DYNAMS: DYANMIC MARGIN SELECTION FOR EFFICIENT DEEP LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The great success of deep learning is largely driven by training over-parameterized models on massive datasets. To avoid excessive computation, extracting and training only on the most informative subset is drawing increasing attention. Nevertheless, it is still an open question how to select such a subset on which the model trained generalizes on par with the full data. In this paper, we propose dynamic margin selection (DynaMS). DynaMS leverages the distance from candidate samples to the classification boundary to construct the subset, and the subset is dynamically updated during model training. We show that DynaMS converges with large probability, and for the first time show both in theory and practice that dynamically updating the subset can result in better generalization over previous works. To reduce the additional computation incurred by the selection, a light parameter sharing proxy (PSP) is designed. PSP is able to faithfully evaluate instances with respect to the current model, which is necessary for dynamic selection. Extensive analysis and experiments demonstrate the superiority of the proposed approach in data selection against many state-of-the-art counterparts on benchmark datasets.

1 INTRODUCTION

Deep learning has achieved great success owing in part to the availability of huge amounts of data. Learning with such massive data, however, requires clusters of GPUs, special accelerators, and excessive training time. Recent works suggest that eliminating non-essential data presents promising opportunities for efficiency. It is found that a small portion of training samples[1] contributes a majority of the loss (Katharopoulos & Fleuret, 2018; Jiang et al., 2019), so redundant samples can be left out without sacrificing performance. Besides, the power law nature (Hestness et al., 2017; Kaplan et al., 2020) of model performance with respect to the data volume indicates that loss incurred by data selection can be tiny when the dataset is sufficiently large. In this sense, selecting only the most informative samples can result in better trade-off between efficiency and accuracy.

The first and foremost question for data selection is about the ***selection strategy***, how to efficiently pick training instances that benefit model training most. Various principles have been proposed, including picking samples that incur the largest loss or gradient norm (Paul et al., 2021; Coleman et al., 2020), selecting those most likely to be forgotten during training, as well as utilizing subsets that best approximate the full loss (Feldman, 2020) or gradient (Mirzasoleiman et al., 2020; Killamsetty et al., 2021). Aside from selection strategies, existing approaches vary in the ***training schemes*** which can be divided roughly into two parts: static ones and dynamic (or adaptive) ones. Static methods (Paul et al., 2021; Coleman et al., 2020; Toneva et al., 2019) decouple the subset selection and the model training, where the subset is constructed ahead and the model is trained on such a fixed subset. Dynamic methods (Mindermann et al., 2022; Killamsetty et al., 2021), however, update the subset in conjunction with the training process. Though effectively eliminates amounts of samples, it is still not well understood how these different training schemes influence the final model.

In this paper, we propose dynamic margin selection (DynaMS). For the selection strategy, we inquire the classification margin, namely, the distance to the decision boundary. Intuitively, samples close to the decision boundary influence more and are thus selected. Classification margin explicitly utilizes the observation that the decision boundary is mainly determined by a subset of the data.

[1] We use the terms data, sample, and instance interchangeably

For the training scheme, we show the subset that benefits training most varies as the model evolves during training, static selection paradigm may be sub-optimal, thus dynamic selection is a better choice. Synergistically integrating classification margin selection and dynamic training, DynaMS is able to converge to the optimal solution with large probability. Moreover, DynaMS admits theoretical generalization analysis. Through the lens of generalization analysis, we show that by catching the training dynamics and progressively improving the subset selected, DynaMS enjoys better generalization compared to its static counterpart.

Though training on a subset greatly reduces the training iterations, the overhead introduced by data evaluation undermines its significance, especially when the model is complex and selection is carried out frequently. Previous works explored utilizing a lighter model as a proxy for data evaluation. Utilizing a separate proxy trained a priori (Coleman et al., 2020), however, is insufficient for dynamic selection, where the proxy should be able to agilely adapt to model changes. We thus propose parameter sharing proxy (PSP), where the proxy is constructed by multiplexing part of the underlying model parameters. As parameters are shared all along training, the proxy can acutely keep up with the underlying model. To train the shared network, we utilize slimmable training (Yu et al., 2019) with which well performing PSP and the underlying model can be obtained in just a single train.

Extensive experiments are conducted on benchmarks CIFAR-10 and ImageNet. The results show that our proposed DynaMS effectively pick informative subsets, outperforming a number of competitive baselines. Note that though DynaMS is primarily designed for supervised learning tasks, it is widely applicable as classifiers have become an integral part of many applications including foundation model training (Devlin et al., 2019; Brown et al., 2020; Dosovitskiy et al., 2021; Chen et al., 2020), where hundreds of millions of unlabeled data are consumed.

In summary, the contributions of this paper are three-folds:

- We establish dynamic margin select (DynaMS), which selects informative subset dynamically according to the classification margin to accelerate the training process. DynaMS converges to its optimal solution with large probability and enjoys better generalization.
- We explore constructing a proxy by multiplexing the underlying model parameters. The resulting efficient PSP is able to agilely keep up with the model all along the training, thus fulfill the requirement of dynamic selection.
- Extensive experiments and ablation studies demonstrate the effectiveness of DynaMS and its superiority over a set of competitive data selection methods.

2 Methodology

To accelerate the training process through data selection, we propose dynamic margin selection (DynaMS) whose framework is presented in Figure 1. Instances closest to the classification decision boundary are selected for training. We show that the most informative subset changes as the learning proceeds, so that a dynamic selection scheme that progressively improves the subset can result in better generalization. Considering the computational overhead incurred by selection, we then explore parameter sharing proxy (PSP), which utilizes a much lighter proxy model to evaluate models. PSP is able to faithfully keep up with the underlying model in the dynamics selection scheme. The notations used in this paper are summarized in Appendix F.

