A. Motivation: Privacy for Restricting Memory

Ref.	Dataset	Ordering	Memory	Cost	Iters	Cost
[9]	CIFAR10	Cls Inc	1-25MB	0.05¢	250K-375K	20$
[44]	CIFAR100	Cls Inc	10 MB	0.02¢	50K	8$
[19, 25]					125K	15$
[9]	TinyImageNet	Cls Inc	5-20 MB	0.04¢	350K-500K	25$
[19, 25]	ImageNet100	Cls Inc	0.3-1 GB	2¢	100K	50$
[19, 25]	ImageNet1K	Cls Inc	33GB	66¢	1M	500$
[29]	CLEAR	Dist Shift	0.4-1.2GB	2¢	300K	100$
[24]	ResNet50 (bs=256)		22GB			
Ours	GLDv2-CL	Dist Shift	**90GB**	2$	2K	10$
	ImageNet21K-CL	ClsInc, DataInc	**400GB**	10$	8K	35$

Table 3. **Cost of Memory vs Computation.** Google Cloud Standard Storage (2¢per GB per month for 1 month) and Compute Cost measured as running cost of an A2 instance (3$ per hour for 1 GPU). Number of iterations (forward+backward passes) for training a CL model on that dataset listed for comparison invariant to input image and model sizes. We observe that computational costs for running an experiment far outweigh the costs for storing replay samples.

Prior art on continual learning [5, 9, 14, 29, 32, 44] motivate the problem from the aspect of prohibited access to previously received data; except for a small portion that is allowed to store in memory. The two principal motivations behind restricting the access to past samples in the literature are two folds. (**i**) Storage space is expensive. (**ii**) Access to previous data is prohibitive due to privacy and GDPR constraints.

As for the first argument used as a motivation for limiting the memory size, as we have elaborated in Section 1 of the main paper and similarly further detail in Table 3, the cost of storing data is insignificant. This is particularly the case when considering the associated computational costs of training deep models. To that end, the argument of to restricting the memory size is a an enough justifiable reason. For example, as per Table 3, it costs 2 cents to store the entirety of the CLEAR dataset, among the largest datasets for continual learning, while it costs about 100$ to train a model continually on the same dataset. If reducing costs are the key issue, limited computational budget and not memory, as argued earlier, is the way forward.

As for privacy considerations, this is too a not well-motivated reason behind limiting memory. A classical argument is that due to GDPR requirements, data needs to be removed or company privacy policies, it can no longer accessible after x number of months. First, any previous benchmark with memory constraints already violates this privacy consideration. This is since, which data is to be made private and shall be deleted should not be up to the learning algorithm to decide. To that end, restricting memory samples does not help solving the privacy considerations. Even if the samples stored in memory were selected such that they do not violate any privacy constraints, which none of the prior art address, it remains a question on whether the trained models preserve any sensitive information after training on private data. Without imposing such restriction on the learning algorithm and the underlying model, restricting the memory for the argument of privacy does not meet its stated objectives. This is particularly the case, as Haim *et al*. [23] have found that models retain a lot of information of the training samples. This is to an extent that large number of training samples can be reconstructed only given the trained model. Goel *et al*. [21] presents a catastrophic forgetting baseline, indicating forgetting might be the very objective of sustaining privacy which is antithetical to the objective of continual learning.

In conclusion, restricting access to previous samples by restricting memory do not contribute to solving the privacy problem. Instead, we consider the more realistic setting where only limited amount of computation is given due to cost restraint or the need to predict every sample in a high throughput stream. This in any how imposes an implicit restraint on access of past memory samples.

B. Dataset Construction

B.1. Constructing Imagenet2K

ImageNet2K train set is constructed using all training images in ImageNet1K dataset [18] for 1K classes as an initialization, with selecting an additional 1K non-overlapping classes from ImageNet21K dataset [18] to form the ImageNet2K dataset. We illustrate the creation of the test, validation and train sets below:

Test Set: We use the ImageNet1K val set as the test set to be consistent with test sets used in previous literature using ImageNet1K. We separate 50 images per class from the sample set of the new 1K classes. We combine these two sets to create the overall test set for experiments. The test set for every timestep consists of classes from this test set which have been seen so far.

Validation Set: We use ImagenetV2 dataset [45] as the validation set from Imagenet1K data. We seperate 50 images per class, not used in the test set to create the val set for the new 1K classes. We combine these two sets to create the overall

validation set for experiments. The validation set for every timestep consists of classes from this validation set which have been seen so far.

Train Set: We order all the samples from the new 1K classes not used for creating the test and val sets for training. We order them by classes to form the CI-ImageNet2K stream and randomly shuffle all these images to form the DI-ImageNet2K stream. Note that the stream order is provided samplewise, allowing the stream size N to be adjusted. In the standard experiments, data equivalent to 50 classes is sampled every timestep, for 20 timesteps.

B.2. Constructing Continual Google Landmarks V2

Continual Google Landmarks V2 (CGLM) consists of 580K samples. To obtain this subset, we start with the train-clean subset of the Google Landmarks V2 available from the Google Landmarks V2 dataset website[2]. We apply the following preprocessing steps in order:

1. Filter out images which do not have timestamp metadata available.
2. Remove images of classes that have less than 25 samples in total
3. Order data by timestamp.
4. Randomly sample 10% of data from across time as the test set

We get the rest $580K$ images as the train set for continual learning over 10788 classe, with rapid temporal distribution shifts. We do not have a validation set here as we benchmark transfer of hyperparameters used from ImageNet to this dataset.

