Samples with Low Loss Curvature Improve Data Efficiency

Isha Garg
Purdue University
West Lafayette, USA
gargi@purdue.edu

Kaushik Roy
Purdue University
West Lafayette, USA
kaushik@purdue.edu

Abstract

In this paper, we study the second order properties of the loss of trained deep neural networks with respect to the training data points to understand the curvature of the loss surface in the vicinity of these points. We find that there is an unexpected concentration of samples with very low curvature. We note that these low curvature samples are largely consistent across completely different architectures, and identifiable in the early epochs of training. We show that the curvature relates to the 'cleanliness' of the data points, with low curvatures samples corresponding to clean, higher clarity samples, representative of their category. Alternatively, high curvature samples are often occluded, have conflicting features and visually atypical of their category. Armed with this insight, we introduce SLo-Curves, a novel coreset identification and training algorithm. SLo-curves identifies the samples with low curvatures as being more data-efficient and trains on them with an additional regularizer that penalizes high curvature of the loss surface in their vicinity. We demonstrate the efficacy of SLo-Curves on CIFAR-10 and CIFAR-100 datasets, where it outperforms state of the art coreset selection methods at small coreset sizes by up to 9%. The identified coresets generalize across architectures, and hence can be pre-computed to generate condensed versions of datasets for use in downstream tasks. Code is available at https://github.com/isha-garg/SLo-Curves.

1. Introduction

Deep learning applications have exploded due to access to big data and computational resources. However, data is expensive to gather, annotate and store, and directly influences the computational resources required. Storing more data also runs higher risks of data leakage and privacy violation. It also influences parallelism and communication bottlenecks between machines [17]. There is limited understanding of the mechanism through which deep learning models process complex datasets, what constitutes 'good' or 'easy' examples for learning, and how large a number of samples are indeed beneficial. Research on data efficiency often focuses on doing more with less data. Standard training methods assume all data points from the training set to be independently and identically distributed from the true training distribution. Data points are sampled uniformly during training and treated as having equal significance. An alternative is to identify representative data points from the training distribution that are more beneficial to learning than random uniform sampling [30, 31]. They can be used as smaller condensed datasets, or upweighted during training as per their significance. These subsets of important points are also known as coresets.

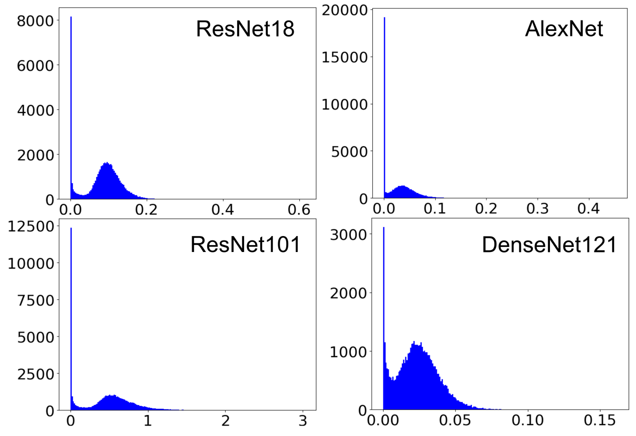
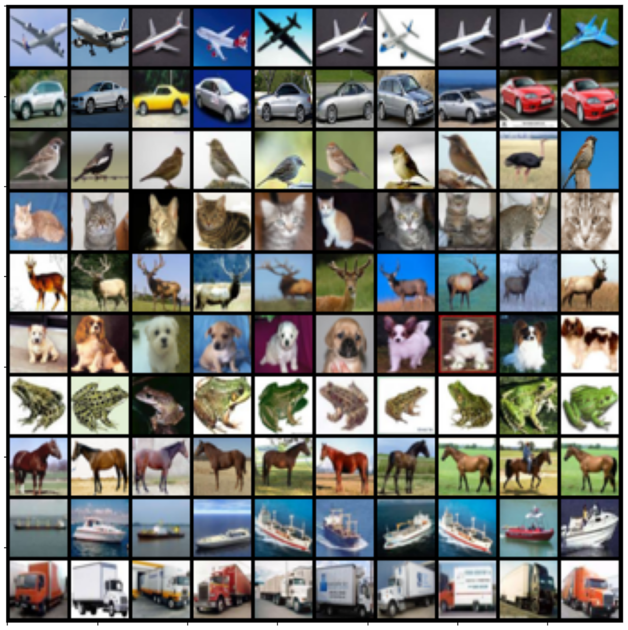


Figure 1. Histograms of training dataset's curvature for various networks trained on CIFAR-10.

Coresets prove useful in many downstream applications. They are data-efficient and can serve as a good choice of proxy data for Neural Architecture Search (NAS) [45, 52], as episodic memory in Continual Learning [2, 6, 60], and as the choice of augmentation samples during self supervised learning [37]. They can be utilized in many regimes that may have memory or compute resources, such as for compute-efficient learning [35,43,46], dataset condensation [8, 58, 61], efficient Hyper-parameter Optimization (HPO) [33], and speeding up the training of Generative Adversarial Networks (GANs) [54]. The utility of identifying im-


