Mamba-PTQ: Outlier Channels in Recurrent Large Language Models

Alessandro Pierro* [1 2] **Steven Abreu*** [1 3]

Abstract

Modern recurrent layers are emerging as a promising path toward edge deployment of foundation models, especially in the context of large language models (LLMs). Compressing the whole input sequence in a finite-dimensional representation enables recurrent layers to model long-range dependencies while maintaining a constant inference cost for each token and a fixed memory requirement. However, the practical deployment of LLMs in resource-limited environments often requires further model compression, such as quantization and pruning. While these techniques are well-established for attention-based models, their effects on recurrent layers remain underexplored.

In this preliminary work, we focus on post-training quantization for recurrent LLMs and show that Mamba models exhibit the same pattern of outlier channels observed in attention-based LLMs. We show that the reason for difficulty of quantizing SSMs is caused by activation outliers, similar to those observed in transformer-based LLMs. We report baseline results for post-training quantization of Mamba that do not take into account the activation outliers and suggest first steps for outlier-aware quantization.

1. Introduction

Attention-based models, also known as Transformers (Vaswani et al., 2023), constitute the current state-of-the-art backbone for large language models (LLMs) (Brown et al., 2020). However, their powerful modeling capabilities come with significant computational requirements, resulting in high inference costs and limiting the deployment on edge and low-power devices. Novel recurrent neural network (RNN) architectures, informed mainly by recent work on state space models (SSMs) (Gu et al., 2020; 2022), are now emerging as promising alternatives for sequence modeling tasks, either in isolation (Poli et al., 2023; Gu & Dao, 2023; Peng et al., 2023) or as hybrid models interleaving recurrent and attention blocks (De et al., 2024; Lieber et al., 2024; Botev et al., 2024). In particular, RNNs compress the input sequence into a finite-dimensional representation, decoupling the computational and memory cost of each token's forward pass from the sequence's length. Hence, they provide better scalability to long context scenarios than vanilla self-attention, which scales quadratically with sequence length.

However, similarly to Transformers, deploying recurrence-based LLMs at scale or in resource-constrained environments requires advanced model optimization techniques, such as quantization, pruning, and knowledge distillation. While applying these techniques starts to be well understood in the context of attention-based LLMs, model optimization for recurrent and hybrid architectures remains an important yet underexplored topic. In this paper, we focus on quantization and analyze its impact on Mamba (Gu & Dao, 2023) model family, drawing connections to previous work on quantized LLMs.

2. Quantization and outlier channels in LLMs

Quantization is a compression technique that reduces the numerical precision of a model's weights and activations to integer datatypes in order to facilitate inference (Jacob et al., 2017). We adopt symmetric per-tensor quantization: given a tensor $\mathbf{x}$ and a bit precision n, its quantized representation is computed as:

$$\bar{\mathbf{x}}_n = \left\lfloor \frac{(2^{n-1}-1)\mathbf{x}}{\max|\mathbf{x}|} \right\rceil = \lfloor s_x \mathbf{x} \rceil \tag{1}$$

where the *quantization scale* s_x is a scalar. The benefits of quantization for inference efficiency are twofold. Firstly, weight quantization reduces the memory footprint of the model, which is especially beneficial in the memory-bound regime of autoregressive generation. Secondly, when both weights and activations are quantized, matrix multiplications can be offloaded to the integer processing units, which typically offer higher throughput and energy efficiency than floating point units.

*Equal contribution [1]Neuromorphic Computing Lab, Intel Labs, Neubiberg, Germany [2]Institute of Informatics, LMU Munich, Germany [3]CogniGron Center & Bernoulli Institute, University of Groningen, Groningen, Netherlands. Correspondence to: Alessandro Pierro <alessandro.pierro@intel.com>.

Work presented at the Efficient Systems for Foundation Models Workshop at the 41st International Conference on Machine Learning, Vienna, Austria.