C. Estimation of Equivalent Iterations

In this section, we elaborate on the details of selecting the computational budget for distillation and expensive sampling approaches. The key point for these calculations is the fact that the computational (and time) cost for a forward pass is $1/2$ the cost of a backward pass [3].

Distillation: When distillation approaches have a budget of $2/3^{rd}$ iterations of the naive baseline, the computational cost is as follows: $2/3\mathcal{C}$ for training of the student model, and $1/2 \times 2/3 = 1/3\mathcal{C}$ for the teacher model which only has a forward pass, which sums up to $\mathcal{C}$. Hence, distillation methods have an equivalent computational budget as the naive baseline with $2/3^{rd}$ training iterations.

Sampling: We train the expensive models for $1/2$ the number of iterations as a naive model. To select that subset of training data, we randomly sample $3\times$ the required number of training samples from the stored set and forward pass them through the latest trained model to obtain the features/probabilities. And then we select the best $1/3^{rd}$ of the $3\times$ set for training using different selection functions. We assume the cost of selecting samples given the features/probabilities is negligible.

The computational cost of training for expensive sampling methods is $1/2\mathcal{C}$, as the selected sample set is half the size compared to the naive baseline. The computational cost of selecting the samples is $1/2 \times 3 \times 1/3\mathcal{C} = 1/2\mathcal{C}$ (forward pass requires $1/3^{rd}$ of the total cost, on $3\times$ the required data, the required data size being $1/2$ when compared to naive). The combined cost is the sum of selection and training cost, which is $\mathcal{C}$. Hence, expensive sampling methods have an equivalent budget with $1/2$ training iterations.

D. Additional Results

Due to limited space, some of the experiments in the manuscripts were deferred to the Appendix. In this section we present results for all mentioned costly sampling methods and distillation methods. Additional results for ***Section 4.2: 1 "Do Sampling Strategies Matter?"*** are presented in Figure 11 where all three costly sampling methods are presented, namely Max Loss, Uncertainty Loss, and KMeans. Similarly, additional results for ***Section 4.2: 2 "Does Distillation Matter?"*** are presented in Figure 12 where all four distillation methods are presented, namely BCE, MSE, Cosine, and CrossEntropy. We observe that all previous conclusions consistently hold, *i.e.* the Naive baseline is still leading in comparison to all previous approaches.

We also extend the time steps and the number of iterations sensitivity experiments to all four considered distillation methods in all three setups, DI-ImageNet2K, CI-ImageNet2K and CGLM. We present additional results for ***Section 4.3: 1. Does the Number of Time Steps Matter?***: with results for DI-ImageNet2K presented in Figures 13 and 14 with 50 and 200 time steps respectively, CI-ImageNet2K in Figure 15 and 16 with 50 and 200 time steps respectively and CGLM in Figures 17 and 18 with 50 and 200 time steps respectively. We observe that all previous conclusions consistently hold, *i.e.* the conclusions are robust to changing time steps for a given cost $\mathcal{C}$.

[2] https://github.com/cvdfoundation/google-landmark

[3] https://www.lesswrong.com/posts/jJApGWG95495pYM7C/how-to-measure-flop-s-for-neural-networks-empirically

We present additional results for ***Section 4.3: 2. Does the Compute Budget Matter?***: on DI-ImageNet2K in Figures 19 and 20 for 100 and 1200 iterations respectively, and CI-ImageNet2K in Figures 21 and 22 for 100 and 1200 iterations respectively and on CGLM in Figures 23 and 24 for 40 and 400 iterations respectively. We observe that all previous conclusions consistently hold, *i.e*. the conclusions are robust to changing computational cost, towards both harsher and laxer computational constraint regimes.

