# Grow, prune or select data: which technique allows the most energy-efficient neural network training?

Anais Boumendil, Walid Bechkit, Pierre-Edouard Portier, Frédéric Le Mouël, Malcolm Egan

# Grow, prune or select data: which technique allows the most energy-efficient neural network training?

Anais Boumendil
*Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France*
anais.boumendil@insa-lyon.fr

Walid Bechkit
*Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France*
walid.bechkit@insa-lyon.fr

Pierre-Edouard Portier
*Univ Lyon, INSA Lyon, CNRS, LIRIS UMR5205, 20 Avenue Albert Einstein, Villeurbanne, F-69621, Rhône, France*
pierre-edouard.portier@insa-lyon.fr

Frédéric Le Mouël
*Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France*
frederic.le-mouel@insa-lyon.fr

Malcolm Egan
*Univ Lyon, Inria, INSA Lyon, CITI, EA3720, 69621 Villeurbanne, France*
malcom.egan@inria.fr

***Abstract*—The training energy efficiency of deep neural networks became an extensively studied research topic in the last years. Some of the existing approaches seek to reduce the size of the architecture by either starting the training with a large network and pruning it, or by beginning with a seed architecture and then growing it. Instead of compressing the architecture, other approaches aim to reduce the number of training examples through data selection.**

**While various approaches belonging to these two categories have been proposed, only a few works actually conduct energy measurements. Others merely mention potential gains in efficiency or rely on alternative evaluation metrics such as FLOPs. In this paper, we conduct a series of experiments both on a synthetic dataset and on image classification benchmarks in order to compare the impact of pruning, architecture growing and data selection on training energy consumption and prediction quality.**

**Our results show that growing maintains a high prediction quality but brings limited energy gains when the size of the resulting architecture is large. Pruning can offer high gains, but also impacts accuracy, making it more suited for large models. Data selection provides energy gains correlated with the selectivity rate but causes an accuracy loss. We find that the effectiveness of every technique depends on its hyperparameters and on the architecture size.**

***Index Terms*—Neural networks, data selection, pruning, growing, energy efficiency.**

## I. Introduction

Deep neural networks (DNNs) high prediction quality made them indispensable to many domains. However, their energy consumption, especially during the training phase, is an important obstacle to their integration to resource-constrained devices and has a worrying environmental impact [1], [2]. Therefore, the energy efficiency of neural networks became an extensively studied topic as many approaches were proposed to optimize their energy consumption. The first work dealt mainly with the inference phase, but the number of studies that focus on training, the most energy-intensive step, is growing. These approaches act through different ways as designing accelerators [3], building optimized architectures manually [4] or automatically through neural architecture search [5] or through growing that starts the training with a small architecture and increases its size progressively during training [6], [7]. Some techniques aim to compress existing models, notably by pruning some parameters [8], [9]. Other approaches act on the data level and aim to train the network on fewer examples [10], [11].

Some techniques are more explored than others. For instance, pruning is deeply studied while growing, the opposite technique, is less explored. Data selection has mainly been used to reduce training time, and only few works [11] evaluate its impact on energy. Moreover, many studies use metrics such as FLOPs or MACs (multiply-accumulate) to evaluate efficiency. However, the latter don't provide the best approximation of energy consumption [12], [13].

In this paper, we address DNNs training energy efficiency by providing a comparison between data selection, growing and pruning. We first review some of the existing work on the three approaches. Then, we implement representative methods from the literature for each approach and we study their impact on training energy consumption and prediction quality through two groups of experiments: tests on a synthetic dataset and tests on image classification benchmarks.

## II. Related work

In this paper, we mainly target the enhancement of training energy efficiency, for which many approaches were proposed. Specifically, we study techniques that reduce the dataset size (data selection) or the architecture size (pruning and growing).

Data selection supports model training with fewer examples. It is mainly used to accelerate training [10], [14], [15] which indirectly reduces energy consumption. Few works still apply it with the goal of saving energy [11], [16]. In this paper, we are interested in adaptive data selection that renews the subset of selected examples after a certain number of epochs. Selection can rely on the model's loss [14] or gradients [11]

or even be random. For instance, the work in [16] shows that a random batch skipping with a probability of 50% keeps a high accuracy and reduces training energy consumption by half.

