On the Pitfalls of Instance-Based Dynamic Curricula on Image Classification Benchmarks

Alexandre Galashov
UCL Gatsby
Google DeepMind
London, United Kingdom
galashov.alexandr@gmail.com

Amal Rannen-Triki
Google DeepMind
London, United Kingdom
arannen@google.com

Yee Whye Teh
Google DeepMind
London, United Kingdom
ywteh@google.com

Razvan Pascanu
Google DeepMind
London, United Kingdom
razp@google.com

András György
Google DeepMind
London, United Kingdom
agyorgy@google.com

Abstract

Active learning has provable benefits in learning machine learning models. As such, automatically building an instance-based curriculum to speed up neural network training has been an active area of research. Recent works such as those of Jiang et al. [2019], Zhou et al. [2020], Wang et al. [2024b], Yuan et al. [2025] have shown benefits of using dynamic curricula on standard image classification tasks, where training data is selected adaptively online, based on how the training goes. On the other hand, Wu et al. [2021] demonstrated that static data ordering, designed independently of the actual training run, does not help in learning classifiers for some standard image classification benchmarks (such as CIFAR10 and CIFAR100). In this paper we extend this result to the case of dynamic data selection, still focusing on image classification, showing that, at least on standard benchmarks like CIFAR10, CIFAR100, and ImageNet, the performance of some state-of-the-art instance-based dynamic curriculum selection methods can be traced back to changes in the effective learning-rate schedule due to the subsampling of the data, rather than the careful selection of the exact data points. This highlights the importance of properly controlling all variables in designing experiments and perhaps suggests that standard image classification tasks are not the best benchmarks for studying curriculum learning.

1 Introduction

State-of-art results in machine learning often rely on scale across multiple domains [Bommasani et al., 2021, Zhao et al., 2023, Zhou et al., 2023, Gemini Team et al., 2023], which poses significant computational challenges, limiting their accessibility and even having an environmental impact [e.g., Rillig et al., 2023]. Given the trend towards increasing scale, a significant question for the community has been how one can reduce the computational cost of training these models. Extensive work in this space includes techniques for efficient inference and training, relying for instance on sparsity [Thangarasa et al., 2023, Liu et al., 2022], quantization [Bai et al., 2022, Xiao et al., 2023], and conditional computation [Iuzzolino et al., 2021]. Mixture of experts [Artetxe et al., 2021, Douillard et al., 2024, Jiang et al., 2024], distillation [Liang et al., 2020, Section 3.2],[Gemini Team, 2024] or compression [Zhu et al., 2023, Wang et al., 2024a] methods provide additional tools for reducing model size.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

One particular thread in this rich body of literature focuses on active learning [Balcan et al., 2009], that is, identifying and prioritizing relevant data points. One argument for the resurgence of these ideas is the recent empirical observation that large proportions of typical datasets can be dropped without affecting the final performance of the model [e.g., Paul et al., 2021], pointing out that some data points are less informative than others. It stands to reason then that one can improve data and computational efficiency of deep learning by not wasting compute on non-informative data points.

Furthermore, works such as that of Toneva et al. [2019] show a natural separation of examples into *easy* and *hard* ones, where easy examples can be learned even within the first epoch, while *hard* examples require multiple passes to be classified correctly. There is a rich literature on algorithms trying to exploit this observation in different settings, from standard image classification [e.g., Shalev-Shwartz, 2017, Jiang et al., 2019, Zhou et al., 2020, Wang et al., 2024b, Yuan et al., 2025] to meta-learning or meta-RL [e.g., Arnold et al., 2021, Open-Ended Learning Team et al., 2021].

While the idea of utilizing easy and hard examples seems appealing, Wu et al. [2021] demonstrated that, for the CIFAR10 and CIFAR100 datasets, using a curriculum based on prior information (i.e., independently of the actual training run), such as the easy/hard classification of examples discussed above, has only marginal benefits, mostly coming from modifying the training set size but not the actual selection of the data points. Their curriculum definition consists of ordering the data based on some difficulty score, and selecting the epoch size according to some predetermined criteria, known as a *pacing function* [Bengio et al., 2009, Hacohen and Weinshall, 2019]. Note that the pacing function can be interpreted as a schedule for subsampling the dataset.

On the other hand, recent works [Jiang et al., 2019, Zhou et al., 2020, Yuan et al., 2025] claim success from using *dynamic* curricula in standard image classification tasks, where the selection of data points (and their order) is determined dynamically, based on the actual state of the training procedure, suggesting that dynamic curriculum design might be more powerful than a static curriculum.

In this paper we show that this is not exactly the case for typical image classification datasets, specifically CIFAR10, CIFAR100, and ImageNet. Extending the results of Wu et al. [2021] to the case of dynamic curriculum, we demonstrate that the effects of state-of-the-art methods can be explained by an adaptive schedule for changing the epoch size on these datasets, and the actual choice of the training points for each epoch does not matter. This suggests that these widely used datasets might not be suitable for understanding or building curriculum learning algorithms. It also highlights an important pitfall of such research, which needs to control for any epoch-dependent quantity to ensure the effect is coming from the selection of specific data points rather than other side effects of the learning process. Finally, based on this observation, we present an algorithm which inherits the speed-ups of dynamic curriculum training methods (with changing epoch size) by simply transforming the learning-rate schedule dynamically based on the average performance of the model.

2 Preliminaries

We consider standard supervised classification, defined over instances from a set $\mathcal{X}$ to a finite label set $\mathcal{Y}$. As usual, we are given a training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, and a test set $\mathcal{D}^{\text{test}}$ of the same form. We consider a class of predictors $f_w : \mathcal{X} \to \mathcal{Y}'$ parametrized by $w \in \mathcal{W} \subset \mathbb{R}^d$, whose performance is measured by a loss function $\ell : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}$. For concreteness, we assume that $\mathcal{Y} = \{0, 1\}^C$, and labels are one-hot representations of C classes, and $\mathcal{Y}' = [0, 1]^C$. During training the aim is to find a parameter w minimizing $\mathcal{L}(w; \mathcal{D}) = \sum_{(x,y)\in\mathcal{D}} \ell(f_w(x), y)$ (where the performance in the end is measured on the test set $\mathcal{D}^{\text{test}}$ via $\mathcal{L}(w, \mathcal{D}^{\text{test}})$). In typical settings, an optimization algorithm iteratively improves w, going over the whole training set $\mathcal{D}$ multiple times.

3 Training curriculum

Active learning [Balcan et al., 2009] aims to choose the training examples to be processed adaptively to improve training efficiency, both in terms of convergence speed and the performance of the final model parameter. In the neural network curriculum learning literature, this usually happens at the epoch level [see, e.g., Bengio et al., 2009, Hacohen and Weinshall, 2019]: in every epoch e a subset $\mathcal{D}_e$ of the training data is selected and is given to the learning algorithm to update its parameters.

The selection is usually done through a *score function* s_e, where training points are included in $\mathcal{D}_e$ based on their score (either deterministically or randomly). The size of $\mathcal{D}_e$, which determines the length of epoch e is sometimes referred to as the *pacing function*. It affects learning through the learning-rate schedule: typically an epoch-dependent learning rate η_e is used, which controls the size of the update in epoch e. Thus the update has the form

$$w_e = \texttt{update}(w_{e-1}, \eta_e, \mathcal{D}_e),$$

where a possible dependence on the state of the optimizer is not shown explicitly. For example, in case of gradient descent, the update is computed as $w_e = w_{e-1} - \eta_e \nabla_w \mathcal{L}(w_e; \mathcal{D}_e)$. In the formulation above, the *order of the data* in epoch e does not matter, as the gradient information from each data point is handled exactly the same way. In practice, training usually happens in minibatches, that is, $\mathcal{D}_e$ is split into smaller subsets $\mathcal{D}_{e,k}$, and updates are applied successively over these subsets. In this case, given the iterative change of parameters and that the state of the optimizer depends on the minibatch index, the order in which data points are visited matters.