(a) Top 100 examples with the lowest loss curvature

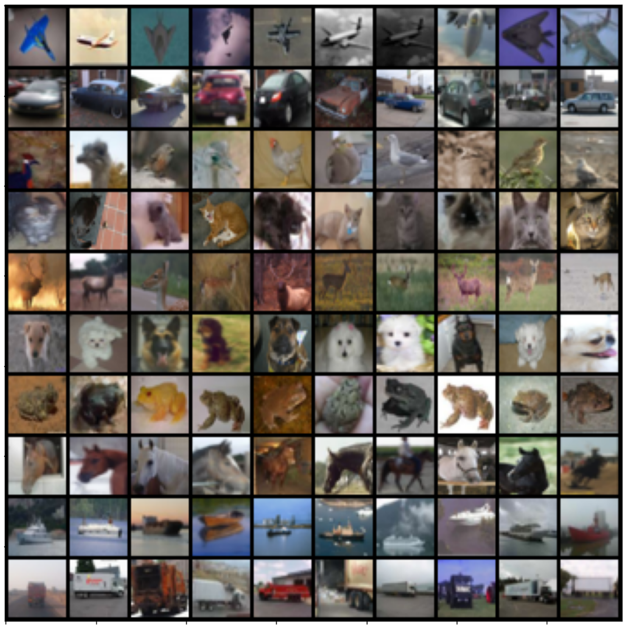

(b) Top 100 examples with the highest loss curvature

Figure 2. We visualize the 10 samples from each class with lowest and highest curvature of loss identified over 10 different randomly initialized ResNet18 models trained on CIFAR-10. Each row corresponds to a class, consistent across both figures, and 10 ordered samples for each class are shown. The network was trained with random horizontal flipping as part of the augmentations.

portant samples can extend beyond efficiency, such as in understanding dataset limitations [5, 55], enabling faster or better convergence by upsampling certain samples during training [12, 40], or in the choice of data to label in active learning [1, 50]. However, it is not always clear what makes a sample informative or good for learning. Methods differ in their definition of significance of samples. Some representative works measure importance via the confidence or margin of the predicted output of the network [1, 11], by clustering samples together in the input or feature space [10, 22], matching the gradient to that of the entire dataset [35, 43, 46], choosing samples that lie closest to the decision boundary [14] or samples that are not forgotten once learnt [57]. Some other representative methods frame it as a as a submodular function optimization problem [27, 27, 43].

In this paper, we look at data efficiency from the lens of the loss landscape around the training samples. While the loss surface with respect to the parameters of the neural network has received considerable attention as a means of analyzing the stability of the solution [19, 47–49], the loss landscape with respect to the data is far less studied. Most of the studies have been associated with adversarial robustness [44]. We are interested in the curvature of the loss surface, or inversely, the smoothness of the decision boundary around the data point. This is captured by the local linearity of the gradient around the sample, measured as the trace of the Hessian. We plot the histogram of an efficiently calculable estimation of this trace for the training dataset of different pretrained models in Figure 1 for the CIFAR-10 dataset [38]. We note they resemble a bimodal distribution, with a spread out Gaussian superimposed upon a very sharp, tall peak around zero. We are interested in the samples that make up the sharp peak, the ones around which the loss surface curvature is very low.

We visualize the sample ordered by curvature accumulated over ten differently initialized ResNet18 [23] models trained on CIFAR-10 in Figure 2. We find that they reveal useful information about the kinds of samples present in the dataset. The low curvature samples, shown in Figure 2a can be considered clean, prototypical and minimal, in that they are strongly representative of their category. On the other hand, Figure 2b shows the samples with the highest curvature of loss. We can see that they do not appear to be characteristic of their category. They have confusing backgrounds with interfering patterns, uncommon viewing angles and incomplete shapes. We show that the low curvature samples appear to be largely consistent not only across networks that were initialized differently, but also across completely different architectures such as MobileNetV3 [24], ResNet101 [23], AlexNet [39], VGG19 [53] and DenseNet121 [25]. We also show that they can be identified quite early on during the training process. As an application of this insight, we show that they make very good coresets.

Many coreset selection methods perform well at large coreset sizes, close to the size of the whole dataset. In this paper, we explore smaller coresets, ranging from single digit images per class, as is often common with few shot learning [16, 18], up to 100 images per class. Depending on the dataset and the number of classes, this can range from 0.1% to 20% of the dataset. We introduce SLo-Curves, a novel coreset selection method which identifies samples with low curvatures and train on these coresets with an additional regularizer that penalizes large curvatures. Both the method for measuring the curvature and the form of the regularizer are inspired directly from CURE [44], which introduced the regularizer as a means to promote adversarial accuracy at the expense of clean accuracy. We show that when there are small sample sizes, chosen appropriately, this regularizer also helps improve clean accuracy. We summarize our contributions below:

1. We study the loss surface with respect to the input data points and identify an unexpected concentration of samples with very low curvature.
2. We show that the curvature of the loss function relates to the notion of visual 'cleanliness'. Low curvature samples are free of conflicting features, while high curvature samples look cluttered, obstructed or unrepresentative of the other samples in the class. To the best of our knowledge, this is a novel observation.
3. We show that these samples are largely architecture and initialization independent. This implies that the Hessian reveals intrinsic properties of the dataset rather than just an artifact of the training procedure, a particular local minima, or the choice of network architecture.
4. We introduce SLo-Curves: an algorithm that selects samples with low curvature as coresets and additionally penalizes the curvature of the loss function while training on them. SLo-Curves outperforms all other choices of coreset selection methods by up to 9% for coresets of few samples per class. To the best of our knowledge, this is the first paper to explore second order statistics of the loss with respect to the data for coreset selection.
5. Due to these samples being consistent across architectures, SLo-Curves does not rely on gradient statistics during training or need to be averaged out over multiple initializations. Coresets can be pre-computed and are performant across architectures. We also find that these points can be identified early on during training, in as few as 5-10 epochs.

Our method is inherently more productive at small coreset sizes since the samples that we identify belong to the limited set that have very low curvature, corresponding to the tall peak at zero in Figure 1. Further study is required to understand what causes the decision boundary to smoothen along the same samples with different networks, and we hope that this paper accelerates interest in this direction.