Pruning starts the training with a large model and aims to compress it by removing parameters. This approach is extensively studied in the literature and has first targeted the inference phase, as early techniques propose to prune the network after training [17]. To generalize pruning benefits to training, it is necessary to prune early during the process or even before it, as done by pruning at initialization techniques [8], [18]. Moreover, pruning can be either unstructured when individual weights are removed [18], [19] or structured when groups of weights such as filters/channels [9], [17], [20] or even layers [21] are removed.

Growing [6], [7], [22], [23] consists in starting the training with a small seed architecture and then increasing its size by adding neurons. It can be seen as the opposite technique of pruning [22]. Growing approaches propose to jointly optimize the architecture and its weights [6] during training. Among growing techniques, the work in [24] proposes to detect expressivity bottlenecks and to fix them by adding neurons while GradMax [6] initializes new neurons in a way that maximizes the gradient norm as it allows a higher loss decrease.

## III. Methodology

In this section, we introduce the implemented approaches as well as the tasks on which we compare the different techniques.

### A. Implemented approaches

We implement representative techniques from the literature for each studied approach. In order to cover multiple pruning types, we implement a structured approach that prunes during training and an unstructured approach that prunes at initialization.

*1) Data selection through random batch skipping:* For data selection, we implement the approach proposed in [16]. This work presents a three-level framework with a low precision backpropagation, a selective layer update and data selection. We only consider the data level of the framework, which provides a very simple way to select examples. During an epoch, each batch is skipped, randomly, with a probability $\alpha$ (always set to 50% in the original paper). When a batch is skipped, we still perform a weight update with the gradients of the latest selected batch. This random selection can achieve a high accuracy and can reduce the training energy consumption by up to 50%. The reason behind this result is that skipping batches adds a sampling random noise, which can help to escape saddle points [16]. We specifically implement the approach of [16] because it has the lowest overhead, when other selection strategies based on more complex criteria perform additional computations and impose higher selection costs.

*2) Growing through maximizing the gradient norm:* For the growing technique, we implement the GradMax approach [6] which provides a way to initialize new neurons. GradMax first preserves the previous information learned by the network by setting the incoming weights of the new neurons to zero[1]. Preserving the learned information is a desirable property for growing algorithms. The GradMax strategy also aims to maximize the gradient norm, since it allows a higher decrease of the training objective [6]. To do so, the outgoing weights are initialized based on the Singular Value Decomposition (SVD)[2].

GradMax starts the training with a small seed architecture (*baseline small*). The network architecture is updated periodically during training and neurons (or filters) are added to the layers. After a given number of growth steps, the size of the network reaches the size of a target model (*baseline big*) and the training continues. The GradMax work provides a simple and a fast growing strategy that achieves a high accuracy, which motivated us to further study this approach.

*3) Unstructured pruning through random weight removal:* For unstructured pruning, we opt for a strategy that removes weights before training. A recent approach [25], proposes to use a random pruning technique. The latter was always seen as a baseline for comparison. The authors of [25] however show that random pruning at initialization with specific layer-wise ratios (percentage of weights to prune from every layer) can be effective. We implement this approach and we apply the ER (Erdös-Rényi) ratio [26] which associates larger layers with bigger pruning rates. This pruning method achieves a high accuracy, especially when the network architecture is deep [25]. Moreover, since the pruning is random, the technique should only add a small overhead.

*4) Structured pruning through removing filters with lowest L1-norm:* For structured pruning, we opt for a strategy that removes filters from a Convolutional Neural Network (CNN). We implement an adapted pruning algorithm to study its impact on training efficiency. The implemented strategy prunes the network during the first part of the training and makes the model thinner by removing filters. As in [17], we prune the filters with the lowest $l_1$ norm.

In order to allow a fair comparison with GradMax [6], we adopt the same architecture updating schedule. Thus, periodically, we remove a given number of neurons (or filters) from the layers. Implementing a pruning method which is exactly the opposite schedule of the GradMax technique allows us to find out if it is better, in terms of the accuracy/energy trade-off, to start with a large architecture and then prune it or to do the opposite.

