8 \pm 0.5$	$60.0 \pm 1.5$	$-15.1 \pm 0.3$	$23.7 \pm 0.6$
DR	$19.9 \pm 1.7$	$61.8 \pm 1.2$	$-15.7 \pm 0.6$	$29.9 \pm 1.6$
GD	$21.3 \pm 0.5$	$59.9 \pm 0.5$	$-13.7 \pm 0.2$	$8.3 \pm 2.7$
DM	$22.4 \pm 1.3$	$65.1 \pm 1.8$	$-16.1 \pm 0.3$	$23.3 \pm 0.9$

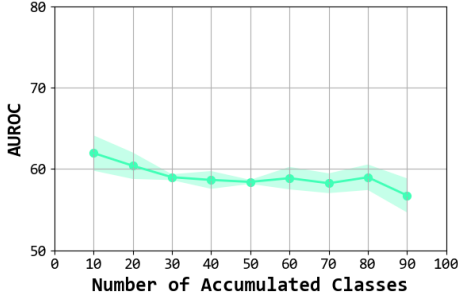
Figure 5:  $\Omega$  curves showing task number  $t$  on the x-axis and  $\Omega$  up to task  $t$  on the y-axis

## E PERFORMANCE OF OOD DETECTION

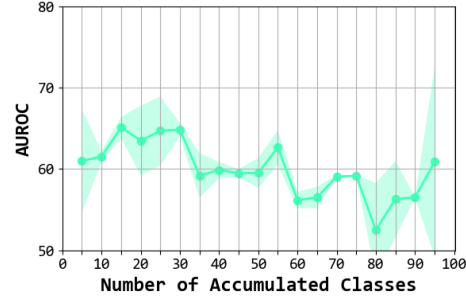
We show AUROC (a metric for OoD detection) over time for DM in both RandomClass Tasks with Uniform Unlabeled Data Distribution (Figure 6a) and ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution (Figure 6b). A high AUROC means the distributions of the ID data and OoD data are separable. As we can see, AUROC is decreasing over time. In the RandomClass scenario, this is a smooth decline (as expected). In the ParentClass scenario, the decline is not smooth, likely due to the correlations between tasks making the task difficulty highly deviate between runs.

Figure 6: AUROC over time for DM showing task number  $t$  on the x-axis and AUROC on the y-axis

(a) RandomClass Tasks with Uniform Unlabeled Data Distribution



(b) ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution



## F SUPER CLASS AND PARENT CLASS ASSOCIATIONS FOR CIFAR-100

We show the relationship between super classes and parent classes for CIFAR-100 (Figure 7) as defined by (Zhu & Bain, 2017).

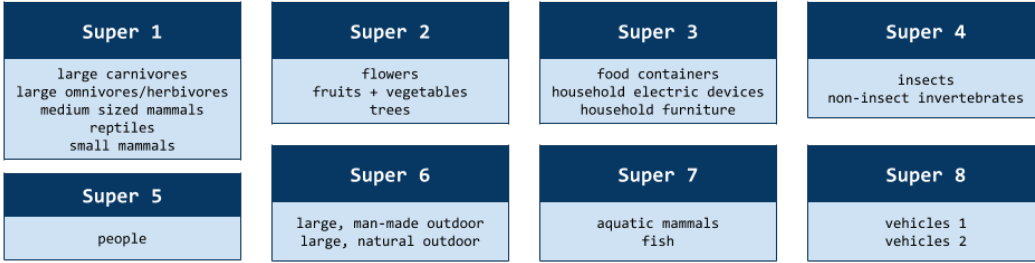


Figure 7: Super-parent class relationships for CIFAR-100

We also visualize example streams for each task sequence in (Figure 8). As a reminder, we use the following terminology to describe the correlations of the tasks (i.e. labeled data): *RandomClass Tasks*, where no correlations exist in task classes, and *ParentClass Tasks*, where tasks are introduced by CIFAR-100 parent classes (i.e. each task is to learn the five classes of a single CIFAR-100 parent class). For the unlabeled data distribution we have: *Uniform Unlabeled*, where all classes are uniformly distributed in unlabeled data for all tasks, *PositiveSuperclass Unlabeled*, where the unlabeled data of each tasks consists of the parent classes in the same super-class as the current task, *NegativeSuperclass Unlabeled*, where the unlabeled data of each tasks consists of parent classes from different super-class as the current task, and *RandomUnlabeled*, where the unlabeled data of each task consists of 20 randomly sampled classes (roughly equal to the average class size in a super-class).

