On data selection for the energy efficiency of neural networks: Towards a new solution based on a dynamic selectivity ratio

Anais Boumendil, Walid Bechkit, Karima Benatchba

On data selection for the energy efficiency of neural networks: Towards a new solution based on a dynamic selectivity ratio

Anais Boumendil
Univ Lyon, INSA Lyon
Inria, CITI, EA3720,
69621 Villeurbanne, France
École nationale
Supérieure d'Informatique
Algiers, Algeria
anais.boumendil@insa-lyon.fr

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

Karima Benatchba
École nationale
Supérieure d'Informatique
Laboratoire de Méthodes
et de Conception des Systèmes
Algiers, Algeria
k_benatchba@esi.dz

***Abstract*—In this paper, we address the energy efficiency of neural networks training through data selection techniques. We first study the impact of a random data selection approach that renews the selected examples periodically during training. We find that random selection should be considered as a serious option as it allows high energy gains with small accuracy losses. Unexpectedly, it even outperforms a more elaborate approach in some cases.**

Our study of the random approach conducted us to observe that low selectivity ratios allow important energy savings, but also cause a significant accuracy decrease. To mitigate the effect of such ratios on the prediction quality, we propose to use a dynamic selectivity ratio with a decreasing schedule, that can be integrated to any selection approach. Our first results show that using such a schedule provides around 60% energy gains on the CIFAR-10 dataset with less than 1% accuracy decrease. It also improves the convergence when compared to a fixed ratio.

***Index Terms*—Neural networks, random data selection, dynamic selectivity ratio, energy efficiency.**

I. Introduction

Deep neural networks (DNNs) bring state-of-the-art performances in many domains as computer vision [1]–[4] and natural language processing [5]–[8]. Two main reasons are behind this high level of DNNs: the increasing number of parameters and the availability of large datasets. These two factors that contribute to the improvement of DNNs performances are also the two main reasons behind their high energy consumption. The latter has a worrying environmental impact [9], [10] and is a major obstacle to neural network integration to resource-constrained devices. Such an integration would be beneficial as it would avoid the use of Cloud [11], [12] and as it would also allow on-device training [13].

Many studies are conducted to enhance the energy efficiency of neural networks, mostly by optimizing the model size. For instance, pruning approaches aim to reduce the model size by removing non-important parameters [14]–[18] while quantization methods seek to reduce the precision by representing the weights, activations or gradients on a smaller number of bits [19]–[22]. Other approaches build efficient models from scratch [23]–[25] and even automate the process through Neural Architecture Search (NAS) [26]–[28].

Instead of optimizing the model size, it is possible to reduce the number of training examples through data selection. The latter has been quite studied in the literature [29]–[34] but most works use this approach to accelerate training. Few studies [13], [35] however show that data selection can allow high energy savings during the training phase.

Most data selection techniques are based on algorithms aiming to find informative examples. If such techniques can be beneficial to keep a high accuracy, they impose additional computations which can limit energy gains. On the other hand, random selection brings the smallest overhead but is usually seen as a naive baseline that any selection strategy should outperform. The work in [13] shows the opposite, as it proves that skipping batches randomly can achieve a high accuracy and that it provides a random sampling noise that can be helpful to escape saddle points.

The result of [13] and parallel studies on randomness positive effects on learning [36], [37] motivated us to further study the random data selection. Instead of skipping batches, we use a finer granularity by selecting individual examples. We compare the random approach to an informative selection based on loss history [38], both in terms of accuracy and training energy consumption, that we measure on the Nvidia Jetson Nano platform [39].

For both the random and loss-based techniques, we explore in detail the effect of the selectivity ratio or the percentage of examples to keep. We find that smaller ratios allow considerable energy gains, but also lead to lower accuracies. To mitigate the negative effect of small ratios on accuracy and to take advantage of their ability to reduce training costs, we propose to use a dynamic selectivity ratio to progressively decrease the number of examples used for training. This new schedule is an alternative to the usual static ratio. It can be

combined with any data selection technique with only few changes. With such a dynamic ratio, we use more examples in early epochs to allow quick accuracy gains before progressively reducing the subset size to maximize energy savings. We implement an exponential schedule and we present the results of our preliminary experiments. The latter show that using a dynamic ratio improves the convergence when compared with a fixed rate and also provides a high accuracy and significant energy savings.