### B. Considered experiments

We study data selection, growing and pruning through two types of experiments: Teacher student tests that involve small fully connected networks and a synthetic dataset, as well as image classification tests on benchmark CNNs and datasets.

[1]The original paper [6] also provides a variant where the outgoing weights are set to zero.

[2]The original paper [6] also provides a variant where weights are initialized with an iterative method such as projected gradient descent.

*1) Teacher student experiments:* The teacher student test [6], [27] involves two neural networks. The first one is called the teacher network $N_t$ and is initialized randomly. This network is not trained and is only used to generate a synthetic dataset. Indeed, we generate $n$ training data points $X$ and we pass them through the teacher network to generate the target points $Y$ [6]. The second network, called student model $N_s$, is trained in order to mimic the teacher's distribution by minimizing the squared loss between $N_s(X)$ and $Y$ [6].

For the teacher student test, we only consider fully connected networks containing $n_i$ input neurons, $n_h$ hidden neurons and $n_o$ output neurons that we denote $n_i : n_h : n_o$ as in [6]. As the implemented structured pruning technique targets CNNs, we do not apply it for the teacher student test. We use a 100:50:10 architecture for the teacher network and for the data selection student model. For GradMax, we start with a 100:25:10 architecture for the student that is grown to match the size of the teacher network. For unstructured pruning, we start with a 100:100:10 architecture for the student model. We prune 50% of the weights through masking in order to obtain an equivalent size to 100:50:10 in terms of non-zero weights. We start the training with a bigger network for pruning in order to obtain a similar final network size (in terms of the number of non-zero weights) between growing and pruning.

As in [6], we sample $n = 1000$ training points from a Gaussian distribution with mean 0 and unit variance. We use a learning rate of 0.01, a batch size of 100, Stochastic Gradient Descent (SGD) as an optimizer and we train the student network for 3500 iterations [6]. For GradMax, we keep the same settings as in [6]. The network growing starts at iteration 500. We have 5 growth steps every 500 iterations. After the last growth, 1000 iterations are performed.

*2) Image classification experiments:* In addition to the teacher student experiments, we also compare the studied approaches on image classification benchmarks. We use the datasets CIFAR-10 and CIFAR-100 and we consider the Wide-ResNet (WRN) [28] and VGG [29] architectures. Specifically, we use the implementations of [6] of the VGG-11 and WRN-28-1 variants (with this implementation of WRN, the second batch normalization layer of the block is removed). This choice of architectures allows us to cover two types of Convolutional Networks, namely CNNs with and without residual connections. For the hyperparameters, we use the same settings as [6]. We train all networks for 200 epochs with SGD and a 0.9-momentum. We use a batch size of 128. We use a learning rate of 0.1 for WRN-28-1. For VGG-11, we had to tune the learning rate and we found that the best value is 0.01 for data selection and 0.05 for the other approaches. For both models, we decay the learning rate with a cosine schedule. For the GradMax approach, we follow [6] and we set the number of filters of the seed architecture to $\beta = 1/4$ of the number of filters of *baseline big*. We shrink the dimensions of all the layers of the VGG-11 model, but we only shrink the first layer of every block in WRN-28-1 model to keep correct dimensions for the skip connections, as done in [6]. We use the same settings for the filter pruning approach but with an opposite network size evolution. The growing/pruning starts at iteration 10000 and we then have 12 grow/prune steps with an interval of 2500 iterations [6]. Growing/pruning finishes by the middle of training. For data selection, we set $\alpha = 50\%$ as in the original work [16] but we also study the performances of lower selectivity ratios.

### C. Experimental settings

We use TensorFlow to implement all the studied approaches. Our code will be available after publication. We perform our experiments on a MacBook Pro with an Apple M2 Max processor and 64 GB of RAM. All the trainings were executed on CPU. To measure energy consumption, we used the "powermetrics"[3] command. The latter is a pre-installed tool on macOS that gathers many statistics as CPU usage and power consumption.

## IV. Teacher student experimental results

In this section, we analyze the results of the teacher student experiments averaged over 300 runs. We plot the training curves in Figure 1. Dividing the train set into batches gave curves with some variations. For more clarity, we plot a smoothed version of the curves in Figure 2 obtained with the SciPy implementation of the Savitzky-Golay algorithm [30].

