Drop it Like it's Hot
Network Pruning with Data Selection

Hersh Gupta[1] **Radhegovind Sriram**[1] **Preetham Pangaluru**[1] **Colton Rowe**[1]
[1]University of California, Los Angeles
{hershgupta,radhesriram,vignesh99,coltonrowe}@g.ucla.edu

Abstract

Deep neural networks have achieved remarkable performance across a range of tasks, but their growing complexity comes at significant computational and storage costs. Model pruning and data subset selection have each shown promise as complementary strategies for improving efficiency without severely degrading accuracy. However, identifying informative data subsets and deciding which weights to remove often requires complex heuristics. In this work, we introduce a novel framework that jointly leverages coreset selection—guided by a forgetting-based metric called forgettability — and structured network pruning to streamline training and inference. Our approach systematically prunes both the model and the training data. By including the most representative samples during early training and pruning network parameters, we can significantly reduce computational overhead and model size while maintaining competitive accuracy. Empirical results on benchmark datasets demonstrate that the combined method outperforms standard pruning or data selection alone, offering a promising pathway to more efficient deep learning.

1 Introduction

Deep neural networks now set the standard across a wide spectrum of machine learning tasks, from image recognition [Krizhevsky et al., 2012] and natural language processing [Devlin et al., 2019] to reinforcement learning [Mnih et al., 2015]. However, their success often comes at a cost: large models with millions or even billions of parameters demand immense computational resources and storage, imposing practical barriers to real-world deployments—especially on resource-constrained edge devices or in time-sensitive applications [Li et al., 2020, Sze et al., 2017].

Two well-studied avenues to alleviate this burden are model pruning and data selection. Model pruning techniques [Frankle and Carbin, 2019, Gale et al., 2019, Han et al., 2016, He et al., 2017] focus on removing redundant parameters from over-parameterized networks, thus simplifying inference, reducing memory footprint, and often speeding up training. For instance, magnitude-based pruning [Han et al., 2016] removes weights below a threshold, and the Lottery Ticket Hypothesis [Frankle and Carbin, 2019] suggests that sparse subnetworks exist which can train effectively from the original initialization. Other methods have refined pruning to be more structured [He et al., 2017] or have conducted large-scale empirical evaluations of sparsity [Gale et al., 2019].

Data selection, on the other hand, deals with the training dataset. Instead of training on the full dataset, data selection methods aim to identify a representative coreset—a small, yet informative subset of data—from which the model can learn effectively [Bachem et al., 2017, Feldman and Langberg, 2011, Mirzasoleiman et al., 2020, Sener and Savarese, 2018]. Such approaches can shorten training time and sometimes improve generalization by mitigating overfitting to outlier samples. By selecting only the most informative examples, coresets can deliver most of the performance benefits of the full dataset at a fraction of the computational cost.

Preprint. Under review.

These strategies, while both beneficial, are typically considered in isolation. Pruning is often applied after or during full-dataset training, while coreset selection focuses on subset identification without explicitly considering model structure. By overlooking their potential synergy, we may miss opportunities to boost efficiency further. A joint approach that simultaneously selects data and prunes the network could streamline resource usage from both ends—fewer parameters and fewer training samples—potentially preserving or even enhancing final accuracy.

In this work, we bridge this gap by jointly applying coreset selection and pruning within a single cohesive framework. A key ingredient in our approach is the concept of forgettability, a forgetting-based metric derived from the idea of "forgetting events" [Chaudhry et al., 2018, Toneva et al., 2019]. Forgetting events identify samples the model repeatedly misclassifies after having classified them correctly, signaling their importance or difficulty. By selecting coresets based on forgettability, we ensure the reduced dataset still provides a rich learning signal.

Alongside coreset selection, we apply network pruning to remove superfluous parameters as training progresses. The interplay between these two actions encourages the network to learn efficiently from the most critical examples while discarding unnecessary weights. Our contributions are as follows:

- We introduce a new framework that integrates coreset selection and model pruning, leveraging forgettability as a guide.
- We empirically demonstrate that this integrated approach achieves significant reductions in training cost and model size, while maintaining competitive accuracy on benchmark datasets.
- We provide insights into how data selection complements pruning, showing that focusing training on carefully chosen samples can retain performance even as large portions of the model are pruned.