2. Literature Review

The importance of samples is a relevant concern in many areas. Importance sampling methods [30, 31] prioritize the important samples for fast and efficient training by approximating the norm of the gradient resulting from each sample. Curriculum learning [4] seeks to order sample exposure to the network during training in order to learn easier samples first and then harder samples for better solutions. Active learning [51] focuses on only labeling the important samples to reduce annotation costs. However, the definition of importance of samples is not fixed. In this section, we focus on coreset selection as a base method for deciding sample importance scores, that can be then used to different ends. We broadly classify techniques for coreset selection in deep learning and discuss representative approaches in each.

Clustering based methods: Points close to each other can be clustered together and represented well by the cluster center. The choice of metric for clustering varies. Authors of [10] clusters samples based on distance in the feature space. Authors of [22] reframe the problem as solving a minimax facility location problem greedily.

Classification Output based methods: These approaches work on the assumption that examples that are classified with lesser confidence or score are more informative than examples classified with strong confidence. Authors of [11] study the effects of choosing softmax scores, entropy and the distance of the top-2 classes in the output space. A different approach is taken by the authors of [57], who show that a large majority of samples are learned correctly early on during training and are never consequently classified incorrectly, or forgotten, during the remaining course of training. They deem the samples that are never forgotten once learnt, as easily learned, and create coresets that consist of examples that need to be re-learnt.

Loss Gradient based methods: This approach categorizes the importance of a sample as its contribution to the loss or the gradient of the loss. Methods vary in their choice of the time of training at which the contribution is measured. Both Grad-Match [35] and CRAIG [43] select weighted subsets such that the appropriately scaled gradient of the subset matches the gradient of the entire dataset. Both reselect the subset and weights every few epochs. They differ in the choice of algorithm for building the dataset, with CRAIG converting the gradient-distance optimization problem into a submodular function and then using a greedy approach and Grad-Match using a different greedy Orthogonal Matching Pursuit algorithm. Both outline convergence guarantees, and show scalability via approximating gradients of the loss with respect to the input with gradients at the last layer [31]. To overcome the requirement for reselecting the coresets during training, authors of [46] introduce GraNd, which approximates the L2 norm of the gradient early on in training (after a few epochs) and averages it over many different training initializations.

Bilevel optimization and submodular function methods: Glister [36] and Retrieve [37] frame the minimization of the loss on selected samples and the selection of samples as a bilevel optimization problem. Authors of [27, 28] explore different submodular optimization functions.

Decision Boundary based methods: These methods assume that the points closest to the boundary are hardest to separate, and hence most informative. Authors of [14] utilize DeepFool, a method to generate adversarial examples, as a way of measuring distance to boundary. If an example is easier to fool, it is presumably closer to the boundary and significant. Authors of [42] select the samples whose classification likelihood is most dissimilar to its neighbors.

Synthetic coreset creation methods: In contrast to selecting points from the dataset, there has been recent interest in methods that create synthetic data to mimic the overall distribution. Dataset distillation [58] was introduced to create synthetic data by matching the gradient of samples averaged over many different initializations. Authors of [8] improve performance by repeating the synthetic set creation over many iterations. However, the datasets do not resemble the data, and this can be an advantage or disadvantage based on the application. In this manuscript we only study the samples present in the training distribution.

Coreset selection libraries: Coreset selection in deep learning is an area of rapidly increasing focus, as evidenced by the release of recent libraries that compare different coreset selection methods, and can be incorporated into other deep learning applications. Recently released libraries are CORDS [34], Minicore [3] and DeepCore [21]. We use DeepCore for our simulations.

Our method falls under both the categories of decision boundary based methods and loss gradient based methods. We select real samples from the dataset, choosing coresets from pre-trained models based on the trace of the Hessian of the loss. Our coresets consist of the samples with low trace of Hessian, or low loss curvature around them. The selected samples are chosen only once, are not re-weighted and no special optimization is followed besides an additional regularizer that encourages the learned decision boundary to be flat by penalizing the curvature estimate. They generalize across architectures and hence, do not need to be calculated real time for every network.

3. Methodology and Observations

We study the curvature profiles of the loss with respect to input for different training data points. The curvature profile can be studied via the Hessian of the loss. Concretely, let's consider a data point $x \in \mathbb{R}^d$. Let H denote the Hessian of the loss, L with respect to the data point x, i.e.

$$H(x) = \left[\frac{\delta^2 L(x)}{\delta x_i \delta x_j}\right]_{i,j=1...d}$$

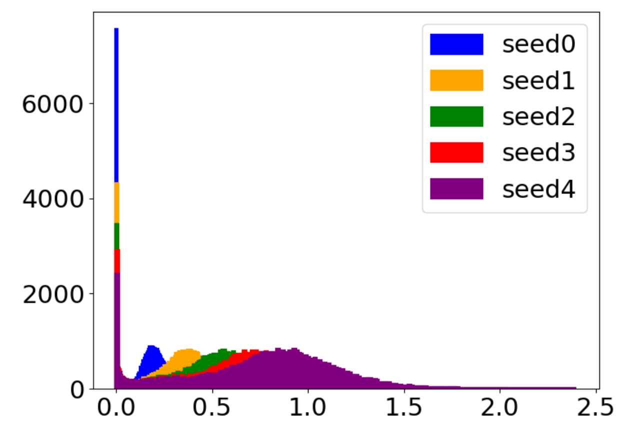
The curvature profile is captured by the eigenvalues. A good measure is the sum of the eigenvalues or the trace of the Hessian [13, 32]. The higher the trace, the more the loss surface curves around the data point, or the less smooth the decision boundary at the data point's vicinity. We plot the histogram of an approximation of the Hessian trace (explained in section 3.1) for the training set of different network architectures in Figure 1, at the end of training on CIFAR-10. We note that they follow a similar trend: a sharp tall peak at very small values, and another Gaussian-like peak later. We focus on the set of samples with very low curvatures that make up the sharp peak at zero. Histograms for different architectures can be found in Appendix A. We note that these networks were trained with Cross Entropy Loss, SGD optimizer with momentum and L2 regularization. There is no term explicitly encouraging a low spread of curvature for some samples.

