Approximate Matrix Multiplication for Energy-Efficient Training of LLMs

Nils Kasper
Hasso Plattner Institute
nils.kasper@student.hpi.uni-potsdam.de

Anna Kazachkova
Hasso Plattner Institute
anna.kazachkova@hpi.de

Rainer Schlosser
Hasso Plattner Institute
rainer.schlosser@hpi.de

Ralf Herbrich
Hasso Plattner Institute
ralf.herbrich@hpi.de

Abstract

Motivated by previous work on approximated matrix multiplication on smaller deep neural networks, we consider their application on large language models (LLMs). Our work investigates whether exact tensor operations can be replaced with less energy-consuming (probabilistic) approximations that preserve accuracy while reducing computation. We introduce the *top-η rate* algorithm, which selects a fixed proportion of components per layer, allowing for rate-based approximation of matrix multiplication. Applied to LLMs, we demonstrate that our method reduces the training energy consumption by about 20% and the inference energy by about 20%, with minimal accuracy degradation and consistently outperforms the state-of-the-art top-k algorithm in both efficiency and stability.

1 Introduction

Matrix multiplication is a basic operation that appears in many algorithms in computer science: it forms the core of linear-algebra kernels in scientific computing [Golub and Van Loan, 2013], and builds the backbone of LLMs [Vaswani et al., 2017]. Therefore, the challenge of performing accurate and energy-efficient matrix operations attracts a great deal of attention in the field of sustainable artificial intelligence [Imanov et al., 2024]. Due to inherent redundancy and variance present in the weight matrices of deep neural networks (DNNs), exact matrix multiplications may not always be necessary, and approximating these operations could significantly enhance energy efficiency.

Another subfield of energy-saving approaches is sampling-based approximations. They are used to accelerate inference [Sun et al., 2017, Drineas et al., 2006], and for training [Adelman et al., 2021b] smaller DNNs like multilayer perceptrons (MLPs) or convolutional neural networks (CNNs). However, using them for larger models, e.g. LLMs, has not been systematically studied. Adelman et al. [Adelman et al., 2021b] investigates several sampling approaches for DNNs, among them the Column-Row Sampling (CRS) algorithm and top-k algorithm, which deterministically select column–row pairs with the largest norms. We adapt the state-of-the-art algorithm top-k to improve the accuracy-energy consumption trade-off and evaluate its performance against the full-finetuning baseline for both inference and training.

Enhancing this, we propose our novel approach to make LLMs training and inference even more energy efficient, and outperform the previous two algorithms on an energy accuracy trade-off. Instead of sampling a fixed number of column–row pairs, our method adapts the number of selected pairs to the size of each matrix multiplication, whose ranks vary widely in magnitude. This makes it

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Rethinking AI: Efficiency, Frugality, and Sustainability.

easy to directly translate the chosen hyperparameter to the relative amount of saved energy. The approximation operates independently for each tensor operation, preserving the overall network architecture and tensor dimensions.

Our contributions are as follows: (1) We adapt existing sampling-based techniques for approximate matrix multiplication to be able to apply them to LLMs and test their performance. (2) We extend the current state-of-the-art top-k algorithm from Adelman et al. [2021b] with a novel sampling algorithm, which uses a rate-based selection approach, and we state the algorithm's theoretical properties. (3) We empirically show that this adaptation makes it possible to outperform existing approximation methods and saves 20% of computation in both inference and training of LLMs. Additionally, we provide a reference implementation (`https://github.com/NilsKasper/approximating-matrix-multiplication-eurips2025`).

2 Related Work

An overview of related work can be found in Appendix B. Among the methods investigated, the most related are the CRS algorithm and the state-of-the-art top-k algorithm.