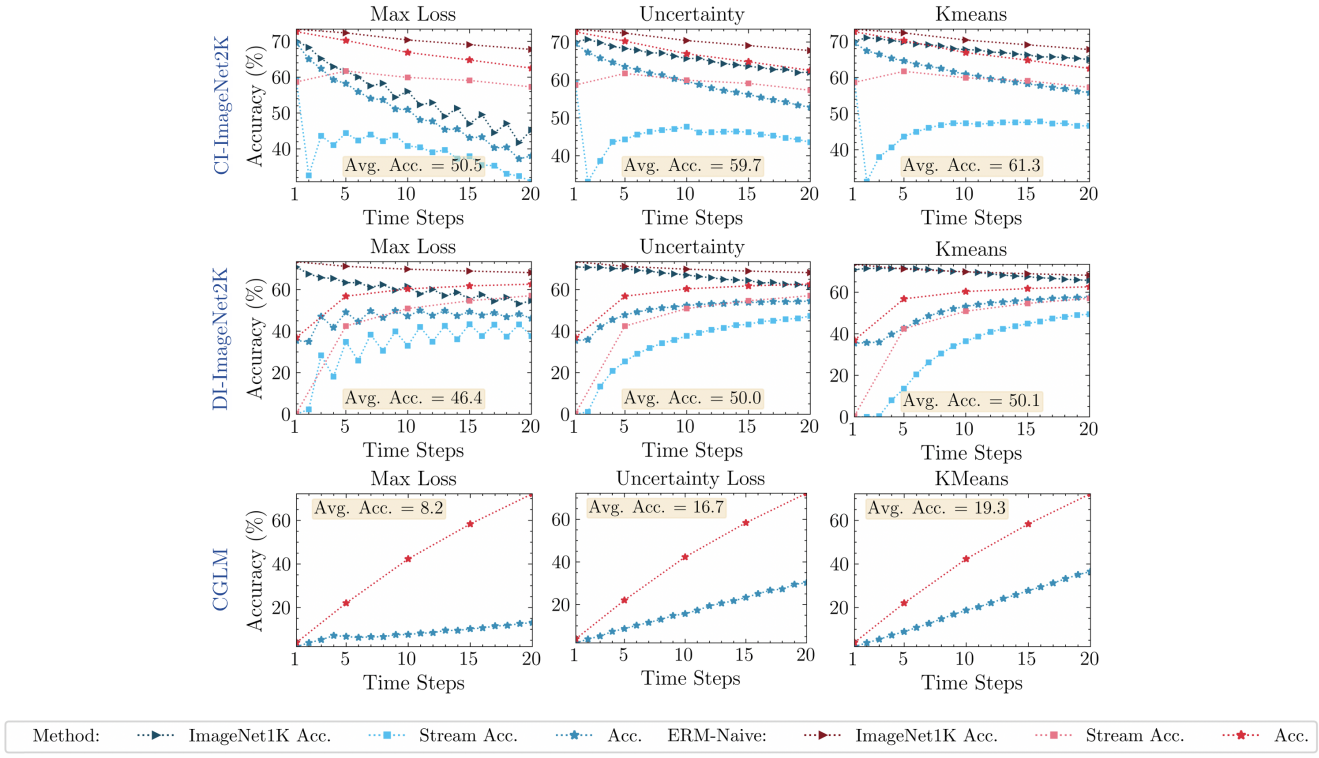

Figure 11. **Expensive Sampling.** As mentioned in the manuscript, KMeans performs the best among expensive sampling techniques such as Max Loss and Uncertainty Loss. Nevertheless, the performance of the expensive sampling methods is worse than simple Naive.

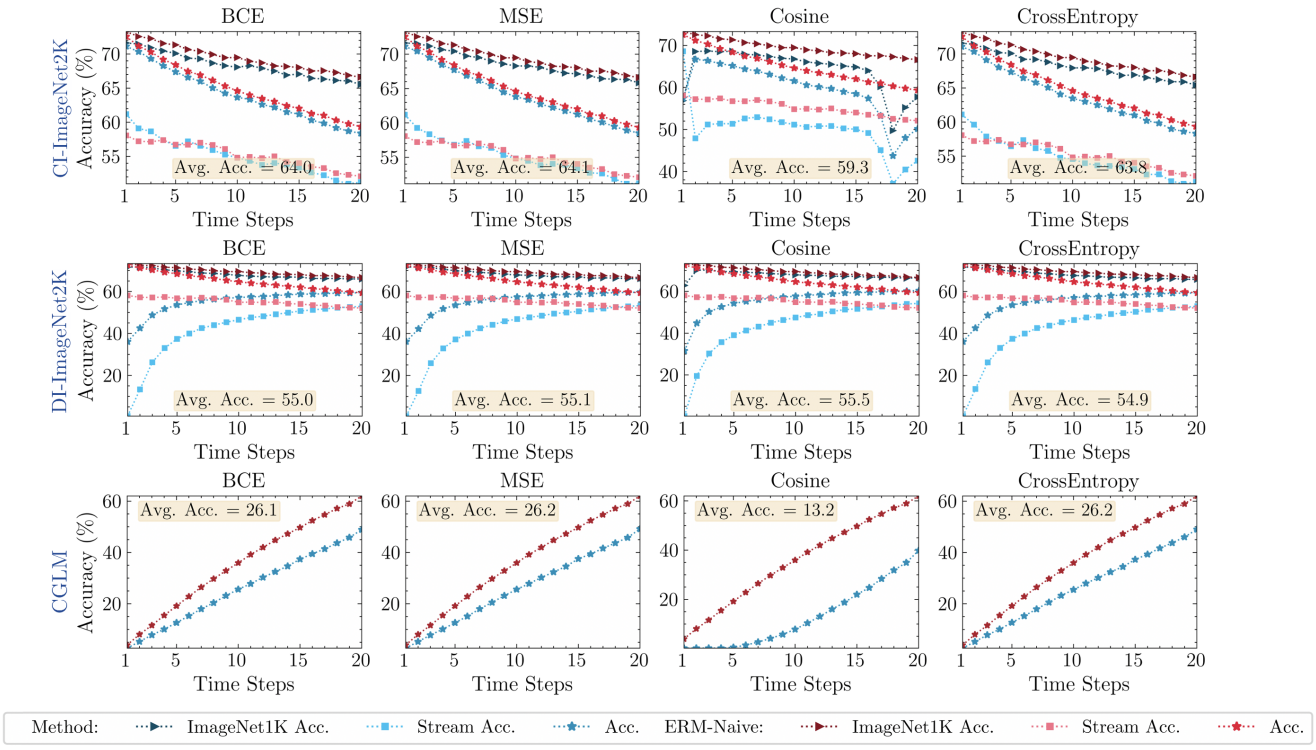

Figure 12. **Distillation Methods.** All four studied distillation methods under perform compared to the simple Naive baseline in all three studied settings. ImageNet experiments are allowed 400 iterations whereas CGLM is allowed 100 iterations.