3.1. Calculating the Curvature

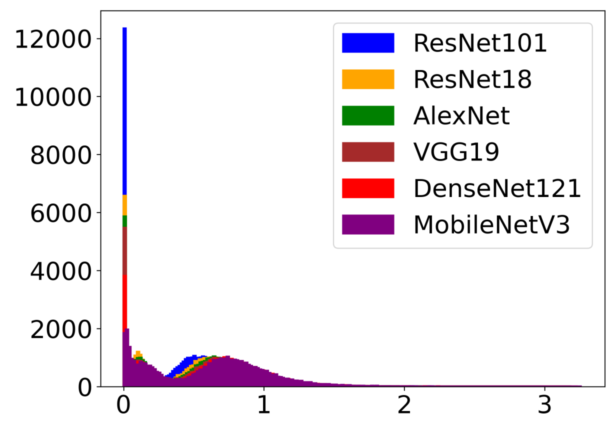
The eigenvalues of the Hessian of the loss with respect to the data point contain information about the curvature profile of the loss surface around the data point. Large absolute eigenvalues imply a large curvature of the loss function around the sample, and small magnitude of eigenvalues imply that the loss function (or the decision boundary) has a small curvature, or is locally linear in the vicinity of the data point. The Hessian of the loss surface with respect to model weights has been utilized to study the properties of the minima the optimization selects. [19, 47–49]. However, in this paper, we instead study the curvature of the loss surface with respect to the training data.

In order to study the curvature profile without having to explicitly calculate the Hessian and its eigenvalues, we estimate instead the trace of the Hessian. The trace is the sum of the eigenvalues and can be calculated efficiently via Hutchinson's trace estimator [26] and finite difference approximation. This is the same form of curvature measure used in CURE [44] as the curvature regularizer for enhancing adversarial robustness during training. We now go into the details of this formulation.

Let $\lambda_i,\ i = 1...d$ be the eigenvalues of the Hessian $H(x)$. We are interested in the spectrum of the magnitude of the eigenvalues, which can be summarized by the sum of the square of the eigenvalues. This can be estimated by the norm of the Hessian-vector product as introduced in the Hutchinson's trace estimator [26]. Using Hutchinsons's trace estimator, the definitions of the Frobenius norm of a matrix, properties of the trace, and the fact that $H(x)$ is symmetric, we get the following estimate:



(a) Curvatures accumulated over differently seeded ResNet-18



(b) Curvatures accumulated over different architectures

Figure 3. Histograms of curvature of the training set of CIFAR-10, with curvature estimates accumulated over different trained models.

$$\begin{aligned}\sum \lambda_i^2 &= Tr(H^2)\\ &= \mathbb{E}_z\left[z^T H^2 z\right]\\ &= \mathbb{E}_z\left[z^T H^T H z\right]\\ &= \mathbb{E}_z\left[(Hz)^T (Hz)\right]\\ &= \mathbb{E}_z\left[\|Hz\|_F^2\right]\end{aligned}$$

where z are Rademacher random variables, equal to +1 or -1 with equal probability, i.e. $z \sim^{i.i.d} \{-1,+1\}^d$. We now use finite approximation to estimate the Hessian vector product efficiently.

$$Hz = \frac{\nabla L(x+hz) - \nabla L(x)}{h} \quad \text{for } h \to 0$$

The cost of calculating this is just the cost of performing a backward pass on the network twice. This formulation is widely used whenever the curvatures of loss surface are discussed [7, 9, 49, 59]. Similar to CURE [44], we do not choose random directions for sampling z to take the expectation over. Instead, we choose the adversarial direction, which is known to lead to high curvature [15, 29], and set z to be $\frac{sign\nabla L(x)}{\|sign(\nabla L(x))\|}$. We found the results to not be significantly impacted by h and set it equal to 3 to match the change used in corresponding adversarial settings [20, 41].

Putting this together, the measure of curvature of the decision boundary or the loss function around a data point is proportional to the following quantity, γ where :

$$\gamma(x) = \|\nabla_x L(x+hz) - \nabla_x L(x)\|; \quad \text{where } z = \frac{sign\nabla_x L(x)}{\|sign(\nabla_x L(x))\|}$$

3.2. Consistency of Low Curvature Samples

We notice that the low curvature samples are largely consistent across architectures and initialization. We show this overlap in Figure 3, with plots of the histogram of cumulative curvature values of samples (estimated by γ), accumulated over different seeds in Figure 3a and over different architectures in Figure 3b. Mathematically, the i^{th} curve corresponds to the histogram of $\sum_{j=0}^{i}\gamma_j$ for all training samples, where j indexes different trained models. The first histogram is shown in blue, with unaccumulated curvature estimates (γ_0 collected from model 'seed0' or 'ResNet101'). Next the curvature estimates of the second model (γ_1 for 'seed1' or 'ResNet18') are added to the first model's curvature estimates for each sample and the resulting plot is shown in orange. As we accumulate curvature estimates over many different models, the sharp peak at zero starts to spread out a bit. Yet, even after accumulating over 5 seeds or 6 different architectures all converging to different minimas, there are more than 2000 common low curvature samples (identified as the smallest, purple peak at zero). This implies that these samples appear to be an intrinsic property of the dataset rather than the initialization or the local minimas the networks converged to. We corroborate this experimentally in Section 4.3, by showing that a ResNet18 model trained on coresets identified on any of these architectures outperforms random sampling by an average of 5.3%.