2.1 Selection with Classification Margin

Given a large training set $\mathcal{T} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{|\mathcal{T}|}$, where $\boldsymbol{x}_i$, y_i are the input and ground-truth label respectively, data selection extracts the most informative subset $\mathcal{S} \subset \mathcal{T}$ trained on which the model $f(\boldsymbol{x})$ yields minimal performance degradation. To pick the most informative subset for efficient training, the first step is to score the candidate samples. Towards this end, we utilize the classification margin, that is, the distance to the decision boundary, to evaluate the informativeness. $|\mathcal{S}|$ examples with the smallest classification margin are selected. Intuitively, these samples should be influential most to the model decision. Following (Mickisch et al., 2020; Emam et al., 2021), the decision boundary between two classes c_1 and $c_2 \in \{1, \dots C\}$ is $\mathcal{B} := \{\boldsymbol{x} \mid f_{c_1}(\boldsymbol{x}) = f_{c_2}(\boldsymbol{x})\}$, where $f_c(\boldsymbol{x})$ is the c entry of model output, indicating the probability of $\boldsymbol{x}$ belongs to class c. the classification

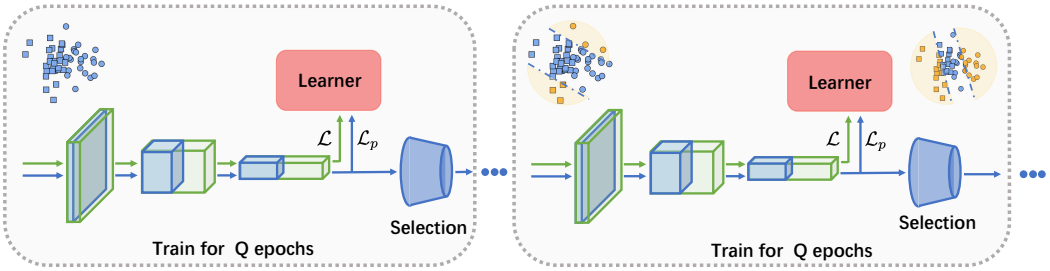


Figure 1: The overall framework of dynamic margin select (DynaMS). The green model indicates the underlying model to be trained and the blue one is the parameter sharing proxy which efficiently evaluates data. Instances are selected every Q epochs, then the model is trained on the selected subset.

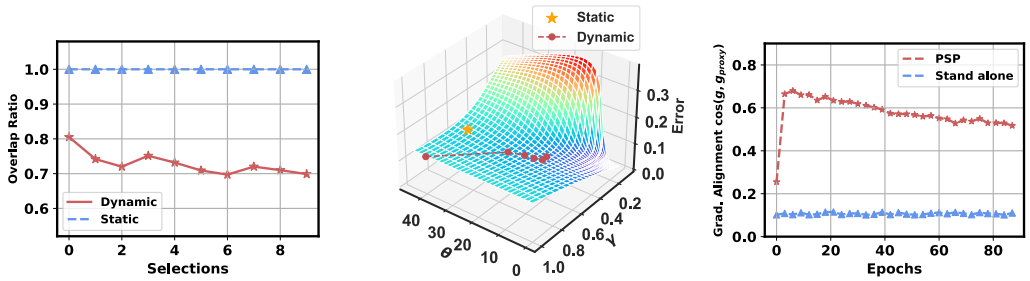
margin is then:

$$M(\boldsymbol{x}, c_1, c_2) = \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_2 \quad \text{s.t. } \boldsymbol{x} + \boldsymbol{\delta} \in \mathcal{B} \tag{1}$$

which is the minimal perturbation required to move $\boldsymbol{x}$ form c_1 to c_2. Directly computing the margin is infeasible for deep neural networks, so scoring is conducted in the feature space instead. Typically neural networks applies a linear classifier on top of the features (Goodfellow et al., 2016), so the classification margin $M(\boldsymbol{x}, c_1, c_2)$ can be easily obtained as: $M(\boldsymbol{x}, c_1, c_2) = (\boldsymbol{W}_{c_1} - \boldsymbol{W}_{c_2})^\top h(\boldsymbol{x}) / \|\boldsymbol{W}_{c_1} - \boldsymbol{W}_{c_2}\|_2$, where $\boldsymbol{W} \in \mathbb{R}^{d \times C}$ is the weight of the linear classifier[2] and $h(\boldsymbol{x})$ is the feature of $\boldsymbol{x}$ before classification. In this way, the classification margin of a labeled sample $(\boldsymbol{x}, y)$ along class c is $M(\boldsymbol{x}, y, c)$ if $y \neq c$ or $\min_{\tilde{c} \neq y} M(\boldsymbol{x}, y, \tilde{c})$ if $y = c$. The former indicates the distance moving $(\boldsymbol{x}, y)$ to class c while the latter is the distance moving $(\boldsymbol{x}, y)$ to the nearest class other than y. To keep the subset samples balanced, during the selection phase, we evenly pick $|\mathcal{S}|/C$ samples with the smallest classification margin along each class. The selection procedure is denoted as $\mathcal{S} = \text{SELECT}(\boldsymbol{w}, \mathcal{T}, |\mathcal{S}|)$.