CRS - Column-Row Sampling Let $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{n\times p}$, where $A^\top$ denotes the transposition of A (the matrix is stored in the transposed format). The matrix product $A^\top B$ can be approximated by a weighted sum of outer products of sampled columns from $A^\top$ and their corresponding rows from B:

$$A^\top B \approx \frac{1}{k} \cdot \sum\nolimits_{t=1}^{k} \frac{1}{p_{i_t}} A^{\top(i_t)} B_{(i_t)} \,, \tag{1}$$

where $A^{\top(i)}$, $B_{(i)}$ indicate the i-th column of $A^\top$ and i-th row of B respectively. Here k denotes the number of sampled pairs, with $1 \leq k \leq n$ and $\{p_i\}_{i=1}^n$ specifies a probability distribution over the pairs of columns and rows from $A^\top$ and B, with indices $i_t \in \{1, \ldots, n\}$. This approach reduces the computational complexity from $\mathcal{O}(mnp)$ to $\mathcal{O}(mkp)$, with $k \ll n$ usually. One important property of this approximation algorithm is that it is unbiased [Drineas et al., 2006], which means that, when executed repeatedly, its expected value converges to the true matrix product:

$$\mathbb{E}\left[\frac{1}{k} \cdot \sum\nolimits_{t=1}^{k} \frac{1}{p_{i_t}} A^{\top(i_t)} B_{(i_t)}\right] = A^\top B \,. \tag{2}$$

Top-k algorithm The top-k algorithm is similar to CRS, but instead of sampling which column-row pairs to choose, the algorithm deterministically selects the top-k column-row pairs. Here, top-k refers to the k column-row pairs with the largest product of norms, that is, $\left|A_{(i)}\right| \cdot \left|B_{(i)}\right|$. Adelman et al. [2021a] claim that if A and B are randomly initialized matrices with

$$\mathbb{E}\left[A^{\top(i)} B_{(i)}\right] = 0 \,, \tag{3}$$

then the mean-squared error of $A^\top B$ for a selection of k column-row pairs is minimized when selecting the top-k column-row pairs with respect to the maximum norm multiplication $\left|A_{(i)}\right| \cdot \left|B_{(i)}\right|$. They show that, if one matrix is drawn from $\mathcal{N}(0,1)$ and one from $\mathcal{N}(1,1)$, then top-k strongly outperforms the CRS algorithm in both compute reduction and accuracy. Thus, we choose the top-k algorithm to compare against in our experiments.

3 Our Approach

Relations to LLMs In the structure of an LLM, the dimensions over which the multiplication is performed differ in their magnitude. For every layer, let us denote by $A \in \mathbb{R}^{n\times m}$ the input or hidden activation matrix and by $B \in \mathbb{R}^{n\times p}$ the weight matrix to the next layer pre-activation. For example, in the *Llama-3.2-1B-Instruct* model, the layer *lm_head* conducts multiplication over a dimension of $128,256$ and the backpropagation through layer *layers.15.self_attn.o_proj* conducts multiplication over the batch size. The main drawback of the CRS and top-k algorithms is that they require a hyperparameter to explicitly set the number of sampled column–row pairs. This creates an inherent trade-off. If the sampling rate is too low, too much information is lost in the multiplication of high-dimensional matrices, leading to poor approximations. However, if the sampling rate is too high, the reduced matrices are even larger than the originals, and little computational benefit is gained. In practice, this makes it difficult for these methods to strike a useful balance, which limits their applicability for LLMs.

Top-η rate algorithm To address this drawback, we propose setting the number of sampled column-row pairs relative to the shared dimension n of the matrices A and B. Given a rate $\eta \in [0, 1]$, we achieve this by selecting the ηn column-row pairs with the largest norm. Thus, η controls the exact ratio of selected column-row pairs rather than the absolute number. More formally, given $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{n\times p}$ and a rate $\eta \in (0, 1]$, the matrix product $A^\top B$ is approximated by a weighted sum of outer products of columns from $A^\top$ and their corresponding rows from B:

$$A^\top B \approx \sum\nolimits_{t=1}^{\lceil n\cdot\eta\rceil} \frac{1}{\lceil n\cdot\eta\rceil} A^{\top(i_t)} B_{(i_t)}\,, \tag{4}$$

where $A^{\top(i_t)} B_{(i_t)}$ is the $t-$th largest product in terms of multiplied norms $\left|A_{(i)}\right|\left|B_{(i)}\right|$. The unbiased property of top-k still holds for the top-η algorithm, namely:

$$\mathbb{E}\left[\sum\nolimits_{t=1}^{\lceil n\cdot\eta\rceil} \frac{1}{\lceil n\cdot\eta\rceil} A^{\top(i_t)} B_{(i_t)}\right] = A^\top B\,. \tag{5}$$