3.3. Identified Coresets

We now visualize the low and high curvature samples (taken from the accumulated scores of 10 randomly seeded ResNet18 models trained on CIFAR10) in Figures 2. Each row corresponds to a class, and contains the top 10 samples for that class with the lowest and highest curvatures in Figures 2a and 2b, respectively. As can be seen from the figure, the low curvature samples are very 'good' samples, in that they have low to no conflicting backgrounds, structure typical of the class, are centered and are not zoomed in or trimmed. The networks are trained with random left to right flipping, and hence we see samples facing both ways. On

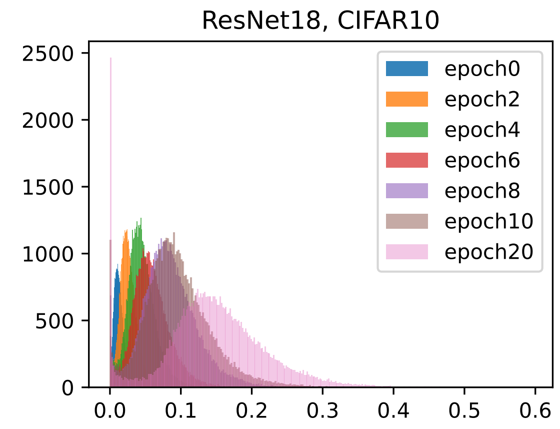


Figure 4. The histogram of curvature estimates of the training dataset as training progresses. For ease of visualization, the opacity of the curves decreases with epochs.

the other hand, the high curvature samples are 'bad' in that they are visually harder to identify by humans as well. They have conflicting backgrounds (a deer in a fiery landscape), uncommon viewing angles (a truck from behind) and colors uncharacteristic of the class (a blue plane or a yellow frog). This implies that the curvature, captured by γ could be useful as a metric for 'learnability' of a sample.

3.4. Early Emergence of Low Curvature Samples

We track γ through the training progress to study the emergence of low curvature samples, and visualize this in Figure 4. As training progresses (lesser opaque histograms in the figure), we can see the peak towards zero becoming more defined. We find that the concentration of samples around low curvatures emerges in as few as 5-10 epochs of training. We corroborate this in Section 4.5 by showing that the coresets identified by our proposed method as early as epoch 10 outperform random sampling by an average of 4.7% accuracy across all considered coreset sizes.

3.5. SLo-Curves: An Algorithm for Data Efficiency

Based on these insights, we propose SLo-Curves, an algorithm to identify coresets of samples with low curvature and train on them for increased sample efficiency. We use curvature estimate, γ as the selection metric, choosing samples with the lowest values to make up our coreset. Additionally, we show that regularizing the learned curvature around these coresets helps increase the accuracy by a few percentage points. Hence, we first select the points with the lowest γ from pretrained networks, and then train a new randomly initialized network on these points using the standard cross entropy loss $L_{CE}(x)$ and γ serving as the additional regularizer term, $L_{reg} = \gamma(x)$. The new loss is thus:

$$L(x) = L_{CE}(x) + \lambda\gamma(x)$$

The tradeoff parameter λ is the only hyperparameter our method induces and is searched over $\{0,0.5,1,5,10,20,50\}$. We note that smaller coresets require larger values of λ and larger coresets require smaller values, ultimately giving better results when the additional regularizer is turned off. This ties with CURE [44], wherein the same regularizer improves adversarial accuracy at the cost of clean accuracy when used with the whole dataset. In our method, it serves instead to improve accuracy at low sample complexities. We anticipate that it is an especially effective form of regularization for small datasets learned with complex models. We retierate that we study small coreset sizes since our significance metric is based on the samples that concentrate near zero curvature, and the number of such samples is a limited percentage of the dataset.

4. Results and Discussion

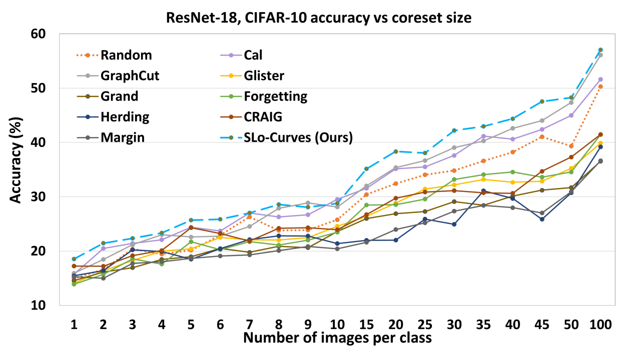
In this section, we outline our network architectures, the training hyperparameters and discuss results.