3.1 Static vs. dynamic curriculum

Several papers considered designing curriculum with various successes. *Static* curriculum design [Bengio et al., 2009, Hacohen and Weinshall, 2019, Wu et al., 2021] considers the case when the quantities discussed above are fixed in advance:

(i) The score function s is fixed in advance, and it is the same for each epoch (i.e., $s_e = s$).

(ii) The learning-rate schedule η_e and the number of points g_e in epoch e (the pacing function), is fixed in advance. Note that the pacing function modulates the learning rate: smaller dataset sizes with each epoch bring later (and smaller) learning rates earlier in the training.

(iii) Data ordering, in which the data is selected to $\mathcal{D}_e$.

Given the above, for every epoch e, $\mathcal{D}_e$ is chosen to be a subset of $\mathcal{D}$ of size g_e comprising – typically – of the easiest or hardest or a random subset of points in $\mathcal{D}$.

In a recent work, Wu et al. [2021] demonstrated through extensive experiments that the above choices have little effect in the standard training regime for CIFAR10/100. They showed that various score functions estimating instance difficulty are broadly consistent over similar predictor families (i.e., neural network architectures), while the pacing function and data ordering has little effect on the final performance. On the other hand, they found that well-chosen curricula can lead to better training performance if the training time is limited (i.e., it speeds up convergence) or the data is noisy.

On the other hand, a number of papers claimed improvements for *dynamic curricula* [Shalev-Shwartz, 2017, Jiang et al., 2019, Zhou et al., 2020, Wang et al., 2024b, Yuan et al., 2025]. In these papers the selection of $\mathcal{D}_e$ is done dynamically, where typically the score function s_e depends on the training procedure and current parameters, and the data selection is determined by the value of the score function, not by selecting a predetermined number of data points based on their rank.

Formally, at each epoch e, the *score* $s_e(x, y, w_0^{e-1})$ of a point $(x, y) \in \mathcal{D}$ may depend on the whole history of the training process so far, where w_0^{e-1} denotes the sequence $(w_1, \ldots, w_{e-1})$. The inclusion of (x, y) in epoch e is determined by a *selection criterion* $\alpha_e(x, y, s_e)$ with values in $\{0, 1\}$, which, depending on the value of the score, decides whether to include a point in the epoch. This allows us to construct a dataset $\mathcal{D}_e$ for epoch e as

$$\mathcal{D}_e = \{(x, y) \in \mathcal{D} : \alpha_e(x, y, s_e) = 1\} \tag{1}$$

This protocol is summarized in Algorithm 1 in Appendix B. One potential computational issue is the computation of the score function for all data points in $\mathcal{D}$. For example, if $s_e(x, y, w_e)$ depends on the loss $\ell(f_{w_e}(x), y)$, a forward pass on the neural network f_{w_e} has to be computed for each datapoint each epoch, which is already an expensive step. In the rest of the paper we examine if the negative findings of Wu et al. [2021] also apply to dynamic curricula.

The Instance-Dependent Early Stopping (IES) algorithm. To make the investigation more specific, we start from the recent paper of Yuan et al. [2025] presenting state-of-the-art speedups through employing a dynamic curriculum. Specifically, their score function s_e^{sd} is the second-order difference of the loss function: at epoch e, they use

$$s_e^{sd}(x, y, w_0^{e-1}) = \ell(f_{w_{e-1}}(x), y) - 2\ell(f_{w_{e-2}}(x), y) + \ell(f_{w_{e-3}}(x), y). \tag{2}$$

Given a loss function which is bounded from below and assuming the loss decreases over the epochs, this is an intuitive measure of how close the optimization is to the achievable minimum. Note that in order to compute (2), we must store in memory the values of the loss from two previous epochs, $\ell(y, f_{w_{e-2}}(x))$ and $\ell(y, f_{w_{e-3}}(x))$ for every data point. Then, for a given threshold $\delta > 0$ (hyperparameter), the *selection criterion* is computed as

$$\alpha_e^{sd}(x, y, s_e^{sd}) = \mathbb{I}\left(\left|s_e^{sd}(x, y, w_0^{e-1})\right| \geq \delta\right) \tag{3}$$

This produces a subsampled dataset $\mathcal{D}_e$ for epoch e via (1), which is used to train the model. The resulting algorithm, called *Instance-Dependent Early Stopping (IES)*, is an instantiation of Algorithm 1 with the above defined quantities, and is given as Algorithm 2 in Appendix B.

Note that to run IES, in every epoch the prediction for all training points in $\mathcal{D}$ has to be computed, and so any computational savings coming from the data selection can only come from saving on the backward pass (and potentially reaching earlier some stopping criterion). Furthermore, even though in principle one could reuse the forward computation of $f_{w_e}(x)$ used in calculating the score $s_e(x, y, w_0^{e-1})$ to compute the loss gradients for the points in $\mathcal{D}_e$ in the backward pass, this would come with a huge memory footprint for storing all intermediate forward computations.[1] Instead of this, in their implementation, Yuan et al. [2025] modified their algorithm by using the stale score s_{e-1}^{sd} in epoch e, which can be computed more efficiently by computing the score for all points in $\mathcal{D}_{e-1}$ in the backward pass and then doing an additional forward pass on points in $\mathcal{D} \setminus \mathcal{D}_{e-1}$. In the experiments, we will use this stale variant of IES. However, even in this case, roughly at most 2/3 of the computation can be saved, since the forward pass is computed for the whole training set. In Algorithm 4 (Appendix B) we introduce another, computationally more efficient variant which uses stale scores pointwise (i.e., the delays are not necessarily the same for all the points) and – in principle – allows to save more than 50% of the computations.

4 Do curricula matter?

In this section, we change the training algorithm in multiple ways to understand the role of the different components of the IES algorithm. First we mention that Yuan et al. [2025] experimentally analyzed the choice of the threshold δ used in (3) and found that the performance is reasonably stable for different choices of δ; hence, we do not examine the effect of this parameter further. While IES chooses the pacing function (the size of $\mathcal{D}_e$) and the actual training set $\mathcal{D}_e$ for epoch e in a single step, these are not necessarily connected. To separate these two choices, we investigate what happens if the points in $\mathcal{D}_e$ are replaced with (approximately) the same number of random points. This random selection strategy dynamically selects a pacing function but then randomly chooses training points according to the pacing function but independently of the individual properties of the training examples. We then extend the methodology into a computationally more efficient and practical method which avoids doing an additional forward pass on the data. Finally, we discuss the importance of the learning-rate schedule and how it is affected by the pacing function.

Experimental setup. We start by analyzing the methods on the CIFAR10 dataset with further experimental evaluations provided in the following sections. We will use the notion of *compute steps* where we count a forward-backward pass on a training example as 1 compute step and only a forward pass as $1/3$ compute steps. This is because the cost of a backward pass is (approximately) twice the cost of a forward pass. This measure is relevant since, IES (as discussed above) and some variants to be defined below do additional forward passes to compute the scores. We follow the protocol of [Yuan et al., 2025]: We train a ResNet-18 neural network[2] with batch size 64 for 200 epochs with SGD, and we use an exponential learning-rate schedule with initial value 0.1 and $\gamma = 0.96$, and threshold $\delta = 0.001$ for the score threshold. Each experiment is run with 3 seeds; in the figures solid lines indicate the means, while the shaded areas show confidence intervals. We consider the performance of the algorithms both as a function of the number of training epochs and as a function of *compute steps*. Experiments are conducted using $A100$ GPUs.