Computational Complexity To accelerate training, the selection strategy must be efficient enough so that the overhead introduced by the selection procedure is negligible compared to the reduced computation for training. The time complexity of selecting with classification margin is $\mathcal{O}(C \cdot (d + \log |\mathcal{S}|) \cdot |\mathcal{T}|)$, where d is the dimension of sample representation before classification, Computing the margins takes $\mathcal{O}(C \cdot d \cdot |\mathcal{T}|)$ computation and pick the top $|\mathcal{S}|$ values among $|\mathcal{T}|$ instances require $\mathcal{O}(C \cdot \log |\mathcal{S}| \cdot |\mathcal{T}|)$ computation. These complexities make this approach rather appealing because it is on par with the fastest existing competitive approaches.

2.2 Dynamic Selection



(a) Selection overlap ratio. (b) Generalization error. (c) Gradient aligment.

Figure 2: (a) Overlap ratio of subsets extracted in two consecutive selections. A near 1 overlap ratio means the selection is converged. (b) Generalization of static as well as dynamic data selection according to classification margin. Selection budget $\gamma_{\text{avg}} = \gamma_s = 60\%$. (c) The averaged gradient alignment $\cos(\boldsymbol{g}, \boldsymbol{g}_{\text{proxy}})$ of parameter sharing proxy and an stand-alone proxy along model training.

[2]Without loss of generality, we omit the bias term for notation clarity

Given the subset selected with the class margin strategy, model is subsequently trained on $\mathcal{S}$. Conventional static training scheme assumes that the optimal subset converges and is not related to the model training dynamic (Paul et al., 2021; Coleman et al., 2020). Though effectively eliminate instances, the "converged optimal subset" assumption may be too strong. To investigate whether the most informative samples vary during training, we plot the overlap ratio of samples selected in two consecutive selections during the training of a 50-layer residual network, shown in Figure 2(a). 10 epochs of training are conducted between two selections. It can be observed that the overlap ratio is around 0.75 rather than 1.0, which means over 20% samples not in the last $\mathcal{S}$ is picked. In other words, samples that most benefit model training vary as the model evolve, a fixed subset may be outdated after parameter updates and thus yields sub-optimal results.

We thus resort to a dynamic scheme where data selection is performed after each Q epochs training[3]. By selecting in conjunction with training, the informative subset gets updated according to the current model status. For the kth selection, the informative subset $\mathcal{S}_k$ is constructed by picking portion γ_k samples so that $|\mathcal{S}_k| = \gamma_k|\mathcal{T}|$. The selection ratio γ_k determines critical margin κ_k, where only samples with classification margin smaller than κ_k are kept. $\mathcal{S}_k$ will then be used for training Q epochs. In the following, we provide a convergence analysis of DynaMS and show that DynaMS achieves better generalization by constantly improving the model used for sample evaluation.

Convergence Analysis We now study the conditions for the convergence of training loss achieved by DynaMS. We use logistic regression (LR) to demonstrate and then show the conditions are well satisfied when LR is used on top of deep feature extractors. We have the following theorem:

Theorem 2.1. *Consider logistic regression $f(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{w}^\top \boldsymbol{x}}}$ with Gaussian input $\boldsymbol{x} \sim \mathcal{N}(0, \Sigma)$. Assume $\|\boldsymbol{w}\|_2 \leq D$. Let $\boldsymbol{w}^*$ be the optimal parameters, d be the dimension of $\boldsymbol{x}$, and λ be the largest eigenvalue of the covariance Σ. For $t \in \{1, \dots T\}$ and constants $\varepsilon > 0, \zeta > 1, \mu > 0$, select subset with critical margin $\kappa_t = (1+\varepsilon)\log(\zeta T - t)$ and update parameters with learning rate $\eta = \frac{D}{E\sqrt{T}}$. Then with probability at least $1 - \frac{1}{\mu d}$*

$$\min_t \mathcal{L}(\boldsymbol{w}_t) - \mathcal{L}(\boldsymbol{w}^*) \leq DE\left(\frac{1}{\sqrt{T}} + \frac{c_{\varepsilon,\zeta}}{T^{1+\varepsilon}} + \frac{c_{\varepsilon,\zeta,\lambda}}{T^{\beta}}\right) \tag{2}$$

where $E = \sqrt{(1+\sqrt{2\mu})d}$, $\beta = \frac{(1+\varepsilon)^2}{2D^2\lambda}$, $c_{\varepsilon,\zeta}$ and $c_{\varepsilon,\zeta,\lambda}$ are constants depending on ε, ζ and λ.

The proof is left in Appendix B. Theorem 2.1 indicates that by smartly manipulating the critical margin κ, dynamically selecting data based on the classification margin is able to converge and achieve the optimal model parameter $\boldsymbol{w}^*$ with large probability. The Gaussian input assumption is overly strong in general, but when this linea r classifier is adopted on top of wide enough feature extraction layers, the condition is well satisfied because infinitely a wide neural network resembles Gaussian process (Lee et al., 2019; Xiao et al., 2018; de G. Matthews et al., 2018).

Generalization Analysis Recently, based on replica method, (Sorscher et al., 2022) developed an analytic theory for data selection. Assume training data $\boldsymbol{x}_i \sim \mathcal{N}(0, \mathbf{I})$ and there exists an oracle model $\boldsymbol{w}_o \in \mathbb{R}^d$ which generates the labels such that $y_i = \text{sign}(\boldsymbol{w}_o \cdot \boldsymbol{x}_i)$ for all i. Following the static selection scheme, when a lower fidelity estimator $\boldsymbol{w}$ is used to pick samples that has small classification margin $|\boldsymbol{w}^\top \boldsymbol{x}|$, the generalization error takes the form $\mathcal{E}(\alpha, \gamma, \theta)$ in the high dimensional limit C. $\alpha = \frac{|\mathcal{T}|}{d}$ indicates the abundance of training samples before selection; γ determines the selection budget and $\theta = \arccos\left(\frac{\boldsymbol{w}^\top \boldsymbol{w}_o}{\|\boldsymbol{w}\|_2 \cdot \|\boldsymbol{w}_o\|}\right)$ shows the closeness of the estimator to the oracle model. The full set of self-consistent equations characterizing $\mathcal{E}(\alpha, \gamma, \theta)$ is given in Appendix C. By solving these equations the generalization error $\mathcal{E}(\alpha, \gamma, \theta)$ can be obtained.