Our paper is organized as follows: we first review some of the previous work on data selection in Section II. Then we discuss the first part of our work, namely the study of the random data selection approach. We describe the implemented random technique and the loss-based method [38] that we use for comparison in Section III. We also introduce the results of both approaches in terms of accuracy and training energy consumption. Afterward, we move to the second part of our work on the dynamic selectivity ratio. In Section IV, we describe the decreasing schedule approach that we propose, as well as the results of our preliminary tests. We summarize our findings, and we discuss future work in Section V.

II. Related work

Many work in the literature [29]–[35], [40], [41] studied the possibility of training neural networks using only a part of the dataset. Indeed, all examples don't have the same importance and impact on learning. A data selection approach allows to choose on which examples the training is performed. It can be either adaptive or non-adaptive. We describe both types in this section, but we focus on the adaptive selection in the rest of the paper.

Non-adaptive approaches use the same subset of examples during all the training. Since the selection is performed only once before training, non-adaptive techniques only add a small overhead but need to perform a very careful selection since no example renewal is allowed and excluding important data can harm accuracy. Among non-adaptive selection strategies, the approach of [32] provides two scores to evaluate the importance of examples: the expected loss gradient norm that can be used at initialization and the norm of the error vector, an approximation of the first score, that can be used after few epochs. The authors of [30] don't provide a new selection criteria, but rather a new framework to select data. They use a small model as a proxy to select a training subset (by applying existing data selection methods) for a bigger model in order to lower selection costs.

Adaptive data selection renews the subset of selected examples periodically during training. The selection process can involve importance sampling [29], [31] or gradient computing, as in the work of [35] that provides a technique to select the subset that matches best the gradients of the full training or validation set. Adaptive data selection allows to adapt the selected examples according to the model's prediction quality but imposes higher selection costs as it is repeated multiple times during training.

Data selection has been mainly used to accelerate training [29], [30], [38]. Only few approaches measured its impact on energy [13], [35]. Specifically, the work in [13] shows that a random data selection, through a 50% random batch skipping, allows to reduce training energy consumption by half while keeping a high accuracy. This result is very interesting as the random selection imposes the smallest selection overhead, which is beneficial for energy. In this paper, we further study the possibility of applying a random approach and its impact on the energy/accuracy trade-off. We use a finer granularity than [13] as we select individual examples instead of batches and we study various selectivity ratios (percentage of examples to keep) while the previous work only uses a value of 50%.

The work in [13] is not the only study that highlights the usefulness of randomness for neural networks training. For instance, the randomness of Stochastic Gradient Descent (SGD) helps the algorithm to escape saddle points [36]. The work in [37] shows that pre-training a neural network with random labels often accelerates its training (in terms of convergence) on another task with random or real labels and rarely causes the opposite effect. Finally, a recent study [42] shows that a random weight pruning at initialization, also seen as a naive baseline, achieves close performances to the dense equivalent model when appropriate layer-wise sparsity ratios (percentage of weights to keep in every layer) are used. In the first part of our paper, we study another aspect of randomness, namely the effects of a random choice of a subset of training examples.

All the works that we studied on data selection use a fixed value for the selectivity ratio. The authors of [35] propose a warm-up mechanism which consist in performing the training on the full dataset for a given number of epochs and then apply data selection for the rest of the training. They observed that such a warm-up improves convergence. The selectivity ratio is thus applied later in training but is not modified dynamically.

A dynamic size of the training set size has been explored in parallel work on curriculum learning, where examples are ordered according to their difficulty [43]. Increasing or decreasing, or even cyclical training set sizes [44] can be used in curriculum learning. The authors of [45] highlight the benefits of an increasing training set size, even with random curriculum (examples ordered randomly). We further investigate the dynamic training set size in the second part of our paper, where we combine it with data selection and we formulate it within a decreasing schedule.