[1] This is due to the sequential order of first computing the loss to determine which data points are selected, and then compute the gradients. Note that the gradient $\frac{\partial l}{\partial W_k} = \delta_k h_{k-1}^T$ is the outer product of the backpropagated signal δ_k and activation h_{k-1} and therefore one needs to store all intermediate activations if they are not to be recomputed.

[2] Experiments with a vision transformer are presented in Appendix D.

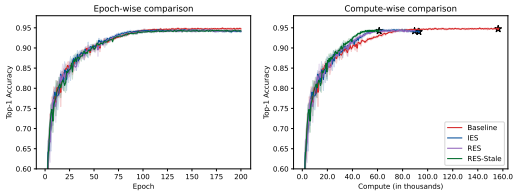


Figure 1: CIFAR10. Performance as a function of epochs (left) and of compute steps (right). Stars indicate the 200-th epoch.

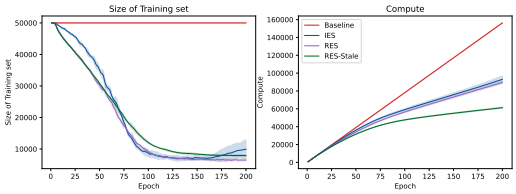


Figure 2: CIFAR10. The size of the training set (left) and the total amount of compute spent (right) as the number of training epochs grows.

Baseline and IES curriculum We start with establishing the performance of our baselines: standard training and the dynamic IES curriculum. In line with the findings of Yuan et al. [2025], Figure 1 shows that IES (blue) achieves slightly worse final accuracy than the baseline (red), where in every epoch the whole training set $\mathcal{D}$ is used, with approximately 20-30% less compute. Note that this is similar to the findings of Wu et al. [2021] for the static curriculum.

4.1 Random subsampling

The IES algorithm can be understood as constructing online a curriculum, an ordering of the data that takes into account the hardness or difficulty of each point. This is in contradiction with the empirical evidence provided by Wu et al. [2021] for the case of static curriculum, who showed that data ordering does not matter. To understand what happens in the dynamic case, we modify the IES algorithm, so that after computing $\mathcal{D}_e$ according to the score, we resample the data points in an i.i.d. fashion from $\mathcal{D}$. Specifically, while we keep the reduction of the dataset the same as the algorithm predicts, we remove the role of selecting certain points.

More formally, at the beginning of epoch e, we compute the selection criterion $\alpha_e(x, y, s_e)$ for every data point $(x, y) \in \mathcal{D}$ as before. However, instead of the deterministic choice of (1), we calculate the proportion of data points to be accepted, and randomly select the elements of $\mathcal{D}_e$ according to this probability. The total (target) number of training points to be used in epoch e thus is $g_e = \sum_{x,y\in\mathcal{D}} \alpha_e(x, y, s_e)$, and we include every $(x, y) \in \mathcal{D}$ in $\mathcal{D}_e$ independently with probability $\frac{g_e}{|\mathcal{D}|}$. This is done by sampling independent Bernoulli random variables $\alpha_e^r(x, y)$ with parameter $\frac{g_e}{|\mathcal{D}|}$ (i.e., $\alpha_e^r(x, y) = 1$ with probability $\frac{g_e}{|\mathcal{D}|}$) and select $\mathcal{D}_e$ as

$$\mathcal{D}_e = \{(x, y) \in \mathcal{D} : \alpha_e^r(x, y) = 1\}.$$

Note that the expected cardinality of $\mathcal{D}_e$ is g_e. The pseudocode of the procedure, which we call *Random Subsamping Early Stopping (RES)* is given in Algorithm 3 in Appendix B. The difference to IES is that now the exact prioritization of data does not matter, only that the epochs become *shorter*.

Experimental results on CIFAR10, shown in Figure 1, demonstrate that IES and RES behave the same way. We used a threshold $\delta = 0.005$ for RES in order to match the dynamics of rejected samples with IES, which uses $\delta = 0.001$ (for completeness, we add the results with $\delta = 0.001$ in Appendix C). Accordingly, the epoch sizes remain similar for both algorithms, as shown in Figure 2, with RES having somewhat smaller epochs (at the beginning and end of the training) due to its increased threshold δ, but the compute used is very similar. Overall, we see that both methods have similar behavior and they achieve essentially the same performance, indicating that *the actual curriculum does not matter, only the choice of the pacing function*.

4.2 Stale random curriculum: random subsampling on the fly

Although both IES and RES achieve performance that is very close to that of the baseline with much less compute, the extra forward pass needed to compute the score functions limits the potential gains of these algorithms. We can avoid doing this additional forward pass if we only update the score of the data points which participate in the training on the current epoch (i.e., the points in $\mathcal{D}_e$), and use a stale score value – saved at some previous epoch – for all the other points. Note that this procedure would not work for any selection mechanism which makes a deterministic decision about every training point based on a threshold on its score (such as α_e^{sd} of IES), because once a point is

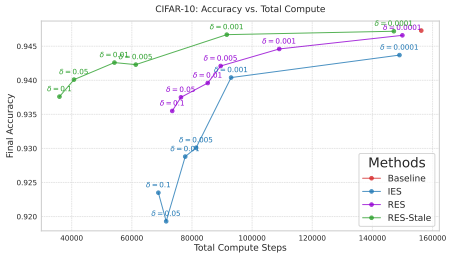


Figure 3: CIFAR10. Final accuracy as a function of total training compute for different parametrizations of the algorithms.

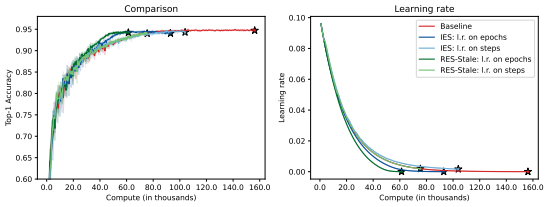


Figure 4: CIFAR10. Performance (left) and learning-rate schedule (right). Stars indicate the 200-th epoch.

excluded, it will never be revisited (since its score will not change), and hence will likely be forgotten as the training goes on. In contrast, the random selection mechanism ensures that points regularly return to $\mathcal{D}_e$, and hence their score will be regularly recomputed as intended, even if the selection is done based on some stale score. To implement the method for the score function s_e^{sd} based on the second-order difference of the loss function, we actually need to store two previous loss values for any point (x, y), say $l'(x, y)$ and $l''(x, y)$, and we use these stale loss values (whose computation could be separated by many epochs) to approximate the score as

$$\hat{s}_e^{sd}(x, y, w_0^{e-1}) = \ell(f_{w_{e-1}}, x, y) - 2l'(x, y) + l''(x, y).$$

For completeness, the pseudocode of the resulting algorithm, called *Random Subsampling Early Stopping with Stale Scores (RES-Stale)* is given in Algorithm 4 in Appendix B.

Figure 1 shows that on CIFAR10, RES-Stale (with the same $\delta = 0.005$ as for RES) achieves similar prediction accuracy as IES and RES but with significant computational savings, due to not needing to recompute the score for all points in every epoch. Naturally, the epoch sizes are similar to those of RES (as shown in Figure 2), although from the second half of the training RES-Stale uses more samples. This is likely because RES-Stale does not use successive loss values to compute the score, hence the score values are likely larger. Overall we see again that the performance of RES-Stale is similar to (computationally even better than) that of IES, confirming again that the curriculum itself is not a crucial component to achieve speedups.

