Selection Via Proxy: Efficient Data Selection For Deep Learning*

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, Matei Zaharia
Stanford University

Abstract

Data selection methods such as active learning and core-set selection are useful tools for machine learning on large datasets, but they can be prohibitively expensive to apply in deep learning. Unlike in other areas of machine learning, the feature representations that these techniques depend on are learned in deep learning rather than given, which takes a substantial amount of training time. In this work, we show that we can significantly improve the computational efficiency of data selection in deep learning by using a much smaller *proxy* model to perform data selection for tasks that will eventually require a large target model (e.g., selecting data points to label for active learning). In deep learning, we can scale down models by removing hidden layers or reducing their dimension to create proxies that are an order of magnitude faster. Although these small proxy models have significantly higher error, we find that they empirically provide useful rankings for data selection that have a high correlation with those of larger models. We evaluate this "selection via proxy" (SVP) approach on several data selection tasks. For active learning, applying SVP to Sener and Savarese [2018]'s recent method for active learning in deep learning gives a $4\times$ improvement in execution time while yielding the same model accuracy. For core-set selection, we show that a proxy model that trains $10\times$ faster than a target ResNet164 model on CIFAR10 can be used to remove 50% of the training data without compromising the accuracy of the target model, making end-to-end training time improvements via core-set selection possible.

1 Introduction

Data selection methods such as active learning and core-set selection are often useful tools for managing machine learning on large datasets. Generally speaking, active learning starts with a small amount of labeled data and selects points to label from a much larger pool of unlabeled data [Settles, 2012, Sener and Savarese, 2018, Lewis and Gale, 1994, Lewis and Catlett, 1994, Settles, 2011]. Through an iterative process, a model is repeatedly trained on the labeled data, and points are selected from the unlabeled pool based on the model's uncertainty or other heuristics. Conversely, core-set selection techniques start with a large labeled or unlabeled dataset and aim to find a small subset that accurately approximates the full dataset for a given task [Har-Peled and Kushal, 2007, Tsang et al., 2005, Huggins et al., 2016, Campbell and Broderick, 2017, 2018]. Each example is selected based on its representativeness or coverage of other points in the input feature space. By identifying these important examples through a model's uncertainty or the input feature representation, active learning and core-set selection techniques improve *data efficiency* or save time in downstream tasks (e.g., training or summarization) by ignoring redundant examples.

Unfortunately, classical data selection methods are often prohibitively expensive to apply in deep learning [Sener and Savarese, 2018]. Unlike other machine learning methods, deep learning models

*updated work from Coleman et al. [2019]

Preprint. Under review.

learn complex internal semantic representations (hidden layers) from raw inputs (e.g., pixels or characters) that enable them to achieve state-of-the-art performance. As a result, much of the computation in training deep learning models is devoted to learning this representation. Unfortunately, many core-set selection and active learning techniques require this feature representation *before* they accurately identify important points. For example, classical active learning methods that label one new data point per iteration based on a model of the previous labeled points would require training a new deep learning model after each data point, which is computationally intractable. Recent active learning work by Sener and Savarese [2018] has proposed methods to request data in large batches, but even this approach requires training a full deep model for every batch.

In this paper, we propose *selection via proxy (SVP)* to make data selection methods for deep learning more computationally efficient. SVP uses the feature representation from a separate, less computationally intensive model as a proxy for the much larger and more accurate target model we aim to train. SVP builds on the idea of heterogeneous uncertainty sampling from Lewis and Catlett [1994], which showed that an inexpensive classifier (e.g., naïve Bayes) can select points to label for a much more computationally expensive classifier (e.g., decision tree). In our work, we show that small deep learning models can similarly serve as an inexpensive proxy for data selection in deep learning, significantly accelerating active learning and core-set selection techniques. To create these cheap proxy models, we can scale down deep learning models by removing layers, reducing their hidden dimensions, or training them for fewer epochs. While these scaled down models achieve significantly lower accuracy than larger models, we empirically find that they still provide useful representations to rank and select points (i.e., high Spearman's and Pearson's correlations with much larger models on metrics such as uncertainty [Settles, 2012], facility location [Wolf, 2011], and forgettability [Toneva et al., 2019]). Because these proxy models are quick to train and apply (often $10\times$ faster), we can identify which points to select nearly as well as the larger target model but significantly faster.

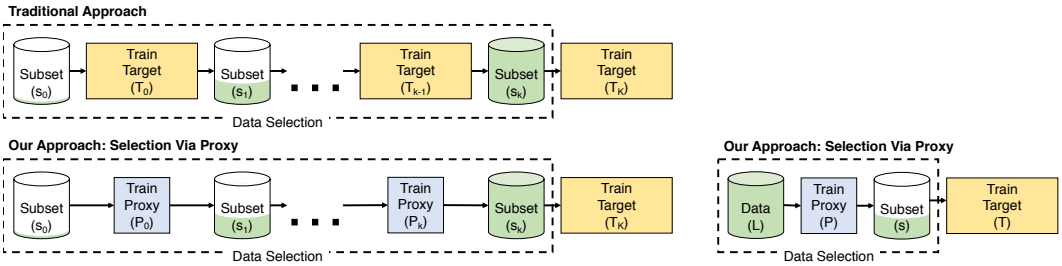
We evaluate SVP using several data selection tasks. For active learning, we extend the recent method by Sener and Savarese [2018]. Augmenting this method with SVP yields up to a $4\times$ speed-up in the data selection process on CIFAR10 and CIFAR100 without reducing accuracy or data efficiency after each round. For core-set selection, we try three methods to identify a subset of points: uncertainty sampling with entropy [Lewis and Gale, 1994, Settles, 2012], facility location [Wolf, 2011], and forgetting events [Toneva et al., 2019]. For each method, we find that smaller proxy models have high ranking correlations with $10\times$ larger models, and perform as well as these large models at identifying subsets of points to train on that yield high accuracy. Thus, core-set selection with SVP could practically be used to reduce the size of large datasets before performing training in domains where data is abundant. To illustrate, we show that SVP lets us remove up to 50% of the data in CIFAR10 without impacting the accuracy of a ResNet164 model trained on it, using a $10\times$ faster model for the selection. This yields an end-to-end training time improvement of about $1.6\times$ for the final ResNet164 (including the time to train and use the proxy). These results demonstrate that SVP is a promising approach to make data selection methods computationally feasible for deep learning.

2 Methods

In this section, we describe SVP and show how it can be incorporated in active learning and core-set selection. Figure 1 shows an overview of SVP in these two contexts: in active learning, we retrain a proxy model P_k in place of the target model T_k after each batch is selected, and in core-set selection, we train the proxy model P rather than the target T over all the data to learn a feature representation and select points. We next describe the specific active learning and core-set techniques that we used in this paper, and how we extended them with SVP.

2.1 Active Learning

Pool based active learning starts with a large pool of unlabeled data $U = \{\mathbf{x}_i\}_{i\in\{1,\dots,n\}}$ from a space $\mathcal{X}$ where each example has an unknown label from a label space $\mathcal{Y}$ and are sampled *i.i.d.* over the space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ as $\{\mathbf{x}_i, y_i\} \sim p_{\mathcal{Z}}$. Initially, methods label a small pool of points $s^0 = \{s^0_j \in \{1,\dots,n\}\}_{j\in\{1,\dots,m\}}$ chosen uniformly at random. Given U, a loss function ℓ, and the labels $\{y_{s^0_j}\}_{j\in\{1,\dots,m\}}$ for the initial random subset, the goal of active learning is to select up to a budget of b points from U to label that will minimize the generalization error of a learning algorithm A_s (i.e., $\min_{s^1:|s^1|\leq b} E_{\mathbf{x},y\sim p_{\mathcal{Z}}}[\ell(\mathbf{x}, y; A_{s_0\cup s_1})]$).