We then extend the generalization analysis to dynamic training scheme. For the kth selection, we use the model $\boldsymbol{w}_{s_{k-1}}$ trained on $\mathcal{S}_{k-1}$ as the estimator $\boldsymbol{w}_k$, which deviates from oracle by angle $\theta_k = \arccos\left(\frac{\boldsymbol{w}_{k-1}^\top \boldsymbol{w}_o}{\|\boldsymbol{w}_{k-1}\|_2 \cdot \|\boldsymbol{w}_o\|_2}\right)$, to evaluate and select samples. The resulting subset $\mathcal{S}_k$ will be used for subsequent training of model $\boldsymbol{w}_{k+1}$, which will later be used as estimator at $k+1$ to produce $\mathcal{S}_{k+1}$. In this way, generalization of dynamic scheme can be obtained by recurrently solving the

[3]For extremely large dataset case where training can be accomplished within just one or a few epochs, the selection can be performed every Q iterations

equations characterizing $(\alpha, \gamma_k, \theta_k)$ with updated keeping ratio γ_k and estimator deviation θ_k. Note that in each round of selection, samples are picked with replacement, so the abundance of training samples α is kept fixed. γ_k, scheduling the subset size in each round of selection, can be adjusted freely to meet various requirements.

We compare the generalization of dynamic selection and its static counter part in Figure 2(b). We show the landscape of $\mathcal{E}(\alpha, \gamma, \theta)$ with different γ and θ by solving the generalization equations numerically. $\alpha = 3.2$ is keep fixed, which means the initial training data is abundant; For static training, Figure 2(b) uses $\theta_s = 40°$ and $\gamma_s = 0.6$. To make sure the averaged number of samples used per epoch in training equals to subset size used in the static scheme, we make sure $\frac{1}{K}\sum_{k=1}^{K}|\gamma_k| = \gamma_s$ (More details scheduling γ_k is given in E). From Figure 2(b), we see that during the dynamic selection process, the estimator gets constantly improved (θ_k decreases) so that the subsets get refined, the dynamic training scheme can thus achieve much better generalization over the static scheme. Discussion of selecting with different $\alpha, \gamma$and θ will be given in Appendix D.

2.3 Parameter Sharing Proxy

With the above dynamic selection framework, the number of updates is reduced. However, the computational overhead incurred by data selection undermines its significance, especially when the model is complex and samples are evaluated frequently. Aside from designing efficient selection strategies, previous works explored utilizing a lighter model as proxy to evaluate the instances so that the problem can be ameliorated. Pretrain a separate proxy and evaluate instances prior to model training (Coleman et al., 2020), however, is insufficient for dynamic selection, as a static proxy can not catch the dynamics of the underlying model. A Proxy that can fulfill the requirements of dynamic selection is still absent.

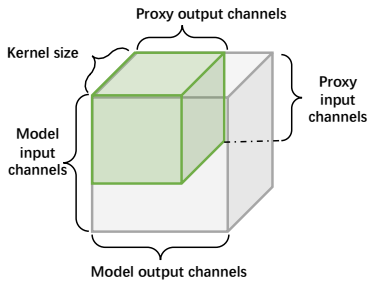


Figure 3: Parameter sharing proxy. The proxy is constructed with part of model parameters, and updated along model training.

We thus propose parameter sharing proxy (PSP), where part of the model is used as the proxy. Taking convolutional neural network as an example, for a layer with kernel $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{c_i \times c_o \times u \times u}$, where c_i, c_o and u are number of input filters, number of output filters and kernel size respectively, the corresponding kernel of proxy is then: $\boldsymbol{\mathcal{W}}_{\text{proxy}} = \boldsymbol{\mathcal{W}}_{1:pc_i, 1:pc_o, :, :}$, where $p \in [0, 1]$ is a slimming factor. As shown in Figure 3, the proxy kernel is constructed with the first pc_i input channels and first pc_o output channels. A p times thinner proxy can be obtained by applying p to each layer.

When separate batch normalization are used for proxy and model, PSP forms a slimmable network (Yu et al., 2019), where multiple shared models of different width are jointly trained and they all yield good performance. As the parameters are completely shared, the proxy can acutely keep up with the model change, thus well satisfy the requirements of dynamic selection. We further investigate the gradients alignment of the proxy and the original model through their cosine similarity:

$$\cos(\boldsymbol{g}, \boldsymbol{g}_{\text{proxy}}) = \frac{\boldsymbol{g}^\top \boldsymbol{g}_{\text{proxy}}}{\|\boldsymbol{g}\|_2 \cdot \|\boldsymbol{g}\|_2}, \text{ where } \boldsymbol{g} = \nabla_{\boldsymbol{\mathcal{W}}} \mathcal{L}\left(\boldsymbol{\mathcal{W}}\right), \boldsymbol{g}_{\text{proxy}} = \nabla_{\boldsymbol{\mathcal{W}}} \mathcal{L}\left(\boldsymbol{\mathcal{W}}_{\text{proxy}}\right) \tag{3}$$

A positive cosine value indicates $\boldsymbol{g}_{\text{proxy}}$ stands in the same side with $\boldsymbol{g}$ thus update on the proxy is also a descent direction for the original model. We compare the gradient alignment of PSP and a stand-alone proxy along training in Figure 2(c) on a 50-layer residual network. With $p = 0.5$, we see that $\cos(\boldsymbol{g}, \boldsymbol{g}_{\text{proxy}})$ for PSP is much larger than the stand-alone proxy. With the well-aligned gradients, update on proxy and the model benefits each other. PSP thus requires fewer training epochs. Overall workflows of DynaMS and DynaMS+PSP is shown in Algorithm 1 and Algorithm 2 of Appendix A.

