

OPTIMIZING MODEL PRUNING FOR ENERGY-EFFICIENT DEEP LEARNING

SAIDJON KAMOLOV^{1*}

ABSTRACT. Deep learning models have become increasingly complex, leading to high energy consumption during training and inference phases. This paper investigates the use of model pruning techniques to significantly reduce the size of deep learning models while preserving predictive accuracy. Our study focuses on optimizing the trade-off between model performance and energy consumption by systematically applying various pruning strategies—magnitude-based, structured, and random pruning—on popular neural network architectures. Using the CIFAR-10 dataset, we evaluate the models' energy consumption during training and inference phases. The results provide insights into achieving substantial energy savings with minimal loss of accuracy, contributing to the development of more sustainable and efficient AI systems.

1. INTRODUCTION

The rapid advancement of deep learning has revolutionized numerous fields, including computer vision, natural language processing, and speech recognition. Convolutional Neural Networks (CNNs), in particular, have demonstrated remarkable performance in image classification tasks. Models such as ResNet-18 and VGG-16 have become benchmarks due to their high accuracy on datasets like CIFAR-10 and ImageNet. However, this increased performance often comes at the cost of substantial computational resources and energy consumption, raising concerns about the environmental impact and sustainability of deploying such models at scale.

Energy consumption in deep learning is not only a financial concern but also an environmental one. Training large neural networks demands significant electricity, which, depending on the energy source, can result in considerable carbon emissions. As the adoption of AI technologies grows, so does the need for more energy-efficient models that can reduce the carbon footprint associated with training and deploying deep learning systems.

Model pruning has emerged as a promising approach to address this challenge. By removing redundant or less significant parameters from neural networks, pruning can reduce model size and computational complexity without substantially affecting performance. This results in faster inference times and lower energy

Date: Received: Oct 18, 2023; Accepted: Dec 2, 2024.

*** Corresponding author.

Key words and phrases. pruning, CNN, deep learning, energy efficiency, ResNet, VGG.

consumption, making it an attractive solution for deploying deep learning models on resource-constrained devices and for large-scale applications where energy efficiency is critical.

Previous research has explored various pruning techniques, including magnitude-based pruning, which removes weights with the smallest absolute values; structured pruning, which removes entire neurons, filters, or layers; and random pruning, which removes weights randomly to serve as a baseline comparison. While these methods have shown potential, a systematic analysis comparing their effectiveness in reducing energy consumption without compromising accuracy is needed.

This paper aims to fill this gap by conducting an in-depth study on the impact of different pruning techniques on the energy efficiency of deep learning models. We focus on two widely used CNN architectures—ResNet-18 and VGG-16—and evaluate them on the CIFAR-10 dataset. Our study measures the trade-offs between model accuracy, size reduction, execution time (used as a proxy for energy consumption), and inference speed. By providing detailed experimental results and analyses, we offer valuable insights into how pruning can be optimized for energy-efficient deep learning.

Our contributions are as follows:

- (1) We implement and compare three pruning techniques—magnitude-based, structured, and random pruning—at various sparsity levels on ResNet-18 and VGG-16 models.
- (2) We measure the execution time during training and inference as a substitute for direct energy consumption measurement, due to limitations in the experimental environment.
- (3) We analyze the trade-offs between model size, accuracy, and execution time to identify the most effective pruning strategies for energy efficiency.
- (4) We provide practical guidelines for applying model pruning to achieve energy savings in deep learning applications.

By addressing the growing need for sustainable AI practices, this research contributes to the development of greener computing solutions in the field of machine learning.

2. LITERATURE

Recent research in model pruning has focused on improving both the energy efficiency and computational performance of deep learning models, particularly in the context of resource-constrained environments. One widely studied technique is magnitude-based pruning, where less significant weights in a network are pruned based on their magnitude. This approach has been shown to significantly reduce the number of parameters while maintaining competitive accuracy, especially in models like ResNet and VGG when trained on datasets such as CIFAR-10 and CIFAR-100. Studies such as those by Li et al. (2020) [5] demonstrate that by pruning redundant filters, models can reduce inference time and energy consumption without significant performance loss. Additionally, using fine-tuning post-pruning further enhances model recovery [5, 4].

Another promising direction is structured pruning, where entire filters or layers are removed, allowing for more substantial reductions in model complexity. This approach has been particularly effective in deep networks like VGG-16 and ResNet, where structured pruning not only reduces the computational cost but also results in smaller memory footprints. Researchers like He et al. (2020) [4] and Han et al. (2016) [3] have found that structured pruning can lead to faster inference times, making it ideal for deployment on edge devices. However, structured pruning may cause more dramatic performance degradation compared to magnitude-based pruning, necessitating more sophisticated fine-tuning strategies [1].