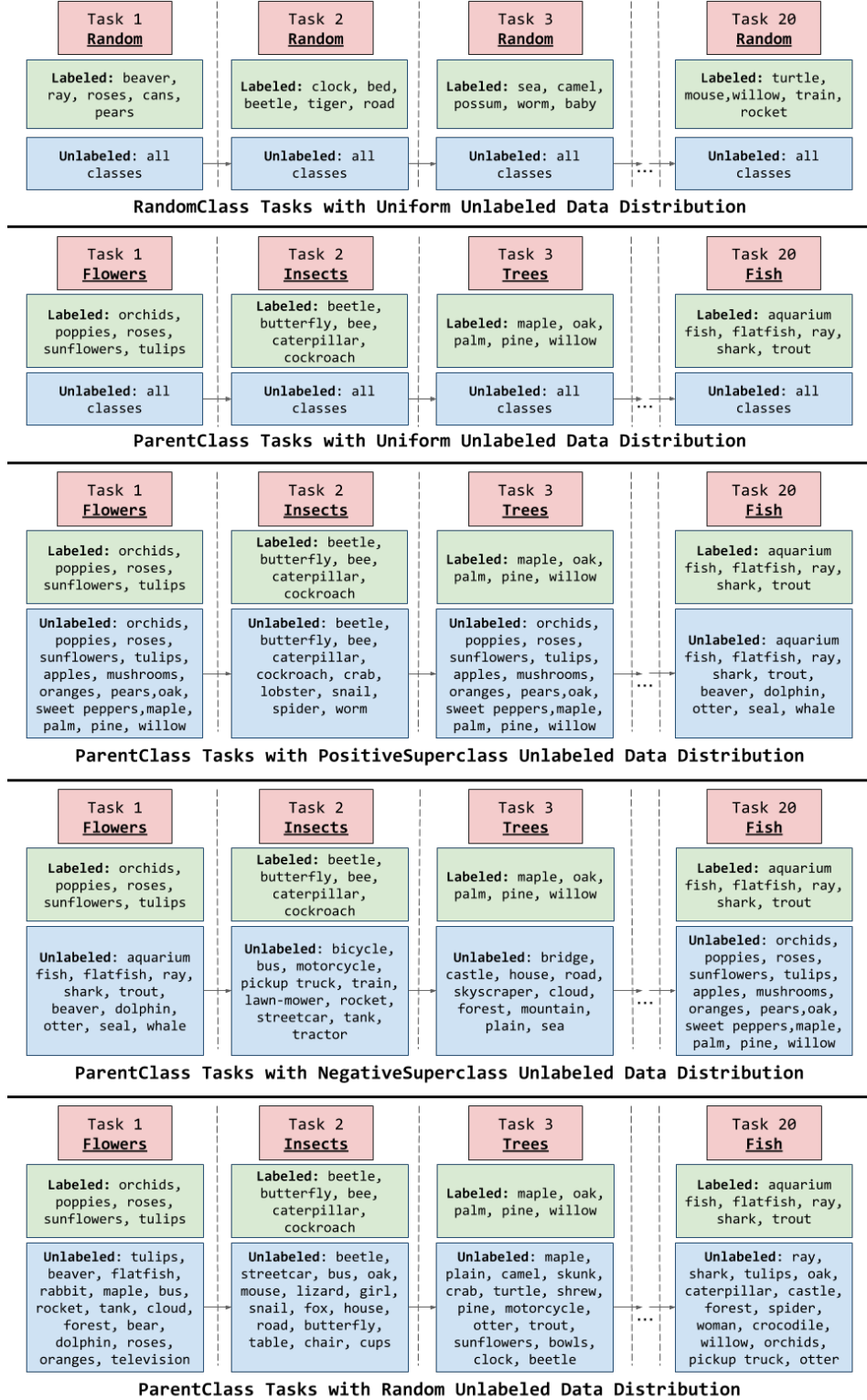


Figure 8: Example streams for each task sequence



## G ADDITIONAL STUDIES

We found that confidence calibration in GD Lee et al. (2019) had mixed effects in our experiments. We ablate this contribution for RandomClass Tasks with Uniform Unlabeled Data Distribution (Table 7a), ParentClass Tasks with PositiveSuperclass Unlabeled Data Distribution (Table 7b), and ParentClass Tasks with Random Unlabeled Data Distribution (Table 7c). We contribute this finding to the assumption made in GD that the unlabeled data does not contain data from the current task (which is heavily violated in some of our experiments). Even though removing this mechanism can boost GD performance for some of the experiments (Tables 7a and 7b) and makes it worse for others (Table 7c), it is still significantly below our method (DM) in each case.