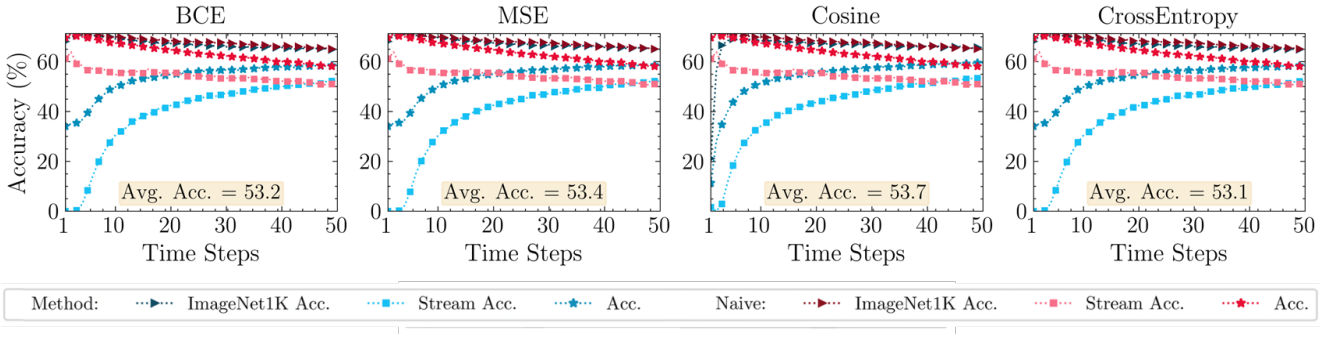

Figure 13. **DI-ImageNet2K 50 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

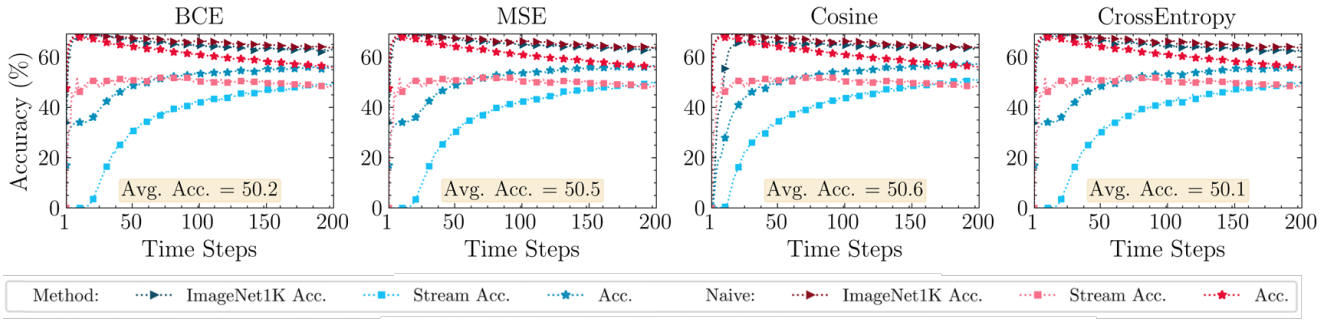

Figure 14. **DI-ImageNet2K 200 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

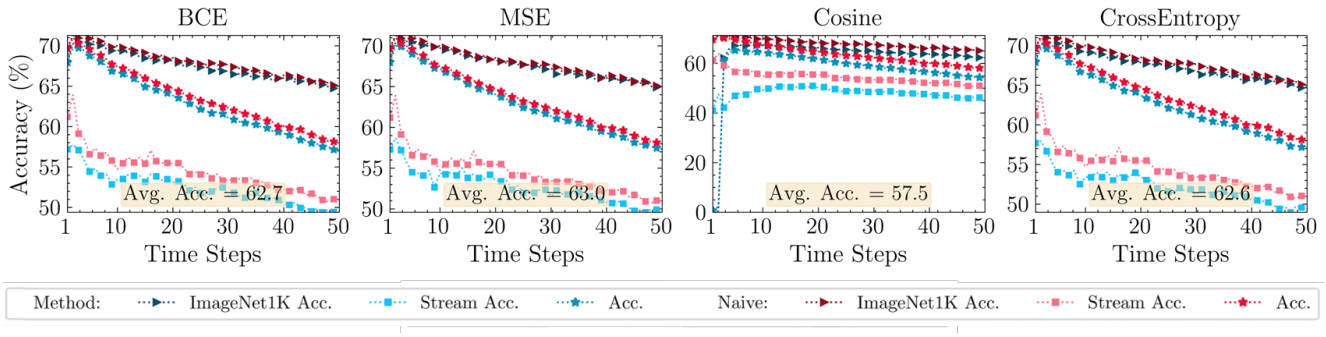

Figure 15. **CI-ImageNet2K 50 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