III. Random vs loss-based data selection

Through this section, we study the impact of random data selection both in terms of accuracy and training energy consumption. In what follows, we first describe the random selection method as well as the loss-based informative approach [38] that we use for comparison. Then, we conduct a series of experiments to study the behavior of the random selection strategy.

TABLE I
FINAL TEST ACCURACY AND TRAINING ENERGY CONSUMPTION ON CIFAR-10

Approach	Selectivity ratio	Final Test accuracy	Test accuracy decrease	Training time	Training energy consumption (Joule)	Energy gains
Random selection	20%	87.22%	4.52%	1 h 56	48954.84	77.56%
	40%	89.7%	1.81%	3 h 37	92018.34	57.83%
	60%	90.39%	1.05%	5 h 26	136184.40	37.58%
	80%	90.61%	0.81%	7 h 17	180286.38	17.37%
Loss-based selection	20%	89.27%	2.28%	2 h 55	68613.12	68.55%
	40%	90%	1.48%	4 h 34	111174.9	49.05%
	60%	90.94%	0.45%	6 h 14	150146.76	31.18%
	80%	90.83%	0.57%	7 h 44	185740.14	14.87%
All dataset	100%	91.35%	/	8 h 46	218184.78	/

A. Solutions description

To train a DNN, the dataset is divided into batches. In every training epoch, a forward phase and a backward phase are performed on each batch before updating the model's weights. In a usual training, all the examples of the batch are considered. We study a random selection strategy where only a subset of the batch, selected randomly, is considered for the forward and backward phases. This strategy involves two hyperparameters: the selectivity ratio S that defines the percentage of examples to keep and the selection frequency R that defines after how many epochs the selected examples are renewed. Indeed, we mainly study the random selection in an adaptive scheme.

Throughout the paper, we compare the random approach to an informative technique, based on loss history to select examples [38]. The latter aims to accelerate training by only applying the backward phases on examples with a high loss. To do so, additional forward phases (selection forward phases) are performed to compute the loss of every example. This technique keeps a history of the last losses and computes for every example the percentile that its loss represents with respect to the loss history. The selection probability of an example is its corresponding percentile raised to a power β, representing the selectivity. We also renew the subset every R epochs.

We specifically compare the random approach to the loss-based one [38] as both techniques perform the selection with simple criteria. Moreover, the loss-based technique is not very costly as it only performs additional forward phases and doesn't require further computations as gradients.

B. Experimental settings

Architectures and datasets: We conduct our experiments on the ResNet-20 architecture [4]. We consider the CIFAR-10 and CIFAR-100 datasets [46]. Both contain 50000 training examples and 10000 examples in the test set. They contain images with a size of 32x32. CIFAR-10 contains 10 classes, while CIFAR-100 gathers 100 classes.

In terms of hyperparameters, we follow the configuration in [4]. We set the number of epochs to 160, the batch size to 128 and we use the SGD optimizer with a 0.9 momentum and a weight decay of 1e-4. For the learning rate, we set it at 0.1 at the beginning of the training and we divide it by 10 at the epochs 80 and 120 as recommended by [4]. Moreover, we follow [13] for the preprocessing of the two datasets. We apply two data augmentation techniques: random crop and random horizontal flip before normalizing. Finally, as in the implementation of [47], we use the Kaiming Normal initialization [48].

We use the PyTorch Framework [49] to implement all the studied approaches. Our code will be shared after publication.

Evaluation metrics and training environment: To evaluate the performances of the selection approaches, we first consider their test accuracy. We also measure the training time and the training energy consumption. For the hardware platform, we use the Nvidia Jetson Nano with 4GB of RAM [39], to simulate a resource-constrained environment. We use the Tegrastats command, included in the device system, to measure energy. The overall energy consumption is hence deduced based on the training time and the average power consumption given by the Tegrastats command. For the experiments targeting hyperparameter tuning, measuring energy is not necessary. Therefore, we use a server, instead of the Jetson Nano to accelerate training. The server contains an Intel Xeon Gold CPU, 64 GB of RAM and uses Ubuntu 20.04 as an operating system.