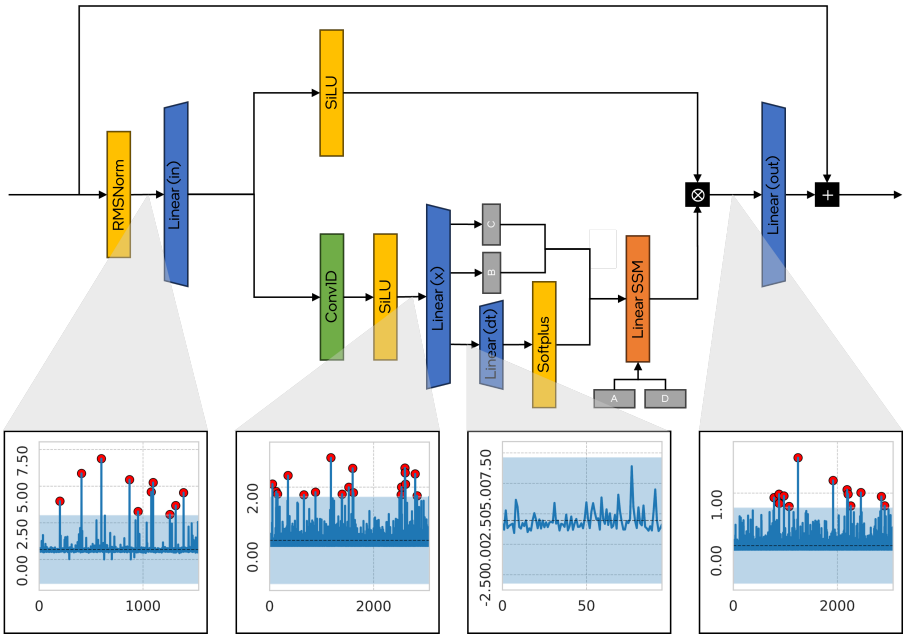

Figure 1: Architecture diagram of the Mamba block and details on the absolute maximum activation (on the y-axis) across channels (x-axis), measured on a subset of WikiText-2 (Merity et al., 2016) for Mamba-130m. Shaded regions account for six standard deviations.

Most state-of-the-art techniques for quantizing LLMs are based on the empirical observation of *outlier channels* (Bondarenko et al., 2021), a small percentage of model dimensions with a dynamic range that is consistently larger than the rest. This phenomenon complicates activation quantization since the large abs max values from the outlier channels deteriorate the effective bit precision of the remaining channels. A possible solution would be maintaining a different quantization scale for each channel, which is not hardware-friendly on current GPU architectures (Xiao et al., 2024). Various strategies have been proposed to circumvent this issue. For instance, some methods treat outlier channels separately, either by maintaining them in floating point format (Dettmers et al., 2022) or by representing them with two integer channels each (Zhang et al., 2024). Other approaches modify the transformer architecture to prevent the emergence of outliers (Bondarenko et al., 2023), while some partially shift the quantization difficulty to the weights, thereby mitigating the impact of outliers (Xiao et al., 2024).

We make the first steps towards post-training quantization for recurrent LLMs, focusing on the Mamba (Gu & Dao, 2023) model family. We analyse the activation patterns of Mamba to assess the presence of outliers, which we define as those channels having an absolute maximum activation beyond six standard deviations from the layer mean, following prior practice (Bondarenko et al., 2021). Figure 1 reports the pre-activations of the linear block of a layer from Mamba-130m (similar results were observed for the other model sizes), measured running the model on a subset of WikiText-2 (Merity et al., 2016). We observe distinct outlier patterns. The pre-activations of the three largest linear layers (*in*, *x*, and *out*), consistently with what was observed for attention-based LLMs, show outliers accounting for less than 1% of channels. However, while the outliers of the first linear block are mostly consistent across layers, the remaining two blocks exhibit no regular behavior. The linear layer projecting the SSM's time steps (*dt*) shows almost no outliers. Similarly to (Dettmers et al., 2022), we further assess the importance of the outlier channels for the model's predictions by evaluating the impact of zeroing out the outliers on downstream accuracy. For Mamba-130m and Mamba-2.8b, we observe a drop in average accuracy of 12.61% and 17.49%, respectively, suggesting that these channels play a significant role in the model dynamics. Extended results are available in the Appendix in Table 2.

3. Method

3.1. Mamba model

Existing state space models are described by the following dynamics:

$$x^k = Ax^{k-1} + Bu^k \tag{2}$$

$$y^k = Cx^k + Du^k \tag{3}$$

where A is the recurrent matrix, B is the input matrix, C is the output matrix, and D is the residual matrix from

the input to the output. State space models of this form must treat every input token equally, as the input matrix B is fixed, such that the SSM cannot focus on or ignore specific tokens. This is a major shortcoming compared to transformer architectures, whose attention mechanism allows for such interactions and thus limits the performance of SSMs, especially on language tasks.