Energy-aware pruning, which directly takes into account the energy consumption of different parts of the network, has also gained attention in recent years. By evaluating the energy contributions of individual filters or neurons, this method allows for more targeted pruning, ensuring that only the least energy-efficient components are removed. Studies such as those by Yeom (2021) [10] have introduced decomposition methods like Singular Value Decomposition (SVD) to identify energy-heavy filters, leading to more energy-efficient models. These methods have been validated on benchmarks like CIFAR-10 and have shown promise in balancing the trade-offs between energy savings and model performance [1].

Finally, recent surveys of pruning techniques have highlighted the importance of choosing the appropriate pruning strategy depending on the deployment context. For instance, unstructured pruning may achieve high levels of sparsity but often requires specialized hardware for efficient execution. In contrast, structured pruning offers more universal acceleration across a range of hardware architectures. The choice of whether to prune before, during, or after training, as well as the criteria for selecting which weights or filters to prune, continues to be an area of active research, with dynamic and static pruning approaches being explored [1, 8]. These recent advancements in model pruning have established a strong foundation for further research in energy-efficient deep learning.

The issue of increased energy consumption is particularly acute in the case of imbalanced data. The most common approach to dealing with imbalanced data is to oversample the minority class to match the size of the majority class [6, 2, 7, 9]. This can increase the size of the training data by several folds and lead to greater computational and energy load.

3. METHODOLOGY

The methodology of this study is structured into three subsections: the experimental setup, pruning techniques, and evaluation methods. These sections outline the overall approach taken to investigate the impact of various pruning techniques on model efficiency and performance. Specifically, we examine how pruning can reduce execution time, used as a proxy for energy consumption, while maintaining acceptable levels of model accuracy.

3.1. Experimental Setup. The experimental setup includes details on the dataset used, the neural network models selected for the study, and the procedure for training and testing both the baseline and pruned models. The dataset chosen

for this study is CIFAR-10, a widely used benchmark for image classification tasks in computer vision. CIFAR-10 consists of 60,000 color images, each of size 32x32 pixels, divided into 10 classes (e.g., airplane, car, bird, etc.). Of these, 50,000 images are used for training and 10,000 for testing. This dataset provides a sufficiently challenging yet manageable environment for conducting experiments on model pruning within a reasonable timeframe.

For the models, we employ two popular Convolutional Neural Network (CNN) architectures: ResNet-18 and VGG-16. ResNet-18 is a residual network with 18 layers that uses skip connections to improve the flow of gradients, making it well-suited for deeper architectures. VGG-16 is a deeper network with 16 layers and is known for its uniform architecture, characterized by small convolutional filters (3x3) and a sequence of fully connected layers at the end. Both models are modified for the CIFAR-10 dataset by adapting the output layers to have 10 neurons, corresponding to the 10 classes in CIFAR-10. These architectures were chosen because they are representative of deep learning models commonly used in image classification and provide a balanced mix of depth and complexity.

To train the baseline models, we use a standard training pipeline with the Stochastic Gradient Descent (SGD) optimizer, a learning rate of 0.1, momentum of 0.9, and a weight decay of 5×10^{-4} . The learning rate is adjusted throughout training using a step scheduler, which decreases the learning rate by a factor of 0.1 after 30 epochs. The models are trained for 50 epochs on the full CIFAR-10 training set to establish a baseline performance before pruning. Once trained, the baseline models are evaluated on the CIFAR-10 test set, and their accuracy, training time, and inference time are recorded. These values serve as reference points for comparison with the pruned models.

3.2. Pruning Techniques. This study explores three distinct pruning techniques: magnitude-based pruning, structured pruning, and random pruning. Each pruning technique is applied to both ResNet-18 and VGG-16 at different sparsity levels: 10%, 30%, 50%, and 70%. These pruning techniques vary in their approach to removing weights or structural components from the model, and each has different implications for the resulting model size, accuracy, and computational efficiency.

Magnitude-based pruning involves removing weights with the smallest absolute values, based on the assumption that smaller weights contribute less to the model’s predictions. This method is applied globally across the network, meaning that the least significant weights from any layer may be pruned. After pruning, the model is fine-tuned to recover any accuracy loss caused by the removal of weights. Magnitude-based pruning is a popular approach because it directly targets less important parameters, potentially achieving significant reductions in model size with minimal impact on performance.

Structured pruning, on the other hand, removes entire neurons, filters, or layers rather than individual weights. In this study, structured pruning is applied to convolutional filters in the case of ResNet-18 and VGG-16. By removing whole filters, structured pruning reduces the complexity of the network’s architecture, potentially leading to faster inference times due to the smaller size of the pruned

network. However, this method often results in more significant accuracy degradation if not fine-tuned appropriately, as the removal of structural components can have a larger impact on the network’s overall capacity to learn and represent features.