(a) Active Learning (b) Core-Set Selection

Figure 1: **SVP applied to active learning** (left) **and core-set selection** (right). In active learning, we follow the same iterative procedure of training and selecting points to label as traditional approaches but replace the target model with a cheaper to compute proxy model. For core-set selection, we learn a feature representation over the data using a proxy model and use it to select points to train a larger, more accurate model. In both cases, we find the proxy and target model have high rank-order correlation, leading to similar selections and downstream results.

Baseline. In this paper, we extend the algorithm of Sener and Savarese [2018]. Like that work, we consider a batch setting with K rounds where we select $\frac{b}{K}$ points in every round aside from the first round where we select $(\frac{b}{K} - m)$ points. Sener and Savarese [2018] select each batch of points to label using the minimax facility location method from Wolf [2011], as shown in Algorithm 1. For each round k of data selection, Sener and Savarese [2018] retrain a target model T_k from scratch on all of the labeled data collected over previous rounds, $s_0 \cup \ldots \cup s_{k-1}$, extract a feature representation from the model's final hidden layer, and then compute the distance between examples (i.e., $\Delta(\mathbf{x_i}, \mathbf{x_j}; T_k)$) to select points. The same model is trained on the final b labeled points to yield the final model, T_K, which is then tested on a held-out set to evaluate test error and quantify the quality of the selected data.

Algorithm 1 FACILITY LOCATION

Input: data $\mathbf{x}_i$, existing pool s^0, trained model T_0, and a budget b

1: Initialize $s = s^0$
2: **repeat**
3: $\quad u = \arg\max_{i \in \{1,\ldots,n\} \setminus s} \min_{j \in s} \Delta(\mathbf{x_i}, \mathbf{x_j}; T_0)$
4: $\quad s = s \cup \{u\}$
5: **until** $|s| = b + |s^0|$
6: **return** $s \setminus s^0$

2.2 Core-Set Selection

Core-set selection can be broadly defined as techniques that find a subset of data points that maintain a similar level of quality (e.g., generalization error of a trained model or minimum enclosing ball) as the full dataset. Specifically, we start with a labeled dataset $L = \{\mathbf{x}_i, y_i\}_{i \in \{1,\ldots,n\}}$ sampled *i.i.d.* from $\mathcal{Z}$ with $p_{\mathcal{Z}}$ and want to find a subset of $m \leq n$ points $s = \{s_j \in \{1, \ldots, n\}\}_{j \in \{1,\ldots,m\}}$ that achieves comparable quality in terms of loss to the full dataset:

$$\min_{s:|s|=m} E_{\mathbf{x},y \sim p_{\mathcal{Z}}} \left[\ell(\mathbf{x}, y; A_s)\right] - E_{\mathbf{x},y \sim p_{\mathcal{Z}}} \left[\ell(\mathbf{x}, y; A_L)\right]$$

Baseline. To find s for a given m, we implement three core-set selection techniques: facility location [Wolf, 2011, Sener and Savarese, 2018], forgetting events [Toneva et al., 2019], and uncertainty sampling with entropy [Lewis and Gale, 1994, Settles, 2012]. Facility location is described above and in Algorithm 1. Forgetting events are defined as the number of times an example is incorrectly classified after having been correctly classified earlier during training a model T. To select points, we follow the same procedure as Toneva et al. [2019]: we keep the points with the m highest number of forgetting events. Points that are never correctly classified are treated as having an infinite number of forgetting events. Similarly, we rank examples based on the entropy from a trained model T and keep the m with the highest entropy. To evaluate core-set quality, we compare the performance of training the large target model T on the selected subset compared to training on the entire dataset, by measuring error on a held-out test set.

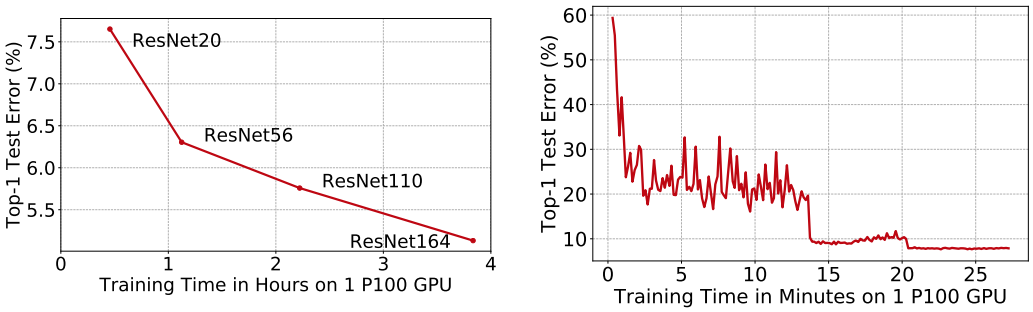


Figure 2: **Diminishing returns for accuracy in model size and training time.** For ResNet models with pre-activation on CIFAR10, we observe diminishing returns for test error as the number of layers increases (left) and as training time increases (right). Notably, during training, ResNet20 reaches 9.0% error in 14 minutes, while the remaining 12 minutes are spent on decreasing error to 7.6%.

2.3 Applying Selection Via Proxy

In general, SVP can be applied by replacing the models used to compute data selection metrics such as uncertainty with proxy models. In this paper, we applied SVP to the active learning and core-set selection methods described in Sections 2.1 and 2.2 as follows:

- For active learning using Sener and Savarese [2018], we replaced the model trained at each batch (T_k) with a proxy (P_k), but then trained the same final model T_K once the budget b was reached to evaluate the quality of the data selection.
- For core-set selection, we used a proxy model to compute facility location, entropy and forgetting event metrics and select our data subsets.

We explored two main methods to create our proxy models:

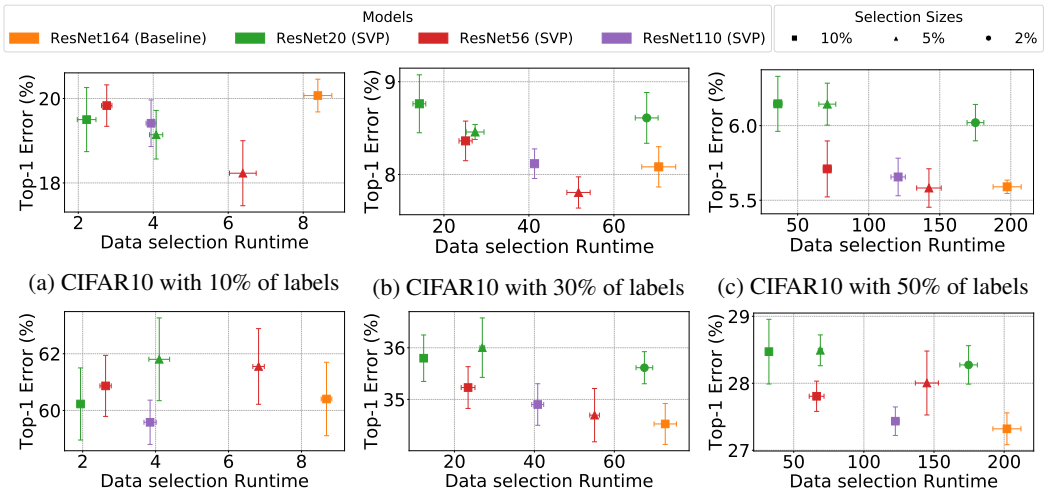
Creating a proxy by scaling down the target model. For deep models with many layers, reducing the dimension (narrowing) or the number of hidden layers (shortening) reduces training times considerably with only a small drop in accuracy. For example, in image classification, the accuracy of deep ResNet models only slightly diminishes as layers are dropped from the network [He et al., 2016b,a]. As Figure 2 shows, a ResNet20 model with 20 layers achieves a top-1 error of 7.6% in 26 minutes, while a larger ResNet164 model with 164 layers only reduces error by 2.5%, but takes 3 hours and 50 minutes to train.