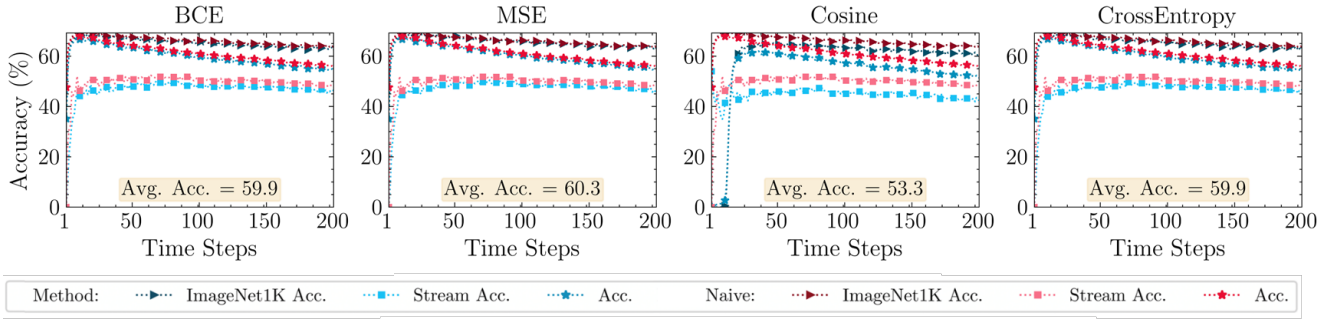

Figure 16. **CI-ImageNet2K 200 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

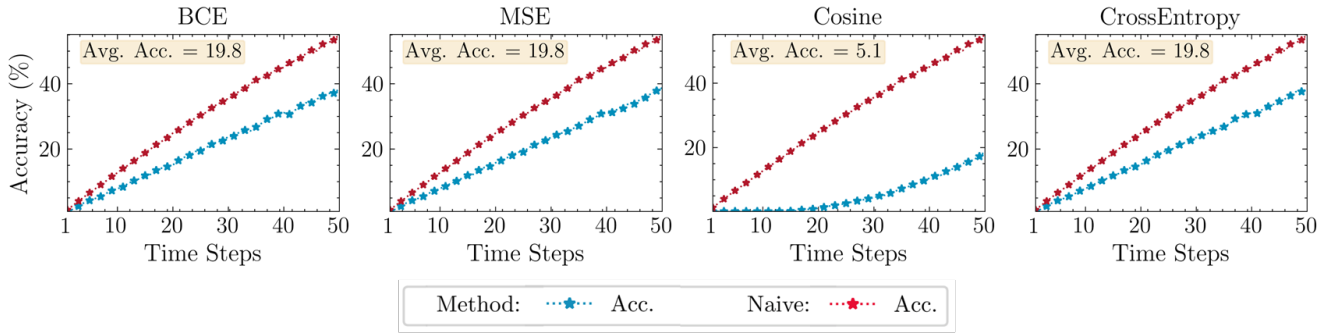

Figure 17. **CLGM 50 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

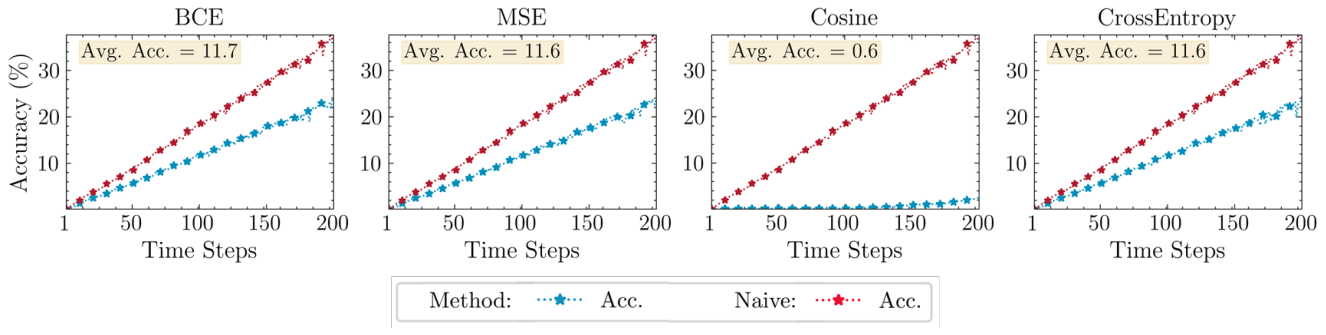

Figure 18. **CGLM 200 Time Steps.** As observed in the manuscript, when the number of time steps, distillation methods still under perform compared to the Naive baseline.