3 Related Work

Accelerating training by eliminating redundant training instances has long been a research focus in academia. This is accomplished by adopting an effective selection strategy and an appropriate training scheme. We summarize the related literature from these two strands of research in the following.

Selection Strategy Selecting samples for efficient learning can be accomplished with various principles. (Loshchilov & Hutter, 2015; Jiang et al., 2019; Paul et al., 2021) believes informative samples are "harder" and are able to largely change the model, so samples that incur large loss or gradient norm are picked. (Toneva et al., 2019) showed that there exist "unforgettable" examples that are rarely misclassified once learned, these instances can be omitted without much performance degradation. Other works adopt uncertain sampling. Samples that are predicted the least confidence are preferred (Settles, 2010). Recently, (Mirzasoleiman et al., 2020; Killamsetty et al., 2021) propose to select subsets that best approximate the full gradient. However, this requires per-sample gradient computation as well as an additional optimization which is expensive both in run-time and in memory efficiency ($\mathcal{O}(|\mathcal{T}|\cdot C\cdot d)$ space complexity[4]). Our work utilizes the class margin to identify informative samples, which is efficient to compute and can synergistically adapt to various training schemes.

Another branch of research related is importance sampling (Alain et al., 2015; Needell et al., 2014). Samples are sampled with probability proportional to their corresponding per-sample gradient norm, which is proved to be able to reduce the variance caused by stochastic training. Instead of accelerating by explicitly eliminating redundant samples, importance sampling methods improves the convergence of optimization. The importance weights are supposed to be updated in each iteration, which tends to introduce large computational overhead.

Training Schemes The introduction of data selection brings more options to model training. Conventional methods adopt a static training scheme (Paul et al., 2021; Toneva et al., 2019; Coleman et al., 2020). Under this paradigm, data selection is conducted prior to model update, and the informative subset is kept fixed during the subsequent model training. Contrastive to the static training scheme, online batch selection picks batch data each iteration in training (Loshchilov & Hutter, 2015; Alain et al., 2015; Zhang et al., 2019; Mindermann et al., 2022). Though sufficiently considered the training dynamics, the overly frequent evaluation and selection incurs prohibitive computational overhead, making them impractical for large datasets. Recently, (Killamsetty et al., 2021) tried selecting after several epochs' training, which is similar to our dynamic training scheme. However, the dynamic training scheme is just utilized as a compromise to avoid overly frequent data selection, as the selection strategy used in (Killamsetty et al., 2021) is expensive both in run-time and in memory efficiency. Formal analyses of its advantage over static scheme is absent.

By systematically considering the sample evaluation strategy, the model training, as well as proxy suitable for dynamic selection, Our proposed DynaMS form an effective and interpretable data selection framework for efficient training.

4 Experiments

In this section, we first analyse the effectiveness of each design ingredient in Section 4.2. Then we compare to state-of-the-art algorithms in Section 4.3. Codes will be released upon publication.

4.1 Experimental Setup

We conduct experiments on CIFAR-10 Krizhevsky & Hinton (2009) and ImageNet 2012 Jia et al. (2009), following standard data pre-processing techniques in He et al. (2016). A brief summarization of the experimental setup is introduced below, while complete hyper-parameter settings and implementation details can be found in Appendix E.

CIFAR-10 Experiments For CIFAR-10, we take ResNet-18 (He et al., 2016) as base models similar to (Toneva et al., 2019; Zhang et al., 2019). The total training epoch is set to 200. We conduct selection every 10 epochs, so overall the selection is performed 19 times ($K = 19$). For subset size, we adopt a simple linear schedule where $\gamma_k = 1 - k \cdot a$ for $k = 1, 2, \ldots, K$, where γ determines the sample reduction ratio. γ is determined by making sure $\gamma_{\text{avg}} = \frac{1}{K}\sum_{k=1}^{K}\gamma_k = \gamma_s$ where γ_s is the selection ratio when static training scheme is applied. In this way the averaged number of data used in the dynamic scheme is kept equal to that of static training for fair comparison. For $0.6\times$ acceleration, we set $a = 0.042$. We conduct experiments on a single NVIDIA Ampere A-100.

[4]Approximating the very high dimensional gradients for DNNs is prohibitively expensive. In practice, only gradient of the last layer is considered.

ImageNet Experiments For ImageNet experiments, we choose ResNet-18 and ResNet-50 as base models. Following the conventions, the total training epoch is set to 120. Selection is also conducted every 10 epochs, so altogether $K = 11$. For subset size, aside from the linear schedule, we also explore a power schedule where γ_k decays following a power law. $\frac{1}{K}\sum_{k=1}^{K}\gamma_k = \gamma_s$ is also ensured for fair comparison. Please refer to Appendix E for more details on determining the schedule. The power schedule reserves more samples in late training, preventing performance degradation caused by over data pruning. We conduct experiments on four NVIDIA Ampere A-100s.