Finally, random pruning is used as a baseline to compare against the more informed approaches of magnitude-based and structured pruning. In random pruning, weights are removed arbitrarily, without consideration of their contribution to the model’s predictions. This technique serves to highlight the importance of applying a principled approach to pruning. Like the other pruning methods, random pruning is followed by a fine-tuning phase to attempt to recover accuracy.

Once pruning is applied, the models are fine-tuned for an additional 10 epochs using a reduced learning rate of 0.01. Fine-tuning allows the pruned models to adjust to the new, sparse structure and recover any performance lost during the pruning process. The same data augmentation and preprocessing steps are used during fine-tuning as in the original training of the baseline models.

3.3. Evaluation and Metrics. The performance of the pruned models is evaluated using several key metrics: accuracy, execution time during training and inference, model size, and parameter reduction. These metrics provide a comprehensive view of the trade-offs between pruning efficiency and model performance. Accuracy is measured as the percentage of correct predictions made by the model on the CIFAR-10 test set. This metric allows us to assess the extent to which pruning affects the predictive capability of the model. Given that the goal of pruning is to reduce computational resources without severely degrading performance, accuracy is a critical measure for determining the success of each pruning method.

Execution time is used as a proxy for energy consumption in this study. Due to limitations in directly measuring energy usage in the Google Colab environment, we rely on the wall-clock time for both training and inference to estimate the computational efficiency of the models. Training time refers to the total time taken to train the pruned models, including the fine-tuning phase, while inference time measures the time required to make predictions on the test set. These times are recorded using Python’s ‘time’ module, and they provide a practical estimate of the potential energy savings that can be achieved through pruning. Although execution time does not account for variations in power consumption, it serves as a reasonable indicator of computational efficiency under the same experimental conditions.

Model size is evaluated in terms of the number of trainable parameters after pruning. This metric reflects the degree to which each pruning method reduces the network’s complexity and memory footprint. By comparing the pruned model’s size to the baseline, we can quantify the parameter reduction achieved through pruning. This is especially relevant for deployment on resource-constrained devices, where memory and storage limitations are critical considerations.

Finally, parameter reduction is expressed as the percentage decrease in the number of trainable parameters relative to the baseline model. This metric helps

TABLE 1. Pruning results for ResNet

Pruning	Sparsity	Accuracy	Training	Inference	Par Reduction (%)
Baseline	0.00	18.20	4.08	0.20	0.00
magnitude	0.10	19.60	1.00	0.20	7.11
magnitude	0.30	19.60	1.03	0.20	21.25
magnitude	0.50	20.00	1.01	0.22	35.11
magnitude	0.70	21.40	1.10	0.31	48.32
structured	0.10	19.20	1.50	0.33	7.63
structured	0.30	17.20	1.05	0.20	24.05
structured	0.50	16.80	1.01	0.21	42.26
structured	0.70	13.40	1.02	0.22	62.05
random	0.10	20.60	1.00	0.21	6.25
random	0.30	19.80	1.02	0.20	18.70
random	0.50	17.00	1.01	0.21	31.13
random	0.70	12.00	1.03	0.22	43.52

to contextualize the savings in computational resources and to understand the relationship between model size and accuracy. By analyzing these metrics together, we are able to identify the pruning strategies that offer the best trade-offs between execution time, model size, and accuracy.

In summary, this study systematically evaluates the effectiveness of three pruning techniques—magnitude-based, structured, and random—at various sparsity levels on two widely used CNN architectures, ResNet-18 and VGG-16, trained on the CIFAR-10 dataset. By using execution time as a proxy for energy consumption and measuring accuracy, model size, and parameter reduction, we aim to provide a comprehensive analysis of the trade-offs involved in optimizing deep learning models for energy efficiency.

4. RESULTS AND DISCUSSION

The results presented in Table 1 illustrate the effects of different pruning methods on the ResNet model’s performance across varying sparsity levels. Notably, magnitude-based pruning demonstrated an improvement in accuracy compared to the baseline, achieving the highest accuracy of 21.40% at a 70% sparsity level, along with a significant parameter reduction of 48.32%. This suggests that pruning less significant weights can enhance model performance while reducing complexity. In contrast, structured pruning showed a decline in accuracy as sparsity increased, dropping from 19.20% at 10% sparsity to 13.40% at 70% sparsity, indicating that removing entire structural components may impair the model’s ability to learn. Random pruning yielded mixed outcomes, with a slight accuracy improvement at low sparsity levels but a notable decrease at higher levels, mirroring the trend observed with structured pruning. Training times for pruned models were consistently lower than the baseline, reflecting computational efficiency gains due to reduced model size. However, inference times did not decrease proportionally and, in some cases, increased slightly, possibly due to overheads