The runtime of our approach is $\mathcal{O}(m\eta np)$ with $\eta n \ll n$. In other words, the actual runtime is reduced by a factor of η, which makes the parameter η easy to interpret and set in a practical application (see Section 4.1). In our experimental section, we show that if we set $\eta = 80\%$, then the empirical compute reduction is exactly $1 - \eta = 20\%$.

4 Evaluation Results

Experiment Setup We implemented the top-k algorithm and the top-η rate algorithm in PyTorch [Paszke et al., 2019] by overwriting Torch's linear functionality. Our implementation allows us to control the sampling parameter k or η and the application of approximation in the forward or backward passes. To make it applicable to LLMs, we used a pretrained model (in our case, Meta's *Llama-3.2-1B-Instruct* model [Touvron et al., 2023]) and replace the layer with a specific name and type with our approximated functionality. This implementation also allows for an easy choice of a different set of layers instead of all 113 linear layers. To evaluate the performance of the model, we choose a dataset from the MMMLU (Measuring Massive Multitask Language Understanding) benchmarks [Hendrycks et al., 2020], namely the *qa-mmlu- college-chemistry* dataset. We use the ROUGE score to evaluate the accuracy of the model on this dataset, and perform a linear search over the respective hyperparameter for the approach. The experiments were performed on an A40 GPU.

LLM Experiments Training We start with a straightforward experiment, which is replacing every single linear layer with our implementation of the top-k algorithm, and show the results of a variety of k values. The setup is described in Appendix C along with the results in Figure 1.

We clearly see that even for higher k, the model could not reach an accuracy that was as good as the baseline. Further, the experiments empirically show that some operations are conducted over the batch size (in our case 8) and some for the context length (approx. $12\,000$). The run with the best accuracy used $k = 2048$. Although this run still saved about 17% of energy, the ROUGE score was 11% worse than the baseline (0.31 vs 0.35). Motivated by our assumptions and previous observations on LLMs, we conducted the multiplication over a rate of the original dimension instead of a fixed k, see Section 2. We analyzed different rates between 0% and 100% and compared the accuracy. The results of this experiment are shown in Appendix D and Figure 2.

We can clearly see that the top-η rate algorithm works better than the top-k algorithm. Figure 2 shows that up to a rate of $\eta \geq 70\%$, the model does not lose any quality at all. In our runs, probably due to the variance of the model for different values of η, we could actually outperform the baseline model for rates of $\eta \geq 90\%$. These energy reductions and the results are again summarized in Table 1.

Looking at the savings, especially from the top-η rate algorithm, this does not imply that the model can be built with 20% less complexity, but that 20% less computations can be used to fit the model

Approach	Parameter (k or η)	RougeL-Score (Baseline)	Compute Reduction
top-η rate	$\eta = 70\%$	0.35 (0.35)	20.0%
top-η rate	$\eta = 80\%$	0.35 (0.35)	13.3%
top-η rate	$\eta = 90\%$	0.36 (0.35)	6.7%
top-k rate	$k = 512$	0.17 (0.35)	28.8%
top-k rate	$k = 1024$	0.26 (0.35)	25.1%
top-k rate	$k = 2048$	0.31 (0.35)	17.8%

Table 1: Best results for LLM training with the top-k and top-η rate algorithm.

of the same complexity. Note that we fine-tune the complex *Llama-3.2-1B-Instruct* model, and we make use of the fact that the domain on which we fine-tuned seems to require 20% less computation for parameter changes from the baseline model.