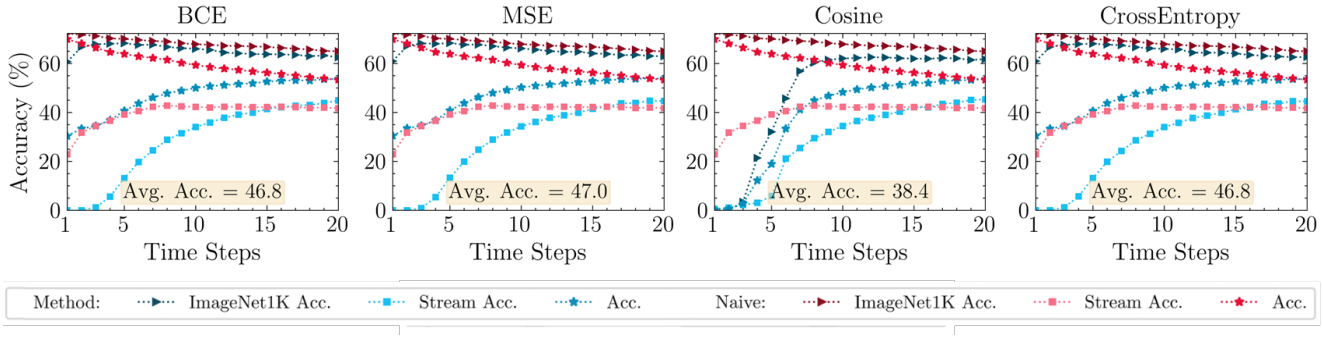

Figure 19. **DI-ImageNet2K - 100 Iterations.** As observed in the manuscript, with reduced compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 100 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 67$ iterations.

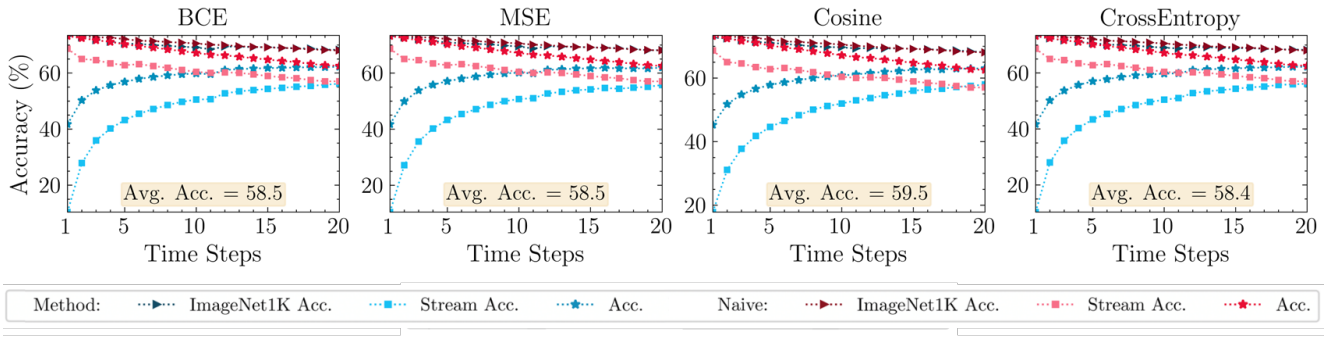

Figure 20. **DI-ImageNet2K - 1200 Iterations.** As observed in the manuscript, with increased compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 1200 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 800$ iterations.

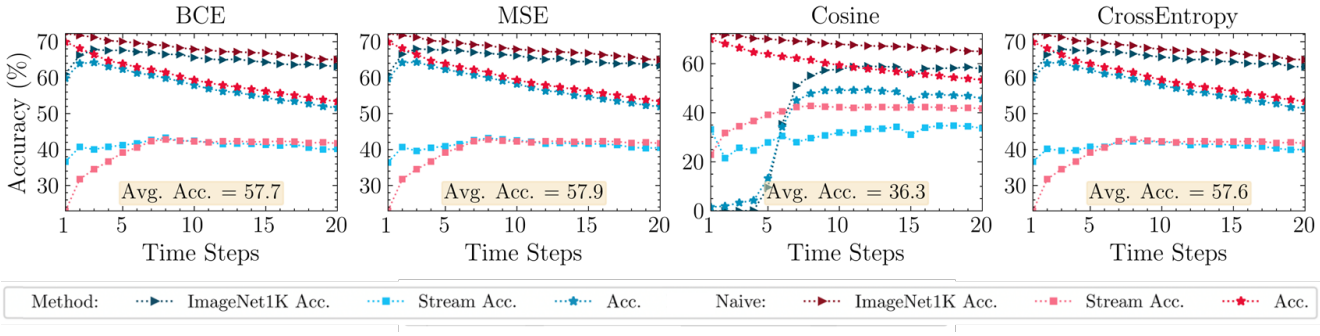

Figure 21. **CI-ImageNet2K - 100 Iterations.** As observed in the manuscript, with reduced compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 100 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 67$ iterations.