By exploring the synergy between pruning and data selection, we aim to open new avenues for constructing efficient, high-performing models that remain feasible in practical, resource-limited environments.

2 Related Works

Research in improving the efficiency of deep neural networks has largely focused on two complementary fronts: model compression, particularly in the form of pruning, and data-efficient training methods, such as coreset selection. While both directions aim to reduce computational overhead and resource usage, they have traditionally evolved in parallel.

Model Pruning. Model pruning techniques aim to remove redundant weights or entire channels without significantly sacrificing accuracy. Early work in this area introduced magnitude-based pruning, which eliminates weights below a certain threshold, achieving considerable compression rates [Han et al., 2016]. The Lottery Ticket Hypothesis [Frankle and Carbin, 2019] further suggested that, within large networks, there exist sparse sub-networks capable of training effectively from the original initialization. Subsequent approaches explored structured pruning [He et al., 2017], where entire filters are pruned to yield more hardware-friendly architectures, and large-scale empirical evaluations have investigated the state of sparsity in neural networks [Gale et al., 2019]. Across these efforts, pruning has demonstrated the potential to reduce both inference time and memory footprint, though it often requires careful tuning or retraining steps to recover performance.

Data Selection and Coresets. In parallel, data subset selection methods, or coreset construction, concentrate on reducing the training set to a representative subset without excessively hurting model accuracy. The concept of coresets originated from computational geometry [Feldman and Langberg, 2011], where small subsets approximate the full dataset. More recent works have tailored coresets for machine learning, providing approximation guarantees and practical constructions [Bachem et al., 2017], identifying informative subsets that can speed up training and improve generalization by focusing the learning process on critical examples. Active learning approaches like the coreset-based selection of Sener and Savarese [2018] adaptively choose samples that best represent the data distribution, and similar ideas have been applied to various tasks, including continual learning and uncertainty-based sampling. Methods like Mirzasoleiman et al. [2020] have shown that carefully chosen subsets can approach full-data performance with significantly fewer training examples.

Forgetting Events and Forgettability. Beyond static data selection heuristics, recent studies have explored dynamic measures of data importance based on training trajectories. For instance, Toneva

et al. [2019] introduced the concept of "forgetting events"—instances where a previously correctly classified sample becomes misclassified in later epochs—highlighting that not all samples are learned equally or at the same pace. Forgetting events have also been linked to sample difficulty and influence on the training process. Drawing from these insights, metrics like forgettability identify consistently challenging or highly influential samples, thus potentially serving as a more principled criterion for data selection.

Combining Pruning and Data Selection. While both pruning and data selection individually contribute to making deep networks more efficient, comparatively little work has investigated their combination. Some works have considered curriculum learning or iterative sample selection strategies alongside network sparsification, but a unified framework that jointly optimizes over which parameters to prune and which samples to keep remains sparse. Since pruning methods typically assume training on the full dataset, they may prune important parameters early or fail to consider that some data points are more crucial than others. Conversely, data selection methods usually operate independently of model structure, ignoring the fact that pruning can change which samples are most beneficial to retain.

Our work addresses this gap by integrating a forgetting-based coreset selection strategy with model pruning. By using forgettability as a guide, we ensure that the data retained during training exerts maximal influence on the pruned network. This joint approach distinguishes our work from prior methods that treat pruning and data selection as separate optimization problems. We show that the synergy between these two paradigms can lead to highly efficient training without a significant drop in accuracy, thus pushing the frontier of resource-efficient deep learning.

3 Problem Formulation

Training models on coresets has been shown to reduce computation and allow for greater model generalization. Neural network pruning reduces parameter count and saves compute on forward passes, but often hinders model performance. We aim to see how training on coresets will affect model accuracy after pruning. The hope is that models trained on coresets will be more general and thus be more robust to pruning.

We consider a supervised learning setting with a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where each $x_i \in \mathcal{X}$ is an input (e.g., an image) and $y_i \in \mathcal{Y}$ is the corresponding label. Let $f_\theta : \mathcal{X} \to \mathcal{Y}$ be a neural network parameterized by weights $\theta \in \mathbb{R}^p$, where p is the number of parameters. The goal is to learn parameters θ that minimize a training loss:

$$\min_\theta \ \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(x_i), y_i),$$

where $\ell(\cdot, \cdot)$ is a loss function, such as cross-entropy.