C. Experimental results

We present in the following the results of our experiments on CIFAR-10 and CIFAR-100. Through our tests, we vary the value of the selectivity ratio S to find out how low it can be. For the update frequency R, the values 1, 2 and 3 were studied in the original work of the loss-based approach [38]. The results show that setting R to 3 is a good option as it allows to keep a high accuracy and it brings a significant training speed up. Therefore, we set R to 3 in our first experiments. We study the usage of other values further in the section.

1) Results on CIFAR-10: On the CIFAR-10 dataset, we apply 4 selectivity ratios: 20%, 40%, 60% and 80% for both the random and loss-based [38] approaches. We use an update frequency $R = 3$ as in [38]. We report the final test accuracy, training time and training energy consumption in Table I where we compare both approaches to a standard training on the entire dataset.

TABLE II
FINAL TEST ACCURACY AND TRAINING ENERGY CONSUMPTION ON CIFAR-100

Approach	Selectivity ratio	Final Test accuracy	Test accuracy decrease	Training Time	Training energy consumption (Joule)	Energy gains
Random selection	20%	59.92%	11.15%	1 h 54	47855.28	78.04%
	50%	65.02%	3.59%	4 h 31	112013.64	48.60%
Loss-based selection	20%	59.01%	12.5%	2 h 54	68312.46	68.65%
	50%	64.86%	3.83%	5 h 37	134360.34	38.34%
All dataset	100%	67.44%	/	8 h 42	217918.20	/

For both approaches, we observe that as the selectivity ratio increases, the energy gains become smaller while the final test accuracy generally improves. We also notice that the ability of data selection to reduce the training time leads to a smaller energy consumption.

Moreover, we observe that the loss-based selection outperforms the random strategy in terms of final accuracy. However, for ratios higher than 20%, the accuracies of both techniques remain close. The energy gains brought by the random approach are higher than those of the loss-based one, as the random strategy only imposes a very small selection overhead while the loss-based one performs additional forward phases to select examples. Particularly, for 40% ratios, the difference in energy gains between the two approaches is around 10% while the final test accuracies of the two approaches are very close, which suggests that the random strategy can bring an interesting trade-off. When the selectivity ratio is above 60%, the energy gains are smaller, as most examples are considered for training.

2) Results on CIFAR-100: On the CIFAR-100 dataset, we apply 2 selectivity ratios: 20% and 50% for both the random and loss-based approaches. We set the update frequency to $R = 3$, as for the previous dataset. We report the final test accuracy, training time and training energy consumption in Table II.

On CIFAR-100, we observe that the random strategy outperforms the loss-based technique with the considered hyper-parameters. Indeed, the loss-based strategy performs better on datasets with higher redundancies, as CIFAR-10 [38]. Since CIFAR-100 contains fewer examples per class, it is a harder task for this approach [38]. The random selection also causes a higher accuracy loss on CIFAR-100, but it remains less sensitive than the loss-based strategy.

We notice that the accuracy decrease when a 20% ratio is applied is higher than the one observed with CIFAR-10. Therefore, it is better to keep the selectivity ratio higher on CIFAR-100 to limit the impact of data selection on accuracy, as it is a more challenging dataset. With a 50% ratio, the accuracy is close to an entire dataset training and the energy gains brought by the random approach are around 50%.

3) Impact of update frequency: In the previous experiments, we set the update frequency R to 3, a value that allows significant energy gains and moderate accuracy decrease. We perform additional experiments on CIFAR-10 and CIFAR-100 with R set to 5 or 10 to study the impact of less frequent updates. We also perform experiments with non-adaptive versions of the loss-based and random approaches, in which we select a subset at the first epoch that we keep unchanged for the rest of the training (R $>$ number of epochs). For all these tests, we use a selectivity ratio of 20%. We report the final test accuracy in Table III.