The key innovation of Mamba over previous SSMs is its ability to perform such content-based reasoning. By letting Mamba's parameters depend on the input, the model effectively gains the ability to filter out irrelevant information so that the relevant context can be compressed more efficiently into the hidden state. Specifically, the parameters for Mamba's SSM block are obtained by:

$$B_t, \Delta_t, C_t = W^{proj} u_t \tag{4}$$

$$\overline{\Delta_t} = \sigma^+(W^{dt}\Delta_t) \tag{5}$$

$$\overline{A_t} = \exp\left(-\exp\left(A^{log}\overline{\Delta_t}\right)\right) \tag{6}$$

$$\overline{B_t} = \overline{\Delta_t} \odot B_t \tag{7}$$

where u_t is the input to the SSM block, W^{proj}, A^{log} and W^{dt} are time-invariant weight matrices, σ^+ denotes the softplus function and $\odot$ denotes element-wise multiplication. The weight matrices B_t and C_t are thus directly dependent on the input u_t, whereas the recurrent matrix A_t is dependent on the input u_t only through the input-dependent timescale parameter Δ_t. The hidden state h_t and output y_t of the SSM block is then computed as:

$$h_t = \overline{A_t} \odot h_{t-1} + \overline{B_t} \odot u_t \tag{8}$$

$$y_t = C_t h_t + D_t \odot u_t \tag{9}$$

As shown in Figure 1, each layer in the Mamba architecture also includes additional gating, nonlinearities, normalization, causal convolution, and linear blocks.

3.2. Baseline quantization

In order to quantize Mamba, we distinguish between Mamba's pre-trained weights and its activations. Importantly, due to the input-dependent parameterization, we consider only input-independent parameters as weights, such as A^{log}, while we consider input-dependent parameters like $\overline{A_t}$ as activations.

We adopt symmetric, per-tensor quantization for weights and activations as described in section 2, using the absolute maximum (absmax) of the tensor for calibration.

For our experiments using naive quantization on the activations, we quantize the output from all linear layers (including the matrices B_t, Δ_t, C_t from Equation 4), but we do not quantize the effective weight matrices $\overline{A_t}, \overline{\Delta_t}, \overline{B_t}$. We further do not quantize the output from the SSM block y_t but only quantize the output from the downstream out projection linear layer.

We use standard notation to denote quantization with n-bit integers for weights as Wn and quantization with n-bit integers for activations as An. For example, 8-bit weight quantization and 4-bit activation quantization is denoted by W8A8.

3.3. Outlier-aware quantization (e.g., SmoothQuant)

The naive absmax quantization is sensitive to outliers. A large value in the tensor $\mathbf{x}$ will yield a small scale $s_x = \frac{2^{n-1}-1}{\max|\mathbf{x}|}$, thus leading to larger rounding errors for the same n-bit quantization precision. As discussed in section 2, outliers (particularly in activations) are the subject of research in LLM quantization.

Most notably, the SmoothQuant method proposed by Xiao *et al.* (Xiao et al., 2024) exploits the fact that outliers exist in activations but not in the weights. SmoothQuant smooths the activation outliers by partially taking them into the preceding weights. Because activation outliers typically persist in the same activation channels, a weight matrix with per-channel quantization can absorb part of the quantization difficulty from the subsequent activations. As such, SmoothQuant introduces a per-channel smoothing factor $s \in \mathbb{R}^{C_i}$ where C_i is the dimension of the activations X and, equivalently, the number of output channels of the weight matrix W. This smpoothing factor is used to scale the weights and activations:

$$(\mathbf{X}\text{diag}(s)^{-1}) \cdot (\text{diag}(s)\mathbf{W}) = \hat{\mathbf{X}}\hat{\mathbf{W}} \tag{10}$$

The aim is to choose a smoothing factor s so that $\hat{\mathbf{X}} = \mathbf{X}\text{diag}(s)^{-1}$ is easy to quantize. However, simply choosing $s_j = \max(|\mathbf{X}_j|)$ where $j = 1, \ldots, C_i$ to minimize the difficulty in quantization activations, will push all these difficulties into the weights. On the other hand, we can choose $s_j = 1/\max(|\mathbf{W}_j|)$ to move all the quantization difficulty from the weights into the activations. The authors propose a new hyperparameter, the migration strength α, to control how much difficulty we want to migrate from activations to weights, using the equation:

$$s_j = \frac{\max(|\mathbf{X}_j|)^\alpha}{\max(|\mathbf{W}_j|)^{1-\alpha}} \tag{11}$$

where a smaller α will leave more difficulty with the activations, and a larger α will migrate more difficulty to the weights. The authors suggest to use a default value of $\alpha = 0.5$ and a larger α for models where activation outliers are more significant such that more quantization difficulty is moved into the weights.

Table 1: One-shot accuracy on downstream tasks for Mamba-1.4b across different quantization configurations.