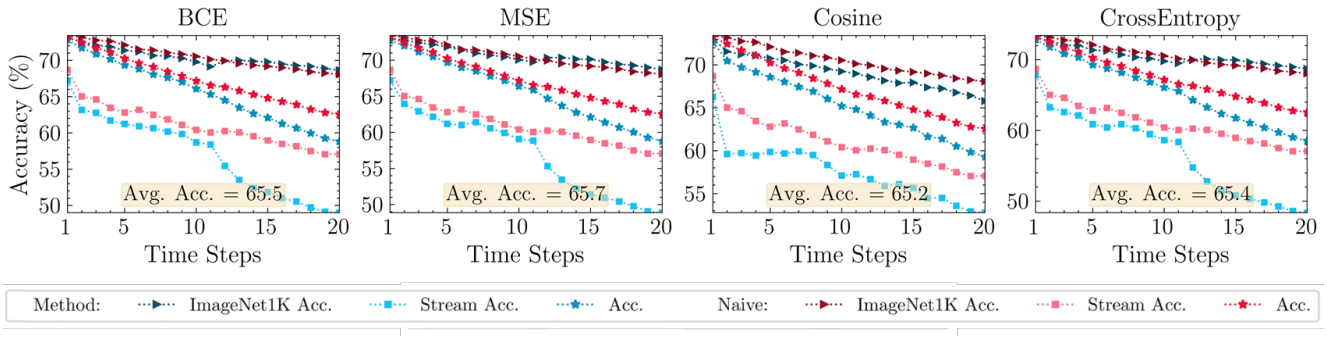

Figure 22. **CI-ImageNet2K - 1200 Iterations.** As observed in the manuscript, with increased compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 1200 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 800$ iterations.

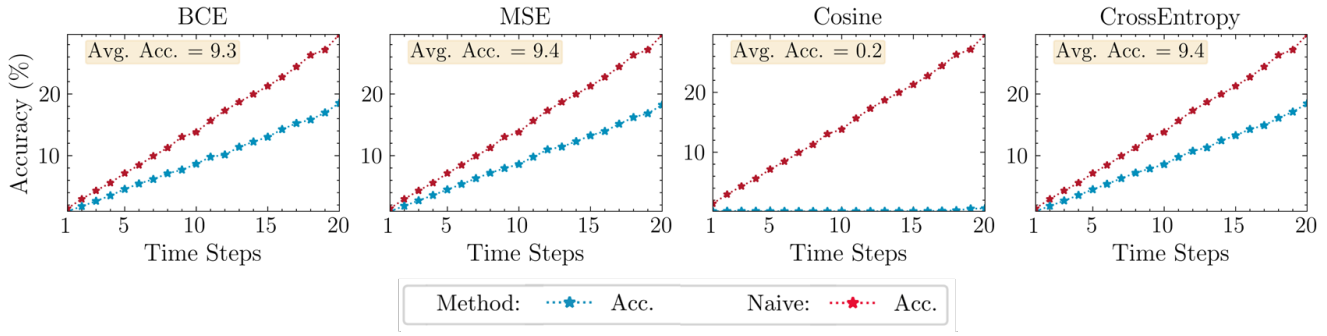

Figure 23. **CGLM 40 - Iterations.** As observed in the manuscript, with reduced compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 40 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 27$ iterations.

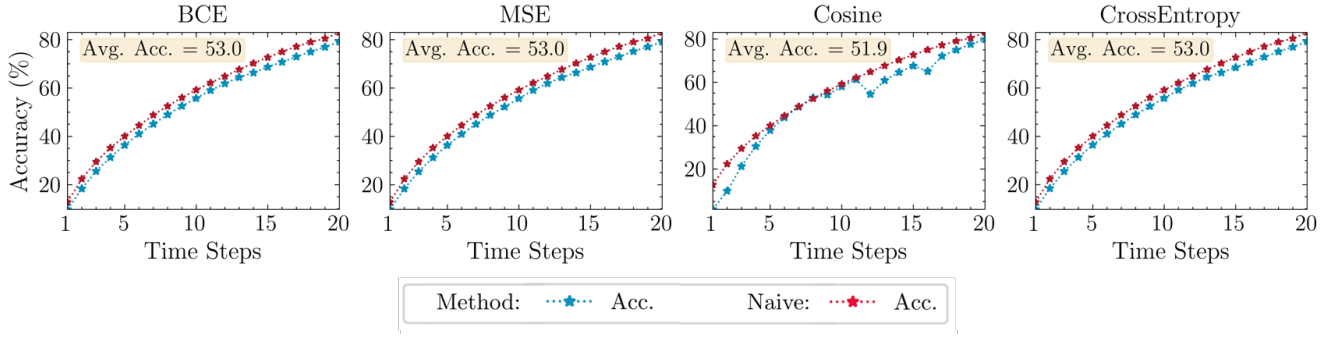

Figure 24. **CGLM 400 - Iterations.** As observed in the manuscript, with increased compute, distillation methods still under perform compared to the Naive baseline. The compute budget of the Naive baseline, $\mathcal{C}$, is set to 400 iterations whereas that of the distillation methods is $^2/_3\,\mathcal{C} = 267$ iterations.