4.2 Ablation Studies

Table 1: Comparison of different data selection strategies. Except for DynaMS, all the other methods conduct training in the conventional static scheme.

Strategy	Top1 Acc.	Top5 Acc.
CE-loss	72.073	91.727
EL2N	72.032	91.778
MS	72.888	91.807
DynaMS	**74.558**	**92.334**

Table 2: Accuracy of utilizing parameter sharing proxy (PSP) with different width configurations. FLOPs↓ is the reduction of FLOPs required when use proxy for sample evaluation.

Width	Top1 Acc.	Proxy Top1 Acc.	FLOPs ↓
1.00×	74.558	-	-
0.75×	73.694	72.882	43.75%
0.50×	73.401	70.720	75.00%
0.25×	73.390	62.349	93.75%

We use ResNet-50 on ImageNet to illustrate the effect of each ingredient in DynaMS, that is, the classification margin criteria, the dynamic training scheme as well as the parameter sharing proxy.

The effect of classification margin selection To inspect the effect of classification margin selection (MS), we compare MS against two widely applied selection strategies **CE-loss** (Loshchilov & Hutter, 2015; Jiang et al., 2019) and **EL2N** (Paul et al., 2021). CE-loss selects samples explicitly through the cross-entropy loss they incur while EL2N picks samples that incur large L2 error. We compare the three with the conventional static scheme so any other factors aside from the selection strategy is excluded. Samples are evaluated after 20 epochs of pretraining. The model is then reinitialized and trained on the selected subset, which contains 60% original samples. As shown in Table 1, MS achieves the best accuracy among the three, validating its effectiveness.

The effect of dynamic training We then apply dynamic selection on MS, where the average subset size is also kept to be 60% of the original dataset. From Table 1 we see that DynaMS outperforms MS by 1.67%, which is significant on large scale dataset like ImageNet. The superiority of DynaMS validates that by constantly improving the model and updating the subset, dynamic selection scheme can result in better performance. Note that DynaMS can be more practical since it does not require the 20 epochs training prior to selection as that in the static scheme.

The effect of parameter sharing proxy We now study the parameter sharing proxy (PSP). An effective proxy is supposed to be faithful to the underlying model, and can agilely adapt to the model update. In Figure 4, we plot the Spearman rank correlation as well as the overlap ratio of selected samples between the proxy and the original model. We see that all along the training, the proxy and the model are fairly similar with rank correlations around 0.68, and over 78% samples selected are the same, indicating that the proxy and the model are fairly consistent. We then investigate how will the complexity of proxy affect dynamic selection. The complexity of the proxy is measured by floating point operations (FLOPs). We enumerate over the slimming factor $p \in \{0.25, 0.5, 0.75, 1.0\}$ to construct proxies of different widths, the corresponding FLOPs of the proxies are $6.25\%, 25.00\%, 56.25\%, 100\%$ respectively. We see that significant computation reduction can be achieved with a thinner proxy with moderate performance degradation.

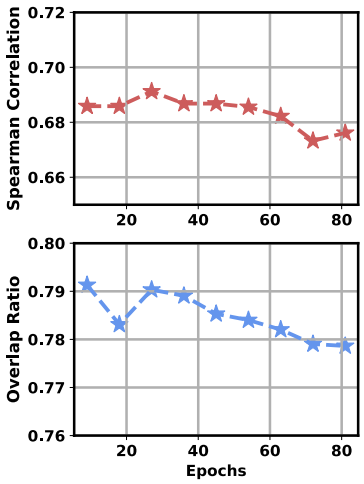
4.3 Comparisons with state-of-the-arts

Finally, we compare the proposed DynaMS against various state-of-the-art methods ranging from static training to dynamic training. Aside from **CE-loss** and **EL2N**, **Random** picks samples uniformly at random. **GraNd** (Paul et al., 2021) select samples that incur large gradient norm. Samples are also evaluated after 20 epochs training. **Forget** (Toneva et al., 2019) counts how many times a sample is mis-classified (forget) after it is learned. Samples more frequently forgotten are preferred. We

Table 3: Comparison for ResNet-18 on Cifar-10.

Methods	Types	Budget	Schedule	Accuracy.
Original	-	100%	-	$95.52_{\pm 0.09}$
Random	Stat.	60%	-	$94.09_{\pm 0.23}$
EL2N_1	Stat.	60%	-	$94.55_{\pm 0.15}$
EL2N_{10}	Stat.	60%	-	$95.34_{\pm 0.13}$
GraNd_{10}	Stat.	60%	-	$95.21_{\pm 0.15}$
Forget_{10}	Stat.	60%	-	$95.29_{\pm 0.12}$
OnlineMS	OLBS.	60%	Const.	$95.21_{\pm 0.18}$
Auto-assist	OLBS.	60%	Const.	$92.37_{\pm 0.24}$
DynaRandom	Dyna.	60%	Linear	$94.45_{\pm 0.17}$
DynaCE	Dyna.	60%	Linear	$94.96_{\pm 0.21}$
Craig	Dyna.	60%	Const.	$94.36_{\pm 0.19}$
GradMatch	Dyna.	60%	Const.	$94.84_{\pm 0.17}$
DynaMS	Dyna.	60%	Linear	$95.28_{\pm 0.15}$

Figure 4: Correlation of proxy and model.



evaluate the forget score after 60 epochs training. To avoid noisy evaluation, many of these static selection approaches ensembles networks before selection. The number of ensambled models is given by the subscription. **Auto-assist** (Zhang et al., 2019) select samples that incur large loss value with a small proxy. Selection is conducted in each iteration thus forming an online batch selection (OLBS) scheme. **DynaCE** and **DynaRandom** apply the corresponding selection strategy, but are trained in a dynamic way. **CRAIG** and **GradMatch** propose to reweight and select subsets so that they best approximate the full gradient. In the experiments, we use the per-batch variant of CRAIG and GradMatch proposed in (Killamsetty et al., 2021) with 10 epoch warm start[5]. The two approaches utilize dynamic selection scheme, all the training settings are kept the same as our DynaMS.