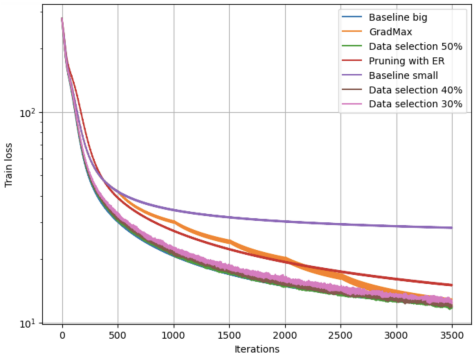


Fig. 1. Training curves averaged over 300 runs of the teacher student test before smoothing.

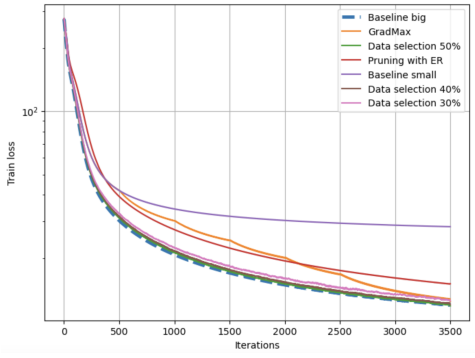


Fig. 2. Training curves averaged over 300 runs of the teacher student test after smoothing.

[3]https://www.unix.com/man-page/osx/1/powermetrics/

TABLE I
ENERGY MEASUREMENTS FOR THE TEACHER STUDENT TESTS

| Approach | Energy consumption (joule) |
|---|---|
| *Baseline small* | 17.94 |
| Data selection 30% | 6.03 |
| Data selection 40% | 8.85 |
| Data selection 50% | 10.21 |
| GradMax | 18.41 |
| Unstructured pruning | 20.58 |
| *Baseline big* | 18.47 |

We observe that all the approaches outperform *baseline small*. We mainly notice a very similar evolution between *baseline big* and the data selection with 50% and 40% rates. We notice a small loss increase with a ratio of 30%. Meanwhile, ratios of 20% and 10% can't be considered for this task as they cause an important increase in loss and harm the learning process.

For the growing technique, the number of neurons increases as the training progresses and the resulting training loss gradually decreases until it becomes lower than that of pruning. After iteration 2500 (last growth step), GradMax outperforms pruning, which gives the highest final training loss among all the studied approaches despite starting the training with a bigger model. This can be related to the small size of the network. Indeed, it is well known that pruning works better on over-parameterized architectures.

Table I contains the average training energy consumption for all approaches. We observe that data selection offers the highest energy gains and that the lower the selection rate, the lower the energy consumption. GradMax offers a smaller energy consumption when compared to *baseline big* but the savings brought by the approach are limited. Indeed, even the energy consumption of *baseline small* and *baseline big* are close due to the small size of the networks used in experiments. Pruning provides a higher energy consumption than *baseline big* as it starts with a larger model and consumes energy for executing the pruning step at initialization. Moreover, unstructured pruning only sets pruned weights to zero and doesn't shrink dimensions. Therefore, it is necessary to use specialized hardware or libraries to optimize the sparse matrices generated by unstructured pruning to obtain energy savings.

## V. IMAGE CLASSIFICATION EXPERIMENTAL RESULTS

As in [6], we consider three combinations of datasets and CNNs: (WRN-28-1, CIFAR-10), (WRN-28-1, CIFAR-100) and (VGG-11, CIFAR-10) [28], [29], [31]. All the results that we provide in this section are averaged over 3 runs. As unstructured pruning requires specialized hardware or libraries to take advantage of sparse matrices to show energy gains [12], we do not consider this approach in the following tests. We focus on the structured technique presented in section III-A4.

### A. Results on WRN-28-1

We compare the final test accuracy, the training time and training energy consumption for all the studied approaches, as well as *baseline big* and *baseline small*.