4.1. Experimental Details

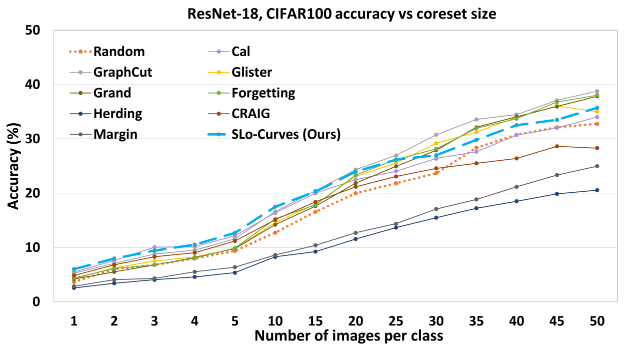
We run experiments on CIFAR-10 and CIFAR-100 datasets [38]. Both datasets have 50,000 training samples with CIFAR-10 having 5000 samples for each of its 10 classes and CIFAR-100 having 500 samples for each of its 100 classes. The networks used are ResNet-18 [23], ResNet-101 [23], VGG-19 [53], AlexNet [39], MobileNetV3Small [24] and DenseNet-121 [25]. We consider coreset of small sizes, ranging from 1 to 100 samples per class. For all networks, we use a learning rate of 0.1 scaled by 0.1 at epochs 81 and 121, SGD optimizer with a momentum of 0.9, with Nesterov momentum [56] turned on and a weight decay of 5e-4. We train all networks for 164 epochs and report mean and variance for 5 randomly seeded runs. All graphs show mean values, with corresponding variances shown in Appendix B. We compare with representative works from the broad categories of coreset selection methods. In particular, we compare with 9 other methods: Random uniform sampling, Glister [36], Forgetting [57], CRAIG [43], GraphCut [27], Cal [42], GraNd [46], Herding [10] and Margin [11]. For methods that require training before coreset selection, we train the network for 40 epochs to ensure a good convergence point before we select the coresets. We have run the baselines using the Deepcore [21] library (building upon it for methods and networks as needed), and find that it is an excellent repository of state of the art methods. The only hyperparameter that we tune for our method is the strength of the regularization, λ. We tune this over the search space $\{0,0.5,1,5,10,20,50\}$ and report the best results.

4.2. Comparison with Coreset Selection Methods

We now discuss the results of our method compared to random sampling and other state of the art coreset selec-

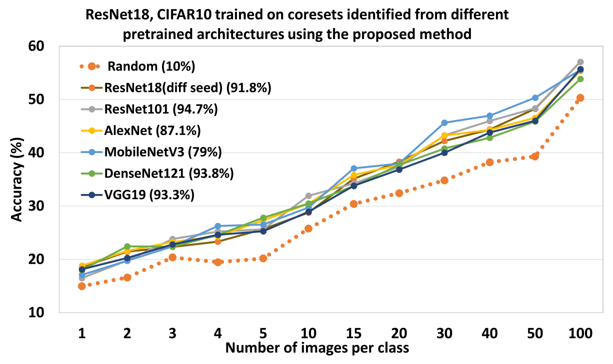


(a) CIFAR-10

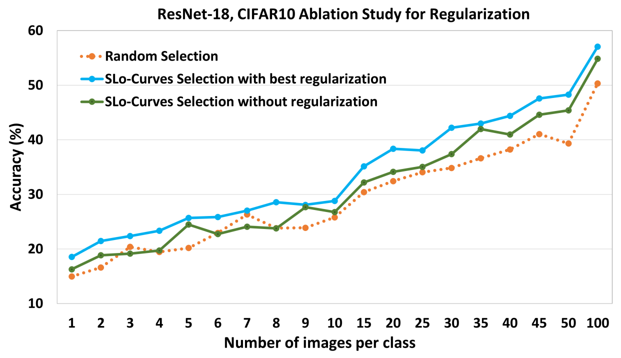


(b) CIFAR-100

Figure 5. ResNet18 trained on coresets identified by various methods. Note that the x-axis is not to scale to clearly show performance at a few samples per class. The numbers shown are the mean of 5 runs. Variances are reported in Appendix B.1 and B.2



(a) Cross architecture results



(b) Ablation Study for selection without regularization

Figure 6. ResNet18 trained on coresets identified by our methods a) from different pretrained architectures and b) with and without regularization. Note that the x-axis is not to scale to clearly show performance at a few samples per class. The numbers shown are the mean of 5 runs. Variances are reported in Appendix B.3 and B.4.

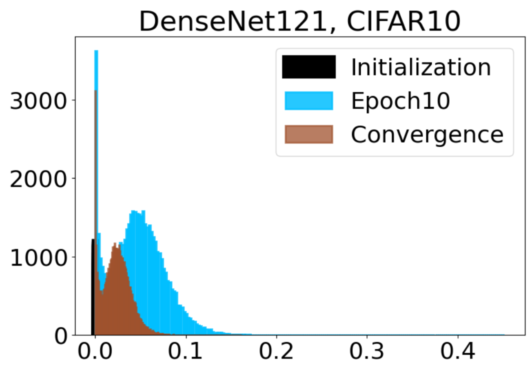
tion methods. The results for ResNet18 networks trained on CIFAR-10 and CIFAR-100 are shown in 5a and 5b respectively. The corresponding tables with mean and variances are shown in Appendix B.1 and B.2.

CIFAR-10: Figure 5a shows that selection by the proposed SLo-Curves method outperforms all other coreset methods studied all the way up to 100 samples per class, or 2% of the total dataset (1000 samples). We note that at such coresets sizes, random sampling forms a surprisingly strong baseline (also noted in [21]). The strongest competitors are the submodular function based method GraphCut [27] and decision boundary based method Cal [42]. SLo-Curves shows the best performance in 16 out of the 19 coreset sizes studied and the second best in the remaining 3 cases. It outperforms random sampling by 1-9%.