4.1 LLM Experiments Inference

Saving energy in the forward pass is also of high interest. Our implementation allows replacing only the layers' forward functions to test inference with the top-k, see Section 2, or our top-η rate algorithm, see Section 3. The full setup is in Appendix E. We first started with an exponential search of values over k for the top-k algorithm on our first subexperiment E.1. To directly compare the response to the baseline model, we set a specific seed and set the model's temperature to 0. We find that, with a specific k of (i) $k = 64$ the model is spamming random characters; (ii) $k = 256$ the model is spamming random words; (iii) $k = 1024$ the model gives an answer, which is wrong; (iv) $k = 4096$ the model is giving a good and correct answer; and (v) $k = 16384$ the model generates the same response as the baseline model. This result also provides a valuable insight into the general domain dimensionality of the pretrained model. For $k \geq 2048$, the model produces reasonable outputs., and by selecting this value of k, we are able to save at least 11.6% of energy. However, with $k = 16384$ the model's response was the same as the pretrained model, and no energy savings were achieved.

We also experimented with distinctive values for η for the top-η rate algorithm on our first subexperiment E.1. Again, we set a specific seed and set the model's temperature to 0. We find that, with a specific η of (i) $\eta = 10\%$ the model is spamming random characters; (ii) $\eta = 50\%$ the model is spamming random words; (iii) $\eta = 90\%$ the model is giving a good and correct answer; and (iv) $\eta = 100\%$ the model generates the same response as the baseline model.

To find the breaking point, we conduct the Experiment E.2. Here we found that, at least for our *Llama-3.2-1B-Instruct* model and our prompts, with $k > 2500$ for the top-k algorithm, our model produces good and correct answers and saves approximately 16% of energy consumption. For $\eta > 80\%$, the top-η algorithm produces correct answers while saving roughly 20% of energy. Since only matrix multiplications are modified in the forward pass, the specified rate η directly translates into predictable energy savings For example, setting $\eta = 80\%$ achieves a compute reduction of exactly $1 - \eta = 20\%$. For more discussions, see Appendix F.

5 Conclusion

The central takeaway of this work is that substantial energy savings are possible without sacrificing accuracy. Previous sampling-based methods were not applicable for LLMs because the selection was set to a fixed amount, which was not appropriate for the respective architecture. We solved this limitation by proposing a rate-based selection, the top-η rate algorithm, which is suitable for LLM architectures. Our results show that the top-k algorithm is ineffective for training, as it fails to reach baseline accuracy even at high k. In contrast, our top-η rate algorithm consistently matches or exceeds baseline performance while reducing compute, achieving up to 20% savings at $\eta = 70\%$. For inference, the top-k approximation only begins to produce correct answers at larger k with limited savings of about 11.6%. By comparison, top-η delivers good to high-quality outputs from $\eta \geq 80\%$ onward, while offering predictable and higher efficiency gains of up to 20%. This demonstrates that top-η clearly outperforms top-k in both training and inference.

Our findings indicate that rate is a more promising parameter for stochastic approximation algorithms compared to a parameterization in terms of the number of column-row pairs.

References

Menachem Adelman, Kfir Levy, Ido Hakimi, and Mark Silberstein. Faster neural network training with approximate tensor operations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27877–27889. Curran Associates, Inc., 2021a. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/eaa1da31f7991743d18dadcf5fd1336f-Paper.pdf`.

Menachem Adelman, Kfir Levy, Ido Hakimi, and Mark Silberstein. Faster neural network training with approximate tensor operations. *Advances in Neural Information Processing Systems*, 34:27877–27889, 2021b.

Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. *Advances in neural information processing systems*, 27, 2014.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332, 2022.

Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices i: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *journal of machine learning research*, 18(187):1–30, 2018.

Elbrus Imanov, Louisa Iyetunde Aiyeyika, and Gunay E Imanova. Development and assessment of energy-efficient approaches for ai-based green computing. In *International Conference on Smart Environment and Green Technologies*, pages 179–187. Springer, 2024.

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up convolutional neural networks with low rank expansions. *arXiv preprint arXiv:1405.3866*, 2014.