While training on the entire dataset $\mathcal{D}$ with a large, unpruned model f_θ can yield high accuracy, it is computationally and memory expensive. We seek to reduce both the size of the dataset used for training and the number of parameters in the model, without significantly compromising accuracy.

3.1 Coreset Selection

A coreset $\mathcal{C} \subseteq \mathcal{D}$ is a small but representative subset of the full dataset. Instead of using all N samples, we want to identify a coreset $\mathcal{C}$ of size $M \ll N$ that preserves the essential information needed for training. Formally, we aim to find:

$$\mathcal{C} = \arg \min_{\mathcal{C} \subseteq \mathcal{D}, |\mathcal{C}| = M} \mathcal{E}(f_\theta, \mathcal{C}),$$

where $\mathcal{E}(f_\theta, \mathcal{C})$ is some measure of how well a model trained on $\mathcal{C}$ approximates training on the full dataset. In practice, identifying the optimal coreset is computationally challenging; heuristics based on data importance, diversity, or "forgetting events" are employed.

3.2 Forgetting-Based Importance Metric (Forgettability)

To guide coreset selection, we leverage forgettability, which is defined as a forgetting event as an epoch in training where a sample (x_i, y_i) transitions from being correctly classified to incorrectly classified. Let f_θ^t represent the model parameters after the t-th epoch of training. A forgetting event occurs for sample i if:

$$\mathbb{I}\big[\arg\max f_\theta^{t-1}(x_i) = y_i\big] = 1 \quad \text{and} \quad \mathbb{I}\big[\arg\max f_\theta^{t}(x_i) = y_i\big] = 0.$$

The total forgetting score of sample i is the count of such events over training. Samples with high forgetting scores can be considered more "forgettable" or challenging, indicating they carry crucial information for shaping the decision boundary. Thus, we define a function $\mathcal{F} : \mathcal{D} \to \mathbb{R}$ that assigns each sample a forgettability score. A coreset constructed by selecting the top M samples with the highest forgettability scores focuses on the most influential and informative samples.

The main disadvantages of using forgetting score is that it requires excessive computation, and that it does not have theoretical guarantees of selection like CRESTYang et al. [2023]. However, forgetting score is conceptually clear and easy to implement so we use it to gain preliminary insights into our question. In CREST and related works, forgetting score is used to compare methods, so there is precedent for using forgetting scores as a coreset proxy. In our testing, we select different proportions of the highest and lowest forgetting scores in our dataset to use as coresets. In addition to this, forgetting score helps us retain the harder-to-learn examples, as they are the ones that are forgotten after being identified correctly. This helps support a model during pruning, as the pruning is less likely to do worse on hard-to-learn examples.

3.3 Pruning

Simultaneously, we want to reduce the number of parameters in θ. A pruned model θ' has significantly fewer nonzero parameters than θ:

$$\|\theta'\|_0 = p' \ll p.$$

We can impose a pruning mask $m \in \{0, 1\}^p$ that selects a subset of parameters to keep. The pruned model is $f_{\theta \odot m}$, where $\odot$ denotes element-wise multiplication. Our aim is to find a pruning pattern m that maintains model accuracy while drastically reducing parameter count.

We chose to use one-shot pruning because of its quicker speed and requiring less compute than other sophisticated pruning methods. Furthermore, since our project investigates the effects of pruning on neural networks which make use of coreset selection methods we chose to use one-shot pruning to better gauge if pruning methods without iterations would worsen accuracy.

3.4 Joint Optimization Objective

Combining coreset selection and pruning, we seek a tuple $(\mathcal{C}, m)$ that simultaneously reduces dataset size and model complexity. We want:

$$\min_{m,\mathcal{C}} \frac{1}{|\mathcal{C}|} \sum_{(x_i,y_i)\in\mathcal{C}} \ell(f_{\theta\odot m}(x_i), y_i),$$

subject to $|\mathcal{C}| = M \ll N$ and $\|m\|_0 = p' \ll p$.

The challenge lies in finding this reduced training set $\mathcal{C}$ and pruning pattern m that preserve performance. By integrating forgettability-based selection to identify the most challenging and informative samples, we can ensure that the reduced dataset still provides a rich learning signal.