We find that increasing the values of R leads to higher accuracy losses. Moreover, we notice an important accuracy degradation when the two techniques are applied in non-adaptive setting. For the loss-based approach, the original work [38] mentions that the loss varies through training. If an example gives a low loss at a given time in training and is then ignored for many epochs, its loss will increase [38], [50]. Therefore, it is necessary to recompute the loss frequently in order to obtain accurate selection probabilities with the loss-based approach.

This result also suggests that the effectiveness of the random approach can be related to the frequent subset update. Indeed, the results of Table III show that a random non-adaptive technique is not an option to consider, as it leads to an important accuracy deterioration. Therefore, the frequent updates of selected examples is a key element behind the effectiveness of the random approach.

4) Impact of low selectivity ratios: We observed in the previous experiments that it is possible to keep the accuracy loss moderate despite using small selectivity ratios. On CIFAR-100, the accuracy decrease is more important for a 20% ratio, but the model is still able to learn with these settings. In the following, we perform additional experiments to find out how low the selectivity ratio can be.

We apply selectivity ratios of 5% and 10% with an update frequency of $R = 3$ on both dataset. We summarize the results in Table IV.

We observe that the 5% and 10% selectivity ratios lead to an important accuracy loss. Interestingly, the loss-based technique is much more sensitive than the random selection. The random approach achieves higher accuracies, which shows that the model still learns under these settings. On the other hand, the loss-based technique provides very small accuracies that are around 10% which shows that using this approach with such small ratios harms the learning process.

TABLE III
IMPACT OF UPDATE FREQUENCY

Dataset	Approach	Update Frequency	Final Test accuracy
CIFAR-10	Random	R=3	87.77%
	Random	R=5	85.77%
	Random	R=10	82.12%
	Random	R > number of epochs	68.65%
CIFAR-10	loss-based	R=3	89.09%
	loss-based	R=5	87.68%
	loss-based	R=10	85.6%
	loss-based	R > number of epochs	66.85%
CIFAR-100	Random	R=3	60.13%
	Random	R=5	57.62%
	Random	R=10	48.91%
	Random	R > number of epochs	27.39%
CIFAR-100	loss-based	R=3	57.55%
	loss-based	R=5	55.73%
	loss-based	R=10	50.09%
	loss-based	R > number of epochs	24.61%

TABLE IV
IMPACT OF LOW SELECTIVITY RATIOS

Dataset	Approach	Selectivity ratio	Final Test accuracy
CIFAR-10	Random	5%	78.08%
	Loss-based	5%	10%
CIFAR-100	Random	10%	52.73%
	Loss-based	10%	13.69%

IV. TOWARDS A NEW DATA SELECTION APPROACH BASED ON A DYNAMIC SELECTIVITY RATIO

In the previous section, we observed that using small selectivity ratios leads to high accuracy drops. Therefore, such ratios can't be considered despite allowing important energy gains during training.

To mitigate the accuracy loss caused by small ratios, we propose a simple yet efficient technique that can be integrated with any adaptive data selection method with only few additional lines of code. This technique consists in using a decreasing selectivity ratio in order to perform training on large subsets in early epochs and use very small selectivity ratios later in training. In what follows, we present one possible implementation of this decreasing schedule, as well as our first experiments.

A. Proposed solution for the dynamic selectivity ratio

We propose to replace the fixed selectivity ratio by the usage of a formula allowing to decrease the size of the selected subset all along training. Since bigger ratios lead to higher accuracies, we propose to use these values in first epochs to improve prediction quality and give a good start to training. In the later epochs, we propose to decrease the subset size in order to maximize the energy gains.

This decreasing selectivity ratio allows to divide the training process into two phases. The first phase aims to maximize accuracy gains through the usage of high ratios but brings limited energy gains. The second phase targets higher efficiency by using small subset sizes. The accuracy loss that such sizes can cause is mitigated by the good start given by the first phase.