In table 3, average (± 1 std) accuracy from 5 runs of sample selection on CIFAR-10 are reported. We see that DynaMS achieves comparable performance against the strongest baselines. CIFAR-10 is relatively simple so many approaches can achieve performance similar to full training. Note that all the static methods aside from Random require pretraining one or several models for 20 epochs before subset selection, the overall acceleration is less significant. We also compare two online batch selection methods, OnlineMS and Auto-assist (Zhang et al., 2019). OnlineMS picks samples according to the classification margin as DynaMS, but the selection is conducted each iteration. OnlineMS didn't outperform DynaMS, meaning more frequent selection is not necessary. Rather, selecting at each optimization step incurs prohibitive computational overhead. Auto-assist didn't get good performance in this experiment. This may result from the overly simple proxy. The logistic regression proxy adopted may not sufficiently evaluate the candidate samples.

For ImageNet, DynaMS outperforms all the baselines. For instance, it achieves 68.65% and 74.56% top-1 accuracy given on average 60% samples for ResNet-18 and ResNet-50 respectively, surpassing the most competitive counterpart Forget by 0.81% and 1.06%. Note that Forget requires pretraining the model for 60 epochs before subset selection, DynaMS is much more efficient. CRAIG and GradMatch didn't get good performance on ImageNet. We hypothesize this is because we use the per-batch variant proposed in (Killamsetty et al., 2021) with batch size 512 in order to fit the per-sample gradients into memory. The per-batch variant treats each mini-batch as one sample and selects mini-batches during the gradient matching process. So a larger batch size means more coarse grain selection which may lead to inferior performance. We also compare a variant DynaRandom. DynaRandom adopts the dynamic selection scheme but a random subset is constructed at each selection. DynaMS outperforms DynaRandom by 1.06% and 1.93% for ResNet-18 and ResNet-50 respectively, indicating that the superiority of DynaMS over static methods comes from effectively identifying informative samples instead of witnessing more data.

[5]We use the published implementation from https: //github.com/decile-team/cords

Table 4: Comparison for ResNet-18 and ResNet-50 on ImageNet

	Types	Budget	Schedule	ResNet 18		ResNet 50	
				Top1 Acc.	Top5 Acc.	Top1 Acc.	Top5 Acc.
Original	-	100%	-	$70.56_{\pm 0.04}$	$89.95_{\pm 0.02}$	$75.96_{\pm 0.04}$	$92.75_{\pm 0.02}$
Random	Stat.	60%	-	$67.16_{\pm 0.10}$	$87.50_{\pm 0.07}$	$72.46_{\pm 0.10}$	$90.85_{\pm 0.04}$
EL2N_1	Stat.	60%	-	$66.38_{\pm 0.09}$	$88.56_{\pm 0.05}$	$72.03_{\pm 0.12}$	$91.78_{\pm 0.04}$
EL2N_{10}	Stat.	60%	-	$66.46_{\pm 0.10}$	$88.73_{\pm 0.04}$	$72.18_{\pm 0.10}$	$92.02_{\pm 0.06}$
GraNd_{10}	Stat.	60%	-	$66.50_{\pm 0.06}$	$88.76_{\pm 0.03}$	$72.14_{\pm 0.06}$	$92.16_{\pm 0.03}$
Forget_{10}	Stat.	60%	-	$67.84_{\pm 0.08}$	$87.50_{\pm 0.03}$	$73.50_{\pm 0.06}$	$91.41_{\pm 0.04}$
SVP+Forget	Stat.	60%	-	-	-	$72.90_{\pm 0.10}$	$91.37_{\pm 0.04}$
SVP+Entropy	Stat.	60%	-	-	-	$73.00_{\pm 0.01}$	$91.52_{\pm 0.01}$
DynaRandom	Dyna.	60%	Power	$67.59_{\pm 0.05}$	$87.62_{\pm 0.03}$	$72.63_{\pm 0.12}$	$90.91_{\pm 0.08}$
DynaCE	Dyna.	60%	Power	$67.58_{\pm 0.10}$	$88.10_{\pm 0.04}$	$72.80_{\pm 0.08}$	$91.31_{\pm 0.03}$
Craig	Dyna.	60%	Const.	$65.32_{\pm 0.08}$	$86.92_{\pm 0.04}$	$70.69_{\pm 0.07}$	$90.72_{\pm 0.02}$
GradMatch	Dyna.	60%	Const.	$66.48_{\pm 0.11}$	$88.61_{\pm 0.04}$	$71.79_{\pm 0.07}$	$91.67_{\pm 0.03}$
DynaMS	Dyna.	60%	Linear	$68.12_{\pm 0.13}$	$88.93_{\pm 0.06}$	$74.10_{\pm 0.09}$	$92.25_{\pm 0.03}$
DynaMS	Dyna.	60%	Power	$\mathbf{68.65}_{\pm \mathbf{0.11}}$	$\mathbf{89.21}_{\pm \mathbf{0.04}}$	$\mathbf{74.56}_{\pm \mathbf{0.09}}$	$\mathbf{92.33}_{\pm \mathbf{0.03}}$
DynaMS+PSP	Dyna.	60%	Linear	-	-	$73.59_{\pm 0.09}$	$91.79_{\pm 0.07}$
DynaMS+PSP	Dyna.	60%	Power	-	-	$73.40_{\pm 0.08}$	$91.80_{\pm 0.01}$