To check the robustness of this observation, in Figure 3 we show the final performance (after epoch 200) of the different methods as a function of the total training compute for various choices of δ, which shows that the final accuracy improves as the training compute is increased, and confirms that the compute-accuracy tradeoff is better for the random selection methods compared to IES. Experiments with a vision transformer instead of a ResNet-18 architecture, presented in Appendix D, demonstrate the same results.

4.3 The impact on the learning rate

The experiments above confirm that the observed gains compared to the baseline come from reducing the size of the dataset in each epoch, rather than the exact data points we pick. The main aspect which differentiates the baseline and RES-Stale is the resulting learning-rate schedule, since it is updated at every epoch. By reducing the epoch size, the learning rate is changed (reduced) more aggressively.

Alternatively, one could define the learning rate as a function of the number of minibatches (or updates). In case the epoch size is fixed during training, defining the learning-rate schedules per epoch or per minibatch update are equivalent. However, when the epoch size changes, this creates a discrepancy between the two definitions. To this end, in Appendix E, we explicitly show how a generic version of RES-Stale (for arbitrary score and selection functions) can be mapped into a modified learning-rate schedule applied to a standard, fixed epoch size method.

Remaining with the current formulation, to clearly demonstrate that the change in performance is due to the updated learning-rate schedule, we carried out another experiment where the learning-rate schedule in RES-Stale copied the learning-rate schedule of the baseline per minibatch. The resulting algorithm is very similar to the baseline, although the data order is different (instead of going over the datasets sequentially, here the data is randomized), and the optimization stops earlier as RES-Stale

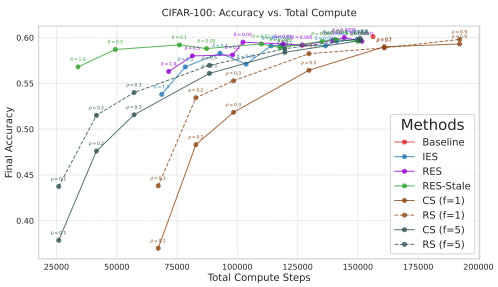


Figure 5: CIFAR100. Final accuracy as a function of total training compute for different parametrizations of the algorithms.

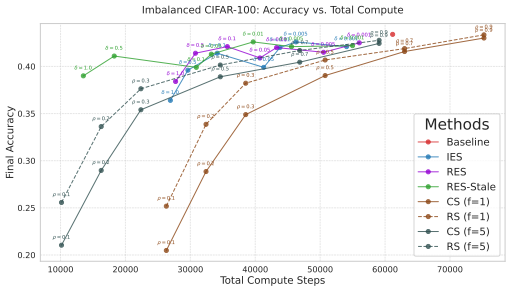


Figure 6: Imbalanced CIFAR100. Final accuracy as a function of total training compute for different parametrizations of the algorithms.

finishes the epochs earlier. The results, presented in Figure 4, confirm the expectations: We can see that the performance of RES-Stale with a step-wise matching learning-rate schedule (green curve), basically matches the performance of the baseline (red curve), while the variant where the learning rate is a function of the epochs (light green), outperforms the baseline. Since in both cases the data is sampled in an i.i.d. fashion, this *confirms that the main performance gain comes primarily form the more aggressively decayed learning-rate schedule*.

5 Robustness studies: experiments on CIFAR100

In this section we carry ablation on the more challenging CIFAR100 as well as considering class imbalance and label noise on CIFAR10 and diversity-based curriculum methods to understand the robustness of previous observations.

5.1 Coreset-based curriculum

As an alternative curriculum method, we consider the coreset-selection method of Sener and Savarese [2017], where training points (a coreset) are selected with the aim that a model learned over the selected subset is competitive for the remaining data points: the training set for $\mathcal{D}_e$ for an epoch is selected by the K-center coreset selection (CS) method [Sener and Savarese, 2017], which utilizes the penultimate-layer embeddings extracted via a forward pass over the full dataset to select a data subset using the K-Center Greedy algorithm. The method is controlled by two primary parameters: the coreset fraction (ρ), representing the proportion of the dataset retained, and the re-selection frequency (f), which dictates how often (in epochs) the subset is recomputed. As a baseline, we compare this against a Random Subset (RS) selection method, which mirrors the CS method exactly in terms of frequency and size but selects samples uniformly at random rather than using the greedy selection strategy. To ensure a fair comparison, compute is measured consistently across all methods: as before, a forward-backward training pass counts as 1.0 step, while a forward-only pass (required for embedding extraction) counts as 1/3 steps. We further compare this method to IES, RES and RES-Stale. For completeness, the results for CIFAR10 are presented in Figure 13 in the appendix, in Figure 5 for CIFAR100. They reinforce our previous observations: specialized coreset selection and its random selection variant perform largely on par, with random selection frequently yielding slightly better results. Consistently, also for CIFAR100, random strategies (RES, RES-Stale or RS) achieve better compute trade-offs compared to their non-random counterparts.

5.2 Class imbalance

We evaluated our approach on a long-tailed version of CIFAR100 with a 10x imbalance ratio [see Cui et al., 2019]. This setup simulates real-world skewed distributions where class sizes range from 500 (head classes) down to 50 (tail classes), totaling 19,573 training samples. The results confirm that randomized pacing remains the superior strategy for compute-efficient training, even in skewed distributions, yet the final best performance is achieved by the baseline. We compare RES, RES-Stale, IES and coresets. The results are given in Figure 6. Overall, we see similar behavior as before.

Perhaps surprisingly, coreset-based selection is still not better than the instance-based curriculum methods; therefore, we only consider the latter in further experiments.

5.3 Label noise

We repeat the experiment on CIFAR100 with 40% symmetric label noise, with the results given in Figure 7. Consistently with all previous findings, the randomized methods (RES-Stale and RES) perform better than the instance-specific selection IES, and the curriculum methods cannot improve upon the baseline, which is consistent with what was observed by [Yuan et al., 2025].

Note that Wu et al. [2021] report that the use of curriculum is helpful when label noise is present. In contrast, we do not see this advantage, potentially due to the selection process. Yuan et al. [2025], which we rely on, report (see Appendix G in their paper) that the second derivative criterion is not helpful in scenarios with label noise.

6 Empirical validation on ImageNet

We outlined the hypothesis that most gains observed by the recent instance-based curriculum algorithm of Yuan et al. [2025] on CIFAR10 and CIFAR100 are explained by the reduced epoch size and the resulting modified learning-rate schedule. We note that other methods providing acceleration results through curriculum [e.g., Jiang et al., 2019, Zhou et al., 2020], also use learning rate defined in terms of epochs (as usual for standard learning-rate schedules), and hence result in similar more aggressive learning rate modifications.

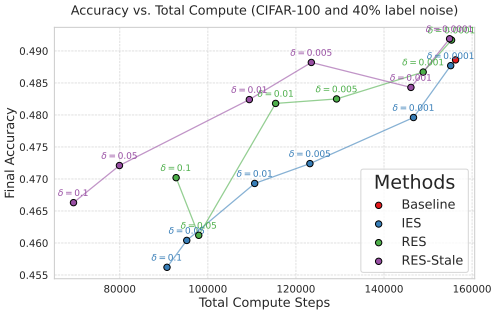


Figure 7: CIFAR100 with label noise. Final accuracy as a function of total training compute for different parametrizations of the algorithms.