The dynamic selectivity ratio can be implemented through different decreasing schedule. We present an example using an exponential evolution:

$$S_t = S_0 \times e^{-\alpha \lfloor \frac{t}{R} \rfloor} \tag{1}$$

where t is the current epoch, S_0 is the initial selectivity ratio and R is the update frequency. The selectivity ratio is only decreased when the subset is updated (every R epochs). The parameter α defines how fast the selectivity ratio is decreased. The value of α can be chosen either by fixing the target average selectivity or by fixing the target final selectivity ratio S_N (value that the schedule should lead to at the end of training).

We summarize the few modifications to make to the usual training procedure to integrate an adaptive selection approach with a decreasing selectivity ratio in Algorithm 1. As in a usual training, the neural network is trained for N epochs on a dataset D. Every R epochs, the selectivity ratio S_t is updated and a new subset of examples is selected according to a given approach. We perform the forward and backward phases on the selected examples before updating the model.

B. Experimental Results

In this section, we present the results of our first tests of the decreasing selectivity ratio. We use the experiment settings and the training environment described in Section III-B. As we observed in Section III-C4 that the loss-based approach is more sensitive to small selectivity ratios, we consider it for the

TABLE V
FINAL TEST ACCURACY AND ENERGY CONSUMPTION OF THE DECREASING SELECTIVITY RATIO ON CIFAR-10

Approach	Test accuracy	Test Accuracy decrease	Training energy consumption (joule)	Energy gains
Loss-based with decreasing selectivity ratio $S_0 = 80\%$ and $\alpha = 0.05$ $R = 3$	90.56%	0.86%	88248.06	59.55%
Loss-based with fixed equivalent average selectivity ratio $S = 28.34\%$ and $R = 3$	89.63%	1.88%	86900.16	60.17%
All dataset	91.35%	/	218184.78	/

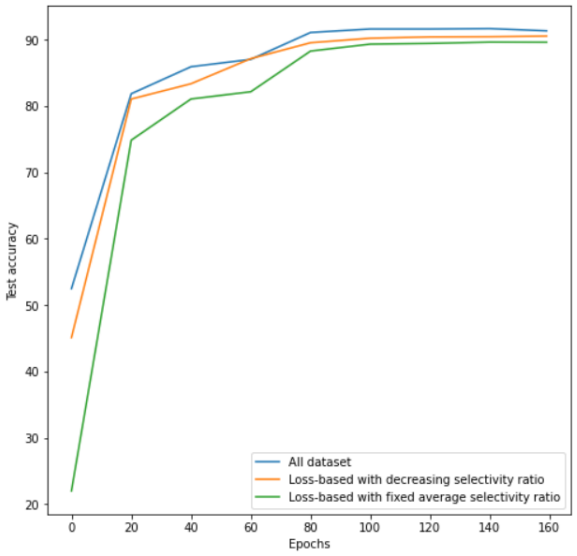

Fig. 1. Comparing the test accuracy evolution according to epochs for a training with a decreasing selectivity ratio to training with a fixed average ratio and to training on all the dataset.

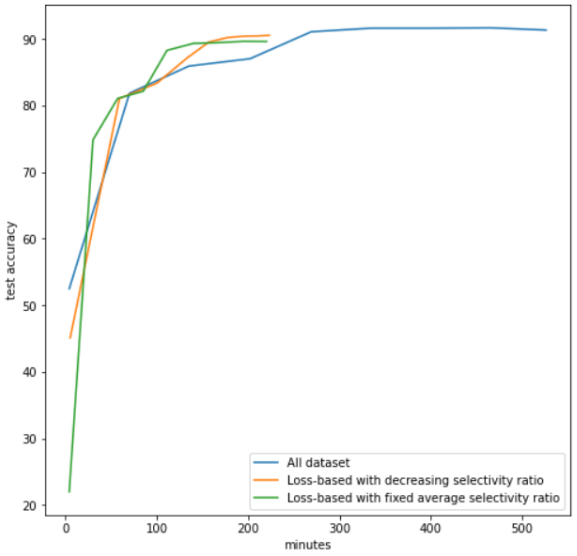

Fig. 2. Comparing the test accuracy evolution according to minutes for a training with a decreasing selectivity ratio to training with a fixed average ratio and to training on all the dataset.