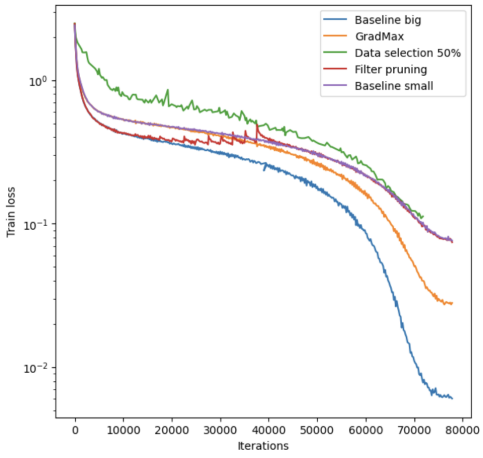


Fig. 3. Training loss curves for (WRN-28-1, CIFAR-10).

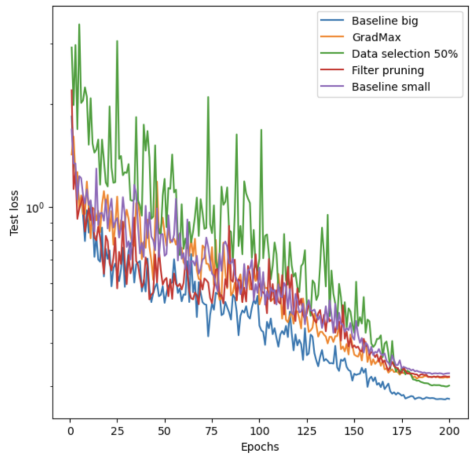


Fig. 4. Test loss curves for (WRN-28-1, CIFAR-10).

We plot the training and test loss curves respectively in Figure 3 and 4 for (WRN-28-1, CIFAR-10). We also summarize the results in Table II for the configuration (WRN-28-1, CIFAR-10) and in Table III for the configuration (WRN-28-1, CIFAR-100).

We observe that the data selection technique brings the highest energy gains as it reduces the costs by half. Moreover, since the random selection has a negligible overhead, the energy gain is almost equal to the selectivity ratio. We observe that data selection causes an accuracy decrease and presents higher train and test losses than the other approaches. It however leads to a lower test loss by the end of training (Figure 4). We tried smaller selectivity ratios to find out how low it could be. We observed that using a rate of 40% increases the loss. A ratio of 30% leads to an even higher loss and harms the learning process. It is thus better to keep the selectivity above 50% for this configuration.

GradMax provides the closest accuracy to *baseline big* on both datasets with the WRN architecture. In terms of energy consumption, it provides limited gains because of the growing schedule. Indeed, the growing stops in the middle of training

TABLE II
FINAL TEST ACCURACY AND ENERGY CONSUMPTION OF THE DIFFERENT APPROACHES FOR (WRN-28-1, CIFAR-10)

| Approach | Final Test accuracy | Test accuracy loss | Training time (minutes) | Training energy consumption (Joule) | Energy gains |
|---|---|---|---|---|---|
| *Baseline small* | 90.02% | 3.09% | 202 | 234844.92 | 49.87% |
| Data selection 50% | 90.41% | 2.67% | 178 | 245385.68 | 47.62% |
| GradMax | 91.25% | 1.77% | 305.33 | 398728.29 | 14.89% |
| Filter pruning | 90.36% | 2.72% | 280 | 323517.64 | 30.94% |
| *Baseline big* | 92.89% | / | 345 | 468462.06 | / |

TABLE III
FINAL TEST ACCURACY AND ENERGY CONSUMPTION OF THE DIFFERENT APPROACHES FOR (WRN-28-1, CIFAR-100)

| Approach | Final Test accuracy | Test accuracy loss | Training time (minutes) | Training energy consumption (Joule) | Energy gains |
|---|---|---|---|---|---|
| *Baseline small* | 63.66% | 8.22% | 197.67 | 234298.18 | 50.03% |
| Data selection 50% | 65.63% | 5.38% | 178.33 | 242617.88 | 48.25% |
| GradMax | 66.53% | 4.08% | 310.67 | 404455.93 | 13.74% |
| Filter pruning | 64.35% | 7.22% | 264.67 | 319291.57 | 31.90% |
| *Baseline big* | 69.36% | / | 338.33 | 468855.34 | / |

and thus the second half of the process is done with the size of *baseline big*.