Similar results have been shown for scaling down networks with a variety of model architectures [Huang et al., 2016, Xie et al., 2017, Huang et al., 2017] and many other tasks including language modeling, neural machine translation, text classification, and recommendation [Conneau et al., 2016, He et al., 2017, Jozefowicz et al., 2016, Dauphin et al., 2017, Vaswani et al., 2017]. We exploit the diminishing returns property between training time and reductions in error to scale down a given target model to a small proxy that can be trained quickly but still provides a good approximation of the decision boundary of the target model.

Training for a smaller number of epochs. As shown in Figure 2, a significant amount of training is spent on a relatively small reduction in error. While training ResNet20, almost half of the training time (i.e., 12 minutes out of 26 minutes) is spent on a 1.4% improvement in test error. Based on this observation, we also explored training proxy models for a smaller number of epochs to get good approximations of the decision boundary of the target model even faster.

3 Results

To demonstrate the effectiveness of SVP, we first apply SVP to active learning using methods from Sener and Savarese [2018] in Section 3.1. We find that across labeling budgets SVP achieves similar or higher accuracy and up to a $4\times$ improvement in data selection runtime (i.e., the time it takes to repeatedly train and select points, as shown in Figure 1). Next, we apply SVP to the core-set selection problem described in Section 3.2. For all selection methods, the target model performs nearly as well as or better with SVP than the oracle baseline that trains the target model on all of the data before



(a) CIFAR10 with 10% of labels (b) CIFAR10 with 30% of labels (c) CIFAR10 with 50% of labels

(d) CIFAR100 with 10% of labels (e) CIFAR100 with 30% of labels (f) CIFAR100 with 50% of labels

Figure 3: **SVP performance on active learning.** Average ($\pm$ 1 std.) top-1 test error for ResNet164 versus runtime in minutes of data selection for 5 runs of active learning with varying budgets, proxies, and selection sizes on CIFAR10 (top) and CIFAR100 (bottom). The orange marker represents the baseline performance of using ResNet164 for both data selection and the final task. Across datasets and labeling budgets, SVP achieves similar accuracy and up to a $4\times$ improvement in runtime.

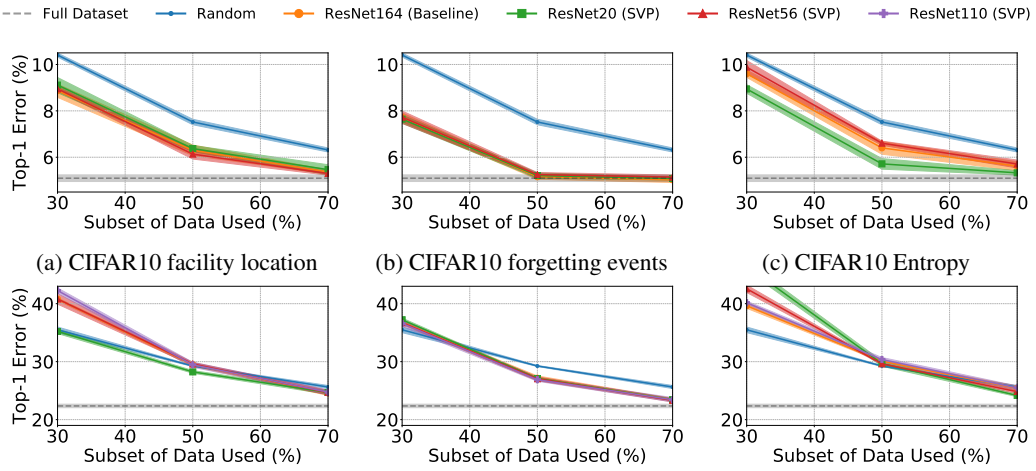
selecting examples. The proxy trains in as little as 7 minutes compared to the 3 hours 50 minutes the target model takes to train, making SVP feasible for end-to-end training time speed-ups. Finally, we illustrate why proxy models perform so well by evaluating how varying depths of ResNet models and three different methods rank examples (see Section 3.3). On both datasets, the correlation across model architectures is nearly as high as between runs of the same architectures, indicating that proxy models provide as good of a ranking for data selection as the target model.

Datasets. To investigate the performance of SVP, we perform experiments on two image classification datasets: CIFAR10 and CIFAR100 [Krizhevsky and Hinton, 2009]. CIFAR10 is a coarse-grained classification task over 10 classes, and CIFAR100 is a fine-grained task with 100 classes. Both datasets contain 50,000 32×32 color images for training and 10,000 32×32 images for testing.

Implementation details. We used ResNet164 with pre-activation from He et al. [2016b] as our large target model for both CIFAR10 and CIFAR100. The smaller, proxy models are also ResNet architectures with pre-activation, but they use pairs of 3×3 convolutional layers as their residual unit rather than bottlenecks as originally proposed in He et al. [2016a] and achieve lower accuracy as shown in Figure 2. We followed the same training procedure, initialization, and hyperparameters as He et al. [2016b] with the exception of weight decay, which was set to 0.0005 and decreased the model's validation error in all conditions. Throughout this section, we report the mean error and standard deviation of at least 5 runs for each experiment.

3.1 Active Learning

We explored the impact of several types of proxy models on the active learning technique in Sener and Savarese [2018], where the target model was configured to be ResNet164. As shown in Figure 3 and Table 2 in the supplementary material, significantly cheaper proxies lead to similar final model accuracy across a range of data labeling budgets. We varied both the size of the model and the number of selection rounds (% of data selected in each round), because the proxy models are so much faster to train that one can afford to run more rounds while still finishing faster than the original method. Across datasets and labeling budgets, we find that SVP can achieve a similar accuracy with up to a $4\times$ improvement in data selection runtime (i.e., the time it takes to repeatedly train and select points up to the given budget size) compared to the baseline method. Small budgets show the best speedups. This happens because in addition to the proxy being faster to train, the dimension of the final hidden



(a) CIFAR10 facility location (b) CIFAR10 forgetting events (c) CIFAR10 Entropy

(d) CIFAR100 facility location (e) CIFAR100 forgetting events (f) CIFAR100 entropy

Figure 4: **SVP performance on core-set selection.** Average ($\pm$ 1 std.) top-1 error of ResNet164 over 5 runs of core-set selection with different selection methods, proxies, and subset sizes on CIFAR10 (top) and CIFAR100 (bottom). We find subsets using facility location (left), the number of forgetting events (middle), and entropy of the output predictions (right) from a proxy model trained over the entire dataset. Across datasets and selection methods, SVP performs as well as an oracle baseline where ResNet164 trained on the full dataset selects the subset.

layer is smaller for the proxies than ResNet164 because they do not use a bottleneck as their residual unit. This reduction significantly speeds-up all comparisons in Algorithm 1, which are a considerable component of the runtime for small budgets.