Table 7: Results (%) for GD Confidence Calibration Ablation on CIFAR-100 with 20% Labeled Data. Results are reported as an average of 3 runs with mean and standard deviation.

(a) RandomClass Tasks with Uniform Unlabeled Data Distribution, 10 Tasks, no Coreset

Confidence Calibration	$A_N$	$\Omega$	BWT	FGT
✓	$21.4 \pm 0.6$ $23.7 \pm 1.2$	$60.0 \pm 1.9$ $67.0 \pm 3.1$	$-14.6 \pm 0.1$ $-5.5 \pm 1.8$	$18.4 \pm 1.5$ $20.3 \pm 2.0$

(b) ParentClass Tasks with PositiveSuperclass Unlabeled Distribution, 20 Tasks, 400 image coreset

Confidence Calibration	$A_N$	$\Omega$	BWT	FGT
✓	$17.9 \pm 0.8$ $19.5 \pm 0.4$	$50.2 \pm 0.8$ $54.4 \pm 3.8$	$-10.6 \pm 0.8$ $-12.6 \pm 1.0$	$-2.1 \pm 2.0$ $7.2 \pm 3.5$

(c) ParentClass Tasks with Random Unlabeled Distribution, 20 Tasks, 400 image coreset

Confidence Calibration	$A_N$	$\Omega$	BWT	FGT
✓	$21.3 \pm 0.5$ $18.1 \pm 0.9$	$59.9 \pm 0.5$ $54.1 \pm 0.7$	$-13.7 \pm 0.2$ $-12.0 \pm 1.2$	$8.3 \pm 2.7$ $20.3 \pm 2.8$

## H ADDITIONAL BACKGROUND AND RELATED WORK

**Continual Learning Approaches:** Approaches to mitigate catastrophic forgetting in continual learning can be broadly organized into three types: *rehearsal*, *architectural*, and *regularization* (Parisi et al., 2019). Rehearsal methods include storage to "replay" data or experiences from previous tasks to mitigate catastrophic forgetting (Aljundi et al., 2019a,b; Chaudhry et al., 2019a,b; Gepperth & Karaoguz, 2017; Hayes et al., 2019; Kemker et al., 2018; Lopez-Paz & Ranzato, 2017; Rebuffi et al., 2017; Robins, 1995; Rolnick et al., 2019; von Oswald et al., 2019). Rather than storing raw data, some methods train a generative model (Kamra et al., 2017; Kemker & Kanan, 2018; Shin et al., 2017) or replay compressed data representations in a late layer (Liu et al., 2020). Architectural approaches typically avoid overwriting the current model by expanding the model parameters to make room for knowledge related to novel tasks (Ebrahimi et al., 2020; Lee et al., 2020; Lomonaco & Maltoni, 2017; Maltoni & Lomonaco, 2019; Rusu et al., 2016). Finally, regularization approaches focus on penalizing changes to parameters important to past tasks. Approaches include regularization penalties (Aljundi et al., 2018; Ebrahimi et al., 2019; Kirkpatrick et al., 2017; Titsias et al., 2019; Zenke et al., 2017), meta learning (Javed & White, 2019), model compression (Beaulieu et al., 2020; He et al., 2019; Saha et al., 2020), or knowledge distillation (Castro et al., 2018; Hou et al., 2018; Lee et al., 2019; Li & Hoiem, 2017).