Sagar V Kamarthi and Stefan Pittner. Accelerating neural network training using weight extrapolations. *Neural networks*, 12(9):1285–1299, 1999.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Xu Sun, Xuancheng Ren, Shuming Ma, and Houfeng Wang. meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting. In *International Conference on Machine Learning*, pages 3299–3308. PMLR, 2017.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

A Llama-3.2-1B Architecture

- **Backbone:** LlamaModel
  - **Token Embeddings:** Embedding(128256, 2048)
  - **Decoder:** 16 $\times$ LlamaDecoderLayer
    - **Self-Attention:** LlamaAttention
      - q_proj: Linear(2048 $\rightarrow$ 2048, bias=False)
      - k_proj: Linear(2048 $\rightarrow$ 512, bias=False)
      - v_proj: Linear(2048 $\rightarrow$ 512, bias=False)
      - o_proj: Linear(2048 $\rightarrow$ 2048, bias=False)
    - **MLP:** LlamaMLP
      - gate_proj: Linear(2048 $\rightarrow$ 8192, bias=False)
      - up_proj: Linear(2048 $\rightarrow$ 8192, bias=False)
      - down_proj: Linear(8192 $\rightarrow$ 2048, bias=False)
      - act_fn: SiLU
    - input_layernorm: LlamaRMSNorm(2048, eps=1e-5)
    - post_attention_layernorm: LlamaRMSNorm(2048, eps=1e-5)
  - norm: LlamaRMSNorm(2048, eps=1e-5)
  - rotary_emb: LlamaRotaryEmbedding
  - **LM Head:** Linear(2048 $\rightarrow$ 128256, bias=False)

B Related Work

To our knowledge, we are the first to investigate both approximation and sampling techniques involving a relative rate of the multiplication dimension and the performance of these algorithms on LLMs. However, there is prior work on sampling techniques and their performance on DNNs, which we name below.

Recent advances in accelerating inference increasingly rely on a variety of approximation strategies. Beyond traditional model compression techniques [Jaderberg et al., 2014], which approach this problem by applying Singular Value Decomposition (SVD) on the tensors one wants to approximate [Denton et al., 2014], usually in convolutional neural networks, similar approaches have already been investigated on LLMs, like freezing the pre-trained model weights and injecting trainable rank decomposition matrices into each layer, approximating the model weights by two low-rank matrices, called adapters [Hu et al., 2022]. Other work focuses on using a larger model once to generate data for training a smaller model, and thus also saves energy. This approach is called distillation models [Hinton et al., 2015].

A substantial line of research has focused on quantization and low-precision data types [Han et al., 2015, Micikevicius et al., 2017, Hubara et al., 2018]. Further energy saving approaches directly address the matrix multiplication itself using quantization [Dettmers et al., 2022]. Other methods approximate weights directly, for example, through value extrapolation, thereby reducing computational overhead during both training and inference [Kamarthi and Pittner, 1999]. Regarding LLMs, another efficient approach to save computation is grouping tokens with similar representations using hashing, so that each token only attends to a subset of other tokens [Kitaev et al., 2020].

More recent work has explored sampling-based approximations as an energy-saving strategy. They have previously been applied to speed up inference [Sun et al., 2017, Drineas et al., 2006] and to accelerate training [Adelman et al., 2021b] of smaller deep neural networks such as MLPs and CNNs. Among the investigated methods, the most pertinent were the CRS algorithm and the state-of-the-art top-k algorithm.

C LLM Experiment 1

For the first experiment, we replace every Linear Layer with our approximation of a Linear Layer and perform the top-k algorithm instead of matrix multiplication with $k = [16, 64, 128, 256, 512, 1024, 2048]$. We trained for 5 epochs with the cross-entropy loss and a batch size of 8. We use the *llama-3.2-1B-Instruct model* from Appendix A. We train on the *qa-mmlu-college-chemistry* data set. The results are depicted in Figure 1.