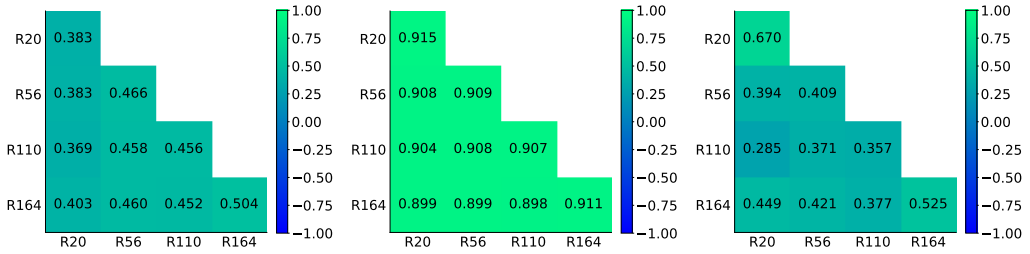
3.2 Core-Set Selection

We apply SVP to core-set selection with three different techniques: facility location [Wolf, 2011, Sener and Savarese, 2018], forgetting events [Toneva et al., 2019], and uncertainty sampling with entropy [Lewis and Gale, 1994, Settles, 2012]. Like Section 3.1, we use ResNet164 as our target model and select points with several different proxy models, as shown in Figure 4 and Table 3 in the supplementary material. We then evaluate the quality of these core-sets by training a ResNet164 model on the selected subsets and measuring its test accuracy. For all methods on both CIFAR10 and CIFAR100, SVP proxy models can perform as well as or better than an "oracle" baseline where ResNet164 itself is used as the core-set selection model.

Using forgetting events on CIFAR10, SVP with ResNet20 as the proxy can remove 50% of data in CIFAR10 without a significant increase in error from ResNet164. The entire process of training ResNet20 on all the data, selecting which examples to keep, and training ResNet164 on the subset only takes 2 hours and 20 minutes (see Table 3), which is a $1.6\times$ speed-up compared to training ResNet164 over all of the data. If we stop training ResNet50 early and remove 50% of the data based on forgetting events from the first 50 epochs, SVP achieves an end-to-end training time speed-up of $1.8\times$ with only a slightly higher top-1 error from ResNet164 (5.4% vs. 5.1%) as shown in Table 1. In general, training the proxy for fewer epochs also maintains the accuracy of the target model on CIFAR10 because the ranking from forgetting events quickly converges (see Figure 9a in the supplementary material). On CIFAR100, partial training does not work as well for proxies for larger subset sizes because ranking from forgetting events takes longer to stabilize (see Figure 9b in the supplementary material). On small subsets, partial training improves accuracy. The lower correlation may be acting as a form regularization that prevents the model from overfitting. The above results show that SVP can make core-set selection viable for deep learning by learning an inexpensive representation to select points from large datasets when data is plentiful.

Table 1: Average ($\pm$ 1 std.) top-1 error and runtime in minutes from 5 runs of core-set selection with forgetting events from ResNet20 trained for a varying number of epochs on CIFAR10 and CIFAR100.

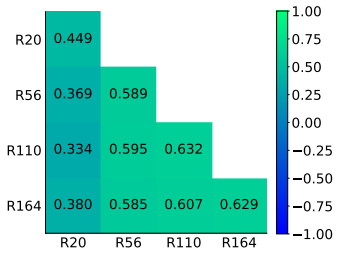
			Error			Runtime			Total Runtime		
		Subset Size	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%
Dataset	**Proxy**	**Epochs**									
CIFAR10	ResNet164 (Baseline)	181	7.7 ± 0.19	5.2 ± 0.11	5.0 ± 0.12	218 ± 1.4	218 ± 1.6	219 ± 1.5	296 ± 3.2	340 ± 6.8	382 ± 4.6
	ResNet20	181	7.6 ± 0.18	5.2 ± 0.11	5.1 ± 0.07	24 ± 1.3	24 ± 1.4	25 ± 1.5	101 ± 2.6	142 ± 2.5	185 ± 5.0
		100	7.1 ± 0.16	5.4 ± 0.22	5.0 ± 0.17	24 ± 1.6	24 ± 1.4	25 ± 1.5	102 ± 1.9	145 ± 1.5	189 ± 2.9
		50	7.2 ± 0.18	5.4 ± 0.09	5.1 ± 0.15	25 ± 1.7	24 ± 1.5	25 ± 1.5	102 ± 2.5	143 ± 1.6	186 ± 1.8
CIFAR100	ResNet164 (Baseline)	181	36.8 ± 0.36	27.1 ± 0.40	23.5 ± 0.19	221 ± 6.1	221 ± 6.1	221 ± 6.1	298 ± 5.7	342 ± 5.5	384 ± 4.7
	ResNet20	181	37.2 ± 0.29	27.1 ± 0.14	23.4 ± 0.16	24 ± 0.7	25 ± 0.7	25 ± 0.7	104 ± 3.3	148 ± 3.6	193 ± 6.1
		100	35.8 ± 0.40	27.7 ± 0.24	24.7 ± 0.33	24 ± 0.5	24 ± 0.5	24 ± 0.5	103 ± 0.5	144 ± 0.7	188 ± 1.3
		50	36.3 ± 0.25	28.2 ± 0.24	24.6 ± 0.28	24 ± 0.5	25 ± 0.7	25 ± 0.7	104 ± 3.6	149 ± 6.2	193 ± 8.4



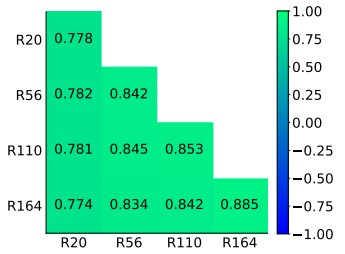
(a) CIFAR10 facility location

(b) CIFAR10 forgetting events

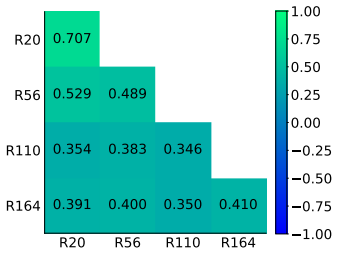
(c) CIFAR10 entropy



(d) CIFAR100 facility location



(e) CIFAR100 forgetting events



(f) CIFAR100 entropy

Figure 5: **Comparing subset selection across model sizes.** We show the average Spearman's rank-order correlation between different runs of ResNet models and a varying number of layers on CIFAR10 (top) and CIFAR100 (bottom). For each combination, we compute the average from 20 pairs of runs. For each run, we compute rankings based on the order examples are added in facility location (left), the number of forgetting events (middle), and entropy of the final model (right). We see a similarly high correlation across model architectures (off-diagonal) as between runs of the same architecture (on-diagonal), suggesting that small models are good proxies for data selection.

3.3 Ranking Correlation Between Models

To understand how well small models serve as an approximation of larger models in data selection, we compare the rankings produced by models of varying depth with various selection methods. Figure 5 shows the Spearman's rank-order correlation between ResNets of varying depth for three selection methods. For facility location, we start with 1,000 randomly selected points and rank the remaining points based on the order they are added to set s in Algorithm 1. Across models, there is a positive correlation similar to the correlation between runs of the same model. We find similar results if we use the same initial subset across runs as shown in Figure 7 in the supplementary material, meaning the variation comes from the stochasticity in training rather than the initial subset.