**Semi-Supervised Learning:** Semi-supervised learning leverages plentiful available unlabeled data to boost model performance when given a (typically small) amount of labeled data. Semi-supervised learning is popular because labeling large datasets is an expensive process. A simple yet popular technique is to provide pseudo-labels (Lee, 2013) for *confident* unlabeled data based on the current model's predictions and to treat this pair (the unlabeled data and pseudo-label) as if it were a labeled data pair. Many following methods build on this idea of using predictions on the unlabeled data to boost performance. For example, mean teachers (Tarvainen & Valpola, 2017) involve averaging model weights for a temporal ensembling approach which encourages consistent label predictions over time. Virtual Adversarial Training (VAT) smooths the decision boundary around each unlabeled data point to be robust against adversarial perturbations. More recent methods include MixMatch (Berthelot et al., 2019), which involves using low-entropy labels and strong data augmentations for a Mix-Up loss, and FixMatch (Sohn et al., 2020), which enforces consistent labeling between weakly and strongly augmented versions of unlabeled data. Other approaches for leveraging unlabeled data is to use it for an auxiliary loss such as generative loss (Kingma et al., 2014; Springenberg, 2015) or self-supervised learning (Jing & Tian, 2020). The reader is referred to (Oliver et al., 2018) for a recent survey of popular techniques and evaluations.

## I DETAILED COMPARISON OF METHODS

In Table 8 we visualize the high level differences of each method, including how the unlabeled data is used by each method. Note that E2E, DR, and GD all use both a coreset (if available) and unlabeled data from the environment, so the comparison is fair.

Table 8: Comparison of distillation methods. **L** refers to *labeled* data from the current task, **C** refers to labeled *coreset* data from past tasks (if available), and **U** refers to *unlabeled* data from the environment

Component	Base	E2E	DR	GD	DM (ours)
Classification Loss	L/C	L/C	L/C	L/C	L/C
Per-Task Distillation over Previous Tasks	-	C/U	-	-	C/U
Single-Task Distillation over Previous Tasks	-	-	C/U	C/U	-
Distill Current Task from Separate Trained Model	-	-	L/U	L/U	-
Soft Global Distillation (All-Tasks)	-	-	-	U	-
Hard Global Distillation (All-Tasks)	-	-	-	-	U
Consistency Loss (All-Tasks)	-	-	-	-	U
Confidence Calibration	-	-	-	C/U	-
OoD Detection	-	-	-	-	U
Fine-Tuning	-	L/C	-	L/C	L/C
Class-Balancing	-	L/C	-	L/C	L/C/U

## J ADDITIONAL EXPERIMENT FOR SCALIBILITY

We report results for the Tiny-ImageNet dataset (Le & Yang, 2015), which contains 200 classes of 64x64 resolution images with 500 training images per class, in Table 9 and Figure 9. We experimented in a similar setting to Table 1a (20% labeled data with RandomClass Tasks, no Coreset, Uniform Unlabeled Data Distribution) with a ten-task sequence (20 classes per task). In this experiment, there are 20,000 labeled images and 80,000 unlabeled images; thus, we both double the number of data and double the image resolution. We find that the conclusions from Table 1a scale to this experiment.

Table 9: Full results (%) on Tiny-ImageNet with 20% Labeled Data for RandomClass Tasks with Uniform Unlabeled Data Distribution (10 Tasks, no Coreset). Results are reported as an average of 3 runs with standard deviation. The results from this table use the set of hyperparameters described in Appendix C

Metric	$A_N$ ( $\uparrow$ )	$\Omega$ ( $\uparrow$ )	BWT ( $\uparrow$ )	FGT ( $\downarrow$ )
UB	$40.7 \pm 0.3$	$100.0 \pm 0.0$	$3.8 \pm 0.5$	$5.2 \pm 0.5$
Base	$6.5 \pm 0.6$	$35.1 \pm 1.5$	$-10.4 \pm 2.4$	$45.1 \pm 2.9$
E2E	$5.8 \pm 0.6$	$30.3 \pm 1.9$	$0.9 \pm 0.6$	$39.3 \pm 3.1$
DR	$6.8 \pm 0.4$	$35.3 \pm 1.1$	$-1.7 \pm 0.7$	$45.0 \pm 2.7$
GD	$11.9 \pm 1.3$	$50.6 \pm 2.9$	$-17.4 \pm 2.6$	$12.5 \pm 1.3$
DM	$24.8 \pm 0.7$	$74.7 \pm 1.6$	$-5.9 \pm 0.4$	$7.6 \pm 0.1$

Figure 9:  $\Omega$  curve for Table 9 showing task number  $t$  on the x-axis and  $\Omega$  up to task  $t$  on the y-axis