Pruning allows higher energy gains than GradMax as half the training is done with the size of *baseline small*. Restricting the prunable layers due to the skip connections of WRN however limits the gains. Pruning achieves a limited accuracy improvement when compared with *baseline small*, which is consistent with the results of [32]. The accuracy achieved by pruning is higher, but we might have hoped that it would have taken better advantage of the weights inherited from the bigger architecture. We also observe that the training curve of pruning (Figure 3) is very similar to *baseline big* at the beginning of training but as filters are removed, it gradually follows the curve of *baseline small*.

### B. Results on VGG-11

We provide for each technique, the final test accuracy, the training time and training energy consumption in Table IV. As for the previous network, data selection allows to reduce the costs by half, as we use a selectivity rate of 50%. In terms of accuracy, the data selection technique performs worse on VGG-11 and provides the lowest accuracy among studied approaches.

GradMax provides a close accuracy to *baseline big*. Since we shrink all the layers of the VGG network, the impact of growing on energy is higher. The gains remain limited because the growing schedule provided in the original paper [6] uses an architecture with the *baseline big* size for half the training.

Pruning allows the highest energy gains that are about 68% since we reduce the dimensions of all layers. It achieves a close test accuracy to *baseline big* and thus performs better on VGG-11. This suggests that the size of the model has an important impact on the effectiveness of the approach. Since VGG-11 contains more parameters, the pruning approach works better.

## VI. CONCLUSION

In this work, we compared the impact of data selection, pruning and growing on the training energy consumption and on the prediction quality. We first studied the behavior of the three techniques on a synthetic dataset before performing experiments on image classification benchmarks.

We observed that data selection through random batch skipping achieves high energy savings, as it provides gains correlated with the selectivity ratio. It however causes an accuracy degradation. Growing through the GradMax technique allows to keep a high accuracy but provides limited energy gains as the network reaches the size of *baseline big*. To make this technique more efficient, it would be interesting to adapt the growing schedule. Since unstructured pruning requires specialized hardware or libraries to show energy gains, we mainly focused on the structured type for the image classification experiments. We observed that structured pruning allows significant energy gains, but that its impact on accuracy varies from an architecture to another. It is indeed more suited for larger models.

The effectiveness of each technique varies according to the architecture size and to the hyperparameters of the method itself, as we observed with the growing schedule and the data selection ratio. Moreover, looking for higher energy gains generally leads to a higher accuracy loss. The applicability of each technique depends on the requirements of the task. If keeping a high prediction quality is more necessary than having high energy saving, growing can be a good choice. Data selection is a good alternative when a decrease in accuracy is allowed to obtain high energy gains, while pruning is more suited for large architectures for which its impact on accuracy remains small. In our future work, it would be interesting to compare the three techniques on a larger set of architectures and datasets.

TABLE IV
FINAL TEST ACCURACY AND ENERGY CONSUMPTION OF THE DIFFERENT APPROACHES FOR (VGG-11, CIFAR-10)

| Approach | Final Test accuracy | Test accuracy loss | Training time (minutes) | Training energy consumption (Joule) | Energy gains |
|---|---|---|---|---|---|
| *Baseline small* | 83.16% | 4.04% | 65.67 | 81540.14 | 92.10% |
| Data selection 50% | 82.48% | 4.82% | 321 | 516966.34 | 49.91% |
| GradMax | 84.17% | 2.87% | 456.67 | 705663.49 | 31.63% |
| Filter pruning | 84.99% | 1.93% | 225 | 333287.40 | 67.71% |
| *Baseline big* | 86.66% | / | 633.67 | 1032068.47 | / |

## ACKNOWLEDGMENT

This work was partially supported by the INSA Lyon -SPIE ICS chair on Artificial intelligence for behavioral flow analysis in digital infrastructures.

## REFERENCES

[1] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[2] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," *arXiv preprint arXiv:2104.10350*, 2021.

[3] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[5] X. Dai, P. Zhang, B. Wu, H. Yin, F. Sun, Y. Wang, M. Dukhan, Y. Hu, Y. Wu, Y. Jia *et al.*, "Chamnet: Towards efficient network design through platform-aware model adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 398–11 407.