To get the computational reduction of the training, we measure the FLOPs it takes to forward and backward a batch of size 8 using the `torch.profiler`. With that, we get:

- normal LLM (baseline): $2,373,281,365,952$ FLOPs
- top-k algorithm, $k = 16$: $952,794,322,960$ FLOPs (59.9% compute reduction)
- top-k algorithm, $k = 64$: $1,437,450,344,464$ FLOPs (39.4% compute reduction)
- top-k algorithm, $k = 128$: $1,609,131,586,752$ FLOPs (32.2% compute reduction)
- top-k algorithm, $k = 256$: $1,636,142,904,512$ FLOPs (31.1% compute reduction)
- top-k algorithm, $k = 512$: $1,690,165,523,648$ FLOPs (28.8% compute reduction)
- top-k algorithm, $k = 1024$: $1,776,735,958,208$ FLOPs (25.1% compute reduction)
- top-k algorithm, $k = 2048$: $1,949,876,729,024$ FLOPs (17.8% compute reduction)

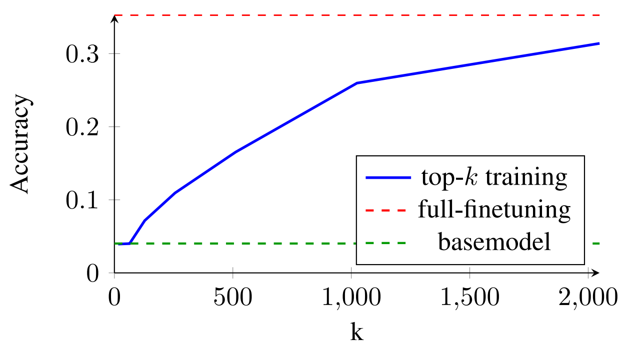

Figure 1: Accuracy vs which k we chose after 5 epochs.

D LLM Experiment 2

For the second experiment, again, every layer was replace by our approximation, but this time we use the top-η Rate algorithm and choose column-row pairs and iterated over $\eta = [1\%, \ldots, 99\%]$. We train for 5 epochs with the cross-entropy loss and a batch size of 8. We use the *llama-3.2-1B-Instruct* model from Appendix A. We train on the *qa-mmlu-college-chemistry* data set. The results are depicted in Figure 2

To get the compute reduction of the training, we measure the FLOPs it takes to forward and backward a batch of size 8 using the `torch.profiler`. With that, we get:

- normal LLM (baseline): $2,373,281,365,952$ FLOPs
- top-η rate algorithm, $\eta = 50\%$: $1,582,120,278,032$ FLOPs (33.3% compute reduction)
- top-η rate algorithm, $\eta = 70\%$: $1,898,362,148,880$ FLOPs (20.0% compute reduction)
- top-η rate algorithm, $\eta = 80\%$: $2,056,545,343,504$ FLOPs (13.3% compute reduction)
- top-η rate algorithm, $\eta = 90\%$: $2,214,729,848,848$ FLOPs (6.7% compute reduction)

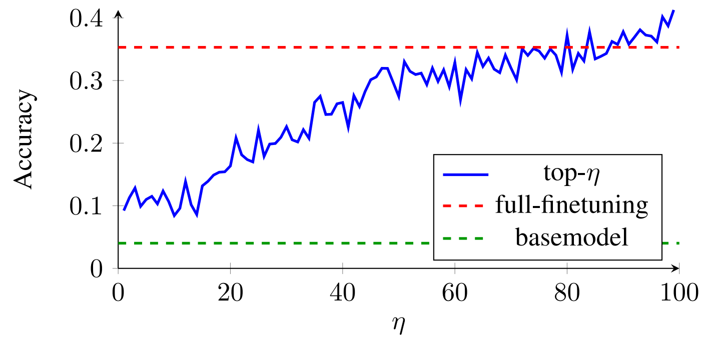

Figure 2: Accuracy full finetuning vs approximated training with top-η rate algorithm after 5 epochs.

E LLM Experiment 3

We already investigate the training performance of our model implementation. But also, a good performance for inference is quite powerful. To do so, we do not look for any benchmarks but just observe and evaluate the model's response for different prompts and different k and η values. We again use the model from Appendix A and replaced every linear layer. To also check at which point we get the same performance as the baseline model, we deactivate any temperature and randomness.