4 Methodology

Our methodology integrates coreset selection and network pruning into a unified framework. The central idea is to leverage a forgettability-driven metric for data selection, ensuring that the reduced training set retains the most informative and challenging samples. Subsequently, we train a model on this coreset, prune it, and finally fine-tune the pruned model on the entire dataset for evaluation. We evaluate this approach using ResNet18 and ResNet50 [He et al., 2016] on CIFAR-10 [Krizhevsky, 2009], and LeNet [LeCun et al., 1998] and a simple fully-connected (FC) network with one hidden layer on MNIST. This diverse set of datasets and architectures allows us to test the generality and robustness of our method.

4.1 Step 1: Calculating Forgetting Scores

We begin by training a baseline model f_θ on the full dataset $\mathcal{D}$ for 10 epochs. During this training phase, we track *forgetting events* for each sample $(x_i, y_i) \in \mathcal{D}$. Let f_θ^t represent the model parameters after the t-th epoch. A forgetting event occurs when a sample transitions from being correctly classified to misclassified. By aggregating these events across training epochs, we compute a *forgettability score* for each sample, reflecting how frequently it is forgotten. Samples with higher forgettability scores are considered more challenging and likely pivotal for shaping the model's decision boundary.

4.2 Step 2: Forgettability-Based Coreset Selection

Using the computed forgetting scores, we construct a coreset $\mathcal{C}$ by selecting the top M samples with the highest forgettability scores, where $M \ll N$. These high-forgettability samples capture the dataset's most informative and challenging examples. Additionally, we also create a coreset by selecting the samples with the lowest forgetability score, which capture the easiest examples in the dataset which might be useful early in training. To investigate the impact of data selection, we evaluate varying data retention ratios $M/N \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. This step ensures that even with significantly reduced datasets, the retained examples provide a strong signal for effective training.

4.3 Step 3: Training the Model on the Coreset

Once the coresets are selected, we train the model f_θ from scratch using only the coreset samples for both coresets for 10 epochs. This reduced training phase enables faster convergence and significantly reduces computational costs compared to training on the full dataset. We apply this step across all architectures (ResNet18, ResNet50, LeNet, and FC net) and datasets (CIFAR-10, CIFAR-100 and MNIST) to ensure broad applicability.

4.4 Step 4: Model Pruning

After training the models on the coresets, we prune them to reduce network complexity. We apply magnitude-based pruning, where weights are ranked by absolute value, and the lowest-magnitude weights are set to zero. Formally, we use a pruning mask $m \in \{0, 1\}^p$ such that:

$$\theta' = \theta \odot m,$$

where $\odot$ denotes element-wise multiplication. This process reduces the number of active parameters in the model, resulting in a smaller, more efficient architecture. We evaluate multiple pruning ratios $\{0, 0.2, 0.4, 0.6, 0.8\}$ to analyze the trade-off between compression and accuracy.

4.5 Step 5: Fine-Tuning on the Full Dataset

Finally, we fine-tune the pruned models $f_{\theta'}$ using the full dataset $\mathcal{D}$. This step helps the pruned model adapt to the complete data distribution, ensuring that the final evaluation reflects its generalization performance. By combining coreset training with fine-tuning, we balance computational efficiency during early training stages with high performance during final evaluation.

4.6 Summary of the Pipeline

- **Step 1:** Train the full model on the entire dataset to compute forgetting scores for each sample.
- **Step 2:** Construct two coresets by selecting top and bottom forgettability samples, varying data ratios in $\{0.1, 0.2, 0.3, 0.4, 0.5\}$.
- **Step 3:** Train two models on both the coresets to efficiently learn from the most challenging examples.
- **Step 4:** Prune the models at different pruning ratios $\{0, 0.2, 0.4, 0.6, 0.8\}$.
- **Step 5:** Fine-tune the pruned models on the full dataset and evaluate their performance.

By systematically varying pruning and data ratios, and experimenting with multiple architectures and datasets, we investigate the synergy between forgettability-driven coreset selection and network pruning. Our results show that carefully selecting hard examples and fine-tuning pruned models can preserve high accuracy while reducing data and model size significantly.