For forgetting events and entropy, we rank points in descending order based on the number of forgetting events and the entropy of the output predictions from the trained model, respectively. Both metrics have comparable positive correlations between different models and different runs of the same model. We also look at the Pearson correlation coefficient for the number of forgetting events and entropy in Figure 8 in the supplementary material and find a similar positive correlation both across different models and different runs of the same model. The consistent positive correlation between different model architecture illustrates why small models are good proxies for larger models in data selection.

4 Related Work

Active learning. In the active learning literature, there are examples of using one model to select points for a different, more expensive model. For instance, Lewis and Catlett [1994] proposed heterogeneous uncertainty sampling and used a Naïve Bayes classifier to select points to label for a more expensive decision tree target model. Tomanek et al. [2007] uses a committee-based active learning algorithm for an NLP task and notes that the set of selected points are "reusable" across different models (maximum entropy, conditional random field, naive Bayes). In our work, we show that this proxy approach also generalizes to deep learning models, where it can significantly reduce the running time of a recent state-of-the-art active learning method (Sener and Savarese [2018]). In addition, we show that this phenomenon extends to core-set selection using metrics such as facility location, entropy, and example forgetting.

Despite deep learning's dependency on large labeled datasets [Halevy et al., 2009, Sun et al., 2017, Hestness et al., 2017], active learning has only recently been applied to deep learning [Sener and Savarese, 2018, Wang and Ye, 2015, Wang et al., 2016, Gal et al., 2017]. To make active learning more feasible for deep learning, Sener and Savarese [2018] investigated the batch setting and proposed a core-set selection approach to active learning that outperformed existing techniques [Wang and Ye, 2015, Wang et al., 2016, Gal et al., 2017]. While this technique reduces sample complexity and makes active learning significantly faster for deep learning, the proposed technique is still computationally expensive because it requires retraining the target model after each round of selection. SVP improves the performance of this technique by up to $4\times$ by using a smaller proxy model to perform selection.

Core-set selection. Core-set selection attempts to find a representative subset of points to speed up learning or clustering; such as k-means and k-medians [Har-Peled and Kushal, 2007], SVM [Tsang et al., 2005], Bayesian logistic regression [Huggins et al., 2016], and Bayesian inference [Campbell and Broderick, 2017, 2018]. However, these examples generally require ready-to-use features as input, and do not directly apply to deep neural networks (DNNs) unless a feature representation is first trained, which can be as expensive as training a full target model. There is also a body of work on data summarization based on submodular maximization [Wei et al., 2013, 2014, Tschiatschek et al., 2014, Ni et al., 2015], but these techniques depend on a combination of hand-engineered features and simple models (e.g., hidden Markov models and Gaussian mixture models) pretrained on auxiliary tasks. In comparison, our work demonstrates that we can use the feature representations of smaller, faster-to-train proxy models as an effective way to select core-sets for deep learning tasks.

Recently, Toneva et al. [2019] showed that a large number of "unforgettable" examples that are rarely incorrectly classified once learned (i.e., 30% on CIFAR10) could be omitted without impacting generalization, which can be viewed as a core-set selection method. They also provide initial evidence that forgetting events are transferable across models and throughout training by using the forgetting events from ResNet18 to select a subset for WideResNet [Zagoruyko and Komodakis, 2016] and by computing the Spearman's correlation of forgetting events during training compared to their final values. In our work, we evaluate a similar idea of using proxy models to approximate various properties of a large model, and show that proxy models closely match the rankings of large models in the entropy, facility location and example forgetting metrics. We show how this similarity can be leveraged for active learning in addition to core-set selection.

5 Conclusion

Classical data selection techniques can be expensive to apply in deep learning because creating an appropriate feature representation is a major part of the computational cost of deep learning. In this work, we introduced selection via proxy (SVP) to improve the computational efficiency of active learning and core-set selection in deep learning by substituting a cheaper proxy model's representation for an expensive model's during data selection. Applied to recent methods from Sener and Savarese [2018]'s work on active learning for deep learning, SVP achieved up to a $4\times$ improvement in runtime with no reduction in accuracy. For core-set selection, we found that SVP can remove up to 50% of the data from CIFAR10 in $10\times$ less time than it takes to train the target model, achieving an end-to-end training speed-up of $1.6\times$ without loss of accuracy. We also showed that the rankings produced for data selection methods with SVP are highly correlated to those of larger models. Our results demonstrate that SVP is a promising approach to reduce the computational requirements of data selection methods for deep learning.

References

Trevor Campbell and Tamara Broderick. Automated scalable bayesian inference via hilbert coresets. *arXiv preprint arXiv:1710.05053*, 2017.

Trevor Campbell and Tamara Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. *arXiv preprint arXiv:1802.01737*, 2018.

Cody Coleman, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Select via proxy: Efficient data selection for training deep networks, 2019. URL `https://openreview.net/forum?id=ryzHXnR5Y7`.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*, 2016.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/dauphin17a.html`.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1183–1192. JMLR. org, 2017.

Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *Discrete & Computational Geometry*, 37(1):3–19, 2007.

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*, pages 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016b.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

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

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable bayesian logistic regression. In *Advances in Neural Information Processing Systems*, pages 4080–4088, 2016.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*, pages 148–156. Elsevier, 1994.

David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

Chongjia Ni, Cheung-Chi Leung, Lei Wang, Nancy F Chen, and Bin Ma. Unsupervised data selection and word-morph mixed language model for tamil low-resource keyword search. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4714–4718. IEEE, 2015.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=H1aIuk-RW`.

Burr Settles. From theories to queries: Active learning in practice. In Isabelle Guyon, Gavin Cawley, Gideon Dror, Vincent Lemaire, and Alexander Statnikov, editors, *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, pages 1–18, Sardinia, Italy, 16 May 2011. PMLR. URL `http://proceedings.mlr.press/v16/settles11a.html`.

Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6 (1):1–114, 2012.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 843–852. IEEE, 2017.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

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 Representations*, 2019. URL `https://openreview.net/forum?id=BJlxm30cKm`.

Ivor W Tsang, James T Kwok, and Pak-Ming Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *Advances in neural information processing systems*, pages 1413–1421, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27 (12):2591–2600, 2016.

Zheng Wang and Jieping Ye. Querying discriminative and representative samples for batch mode active learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(3):17, 2015.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. Using document summarization techniques for speech data subset selection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 721–726, 2013.

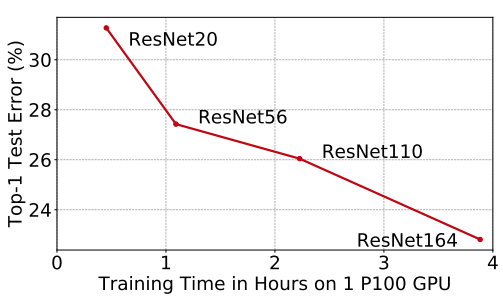
Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. Submodular subset selection for large-scale speech training data. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 3311–3315. IEEE, 2014.

Gert W Wolf. Facility location: concepts, models, algorithms and case studies., 2011.