To further expand the scope of our hypothesis, and to ensure the observed behavior is not due to some pathologies of CIFAR10 and CIFAR100, we ran experiments on ImageNet with different architectures. First, we used the settings of Yuan et al. [2025], with ResNet-34 which we train for 150 epochs, batch size 2048, no label smoothing, SGD optimizer and learning-rate schedule (referred to as *SGD-m*), which starts at 0.1 and is decayed by 0.1 at epoch 50 and epoch 100. The baseline accuracy is 72.35%. Second, in a more optimized setting, we trained a ResNet-50 network for 90 epochs, batch size 4096, label smoothing of 0.1 and SGD optimizer where the learning rate is linearly warmed up to 1.6 from 0 in the first 5 epochs and then decayed three times by 0.1 at epochs 30, 60 and 80. The baseline accuracy is 76.5%. Finally, we train EfficientNet-B0 [Tan and Le, 2019] for 350 epochs with batch size 4096 and the same optimization setting as in the original paper. The baseline accuracy is 76.9%. In all settings, we use $\delta = 1$. We use 3 seeds for all the runs and report the mean and the confidence intervals in shaded areas of all the curves. Experiments are conducted using $A100$ GPUs.

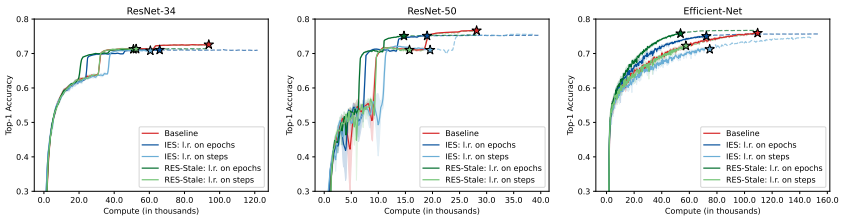


Figure 8: ImageNet. Top-1 test accuracy on ResNet-34, ResNet-50, and EfficientNet, respectively, from left to right, as a function of the total compute. Stars indicate the terminal epoch. Dashed lines correspond to an experiment where training continues after reaching the terminal epoch with the learning rate value at the terminal epoch.

In all three settings, we study the performance of IES (Algorithm 2) and RES-Stale (Algorithm 4), with learning-rate schedule both defined in terms of epochs and minibatches, respectively. The prediction accuracy of the models as a function of compute used in training (also including the cost of the additional forward pass to calculate the score for IES) is shown in Figure 8. We can see that if the learning rate is a function of the number of minibatches (and not epochs), RES-Stale has essentially the same performance as the baseline (up to the number of steps it makes). What is interesting is that for IES, using such learning rate leads to worse performance, which shows the potential problems of the deterministic dataset selection compared to a random choice of $\mathcal{D}_e$. We see that RES-Stale with learning rate as a function of epochs leads to the fastest learning among all the variants. Since here the behavior of IES and RES-Stale are too different, we cannot claim that any benefits of IES can be explained by the change to the learning-rate schedule, but using a random data selection instead of more careful, hardness-based selection seems always preferable, questioning the legitimacy of using an instance-based curriculum.

Note on the final performance. These plots also suggest that the curriculum strategy, by making the epochs shorter and adjusting the learning rate more aggressively, could lead to a final performance which is lower than that of the baseline; see Figure 8, where the red curve corresponding to the baseline is usually slightly higher than the rest, and also Figure 1 for the CIFAR10 and Figure 5 for the CIFAR100 datasets, which show the same phenomenon. One potential explanation for this is that the baseline learning-rate schedule was optimized for final performance, and using the shorter epochs leads to a modified schedule better suited for other training horizons.

To expand our investigations beyond the method of Yuan et al. [2025], inspired by the importance weighting work of Katharopoulos and Fleuret [2018], we also consider a score function called L_1*-error score*.

L_1-error score. Recall that we assume that the labels are one-hot representations of C classes and the output of the network $f_w(x)$ represents a probability distribution over the label space $\mathcal{Y}$. Then the L_1 error of the prediction with parameter w is given as $\Delta_w(x, y) = \frac{1}{2}\|y - f_w(x)\|_1$. The *$L_1$-error score* is defined as

$$s_e^{L_1}(x, y; w_0^{e-1}) = \max\left\{\beta, \Delta_{w_{e-1}}(x, y)\right\}, \tag{4}$$

where $\beta \geq 0$ is a hyperparameter, controlling the minimum of $s_e^{L_1}(x, y, w_0^{e-1})$, that is, the smallest possible score. This score function behaves quite resonably: it is easy to see that for correctly classified example ($f_y(x)$ is the largest coordinate) this score is larger when the confidence is lower, and for incorrectly classified examples the score is higher when the confidence is higher.

We ran an experiment on ImageNet with ResNet-50, and use the L_1-error score with $\beta = 0.3$ for IES and RES-Stale. The results are reported in Figure 15 for this score and we additionally report results for the second-order difference score (2) in Figure 14, in Appendix E. We see that using this score behaves similarly to the second-order difference. This demonstrates that the choice of the score function itself might have only a limited impact on performance.

Additionally, we propose a simple algorithm that uses the expected value of the score function to modulate the learning rate schedule, referred as *minibatch-rescaled LR*. Details as well as pseudo-code (Algorithm 5) are provided in Appendix E. For this method, we use $\beta = 0.8$ for (4) and $\delta = 0.1$ for (2). Results are reported in Figure 14 and in Figure 15. We observe that this simple and computationally cheap approach of just modulating the learning rate schedule leads to similar behavior as RES-Stale, confirming our main hypothesis.

7 Discussion

Our empirical evaluations highlight that in the case of adaptive instance-based curricula, at least for CIFAR10 and ImageNet, the ordering of the data plays little role in the behavior of the system, and the pacing function (i.e. the reduction in size of the training set for each epoch) is what drives the observed faster convergence, due to the impact it has on the learning-rate schedule. This observation extends the results of Wu et al. [2021] from static to dynamic curriculum. We further distill this observation into a cheap and effective adaptive learning-rate schedule that relies on the average of the scoring function, and is readily applicable to standard minibatch training.

An interesting open question is to understand what distributional properties of the data induce this result. Specifically, what data distributions would be ideal to study the impact of the explicit selection of data points? Or whether is it the data alone that drives this behavior or other salient properties of the pipeline? We suspect that the reason for not seeing any positive effect of instance-specific data selection in our experiments is partly due to the fact that we used quite homogeneous (e.g., centered) image data, where it is also known that the features learned in some classification tasks can easily transfer to new classes (of similar nature), as evidenced by the several results on few-shot and transfer learning [Rusu et al., 2019, Bommasani et al., 2021, Galanti et al., 2022], hence learning some instances is generally useful for others as well. Finally, an additional question is whether the lack of success here is merely an issue of scale, and if the curriculum is built at the level of modes of the distribution (or data sources), rather than individual instances, will it be more effective.

Limitations. Our results are limited to CIFAR and ImageNet datasets, which while widely used in the community, might not be representative in general. We also explore only a limited set of algorithms, and while we believe the results are robust, we have not exhaustively explored all possible data selection mechanisms to build curricula.

References

Sébastien M. R. Arnold, Guneet S. Dhillon, Avinash Ravichandran, and Stefano Soatto. Uniform sampling over episode difficulty. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NeurIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, et al. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*, 2021.

Haoli Bai, Lu Hou, Lifeng Shang, Xin Jiang, Irwin King, and Michael R Lyu. Towards efficient post-training quantization of pre-trained language models. *Advances in Neural Information Processing Systems*, 35:1405–1418, 2022.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89, January 2009.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, 2009.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9260–9269, 2019. URL https://api.semanticscholar.org/CorpusID:58014111.

Arthur Douillard, Qixuan Feng, Andrei A Rusu, Adhiguna Kuncoro, Yani Donchev, Rachita Chhaparia, Ionel Gog, Marc'Aurelio Ranzato, Jiajun Shen, and Arthur Szlam. Dipaco: Distributed path composition. *arXiv preprint arXiv:2403.10616*, 2024.