5 Results

5.1 Discussion

Our results vary the most by the model and dataset we train. Training ResNet18 with CIFAR-10 showed no apparent difference in fine-tuning accuracy between the baseline model and the models with data selection. These models reached close to the same accuracy both before and after pruning. Perhaps the dataset is too easy for the model to learn so training on coresets is less impactful in these cases. To contrast, training ResNet18 with CIFAR-100 showed that across the board, models with data selection performed better during fine-tuning (Figure 1), despite under performing before pruning (Figure 2). It is interesting that the high and low forgetting score models behaved similarly with respect to the baseline (Table 1). This configuration shows promise in supporting that training on coresets could improve the model. It might be that training on a smaller dataset allows the model to be more flexible when fine tuned. ResNet50 with CIFAR-10 paints the opposite picture - in these cases, the models with data selection struggle to keep up to the baseline during fine-tuning, and underperform before pruning. Here, fine-tuning convergence is less apparent than in the other models. We trained two additional models on the MNIST dataset - FCNet and LeNet. These behave similar to ResNet50 with CIFAR-10 and ResNet18 with CIFAR-10 respectively. We hesitate to make conclusive statements about these results as they don't cleanly map to our hypotheses, so further experimentation is required. However, these results could suggest that implementing coresets into a pruning workflow improves the accuracy and retrainability of models.

5.2 Code Repository

The following repository contains our full results for LeNet, FCNet, ResNet18, ResNet50 on MNIST, CIFAR-10, CIFAR-100, as well as our project code. `https://github.com/VigneshPreetham/CS-260D_DropHot`

6 Limitations

We did not have enough compute to explore further epochs and learn how pruning a model would cause it to underperform later in training. Furthermore, different datasets and models may give different results because a dataset with mislabeled data can potentially lower accuracy. This is because our forgettability score may select the highest forgotten examples and be less robust against mislabeled data, which can obscure away from the effects of pruning. Additionally, different models can be affected more by one-shot pruning than others, especially because we are not trying to iterate through pruning methods and judging which one performs best.

7 Conclusion

Pruning addresses the increasing complexity of network-based models by reducing parameter size. Through experimentation, our framework shows promise in improving the performance of pruned models by incorporating coresets. In future works, it is necessary to confirm the validity of these results with rigorous statistical hypothesis testing. Future contributors should investigate how re-initializing the models from random effects the performance after pruning. One should also see if introducing coresets through a theoretically rigorous framework like CREST changes the magnitude of the results. Finally, additional controls should be implemented to account for variables like dataset size. These contributions would strengthen the material discussed in this paper.

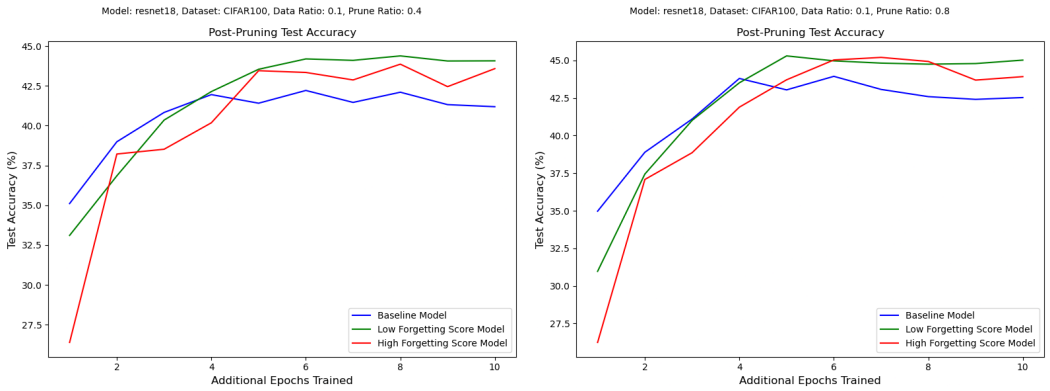

Figure 1: Comparing the test accuracies of post-pruned models to the baseline model over 10 additional training epochs.