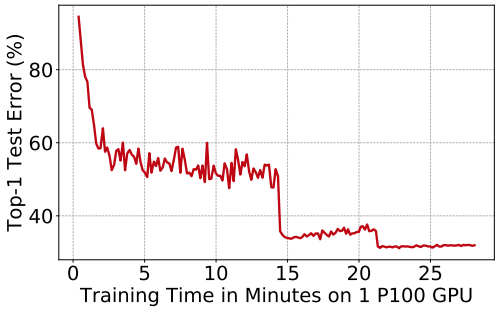
Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Supplementary Material

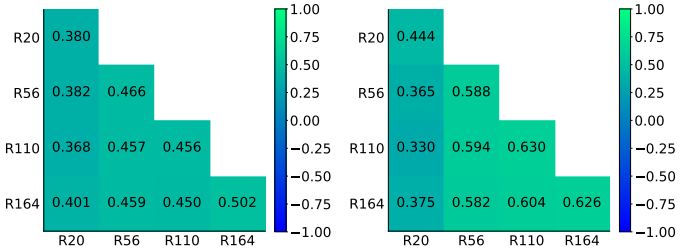


(a) Top-1 test error and training time on CIFAR100 for ResNet with pre-activation and a varying number of layers. There are diminishing returns in accuracy from increasing the number of layers.



(b) Top-1 test error during training of ResNet20 with pre-activation. In the first 15 minutes, ResNet20 reaches 33.9% top-1 error, while the remaining 12 minutes are spent on decreasing error to 31.1%

Figure 6: Top-1 test error on CIFAR100 for varying model sizes (left) and over the course of training a single model (right), demonstrating a large amount of time is spent on small changes in accuracy.

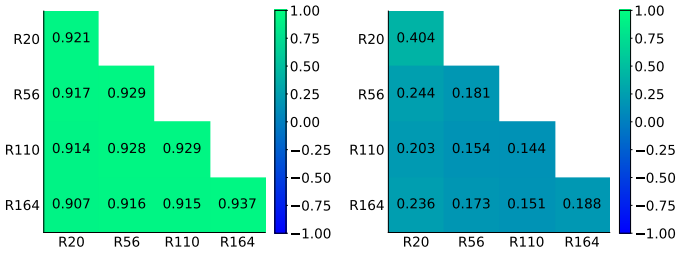


(a) CIFAR10 facility location (b) CIFAR100 facility location

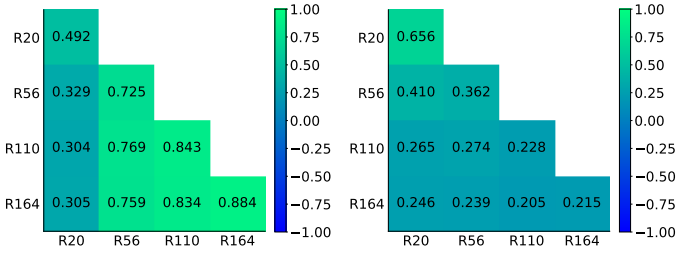
Figure 7: Spearman's rank-order correlation between different runs of ResNet with pre-activation and a varying number of layers on CIFAR10 (left) and CIFAR100 (right). For each combination, we compute the average from 20 pairs of runs. For each run, we compute rankings based on the order examples are added in facility location using the same initial subset of 1,000 randomly selected examples. The results are consistent with Figure 5a and Figure 5d, demonstrating that most of the variation is due to stochasticity in training rather than the initial subset.

Table 2: Average (± 1 std.) top-1 error and data selection runtime in minutes from 5 runs of active learning with varying proxies, selection sizes, and budgets on CIFAR10 and CIFAR100.

			Top-1 Error of ResNet164 (%)					Data Selection Runtime				
		Budget (b)	10.0%	20.0%	30.0%	40.0%	50.0%	10.0%	20.0%	30.0%	40.0%	50.0%
Dataset	**Proxy**	**Selection Size**										
CIFAR10	Random	-	20.3 ± 0.51	12.9 ± 0.37	10.1 ± 0.24	8.5 ± 0.22	7.5 ± 0.11	-	-	-	-	-
	ResNet164 (Baseline)	10%	20.1 ± 0.39	11.3 ± 0.40	8.1 ± 0.22	6.6 ± 0.24	5.6 ± 0.04	8 ± 0.4	31 ± 1.7	71 ± 4.0	126 ± 6.6	197 ± 9.9
	ResNet20	10%	19.5 ± 0.76	12.1 ± 0.44	8.8 ± 0.31	7.2 ± 0.19	6.1 ± 0.18	2 ± 0.2	7 ± 0.8	14 ± 1.5	24 ± 2.1	36 ± 3.0
		5%	19.1 ± 0.58	11.5 ± 0.17	8.5 ± 0.08	7.0 ± 0.26	6.1 ± 0.14	4 ± 0.2	13 ± 0.8	27 ± 2.1	47 ± 3.8	71 ± 6.0
		2%	18.8 ± 0.32	11.3 ± 0.13	8.6 ± 0.27	6.9 ± 0.18	6.0 ± 0.12	9 ± 0.6	32 ± 1.6	68 ± 2.7	116 ± 3.8	175 ± 5.9
	ResNet56	10%	19.8 ± 0.49	11.6 ± 0.16	8.4 ± 0.21	6.3 ± 0.17	5.7 ± 0.19	3 ± 0.1	11 ± 0.7	25 ± 1.5	45 ± 2.0	71 ± 2.8
		5%	18.2 ± 0.77	10.9 ± 0.22	7.8 ± 0.17	6.4 ± 0.18	5.6 ± 0.13	6 ± 0.4	23 ± 1.1	52 ± 2.7	91 ± 5.3	142 ± 8.6
		2%	18.4 ± 0.49	10.8 ± 0.26	7.6 ± 0.18	6.2 ± 0.23	5.5 ± 0.08	16 ± 0.3	64 ± 1.0	141 ± 2.0	248 ± 3.6	381 ± 7.6
	ResNet110	10%	19.4 ± 0.55	11.6 ± 0.16	8.1 ± 0.16	6.4 ± 0.10	5.7 ± 0.13	4 ± 0.1	17 ± 0.1	41 ± 0.7	76 ± 1.9	121 ± 5.0
		5%	18.2 ± 0.53	10.9 ± 0.36	7.7 ± 0.21	6.3 ± 0.08	5.6 ± 0.12	10 ± 1.1	37 ± 5.1	85 ± 9.8	154 ± 17.3	242 ± 27.0
		2%	18.2 ± 0.29	10.7 ± 0.29	7.6 ± 0.31	6.0 ± 0.13	5.5 ± 0.16	25 ± 2.6	101 ± 10.8	227 ± 24.5	412 ± 31.9	649 ± 48.9
CIFAR100	Random	-	60.7 ± 0.81	42.5 ± 0.55	36.0 ± 0.42	31.9 ± 0.48	29.3 ± 0.16	-	-	-	-	-
	ResNet164 (Baseline)	10%	60.4 ± 1.30	42.4 ± 0.57	34.5 ± 0.40	30.2 ± 0.33	27.3 ± 0.24	9 ± 0.1	32 ± 0.6	73 ± 2.8	129 ± 5.9	202 ± 10.0
	ResNet20	10%	60.2 ± 1.27	42.9 ± 0.52	35.8 ± 0.45	31.6 ± 0.31	28.5 ± 0.48	2 ± 0.1	6 ± 0.3	12 ± 0.5	21 ± 1.1	32 ± 1.8
		5%	61.8 ± 1.46	42.7 ± 0.52	36.0 ± 0.57	31.7 ± 0.34	28.5 ± 0.23	4 ± 0.3	13 ± 0.5	27 ± 0.6	46 ± 1.0	69 ± 1.5
		2%	61.6 ± 0.76	43.1 ± 0.23	35.6 ± 0.31	31.4 ± 0.36	28.3 ± 0.29	9 ± 0.5	32 ± 1.2	68 ± 2.1	115 ± 3.4	175 ± 6.2
	ResNet56	10%	60.9 ± 1.08	42.6 ± 0.47	35.2 ± 0.40	30.8 ± 0.25	27.8 ± 0.23	3 ± 0.2	10 ± 0.7	23 ± 1.7	42 ± 3.5	66 ± 5.3
		5%	61.6 ± 1.33	42.6 ± 0.76	34.7 ± 0.52	30.4 ± 0.32	28.0 ± 0.48	7 ± 0.2	25 ± 0.7	55 ± 1.2	96 ± 2.8	145 ± 8.2
		2%	61.2 ± 2.07	42.4 ± 0.28	34.9 ± 0.19	30.8 ± 0.29	27.7 ± 0.21	16 ± 1.0	62 ± 3.2	138 ± 7.7	239 ± 15.7	370 ± 27.3
	ResNet110	10%	59.6 ± 0.78	42.2 ± 0.76	34.9 ± 0.40	30.3 ± 0.46	27.4 ± 0.21	4 ± 0.2	17 ± 0.7	41 ± 1.5	76 ± 2.0	122 ± 2.7
		5%	60.3 ± 1.67	42.4 ± 0.65	35.0 ± 0.86	30.4 ± 0.47	27.5 ± 0.49	10 ± 0.5	41 ± 2.0	93 ± 4.2	166 ± 8.0	260 ± 14.3
		2%	61.1 ± 1.21	42.5 ± 0.60	35.0 ± 0.42	30.2 ± 0.33	27.5 ± 0.49	26 ± 2.9	105 ± 11.1	237 ± 24.2	419 ± 39.2	651 ± 62.3