Tomer Galanti, András György, and Marcus Hutter. On the role of neural collapse in transfer learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=SwIp410B6aQ.

Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, 2019.

Michael Iuzzolino, Michael C Mozer, and Samy Bengio. Improving anytime prediction with parallel cascaded networks and a temporal-difference loss. *Advances in Neural Information Processing Systems*, 34:27631–27644, 2021.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Angela H. Jiang, Daniel L. K. Wong, Giulio Zhou, David G. Andersen, Jeffrey Dean, Gregory R. Ganger, Gauri Joshi, Michael Kaminksy, Michael Kozuch, Zachary C. Lipton, and Padmanabhan Pillai. Accelerating deep learning by focusing on the biggest losers, 2019. URL https://arxiv.org/abs/1910.00762.

Angelos Katharopoulos and Francois Fleuret. Not all samples are created equal: Deep learning with importance sampling. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2525–2534. PMLR, 10–15 Jul 2018.

Kevin J Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, Weizhu Chen, Changyou Chen, and Lawrence Carin. Mixkd: Towards efficient distillation of large-scale language models. *arXiv preprint arXiv:2011.00593*, 2020.

Yuanxin Liu, Fandong Meng, Zheng Lin, Jiangnan Li, Peng Fu, Yanan Cao, Weiping Wang, and Jie Zhou. A win-win deal: Towards sparse and robust pre-trained language models. *Advances in Neural Information Processing Systems*, 35:19189–19202, 2022.

Open-Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michaël Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech M. Czarnecki. Open-ended learning leads to generally capable agents. *ArXiv*, abs/2107.12808, 2021. URL https://api.semanticscholar.org/CorpusID:236447390.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34: 20596–20607, 2021.

Matthias C Rillig, Marlene Ågerstrand, Mohan Bi, Kenneth A Gould, and Uli Sauerland. Risks and benefits of large language models for the environment. *Environmental Science & Technology*, 57 (9):3464–3466, 2023.

Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*, 2019.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv: Machine Learning*, 2017. URL https://api.semanticscholar.org/CorpusID:3383786.

Shai Shalev-Shwartz. Selfieboost: A boosting algorithm for deep learning, 2017. URL https://arxiv.org/abs/1411.3436.

Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/tan19a.html.

Vithursan Thangarasa, Abhay Gupta, William Marshall, Tianda Li, Kevin Leong, Dennis DeCoste, Sean Lie, and Shreyas Saxena. Spdf: Sparse pre-training and dense fine-tuning for large language models. In *Uncertainty in Artificial Intelligence*, pages 2134–2146. PMLR, 2023.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Represenations*, 2019.

Wenxiao Wang, Wei Chen, Yicong Luo, Yongliu Long, Zhengkai Lin, Liye Zhang, Binbin Lin, Deng Cai, and Xiaofei He. Model compression and efficient inference for large language models: A survey. *arXiv preprint arXiv:2402.09748*, 2024a.

Yulin Wang, Yang Yue, Rui Lu, Yizeng Han, Shiji Song, and Gao Huang. Efficienttrain++: Generalized curriculum learning for efficient visual backbone training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):8036–8055, 2024b. doi: 10.1109/TPAMI.2024.3401036.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tW4QEInpni.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

Suqin Yuan, Runqi Lin, Lei Feng, Bo Han, and Tongliang Liu. Instance-dependent early stopping. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=P42DbV2nuV.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*, 2023.

Tianyi Zhou, Shengjie Wang, and Jeffrey Bilmes. Curriculum learning by dynamic instance hardness. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8602–8613. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/62000dee5a05a6a71de3a6127a68778a-Paper.pdf.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*, 2023.

A Comparison of Curriculum Methods

In Table 1, we summarize the main characteristics of the methods considered in this work, highlighting their differences in instance selection, pacing, computational overhead, and memory requirements.

In particular, IES and RES both use instance-level scores to determine the pacing function, but differ in how the training examples are selected: IES uses deterministic thresholding, whereas RES selects examples uniformly at random. RES-Stale further reduces computational overhead by updating scores only for examples encountered during training. We also include coreset selection (CS), random subset selection (RS), and the minibatch-rescaled learning-rate method introduced in Algorithm 5.

Table 1: Comparison of the methods considered in this work. "Changes pacing" indicates whether the number of training examples per epoch is adjusted dynamically. The forward-pass column reports additional computation beyond standard training. For coreset selection, f denotes the re-selection frequency in epochs.

Method	Instance scores	Selection	Changes pacing	Extra forward passes	Persistent state
Baseline	No	Full dataset	No	None	None
IES	Yes	Deterministic threshold	Yes	Full dataset / epoch	Loss history (2nd-order score)
RES	Yes (pacing only)	Random (Bernoulli)	Yes	Full dataset / epoch	Loss history (2nd-order score)
RES-Stale	Yes (pacing only)	Random (Bernoulli)	Yes	None	Cached scores and loss history
CS (coreset)	Embeddings	K-center greedy	No (fixed)	Full dataset every f epochs	None
RS (random selection)	No	Random	No (fixed)	None	None
Minibatch-rescaled LR	Minibatch-level	Full dataset	No (LR only)	None	LR index

B Pseudo-codes for the different algorithms discussed

Algorithm 1 Dynamic curriculum selection

Require: Full training set $\mathcal{D}$, initial parameter w_0, maximum number of epochs E, for every epoch e selection criterion α_e and learning rate η_e.

Initialize model parameters w_0
for $e = 1$ to E **do**
  Compute scores $s_e(x, y, w_0^{e-1})$ for every $(x, y) \in \mathcal{D}$
  Construct $\mathcal{D}_e = \{(x, y) \in \mathcal{D} : \alpha_e(x, y, s_e) = 1\}$
  if $\mathcal{D}_e$ is empty **or** some conventional early stopping criterion is satisfied **then**
    break
  end if
  Update parameters $w_e = \texttt{update}(w_{e-1}, \eta_e, \mathcal{D}_e)$
end for

Algorithm 2 Instance-Dependent Early Stopping (IES)

Run Algorithm 1 with score function s_e^{sd} given in (2) and selection criteria α_e^{sd} given in (3). For epochs $e = 1, 2$, set the score function $s_e^{sd} = \delta$ to ensure that the whole training set is included in the first two epochs, while the corresponding loss values are memorized.

Algorithm 3 Random Subsampling Early Stopping (RES)

Require: Full training set $\mathcal{D}$, initial parameter w_0, maximum number of epochs E, for every epoch e selection criterion α_e and learning rate η_e.

Initialize model parameters w_0
for $e = 1$ to E **do**
  Compute scores $s_e^{sd}(x, y, w_0^{e-1})$ for every $(x, y) \in \mathcal{D}$
  Compute the pacing $g_e = \sum_{x,y \in \mathcal{D}} \alpha_e^{sd}(x, y, s_e)$
  Sample the i.i.d. Bernoulli($g_e/|\mathcal{D}|$) variables $\alpha_e^r(x, y)$ for every $(x, y) \in \mathcal{D}$
  Construct new epoch dataset $\mathcal{D}_e = \{(x, y) \in \mathcal{D} : \alpha_e^r(x, y) = 1\}$
  if $\mathcal{D}_e$ is empty **or** some conventional early stopping criterion is satisfied **then**
    break
  end if
  Update parameters $w_e = \texttt{update}(w_{e-1}, \eta_e, \mathcal{D}_e)$
end for

Algorithm 4 Random Subsampling Early Stopping with Stale Scores (RES-Stale)

Require: Full training set $\mathcal{D}$, initial parameter w_0, maximum number of epochs E, for every epoch e selection criterion α_e with threshold $\delta > 0$ and learning rate η_e.