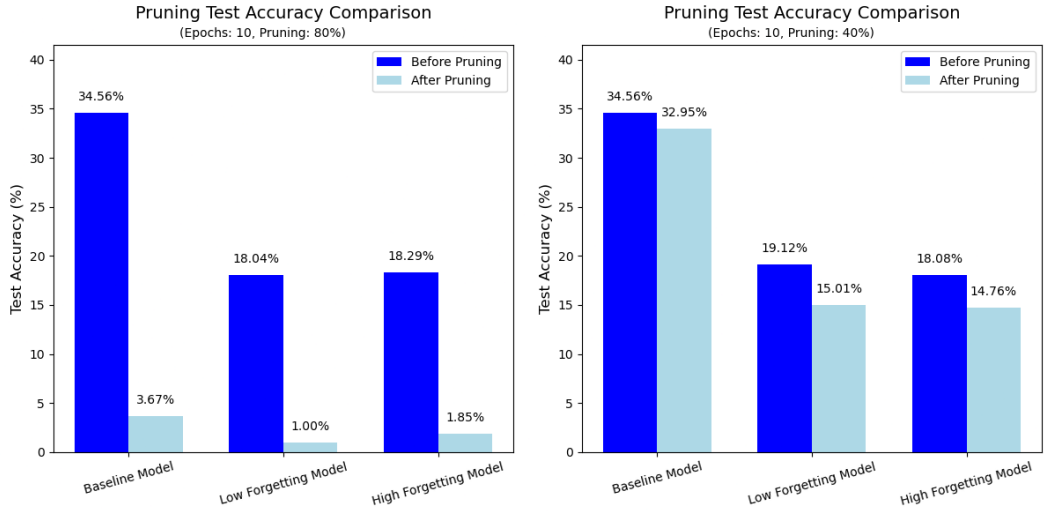

Figure 2: Comparing the test accuracies after training for 10 epochs of ResNet18 models on CIFAR-100, between pre-pruned and post-pruned models trained on 10% data retention using forgetting scores.

Data Ratio	Prune Ratio	Low - Base (%)	High - Base (%)
0.1	0.4	2.88	2.39
0.1	0.6	2.5	1.678
0.1	0.8	2.49	1.39
0.2	0.4	1.74	2.27
0.2	0.6	1.7	0.35
0.2	0.8	2.52	1.76
0.3	0.4	-0.19	0.41
0.3	0.6	0.82	0.81
0.3	0.8	1.021	0.89

Table 1: Difference in test accuracies between models using data selected by low-forgetting, high-forgetting scores and baseline for various data and prune ratios. Accuracies measured after 10 additional epochs of training on post-pruned models (Resnet18 on CIFAR-100)

8 Contributions

Idea: All group members came up with the idea together during a meeting. Implementation: All group members equally contributed to the implementation of the code. Writeup: All group members worked on various different sections of the report. Presentation: We worked on the slides together during a meeting, and they were split among the group members for live presentation.

References

Olivier Bachem, Mario Lucic, and Andreas Krause. Practical coreset constructions for machine learning. In *Advances in Neural Information Processing Systems*, pages 2789–2797, 2017.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision*, pages 532–547, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Dan Feldman and Michael Langberg. Scalable training of mixture models via coresets. In *Annual Symposium on Foundations of Computer Science*, pages 1–10, 2011.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL `https://arxiv.org/abs/1803.03635`.

Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv:1902.09574*, 2019.

Song Han, Jeff Pool, John Tran, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations*, 2016. URL `https://arxiv.org/abs/1510.00149`.

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 (CVPR)*, pages 770–778, 2016.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *International Conference on Computer Vision*, pages 1389–1397, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Y. Li, L. Huang, and et al. SGQuant: Squeezing the last bit on AI accelerator data precision via Sinusoidal Graphs quantization. In *ASPLOS*, pages 733–747, 2020.

Baharan Mirzasoleiman, Jeff Bilmes, and Purnamrita Sarkar. Coresets for data-efficient training of machine learning models. In *Advances in Neural Information Processing Systems*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, and David et al. Silver. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Vivienne Sze, Yu-Hsin Chen, Joel Yang, and Joel S. Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

Mariya Toneva, Alessandro Sordoni, and Rémi et al. Tachet des Combes. An empirical study of forgetting in deep neural networks. In *International Conference on Learning Representations*, 2019.

Yu Yang, Hao Kang, and Baharan Mirzasoleiman. Towards sustainable learning: Coresets for data-efficient deep learning, 2023. URL `https://arxiv.org/abs/2306.01244`.