Algorithm 1 Training procedure with adaptive data selection and decreasing selectivity ratio

Inputs: Dataset D, Number of epochs N, Update frequency R, Initial selectivity ratio S_0, α
for t=1 to N **do**
  if t % R == 0 **then**
    update selectivity ratio $S_t = S_0 \times e^{-\alpha \lfloor \frac{t}{R} \rfloor}$
    renew the subset: $subset$ = *selection_approach* (D, S_t)
  end if
  perform forward propagation on *subset*
  perform backward propagation
  update weights
end for

next experiments to evaluate the effectiveness of the dynamic selectivity ratio. The proposed schedule remains general and applicable to any other selection technique.

On the CIFAR-10 dataset, we use an initial selectivity ratio $S_0 = 80\%$ and we set $\alpha = 0.05$ which leads to a final selectivity ratio of $S_N = 5.65\%$. These settings provide an average selectivity of $S_{avg} = 28.34\%$ (average of all selectivities generated by the schedule). We compare a training using the decreasing ratio to training on the entire dataset and to the usage of a fixed selectivity ratio set to S_{avg}. We plot test accuracy evolution according to epochs and minutes respectively in Figure 1 and Figure 2. We summarize the energy consumption measured on the Nvidia Jetson Nano platform and final test accuracies in Table V.

We observe that the dynamic selectivity ratio schedule reaches a higher final test accuracy than the fixed ratio. The gap between the two techniques remains small as they use the same amount of data but distribute it differently during training. However, using a decreasing ratio allows a faster convergence and mitigates the accuracy loss caused by the small selectivity ratios used in later epochs. In terms of energy consumption, using the loss-based strategy with a decreasing schedule allows high energy gains that are around 60%. Since the accuracy loss is less than 1%, this technique provides an interesting trade-off. The energy gains remains similar with the loss-based technique with a fixed average ratio, as both techniques use the same amount of data. Furthermore, we observe from Figure 2 that the loss-based selection with a dynamic ratio leads to a higher accuracy in a smaller

training time when compared with a standard training on all the dataset. This approach is therefore interesting when the training resource budget is limited, which is the case in power-constrained devices.

V. Conclusion

In this paper, we first studied the impact of random data selection on both accuracy and training energy consumption. We observed that a random adaptive approach allows significant energy savings while keeping a high accuracy when appropriate selectivity ratios are applied. We showed that the effectiveness of random selection is due to the frequent update of the selected subset. Our work shows that the most naive baseline can be competitive when it is employed in the correct settings. In our future work, we aim to explore the combination of this random technique with informative selection approaches to further limit the accuracy decrease.

In the second part of our paper, we proposed a new dynamic selectivity ratio with a decreasing schedule. Our first results show that this simple technique provides an interesting accuracy/energy trade-off and improves the convergence when compared with a fixed ratio. We aim to further study this promising approach by exploring other schedules and various values for the initial and target selectivity ratios. We also plan to perform a larger number of experiments on different datasets to better understand the behavior of this approach.

Acknowledgment

This work was partially supported by the French Embassy in Algeria through the project FSPI N°2021-016 LISEN 2020-02. It was also partially supported by the French National Research Agency (ANR) under the project ANR-21-CE25-0003 (DRON-MAP) and by the INSA Lyon -SPIE ICS chair on Artificial intelligence for behavioral flow analysis in digital infrastructures.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

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

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[8] OpenAI, "Gpt-4 technical report," 2023.

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

[10] 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.

[11] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[12] J. Tang, D. Sun, S. Liu, and J.-L. Gaudiot, "Enabling deep learning on iot devices," *Computer*, vol. 50, no. 10, pp. 92–96, 2017.