Initialize model parameters w_0 and $l'(x, y) = l''(x, y) = 0$ for all $(x, y) \in \mathcal{D}$
for $e = 1$ to E **do**
  if $e \in \{1, 2, 3\}$ **then**
    $\hat{s}^{sd}(x, y) = \delta$
  end if
  Compute the pacing $g_e = \sum_{x,y \in \mathcal{D}} \alpha_e^{sd}(x, y, \hat{s})$
  Sample the i.i.d. Bernoulli($g_e/|\mathcal{D}|$) variables $\alpha_e^r(x, y)$ for every $(x, y) \in \mathcal{D}$
  Construct new epoch dataset $\mathcal{D}_e = \{(x, y) \in \mathcal{D} : \alpha_e^r(x, y) = 1\}$
  if $\mathcal{D}_e$ is empty **or** some conventional early stopping criterion is satisfied **then**
    break
  end if
  Update parameters $w_e = \texttt{update}(w_{e-1}, \eta_e, \mathcal{D}_e)$
  for $(x, y) \in \mathcal{D}_e$, during the above **do**
    Compute $\ell(f_{w_{e-1}}(x), y)$
    Update $\hat{s}^{sd}(x, y) \leftarrow \ell(f_{w_{e-1}}, x, y) - 2l'(x, y) + l''(x, y)$
    Update memory $l''(x, y) \leftarrow l'(x, y)$, $l'(x, y) \leftarrow \ell(f_{w_{e-1}}, x, y)$
  end for
end for

C Additional results for CIFAR10

In this section, we add results using the same threshold $\delta = 0.001$ for all the methods on CIFAR10. The results are given in Figure 9 and in Figure 10. We see that while the behavior of size of training set is different, the performance still remains roughly the same.

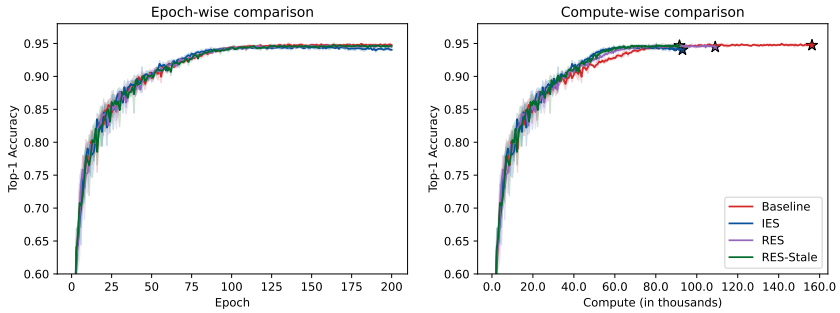


Figure 9: CIFAR10. Left shows performance comparison as a function of epoch, while on the right shows performance as a function of compute steps. Stars indicate where each of the methods reached epoch 200. We use $\delta = 0.001$ for all the methods.

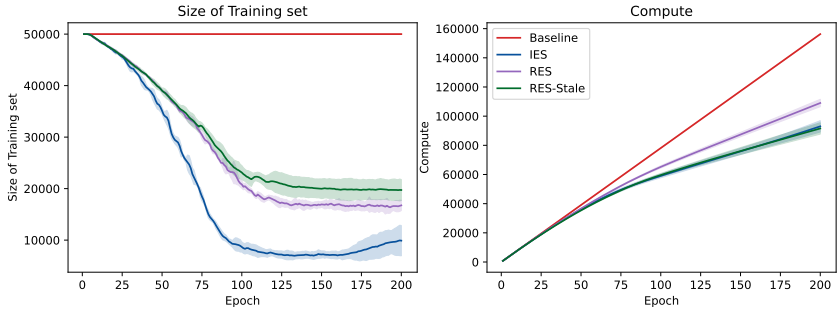


Figure 10: CIFAR10. Left, size of training set as a function of epochs. Right, the total amount of compute spent as the number of training epochs grows. We use $\delta = 0.001$ for all the methods.

D Vision Transformer on CIFAR10

We repeated our CIFAR10 experiments using a Vision Transformer (ViT). We used a small ViT variant (6 layers, 8 heads, 512 hidden dim, 4×4 patches) specifically tuned for 32×32 images. We trained for 200 epochs using AdamW with a linear warmup (10 epochs) and cosine decay schedule, since using SGD with exponential decay led to lower accuracy (around 60%). The results are given in the Figure 12.

RES-Stale ($\delta = 0.001$) achieves a slightly higher final accuracy than the Baseline while using 15% less compute. Moreover, similarly to the results in Appendix C, we see that RES-Stale consistently achieves better compute-accuracy trade-off than IES.

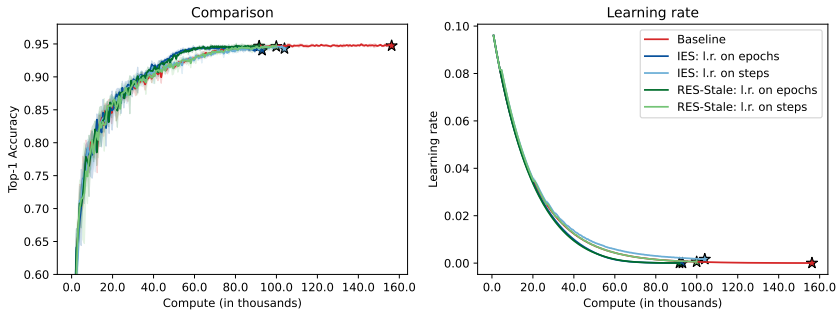


Figure 11: CIFAR10. Left, performance as a function of compute. Right, the learning-rate schedule. Stars indicate the time where algorithm reached 200 epochs. We use $\delta = 0.001$ for all the methods.

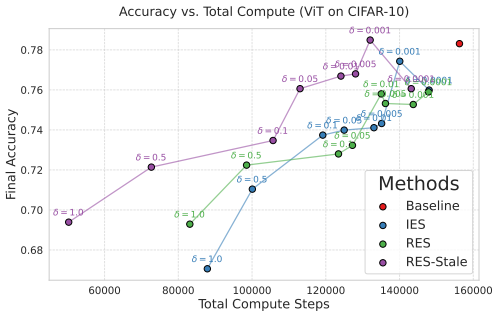


Figure 12: CIFAR10. Final accuracy as a function of total training compute for different parametrizations of the algorithms for the vision transformer (ViT) architecture.

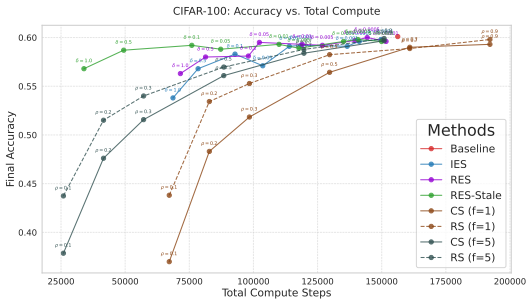


Figure 13: CIFAR10. Final accuracy as a function of total training compute for different parametrizations of the algorithms for ResNet-18, including the coreset-based methods.

E From pacing functions to adaptive learning rate-schemes

We have shown how changing the epoch sizes leads to a change in the learning-rate schedule. Here we will make this connection explicit, mapping a generic version of RES-Stale (for arbitrary score and selection functions) into a modified learning-rate schedule applied to a standard, fixed epoch size training method, with explicit minibatches.