E.1 First Prompt

Our first prompt is: `"How many countries are there in this world?"`. Here is how the several models performed

Basemodel:

There are 196 countries recognized by the United Nations. However, there are many more countries that are not recognized by the UN, and some countries have disputed status.
The number of countries in the world can vary depending on the criteria used to define a "country." [...]

top-k with k of 64

glassglassglassglassglassCodesSuites. swingspecificzonesoriginatria triatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatriatripo [...]

top-k with k of 256:

the one one and and the the time in control with with out one the one the the the the the one for the and and one the the the the a thejeneee support for, and only only the in earth and the theat ech the theorthat thatad in powerjin [...]

top-k with k of 1024

The answer is: 0. The reason why there are no people living on other planets or moons in our universe. The answer is that the universe is not infinite in the sense of physical space, and the universe is not infinite. [...]

top-k with k of 4096:

The answer is simple: 196. This number includes 193 countries recognized by the United Nations, 2 non-member states (Taiwan and Kosovo), and 1 non-sovereign territory (Macau, which is a special administrative region of China). [...]

top-k with k of 16 384

There are 196 countries recognized by the United Nations. However, there are many more countries that are not recognized by the UN, and some countries have a disputed status.
The number of countries in the world can vary depending on the criteria used to define a "country." [...]

top-η rate with η of 10%:

and a a pe in in - and and and the most in all (b and,et the the or 1 a the - a, 1 the in the, 1, and, the - a the the, the thed ust- it, the the for,, and the the the the the [...]

top-η rate with η of 50%

to The first time, the first time, the first time, the first time, the first time, the first time, the first time, the first time, the first time, the first time, [...]

top-η rate with η of 90%:

There are 196 countries recognized by the United Nations. However, there are many more countries that are not recognized or are disputed. [...]

top-η rate with η of 100%

There are 196 countries recognized by the United Nations. However, the number of countries can vary depending on the source and criteria used to define a country. [...]

E.2 Second Prompt

Our first prompt was: `"Who was the first woman to win a Nobel Prize in physics and in what year?"`. Here is how the several models performed

Basemodel:

Marie Curie was the first woman to win a Nobel Prize in physics in 1903, and the first woman to win a Nobel Prize in chemistry in 1911.
She was awarded the Nobel Prize in Physics in 1903 for [...]

top-k with k of 2 048

How many times has she won a Nobel Prize? Who was the first woman to win a Nobel Prize in physics? Who was the first woman to win a Nobel Prize in chemistry? Who was the first woman to win a Nobel Prize in literature? [...]

top-k with k of 2 304:

The answer is Marie Curie, who won the first Nobel Prize in Physics in 1901 for her work on radioactivity. She was awarded the Nobel Prize in Physics in 1903, but she was awarded the Nobel Prize in Chemistry in 1911 for her work on radioactive elements. [...]

top-k with k of 2 560

The answer is Marie Curie, who won the first Nobel Prize in Physics in 1903, and the first woman to win a Nobel Prize in any field in 1911, when she was awarded the Nobel Prize in Chemistry for her pioneering work on radioactivity. [...]

top-η rate with η of 78%:

The first woman to win a Nobel Prize in physics was Dr. Mae J. West, who won the Nobel Prize in Physics in 1990. [...]

top-η rate with η of 79%

The first woman to do so was Marie Curie, who won the Nobel Prize in 1903. [...]

To get the compute reduction of the training, we measure the FLOPs it takes to forward a prompt using the `torch.profiler`. One issue we encounter is that an arbitrary amount of tokens was generated for each prompt. To prevent this, we set the temperature of the model to zero and only generated 20 new tokens, because 20 tokens would always be generated. With that, we get:

- basemodel: $66, 957, 869, 517$ FLOPs
- top-k algorithm, $k = 64$: $1, 746, 313, 677$ FLOPs (97.4% compute reduction)
- top-k algorithm, $k = 256$: $6, 927, 720, 909$ FLOPs (89.7% compute reduction)
- top-k algorithm, $k = 1024$: $27, 653, 349, 837$ FLOPs (58.7% compute reduction)
- top-k algorithm, $k = 4096$: $59, 175, 862, 733$ FLOPs (11.6% compute reduction)
- top-k algorithm, $k = 16384$: $66, 957, 869, 517$ FLOPs (0.0% compute reduction)
- top-k algorithm, $k = 2048$: $55, 283, 548, 621$ FLOPs (17.4% compute reduction)
- top-k algorithm, $k = 2304$: $55, 770, 087, 885$ FLOPs (16.7% compute reduction)
- top-k algorithm, $k = 2560$: $56, 256, 627, 149$ FLOPs (16.0% compute reduction)
- top-η rate algorithm, $\eta = 10\%$: $6, 693, 257, 677$ FLOPs (90.0% compute reduction)
- top-η rate algorithm, $\eta = 50\%$: $33, 491, 821, 005$ FLOPs (50.0% compute reduction)
- top-η rate algorithm, $\eta = 78\%$: $52, 224, 018, 893$ FLOPs (22.0% compute reduction)
- top-η rate algorithm, $\eta = 79\%$: $52, 881, 582, 541$ FLOPs (21.0% compute reduction)
- top-η rate algorithm, $\eta = 90\%$: $60, 263, 397, 837$ FLOPs (10.0% compute reduction)
- top-η rate algorithm, $\eta = 100\%$: $66, 964, 464, 077$ FLOPs (0.0% compute reduction)

F Discussion, Limitations, and Outlook

To our knowledge, we were also the first to expand the top-k algorithm to make it also work for LLMs, where we obtained very good accuracy. In analyzing the behavior of sparsity across transformer layers, we identify a key challenge: different layers induce different multiplication magnitudes, which complicates a constant threshold of a fixed k. To address this, we propose the top-η rate algorithm, a rate-controlled variant that normalizes selection across layers.

On a high level, this shows that the rate-controlled sampling approach works better for larger models and solves the original drawback, the big difference in the magnitude of the models' matrices, which made previous sampling approaches unusable for LLMs.

Our work still offers scope for further research on this approach. First, we evaluate methods through FLOP counts rather than direct hardware measurements. Because conventional matrix multiplication is deeply optimized for CUDA-enabled GPUs, more work could be spent on implementing alternative operations closer to the CUDA architecture, which would automatically yield lower energy use and shorter wall-clock time.

Furthermore, in future research the tests on LLMs could be improved by also testing the accuracy after training on further benchmarks, like the BERT-score. Methodologically, one could also combine the top-k algorithm and the top-η rate algorithm. For example, by retaining at least k dimensions and, above that threshold, applying a rate of η. Another outlook is selective layer sparsification. Rather than replacing all layers uniformly, one could target specific layers. This can be experimented with both smaller models and LLMs. Prior work, e.g., on Layer Pruning, suggests that later layers may be less critical, and tailoring sparsity to layer importance could yield better overall performance.

Finally, the top-η rate algorithm could be evaluated on larger models, because a higher selection rate η can be used while likely maintaining fine-tuning accuracy and therefore can save more energy.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See our approach and its evaluation against baselines.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Appendix F.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

Justification: We do not have theoretical results that require a proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide a code repository (https://anonymous.4open.science/r/anonymous-repo-aaai2026-58B0) to reproduce all experiments, see also further details described in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes] .

Justification: We provide a code repository (https://anonymous.4open.science/r/anonymous-repo-aaai2026-58B0).

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide this information, see https://anonymous.4open.science/r/anonymous-repo-aaai2026-58B0.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We had limited time and resources. Further, this is prototypical work.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See our code repository (https://anonymous.4open.science/r/anonymous-repo-aaai2026-58B0).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Given the considered problem, there are no ethical issues.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes] .

    Justification: We point to potential applications of the solution, e.g. energy-efficient training of LLMs.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: There are no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA] .

    Justification: We use own code and cite relevant other works.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We did not do research with human subjects nor crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA] .

    Justification: We did not use LLMs to derive core methods of the paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.