TABLE 2. Pruning results for VGG

Pruning	Sparsity	Accuracy	Training	Inference	Par Reduction (%)
Baseline	0.00	18.40	8.38	0.26	0.00
magnitude	0.10	17.80	2.05	0.26	2.44
magnitude	0.30	18.60	2.06	0.26	7.53
magnitude	0.50	18.60	2.14	0.37	12.56
magnitude	0.70	18.20	2.06	0.26	17.85
structured	0.10	17.20	2.05	0.26	0.78
structured	0.30	10.60	2.05	0.26	2.57
structured	0.50	10.60	2.04	0.26	4.66
structured	0.70	10.60	2.09	0.35	6.84
random	0.10	17.80	2.20	0.29	2.70
random	0.30	10.60	2.04	0.26	22.40
random	0.50	10.60	2.05	0.26	40.66
random	0.70	10.60	2.00	0.26	60.11

associated with sparse computations. Overall, magnitude-based pruning emerged as the most effective method, balancing parameter reduction and accuracy enhancement, thereby contributing to more efficient deep learning models without substantially compromising performance.

The results detailed in Table 2 reveal the impact of various pruning methods on the VGG model’s performance across different sparsity levels. Magnitude-based pruning consistently maintained accuracy close to the baseline of 18.40%, even at higher sparsity levels; for instance, at 70% sparsity, the accuracy only slightly decreased to 18.20%, with a substantial parameter reduction of 17.85%. This indicates that removing weights with the smallest magnitudes does not significantly impair the model’s predictive capabilities and effectively reduces model complexity. In contrast, structured pruning led to a notable decline in accuracy as sparsity increased, dropping to 10.60% at sparsity levels of 30% and above, suggesting that eliminating entire structural components adversely affects the model’s capacity to learn salient features. Random pruning exhibited a similar trend, with accuracy diminishing to 10.60% at sparsity levels of 30% and higher, despite achieving a higher parameter reduction of up to 60.11% at 70% sparsity. Training times for pruned models were markedly reduced compared to the baseline, decreasing from 8.38 seconds to approximately 2.05 seconds, reflecting enhanced computational efficiency due to the reduced number of active parameters. Inference times remained relatively consistent across all configurations, indicating that the pruning did not significantly affect the model’s runtime performance during prediction. Overall, magnitude-based pruning emerged as the most effective method for the VGG model, achieving a balance between maintaining accuracy and reducing model size, thereby enhancing efficiency without substantially compromising performance.

REFERENCES

- [1] Cheng, H., Zhang, M., & Shi, J. Q. (2024). A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. <https://doi.org/10.1109/TPAMI.2024.3447085>
- [2] Elreedy, D., Atiya, A. F., & Kamalov, F. (2024). A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning. *Machine Learning*, 113(7), 4903-4923. <https://doi.org/10.1007/s10994-022-06296-4>
- [3] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*. <https://doi.org/10.48550/arXiv.1510.00149>
- [4] He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 1389-1397). <https://doi.org/10.1109/ICCV.2017.155>
- [5] Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2022, July). Pruning Filters for Efficient ConvNets. In *International Conference on Learning Representations*. <https://doi.org/10.48550/arXiv.1608.08710>
- [6] Kamalov, F., Thabtah, F., & Leung, H. H. (2023). Feature selection in imbalanced data. *Annals of Data Science*, 10(6), 1527-1541. <https://doi.org/10.1007/s40745-021-00366-5>
- [7] Kamalov, F. (2024). Asymptotic behavior of SMOTE-generated samples using order statistics. *Gulf Journal of Mathematics*, 17(2), 327-336. <https://doi.org/10.56947/gjom.v17i2.2343>
- [8] Liu, Z., Sun, M., Zhou, T., Huang, G., & Darrell, T. Rethinking the Value of Network Pruning. In *International Conference on Learning Representations*. <https://doi.org/10.48550/arXiv.1810.05270>
- [9] Thabtah, F., & Kamalov, F. (2017). Phishing detection: a case analysis on classifiers with rules using machine learning. *Journal of Information & Knowledge Management*, 16(04), 1750034. <https://doi.org/10.1142/S0219649217500344>
- [10] Yeom, S.K., Shim, K.H. & Hwang, J.H., 2021. Toward compact deep neural networks via energy-aware pruning. *arXiv preprint arXiv:2103.10858*. <https://doi.org/10.48550/arXiv.2103.10858>

¹FACULTY OF ENGINEERING, TAJIK TECHNICAL UNIVERSITY, DUSHANBE, TAJIKISTAN.
Email address: said.kamolov@ttu.tj