Consider a minibatch-style algorithm, where in step t, we select a minibatch $\mathcal{B}_t$ of size B uniformly at random from the dataset $\mathcal{D}$, and update the network parameters according to $w_t = \texttt{update}(w_{t-1}, \eta_t, \mathcal{B}_t)$ where η_t is the learning-rate schedule. Previously we mainly considered (and just implicitly mentioned) minibatch-dependent schedules; now we make this connection explicit: given an epoch-based schedule η'_e where each epoch consists of T_e minibatches, batch t falls into epoch $\lfloor t/T_e \rfloor$, and hence the corresponding learning rate is $\eta_t = \eta'_{\lfloor t/T_e \rfloor}$.

Now consider a dynamic curriculum method, starting from the above minibatch-based learning-rate schedule η_t. Due to reducing the epoch size, instead of $|\mathcal{D}|$ samples we only process $g_e = |\mathcal{D}_e|$ samples in epoch e, giving a subsampling ratio $p_e = \frac{g_e}{|\mathcal{D}|}$. Thus, after processing g_e data points in epoch e, the minibatch index of the learning rate should increase not by $|\mathcal{D}_e|/B$ (the number of actual minibatches), but by $|\mathcal{D}|/B$. In other words, after every minibatch in e, the index of the learning rate should increase by $\frac{|\mathcal{D}|}{|\mathcal{D}_e|}$.

Since we process data at the minibatch level, and it is enough to select the datapoints randomly, instead relying on the score function, we can estimate the subsampling ratio $\hat{p}_t \approx p_t$ from the minibatch only, at every step. If the batch size is to small for the estimation to be reliable, we can use a moving average over previous batches. The resulting method is presented in Algorithm 5.

Algorithm 5 Minibatch-rescaled LR-schedule

Require: Full training set $\mathcal{D}$, initial parameter w_0, minibatch size B, max number of minibatches T, selection criterion α learning-rate schedule η_t, schedule time index $\tau_0 = 0$, score function s.

1: Initialize subsampling ratio $p_0 = 1$
2: **for** batch $t = 1, 2, \ldots$ **do**
3: Sample batch $\mathcal{B}_t$ of size B uniformly at random from $\mathcal{D}$.
4: In the forward path of the update, compute score $s(x, y, w_0^{t-1})$ for all $(x, y) \in \mathcal{B}_t$
5: Compute the subsampling ratio estimate $\hat{p}_t = \frac{\sum_{(x,y)\in\mathcal{B}_t} \alpha(x,y,s)}{B}$
6: Update learning rate index $\tau_t = \tau_{t-1} + 1/\hat{p}_t$
7: Cast to integer $\tilde{\tau}_t = \texttt{int}(\tau_t)$
8: Update parameters $w_t = \texttt{update}(w_{t-1}, \eta_{\tilde{\tau}_t}, \mathcal{B}_t)$
9: **end for**

In Figure 14, we show a comparison of this method against IES and RES-Stale on ImageNet with ResNet-50 using the score (2). For the adaptive learning rate method, we use $\delta = 0.1$, while for IES and RES-Stale, $\delta = 1$. We see that using this adaptive learning rate strategy leads to a similar performance compared to other approaches, especially to that of Res-Stale (although the method then runs longer with essentially no improvement).

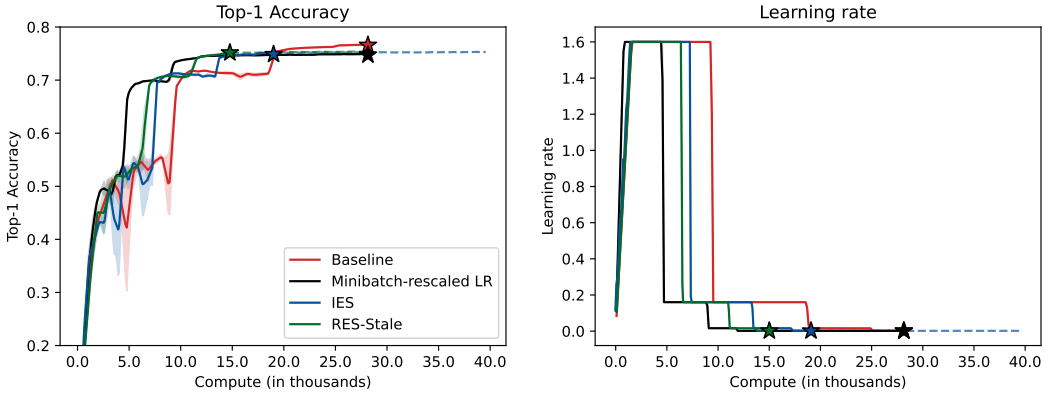


Figure 14: ImageNet with ResNet-50. Adaptive learning rate strategy to IES and RES-Stale using the second-order difference score (2). Left, test accuracy; right, learning rate. Stars indicate terminal epochs. Dashed lines indicate an experiment with continued training after terminal epoch with the final learning rate.

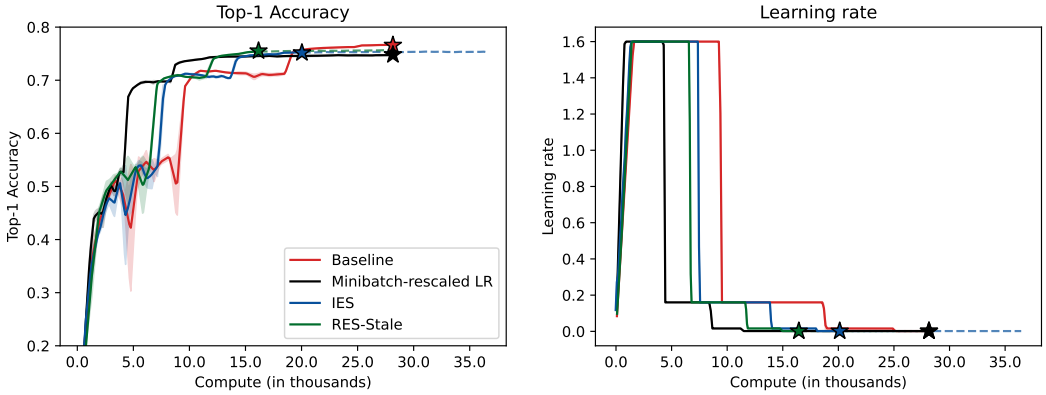


Figure 15: ImageNet with ResNet-50. Adaptive learning rate strategy to IES and RES-Stale using the L_1-error score (4). Left, test accuracy; right, learning rate. Stars indicate terminal epochs. Dashed lines indicate an experiment with continued training after terminal epoch with the final learning rate.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper directly and explicitly explores the hypothesis stated in the abstract, showing empirical evidence for it. Specifically the order of the data does not matter, the observed improvement is given by the impact on the learning rate schedule.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper proposes a hypothesis, providing careful empirical evidence and ablation to support it. However we do not have any theoretical results or proofs. However, even for the empirical exploration, we are explicit about the assumption we make (and contextualize our results accordingly).

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: While we do not include the code implementation, we include extensive experimental details in Section 4, Section 5, Section 6, which allow the reader to reproduce the experimental setting.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: While we do not include the code implementation, we include extensive experimental details in Section 4, Section 5, Section 6, as well as in the appendix. Regarding data, we use popular public datasets.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide all the information about our experiments in Section 4, Section 5, Section 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Shaded areas are used as means of error bars. Experiments are always conducted with 3 random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide all the information about our experiments in Section 4, Section 5, Section 6, including hyperparameters sweep, number of epochs and the used hardware.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We fulfill all requirements.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Our paper conducts no societally harmful research.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: There is no risk of misuse associated to the models used in the present work.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit the creators of each dataset used in our experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: We do not use crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: We do not conduct experiments on humans.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We have not made use of LLMs in any significant way.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.