[6] U. Evci, B. van Merrienboer, T. Unterthiner, F. Pedregosa, and M. Vladymyrov, "Gradmax: Growing neural networks using gradient information," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=qjN4h_wwUO

[7] L. Wu, B. Liu, P. Stone, and Q. Liu, "Firefly neural architecture descent: a general approach for growing neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 373–22 383, 2020.

[8] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, "Pruning neural networks without any data by iteratively conserving synaptic flow," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6377–6389, 2020.

[9] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2736–2744.

[10] C. Coleman, C. Yeh, S. Mussmann, B. Mirzasoleiman, P. Bailis, P. Liang, J. Leskovec, and M. Zaharia, "Selection via proxy: Efficient data selection for deep learning," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=HJg2b0VYDr

[11] K. Killamsetty, D. Sivasubramanian, B. Mirzasoleiman, G. Ramakrishnan, A. De, and R. Iyer, "Grad-match: A gradient matching based data subset selection for efficient learning," *arXiv preprint arXiv:2103.00123*, 2021.

[12] B. R. Bartoldson, B. Kailkhura, and D. Blalock, "Compute-efficient deep learning: Algorithmic trends and opportunities," *Journal of Machine Learning Research*, vol. 24, pp. 1–77, 2023.

[13] L. Heim, A. Biri, Z. Qu, and L. Thiele, "Measuring what really matters: Optimizing neural networks for tinyml," *arXiv preprint arXiv:2104.10645*, 2021.

[14] A. H. Jiang, D. L.-K. Wong, G. Zhou, D. G. Andersen, J. Dean, G. R. Ganger, G. Joshi, M. Kaminksy, M. Kozuch, Z. C. Lipton *et al.*, "Accelerating deep learning by focusing on the biggest losers," *arXiv preprint arXiv:1910.00762*, 2019.

[15] T. B. Johnson and C. Guestrin, "Training deep models faster with robust, approximate importance sampling," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[16] Y. Wang, Z. Jiang, X. Chen, P. Xu, Y. Zhao, Z. Wang, and Y. Lin, "E2-train: Training state-of-the-art cnns with over 80% energy savings," 2019.

[17] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016.

[18] N. Lee, T. Ajanthan, and P. Torr, "SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=B1VZqjAcYX

[19] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5687–5695.

[20] A. Berthelier, Y. Yan, T. Chateau, C. Blanc, S. Duffner, and C. Garcia, "Learning sparse filters in deep convolutional neural networks with a *l*_1/*l*_2 pseudo-norm," in *International Conference on Pattern Recognition*. Springer, 2021, pp. 662–676.

[21] S. Elkerdawy, M. Elhoushi, A. Singh, H. Zhang, and N. Ray, "To filter prune, or to layer prune, that is the question," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[22] L. Wu, D. Wang, and Q. Liu, "Splitting steepest descent for growing neural architectures," *Advances in neural information processing systems*, vol. 32, 2019.

[23] X. Yuan, P. H. P. Savarese, and M. Maire, "Growing efficient deep networks by structured continuous sparsification," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=wb3wxCObbRT

[24] M. Verbockhaven and G. Charpiat, "Spotting expressivity bottlenecks and fixing them optimally," 2023. [Online]. Available: https://openreview.net/forum?id=xBeGd7sAND

[25] S. Liu, T. Chen, X. Chen, L. Shen, D. C. Mocanu, Z. Wang, and M. Pechenizkiy, "The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training," *arXiv preprint arXiv:2202.02643*, 2022.

[26] D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta, "Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science," *Nature communications*, vol. 9, no. 1, p. 2383, 2018.

[27] S. Lawrence, C. L. Giles, and A. C. Tsoi, "Lessons in neural network training: Overfitting may be harder than expected," in *AAAI/IAAI*, 1997, pp. 540–545.

[28] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[30] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures." *Analytical chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.

[31] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset (2014)," *[Online] Available https://www. cs. toronto. edu/˜ kriz/cifar. html*, 2017.

[32] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=rJlnB3C5Ym