(a) CIFAR10 forgetting events (b) CIFAR10 entropy



(c) CIFAR100 forgetting events (d) CIFAR100 entropy

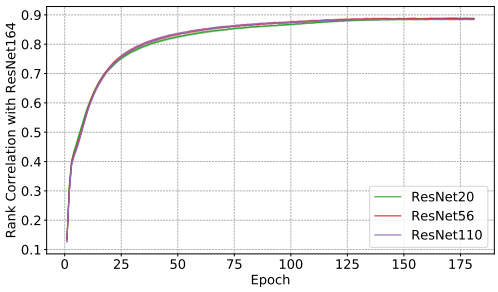
Figure 8: Pearson correlation coefficient between different runs of ResNet with pre-activation and a varying number of layers on CIFAR10 (top) and CIFAR100 (bottom). For each combination, we compute the average from 20 pairs of runs. For each run, we compute rankings based on the number of forgetting events (left), and entropy of the final model (right). Generally, we see a similarly high correlation across model architectures (off-diagonal) as between runs of the same architecture (on-diagonal), providing further evidence that small models are good proxies for data selection.

Table 3: Average ($\pm$ 1 std.) top-1 error and runtime in minutes from 5 runs of core-set selection with varying proxies, selection methods, and subset sizes on CIFAR10 and CIFAR100.

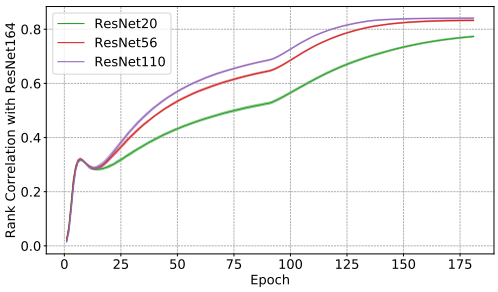
			Top-1 Error of ResNet164			Data Selection Runtime			Total Runtime		
		Subset Size	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%
Dataset	**Method**	**Proxy**									
CIFAR10	Facility Location	ResNet164 (Baseline)	8.9 ± 0.29	6.3 ± 0.23	5.4 ± 0.09	265 ± 48.0	286 ± 91.6	260 ± 42.6	342 ± 47.7	406 ± 94.3	425 ± 41.7
		ResNet20	9.1 ± 0.33	6.4 ± 0.13	5.5 ± 0.21	27 ± 1.1	28 ± 1.4	30 ± 2.2	104 ± 1.9	147 ± 1.0	193 ± 5.7
		ResNet56	8.9 ± 0.09	6.1 ± 0.21	5.3 ± 0.07	65 ± 3.9	67 ± 3.8	68 ± 3.4	142 ± 4.7	187 ± 4.8	230 ± 5.1
	Forgetting Events	ResNet164 (Baseline)	7.7 ± 0.19	5.2 ± 0.11	5.0 ± 0.12	218 ± 1.4	218 ± 1.6	219 ± 1.5	296 ± 3.2	340 ± 6.8	382 ± 4.6
		ResNet20	7.6 ± 0.18	5.2 ± 0.11	5.1 ± 0.07	24 ± 1.3	24 ± 1.4	25 ± 1.5	101 ± 2.6	142 ± 2.5	185 ± 5.0
		ResNet56	7.7 ± 0.27	5.2 ± 0.09	5.1 ± 0.09	63 ± 4.3	63 ± 4.0	63 ± 4.0	141 ± 5.4	184 ± 4.6	226 ± 2.8
	Entropy	ResNet164 (Baseline)	9.6 ± 0.16	6.4 ± 0.27	5.6 ± 0.19	218 ± 1.4	218 ± 1.7	218 ± 1.6	296 ± 1.5	338 ± 2.2	382 ± 3.1
		ResNet20	8.9 ± 0.18	5.7 ± 0.23	5.3 ± 0.09	24 ± 1.3	24 ± 1.5	25 ± 1.5	103 ± 2.2	145 ± 1.3	190 ± 3.7
		ResNet56	9.9 ± 0.29	6.6 ± 0.09	5.7 ± 0.17	63 ± 4.3	63 ± 4.0	63 ± 4.0	141 ± 4.8	182 ± 4.0	226 ± 3.8
CIFAR100	Facility Location	ResNet164 (Baseline)	40.8 ± 0.20	29.5 ± 0.29	24.6 ± 0.42	263 ± 52.2	325 ± 158.7	296 ± 70.2	339 ± 52.7	446 ± 158.1	460 ± 69.1
		ResNet20	35.2 ± 0.37	28.2 ± 0.23	24.7 ± 0.30	27 ± 0.8	28 ± 1.3	30 ± 1.4	105 ± 2.6	151 ± 3.6	198 ± 4.6
		ResNet56	40.8 ± 0.89	29.6 ± 0.28	24.7 ± 0.40	64 ± 1.7	66 ± 1.9	67 ± 2.2	142 ± 1.7	185 ± 1.5	230 ± 3.9
		ResNet110	42.3 ± 0.44	29.5 ± 0.43	24.7 ± 0.38	129 ± 3.7	131 ± 3.6	132 ± 3.5	208 ± 7.3	253 ± 8.5	303 ± 11.6
	Forgetting Events	ResNet164 (Baseline)	36.8 ± 0.36	27.1 ± 0.40	23.5 ± 0.19	221 ± 6.1	221 ± 6.1	221 ± 6.1	298 ± 5.7	342 ± 5.5	384 ± 4.7
		ResNet20	37.2 ± 0.29	27.1 ± 0.14	23.4 ± 0.16	24 ± 0.7	25 ± 0.7	25 ± 0.7	104 ± 3.3	148 ± 3.6	193 ± 6.1
		ResNet56	36.7 ± 0.23	27.0 ± 0.33	23.3 ± 0.28	62 ± 2.4	62 ± 2.6	62 ± 1.9	141 ± 7.1	183 ± 3.8	228 ± 5.2
		ResNet110	36.6 ± 0.51	26.9 ± 0.27	23.4 ± 0.37	127 ± 2.7	127 ± 2.7	127 ± 2.7	207 ± 3.7	250 ± 4.9	293 ± 7.3
	Entropy	ResNet164 (Baseline)	39.6 ± 0.43	30.1 ± 0.12	25.4 ± 0.39	220 ± 6.4	220 ± 6.4	220 ± 6.4	297 ± 7.3	340 ± 7.3	380 ± 7.1
		ResNet20	46.5 ± 0.74	29.7 ± 0.45	24.2 ± 0.21	24 ± 0.6	25 ± 0.7	25 ± 0.7	105 ± 1.7	148 ± 2.6	193 ± 3.6
		ResNet56	42.6 ± 0.63	29.6 ± 0.13	24.8 ± 0.29	62 ± 1.7	62 ± 1.8	62 ± 1.9	142 ± 1.9	186 ± 3.9	230 ± 5.9
		ResNet110	40.2 ± 0.28	30.4 ± 0.35	25.5 ± 0.34	127 ± 3.0	127 ± 3.1	127 ± 3.1	204 ± 3.3	247 ± 3.5	291 ± 3.7