D.1. Effect of Weight Decay

The choice of weight decay, wd $= 0$, in the manuscript was based on result of cross-validation from the set. More specifically, we try weight decays of $\{5 \times 10^{-5}, 1 \times 10^{-4}\}$. We observe a minor difference in performance between various weight decays, with a wd=0 consistently being slightly better. The results are shown in Figure 25

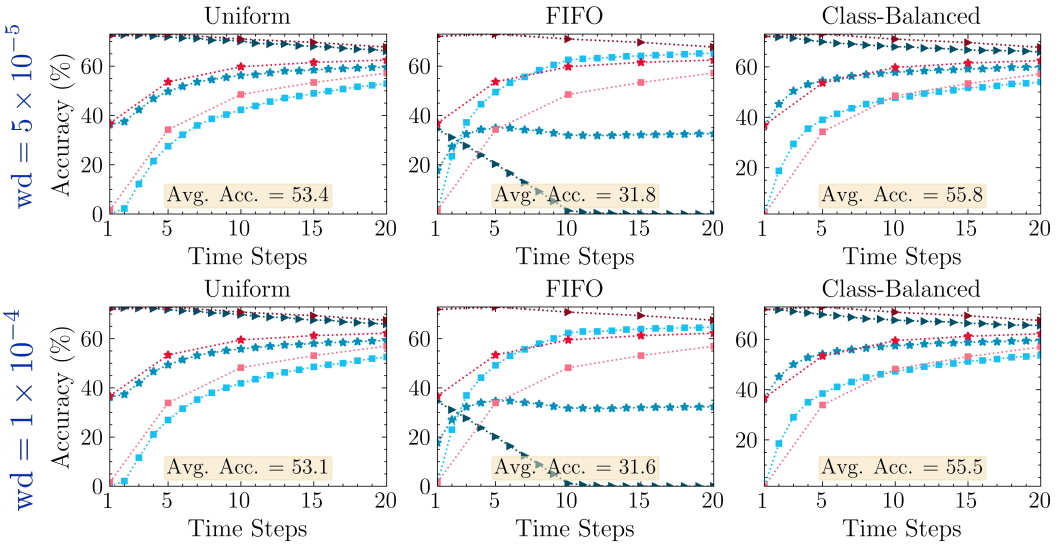

Figure 25. **Effect of Weight Decay.** Increasing the weight decay causes a slight drop in performance. Setting wd=0 gave the best results during our parameter cross-validation.

D.2. Effect of Batch Size

In all experiments, we fixed the batch size (BS) to 1500 to optimize the utilization of our hardware resources and minimize the training time. As shown in the literature, BS and learning rate (LR) are closely related. The selected LR was tuned to fit the selected BS. Regardless, we present varying BS experiments in Fig (1) where we study the latest FC correction method, ACE, and the distillation method, MSE, for BS of 250 and 500 with increased iterations of 600 and 300, respectively, and crossvalidated learning rates. We observe that the Naive baseline is still superior even when the batch size is adjusted. Our findings are summarized in Figure 26.

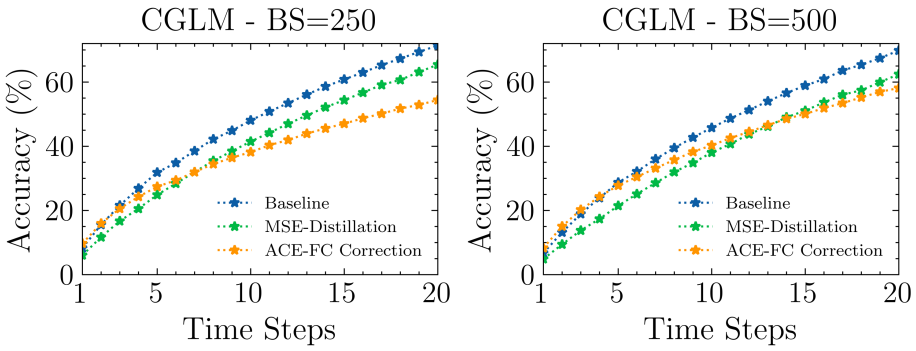

Figure 26. **Effect of Batch Size.** The conclusions presented in our work hold even when the batch size while is changed while the same overall computational budget.

D.3. Effect of Increasing Computational Budget on Distillation

We complement the results in Figure 9 with additional experiments using 800 and 1200 iterations. The observed results align with our previous observations; as long as the computation is normalized across methods, naive, the most simplest among

the considered methods, outperforms existing methods. This is as opposed to prior art comparison that does not normalize compute, which puts the Naive baseline in disadvantage. The results are shown in Figure 27.

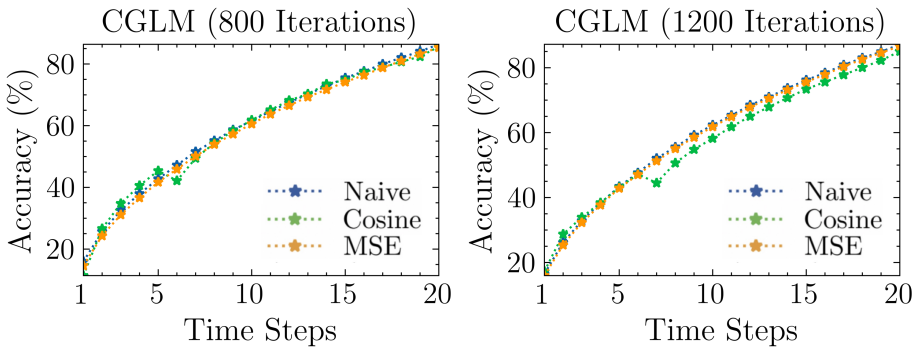

Figure 27. **Effect of Increasing Computation Budget on Distillation.** As the computation budget is increased while maintaining a normalized compute among different methods, Naive baseline still outperforms distillation based methods.