[13] Y. Wang, Z. Jiang, X. Chen, P. Xu, Y. Zhao, Y. Lin, and Z. Wang, "E2-train: Training state-of-the-art cnns with over 80% energy savings," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[14] 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.

[15] H. You, C. Li, P. Xu, Y. Fu, Y. Wang, X. Chen, R. G. Baraniuk, Z. Wang, and Y. Lin, "Drawing early-bird tickets: Toward more efficient training of deep networks," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=BJxsrgStvr

[16] C. Wang, G. Zhang, and R. Grosse, "Picking winning tickets before training by preserving gradient flow," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=SkgsACVKPH

[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] 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.

[19] Y. Fu, H. Guo, M. Li, X. Yang, Y. Ding, V. Chandra, and Y. Lin, "Cpt: Efficient deep neural network training via cyclic precision," *arXiv preprint arXiv:2101.09868*, 2021.

[20] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv preprint arXiv:1606.06160*, 2016.

[21] F. Li, B. Zhang, and B. Liu, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.

[22] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in neural information processing systems*, 2015, pp. 3123–3131.

[23] 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.

[24] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[25] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[26] C. Gong, Z. Jiang, D. Wang, Y. Lin, Q. Liu, and D. Z. Pan, "Mixed precision neural architecture search for energy efficient deep learning," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–7.

[27] 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.

[28] H. You, B. Li, S. Huihong, Y. Fu, and Y. Lin, "Shiftaddnas: Hardware-inspired search for more accurate and efficient neural networks," in *International Conference on Machine Learning*. PMLR, 2022, pp. 25 566–25 580.

[29] T. B. Johnson and C. Guestrin, "Training deep models faster with robust, approximate importance sampling," *Advances in Neural Information Processing Systems*, vol. 31, pp. 7265–7275, 2018.

[30] 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
[31] A. Katharopoulos and F. Fleuret, "Not all samples are created equal: Deep learning with importance sampling," in *International conference on machine learning*. PMLR, 2018, pp. 2525–2534.
[32] M. Paul, S. Ganguli, and G. K. Dziugaite, "Deep learning on a data diet: Finding important examples early in training," *Advances in Neural Information Processing Systems*, vol. 34, 2021.
[33] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," in *ICLR*, 2019.
[34] B. Mirzasoleiman, J. Bilmes, and J. Leskovec, "Coresets for data-efficient training of machine learning models," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6950–6960.
[35] K. Killamsetty, S. Durga, G. Ramakrishnan, A. De, and R. Iyer, "Grad-match: Gradient matching based data subset selection for efficient deep model training," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5464–5474.
[36] R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle points—online stochastic gradient for tensor decomposition," in *Conference on learning theory*. PMLR, 2015, pp. 797–842.
[37] H. Maennel, I. M. Alabdulmohsin, I. O. Tolstikhin, R. Baldock, O. Bousquet, S. Gelly, and D. Keysers, "What do neural networks learn when trained with random labels?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 693–19 704, 2020.
[38] 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.
[39] "Jetson Nano Developer Kit," https://developer.nvidia.com/embedded/jetson-nano-developer-kit, accessed: 2023-06-29.
[40] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1aIuk-RW
[41] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.
[42] 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.
[43] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.
[44] H. T. Kesgin and M. F. Amasyali, "Cyclical curriculum learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.
[45] X. Wu, E. Dyer, and B. Neyshabur, "When do curricula work?" in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=tW4QEInpni
[46] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset (2014)," *URL https://www. cs. toronto. edu/˜ kriz/cifar. html*, 2017.
[47] Y. Idelbayev, "Proper resnet implementation for cifar10/cifar100 in pytorch," *URL https://github.com/akamaster/pytorch_resnet_cifar10 Accessed: 2022-05-13*, 2018, accessed: 2022-05-13.
[48] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.
[49] "Pytorch framework," https://pytorch.org/, accessed: 2023-07-06.
[50] G. E. Hinton, "To recognize shapes, first learn to generate images," *Progress in brain research*, vol. 165, pp. 535–547, 2007.