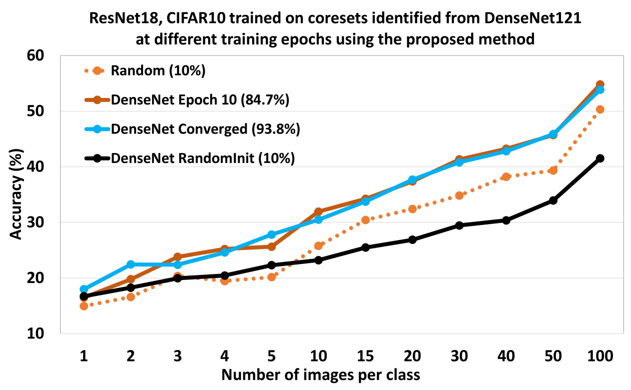
CIFAR-100 Figure 5b shows that our method outperforms or matches the performance of all other coreset methods studied up to 25 samples per class, or 5% of the total dataset (2500 samples). At larger coreset sizes, SLo-Curves shows a few percentage points of accuracy drop from the state of the art methods Cal and GraphCut, but outperforms random sampling, CRAIG [43], Herding [10] and Margin [11] baselines. We note that even at coreset sizes equal to 10% of the dataset (50 samples per class), we outperform random sampling by greater than 3%.

4.3. Cross Architecture Results

From Section 3.2, we know that low curvature samples generalize across architectures. We corroborate this experimentally and show that SLo-Curves consistently chooses good subsets irrespective of pretrained network accuracy or architecture. Figure 6a shows the accuracy of ResNet18 trained on coresets identified from different pretrained models. We show the results of a differently seeded ResNet-18, a lower accuracy AlexNet, completely different architectures MobileNetV3 and VGG19, a deeper similar architecture ResNet101 and a much bigger network, DenseNet121, with pretraining accuracies mentioned in brackets. Coresets chosen by SLo-curves outperform random sampling for all considered architectures by an average of 5.3%. We can see that subsets chosen from lower, similar or higher accuracy and complexity models all consistently perform well. Variances is shown in Appendix B.3.



(a) Histograms



(b) Accuracy

Figure 7. ResNet18 trained on coresets identified from DenseNet121 at initialization, at epoch 10 and at end of training. Note that the x-axis is not to scale. The variances are reported in Appendix B.5.

4.4. Decoupling Selection and Regularization

Here, we show that the criterion for selection of points is beneficial even without the curvature regularization during training. In Figure 6b, we show the results of ResNet18 trained on coresets of CIFAR-10 identified by SLo-Curves selection both with the best regularization strength λ for each coreset size, and without any regularization ($\lambda = 0$). We show the results of random sampling as the baseline. We observe that the best performance is consistently shown by our method with regularization. However, our method without regularization still outperforms random sampling by an average of 2.3% over all coreset sizes. The variances are reported in Appendix B.4.

4.5. Early Emergence of Low Curvature Samples

Many coreset selection methods require training the network for a few epochs before coreset selection. However, we observe from Section 3.4 that low curvature samples can be identified in the early epochs of training. We now study their efficacy for coreset selection for real time usage. We use a DenseNet121 to perform coreset selection, and train a ResNet18 from scratch on the coreset identified by our method. We study the randomly initialized, untrained DenseNet121 (with 10% accuracy), DenseNet121 trained for 10 epochs (to 84.7% accuracy), and DenseNet121 after convergence (93.4% accuracy). Figure 7b visualizes the results, with random sampling shown as a baseline. We show that the coresets identified at epoch 10 perform on par with the coreset identified at convergence. The early emergence of these low curvature samples can also be observed in the corresponding histograms shown in Figure 7a. However, during experimentation, we noted that higher regularization strength (λ) was needed and recommend that where pretrained models are present, it is preferable to extract coresets from them. Surprisingly, choosing low curvature samples selected from an untrained random initialization performs worse than random sampling. We think this may be because the regularizer forces a simplification of the learned boundary that is useful when the samples lend themselves to a simple boundary.

5. Conclusion

We studied the curvature of the loss of trained neural networks with respect to the training data points. We observed that there is a significant concentration of training samples with very low curvature, and found that these samples remain consistent between networks of differing initializations and architectures. We also noted that they can be identified early on during training. We visualize the samples with low and high curvature and find striking visual differences between them. The low curvature samples are clean, devoid of obstructions and prototypical of their category. The high curvature samples have visual clutter, occluding and uncommon backgrounds, and are hard to identify. The results highlight the Hessian of data points as a useful tool in the study of deep learning generalization. We apply these observations to select significant samples and introduce SLo-Curves, a novel coreset identification and training algorithm that can generalize across architectures. SLo-curves constructs coresets from the samples with low curvatures and trains on them with an additional regularizer that penalizes large curvature of loss around them. We show that at small coreset sizes, SLo-Curves beats the state of the art coreset selection methods by up to 9% on CIFAR-10 and CIFAR-100 datasets.

Acknowledgement

The authors wish to thank Dr. Andrew Buchanan for the insightful discussions and suggestions. This work was supported in part by the Center for Brain Inspired Computing (C-BRIC), one of the six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, by the Semiconductor Research Corporation, the National Science Foundation, Intel Corporation, the DoD Vannevar Bush Fellowship, and by the U.S. Army Research Laboratory.