Table 4: Average top-1 error ($\pm$ 1 std.) and runtime in minutes from 5 runs of core-set selection with varying selection methods calculated from ResNet20 models trained for a varying number of epochs on CIFAR10 and CIFAR100.

				Error			Runtime			Total Runtime		
			Subset Size	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%	30.0%	50.0%	70.0%
Dataset	**Method**	**Proxy**	**Epochs**									
CIFAR10	Forgetting Events	ResNet164 (Baseline)	181	7.7 ± 0.19	5.2 ± 0.11	5.0 ± 0.12	218 ± 1.4	218 ± 1.6	219 ± 1.5	296 ± 3.2	340 ± 6.8	382 ± 4.6
		ResNet20	181	7.6 ± 0.18	5.2 ± 0.11	5.1 ± 0.07	24 ± 1.3	24 ± 1.4	25 ± 1.5	101 ± 2.6	142 ± 2.5	185 ± 5.0
			100	7.1 ± 0.16	5.4 ± 0.22	5.0 ± 0.17	14 ± 1.0	14 ± 0.7	14 ± 0.7	92 ± 1.5	135 ± 0.7	178 ± 2.5
			50	7.2 ± 0.18	5.4 ± 0.09	5.1 ± 0.15	7 ± 0.9	7 ± 0.4	7 ± 0.4	85 ± 2.0	126 ± 1.4	169 ± 1.0
	Entropy	ResNet164 (Baseline)	181	9.6 ± 0.16	6.4 ± 0.27	5.6 ± 0.19	218 ± 1.4	218 ± 1.7	218 ± 1.6	296 ± 1.5	338 ± 2.2	382 ± 3.1
		ResNet20	181	8.9 ± 0.18	5.7 ± 0.23	5.3 ± 0.09	24 ± 1.3	24 ± 1.5	25 ± 1.5	103 ± 2.2	145 ± 1.3	190 ± 3.7
			100	8.4 ± 0.14	5.6 ± 0.17	5.2 ± 0.14	14 ± 1.1	14 ± 0.7	14 ± 0.7	92 ± 1.6	134 ± 1.2	176 ± 1.3
			50	10.4 ± 1.19	6.3 ± 0.55	5.2 ± 0.23	7 ± 0.8	7 ± 0.4	7 ± 0.4	84 ± 1.5	126 ± 1.6	169 ± 1.9
CIFAR100	Forgetting Events	ResNet164 (Baseline)	181	36.8 ± 0.36	27.1 ± 0.40	23.5 ± 0.19	221 ± 1.1	221 ± 0.7	221 ± 0.7	298 ± 1.6	342 ± 1.2	384 ± 1.3
		ResNet20	181	37.2 ± 0.29	27.1 ± 0.14	23.4 ± 0.16	24 ± 1.1	25 ± 0.7	25 ± 0.7	104 ± 1.6	148 ± 1.2	193 ± 1.3
			100	35.8 ± 0.40	27.7 ± 0.24	24.7 ± 0.33	14 ± 1.1	14 ± 0.7	14 ± 0.7	92 ± 1.6	134 ± 1.2	177 ± 1.3
			50	36.3 ± 0.25	28.2 ± 0.24	24.6 ± 0.28	8 ± 1.1	8 ± 0.7	8 ± 0.7	87 ± 1.6	132 ± 1.2	177 ± 1.3
	Entropy	ResNet164 (Baseline)	181	39.6 ± 0.43	30.1 ± 0.12	25.4 ± 0.39	220 ± 1.1	220 ± 0.7	220 ± 0.7	297 ± 1.6	340 ± 1.2	380 ± 1.3
		ResNet20	181	46.5 ± 0.74	29.7 ± 0.45	24.2 ± 0.21	24 ± 1.1	25 ± 0.7	25 ± 0.7	105 ± 1.6	148 ± 1.2	193 ± 1.3
			100	46.5 ± 0.52	29.7 ± 0.36	24.1 ± 0.48	14 ± 1.1	14 ± 0.7	14 ± 0.7	91 ± 1.6	135 ± 1.2	176 ± 1.3
			50	43.3 ± 1.83	30.0 ± 0.77	24.7 ± 0.41	7 ± 1.1	7 ± 0.7	8 ± 0.7	85 ± 1.6	128 ± 1.2	170 ± 1.3

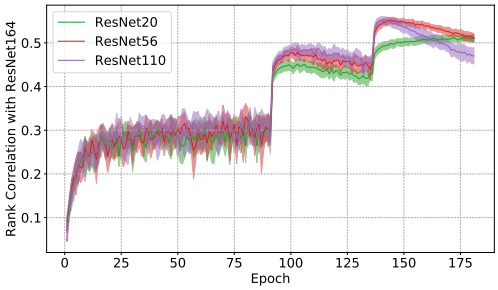


(a) CIFAR10 forgetting events

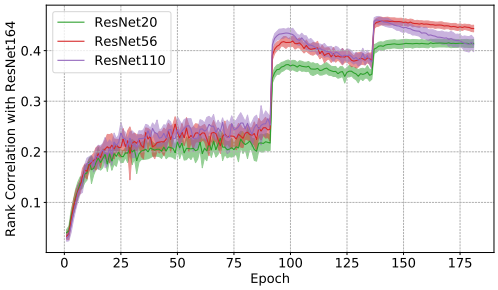


(b) CIFAR100 forgetting events

Figure 9: Average ($\pm$ 1 std.) Spearman's rank-order correlation with ResNet164 during 5 training runs of varying ResNet architectures on CIFAR10 (left) and CIFAR100 (right), where rankings were based on forgetting events.



(a) CIFAR10 entropy



(b) CIFAR100 entropy

Figure 10: Average ($\pm$ 1 std.) Spearman's rank-order correlation with ResNet164 during 5 training runs of varying ResNet architectures on CIFAR10 (left) and CIFAR100 (right), where rankings were based on entropy.