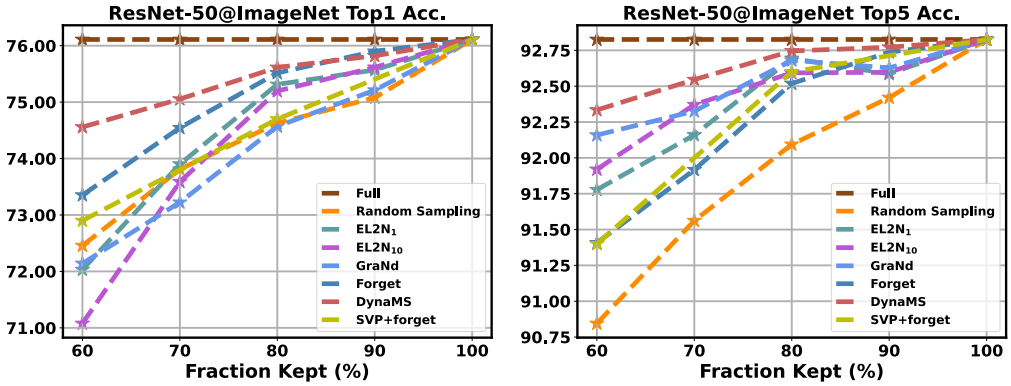


Figure 5: Comparison under different (on average) sample budgets.

ResNet-50 is a rather complex model and the data evaluation time is non-negligible. We thus apply parameter sharing proxy to improve the efficiency. The proxy is $0.5\times$ width so the evaluation requires $0.25\times$ computation compared to the original model. As the gradients of the proxy and the underlying model are well aligned, we only train DynaMS+PSP for 90 epochs. From table 4 we see that though utilizing a proxy harms the model performance compared to DynaMS, it still outperforms all the other baselines. To further demonstrate DynaMS, we draw the accuracy curvature of ResNet-50 against different (on average) sample budgets from 60% to 100% in Figure 5. It can be found that our DynaMS consistently outperforms all the other data selection strategies on different budgets.

5 Conclusion

In this paper, we propose DynaMS, a general dynamic data selection framework for efficient deep neural network training. DynaMS prefers samples that are close to the classification boundary and the selected "informative" subset is dynamically updated during the model training. DynaMS enjoys rigorous convergence and we pioneer to show both in practice and theory that dynamic selection improves the generalization over previous approaches. Considering the additional computation incurred by selection, we further design a proxy available for dynamic selection. Extensive experiments and analysis are conducted to demonstrate the effectiveness of our strategy.

REFERENCES

Guillaume Alain, Alex Lamb, Chinnadhurai Sankar, Aaron Courville, and Yoshua Bengio. Variance reduction in sgd by distributed importance sampling. *arXiv preprint*, arXiv:1511.06481, 2015.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Mark Chen, Alec Radford, Rewon Child, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, 2020.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020.

Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, June 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Zeyad Ali Sami Emam, Hong-Min Chu, Ping-Yeh Chiang, Wojciech Czaja, Richard Leapman, Micah Goldblum, and Tom Goldstein. Active learning at the imagenet scale. *International Conference on Computer Vision and Pattern Recognition*, 2021.

Dan Feldman. Introduction to core-sets: an updated survey. *arXiv preprint*, arXiv:2011.09384, 2020.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,*, pp. 770–778, 2016.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory F. Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint*, arXiv:1712.00409, 2017.

Deng Jia, Dong Wei, Socher Richard, Li Li-Jia, Li Kai, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248—-255, 2009.

Angela H. Jiang, Daniel L.-K. Wong, Giulio Zhou, David G. Andersen, Jeffrey Dean, Gregory R. Ganger, Gauri Joshi, Michael Kaminsky, Michael Kozuch, Zachary C. Lipton, and Padmanabhan Pillai. Accelerating deep learning by focusing on the biggest losers. *aXiv preprint*, arXiv:1910.00762, 2019.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint*, arXiv:2001.08361, 2020.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning*, 2018.

Krishnatejam Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, 2021.

Alex Krizhevsky and Geffery Hinton. Learning multiple layers of features from tiny images. In *Technical report*, 2009.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint*, arXiv:1511.06343, 2015.

David Mickisch, Felix Assion, Florens Greßner, Wiebke Günther, and Mariele Motta. Understanding the decision boundary of deep neural networks: An empirical study. *arXiv preprint*, arXiv:2002.01810, 2020.

Sören Mindermann, Jan Brauner, Muhammed Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. Prioritized training on points that are learnable, worth learning, and not yet learnt. *arXiv preprint*, arXiv:2206.07137, 2022.

Baharan Mirzasoleiman, Jeff A. Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning, ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6950–6960. PMLR, 2020.

Deanna Needell, Rachel Ward, and Nati Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20596–20607. Curran Associates, Inc., 2021.

Burr Settles. From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop in AISTATS*, 2010.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *arXiv preprint*, arXiv:2206.14486, 2022.

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 of Representation Learning*, 2019.

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. In *International Conference on Machine Learning*, 2018.

Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019.

Jiong Zhang, Hsiang-fu Yu, and Inderjit S Dhillon. Autoassist: A framework to accelerate training of deep neural networks. In *Conference on Neural Information Processing Systems*, 2019.