References

[1] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *European Conference on Computer Vision*, pages 137–153. Springer, 2020. 2

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019. 1

[3] Daniel Baker. Minicore: Fast Generic Coresets, 3 2022. 4

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 3

[5] Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. Semantic redundancies in image-classification datasets: The 10% you don't need. *arXiv preprint arXiv:1901.11409*, 2019. 2

[6] Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems*, 33:14879–14890, 2020. 1

[7] Lucas Böttcher and Gregory Wheeler. Visualizing high-dimensional loss landscapes with hessian directions, 2022. 5

[8] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022. 1, 4

[9] Avraam Chatzimichailidis, Franz-Josef Pfreundt, Nicolas R. Gauger, and Janis Keuper. Gradvis: Visualization and second order analysis of optimization surfaces during the training of deep neural networks, 2019. 5

[10] Yutian Chen, Max Welling, and Alex Smola. Supersamples from kernel herding. *arXiv preprint arXiv:1203.3472*, 2012. 2, 3, 6, 7

[11] 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. 2, 3, 6, 7

[12] Sanjoy Dasgupta, Daniel Hsu, Stefanos Poulis, and Xiaojin Zhu. Teaching a black-box learner. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1547–1555. PMLR, 09–15 Jun 2019. 2

[13] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017. 4

[14] Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018. 2, 4

[15] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3762–3770, 2018. 5

[16] Li Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006. 3

[17] Dan Feldman. Introduction to core-sets: an updated survey. *CoRR*, abs/2011.09384, 2020. 1

[18] Michael Fink. Object classification from a single example utilizing class relevance metrics. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. 3

[19] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pages 2232–2241. PMLR, 2019. 2, 4

[20] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 5

[21] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *Database and Expert Systems Applications: 33rd International Conference, DEXA 2022, Vienna, Austria, August 22–24, 2022, Proceedings, Part I*, page 181–195, Berlin, Heidelberg, 2022. Springer-Verlag. 4, 6, 7

[22] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '04, page 291–300, New York, NY, USA, 2004. Association for Computing Machinery. 2, 3

[23] 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, 2016. 2, 6

[24] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 2, 6

[25] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 2, 6

[26] M.F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communication in Statistics- Simulation and Computation*, 18:1059–1076, 01 1989. 4

[27] Rishabh Iyer, Ninad Khargoankar, Jeff Bilmes, and Himanshu Asanani. Submodular combinatorial information measures with applications in machine learning. In Vitaly Feldman, Katrina Ligett, and Sivan Sabato, editors, *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pages 722–754. PMLR, 16–19 Mar 2021. 2, 4, 6, 7

[28] Rishabh K Iyer and Jeff A Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. *Advances in neural information processing systems*, 26, 2013. 4

[29] Saumya Jetley, Nicholas Lord, and Philip Torr. With friends like these, who needs adversaries? *Advances in neural information processing systems*, 31, 2018. 5

[30] Tyler B Johnson and Carlos Guestrin. Training deep models faster with robust, approximate importance sampling. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 1, 3

[31] Angelos Katharopoulos and Francois Fleuret. Not all samples are created equal: Deep learning with importance sampling. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2525–2534. PMLR, 10–15 Jul 2018. 1, 3

[32] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017. 4

[33] Krishnateja Killamsetty, Guttu Sai Abhishek, Alexandre V Evfimievski, Lucian Popa, Ganesh Ramakrishnan, Rishabh Iyer, et al. Automata: Gradient based data subset selection for compute-efficient hyper-parameter tuning. *arXiv preprint arXiv:2203.08212*, 2022. 1

[34] Krishnateja Killamsetty, Dheeraj Bhat, Ganesh Ramakrishnan, and Rishabh Iyer. CORDS: COResets and Data Subset selection for Efficient Learning, 3 2022. 4

[35] Krishnateja Killamsetty, S Durga, 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*, pages 5464–5474. PMLR, 2021. 1, 2, 3

[36] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8110–8118, 2021. 4, 6

[37] Krishnateja Killamsetty, Xujiang Zhao, Feng Chen, and Rishabh Iyer. Retrieve: Coreset selection for efficient and robust semi-supervised learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 14488–14501. Curran Associates, Inc., 2021. 1, 4

[38] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2, 6

[39] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 2, 6

[40] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6781–6792. PMLR, 18–24 Jul 2021. 2

[41] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 5

[42] Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. Active learning by acquiring

contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. 4, 6, 7

[43] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020. 1, 2, 3, 6, 7

[44] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9086, 2019. 2, 3, 4, 5, 6

[45] Byunggook Na, Jisoo Mok, Hyeokjun Choe, and Sungroh Yoon. Accelerating neural architecture search via proxy data. *arXiv preprint arXiv:2106.04784*, 2021. 1

[46] 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, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 20596–20607. Curran Associates, Inc., 2021. 1, 2, 3, 6

[47] Levent Sagun, Léon Bottou, and Yann LeCun. Singularity of the hessian in deep learning. *CoRR*, abs/1611.07476, 2016. 2, 4

[48] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017. 2, 4

[49] Adepu Ravi Sankar, Yash Khasbage, Rahul Vigneswaran, and Vineeth N Balasubramanian. A deeper look at the hessian eigenspectrum of deep neural networks and its applications to regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9481–9488, 2021. 2, 4, 5

[50] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017. 2

[51] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009. 3

[52] Jae-hun Shim, Kyeongbo Kong, and Suk-Ju Kang. Core-set sampling for efficient neural architecture search. *arXiv preprint arXiv:2107.06869*, 2021. 1

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 6

[54] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-GAN: Speeding up GAN training using coresets. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9005–9015. PMLR, 13–18 Jul 2020. 1

[55] Benjamin Spector, Ravi Kumar, and Andrew Tomkins. Preventing adversarial use of datasets through fair core-set construction. *CoRR*, abs/1910.10871, 2019. 2

[56] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013. 6

[57] 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 *ICLR*, 2019. 2, 3, 6

[58] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 1, 4

[59] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael Mahoney. Pyhessian: Neural networks through the lens of the hessian. 2019. 5

[60] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *International Conference on Learning Representations*, 2022. 1

[61] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021. 1