	LAMBADA	HellaSwag	PIQA	WinoGrande	RTE	COPA
Baseline	64.95%	59.11%	74.16%	61.4%	48.01%	79%
W8 (mlp)	64.43%	44.91%	74.32%	60.06%	48.01%	77.00%
W8 (all)	63.01%	44.71%	73.07%	60.06%	51.62%	76.00%
W4 (mlp)	0.02%	25.70%	52.29%	51.54%	52.35%	56.00%
W4 (all)	0.00%	25.72%	52.39%	50.99%	55.23%	64.00%
W8A8 (mlp)	63.11%	44.42%	73.01%	60.06%	51.62%	76%
W8A8 (all)	55.35%	43.84%	70.24%	54.3%	52.71%	75%

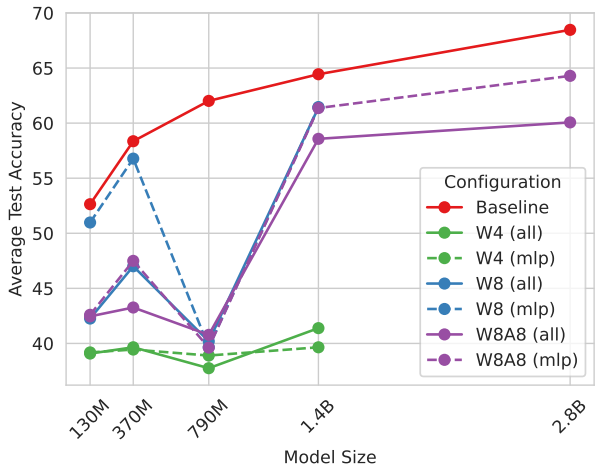

Figure 2: Average one-shot accuracy on downtream tasks across model sizes for Mamba with different quantization configurations. The accuracy is averaged over all tasks shown in Table 1.

4. Experiments

4.1. Experimental setup

We assess the impact of different quantization configurations on the zero-shot accuracy of six downstream tasks: LAMBADA (Paperno et al., 2016), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2019), WinoGrande (Sakaguchi et al., 2019), RTE (Wang et al., 2019), and COPA (Roemmele et al.). We explicitly neglect perplexity benchmarking, since prior work noted how it may not be informative of the actual task performance degradation (Sun et al., 2021).

We run three different experimental conditions:

1. To assess the importance of outlier channels, we analyze the impact of removing outlier channels on downstream task accuracy.
2. We then analyze the effect of naive quantization of only the pre-trained weights on downstream task accuracy.
3. Finally, we analyze the effect of quantization on the pre-trained weights, as well as the activations, without accounting for activation outliers.

Full results for the effect of Experiment **1** for removing outlier channels are found in Table 2 in the Appendix. We present an overview of the findings from Experiment **2** and **3** in Table 1 on the Mamba-1.4b model, while results for all other model sizes are presented in Table 3 in the Appendix.

5. Discussion

In this preliminary work, we make the first steps towards post-training quantization of Mamba, in order to inform future edge deployments of recurrent LLMs based on selective state space models such as Mamba. We have shown that the difficulty of quantizing Mamba is caused by activation outliers, similar to those observed in transformer-based LLMs. We presented baseline results for post-training quantization of Mamba that does not take into account the activation outliers and a first proposal for outlier-aware quantization of Mamba.

5.1. Future work

As this area is under rapid development, several opportunities exist to extend this work. Firstly, a similar analysis could be performed on other recurrent LLMs, such as the RWKV family (Peng et al., 2023), the novel Mamba-2 architecture (Dao & Gu, 2024), or hybrid models such as Griffin (De et al., 2024) and RecurrentGemma (Botev et al., 2024). Secondly, additional work should be done to convert the SSM dynamics fully to integer operations, as previously demonstrated by (Blouw et al., 2021), and explore the use of quantized activations. Lastly, it will be interesting to see how quantized recurrent LLMs perform at the edge in energy-constrained scenarios for real-time multimodal processing (Shrestha et al., 2024), as the specific of the hardware architecture could provide additional guidance on model compression requirements.

Acknowledgements

We thank Jonathan Timcheck, Philipp Stratmann, and Sumit Shrestha for helpful comments and discussions.

References

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. PIQA: Reasoning about Physical Commonsense in Natural Language, November 2019. URL http://arxiv.org/abs/1911.11641. arXiv:1911.11641 [cs].

Blouw, P., Malik, G., Morcos, B., Voelker, A., and Eliasmith, C. Hardware aware training for efficient keyword spotting on general purpose and specialized hardware. In *Research Symposium on Tiny Machine Learning*, 2021. URL https://openreview.net/forum?id=iqcQPOrPcb7.

Bondarenko, Y., Nagel, M., and Blankevoort, T. Understanding and Overcoming the Challenges of Efficient Transformer Quantization, September 2021. URL http://arxiv.org/abs/2109.12948. arXiv:2109.12948 [cs].

Bondarenko, Y., Nagel, M., and Blankevoort, T. Quantizable Transformers: Removing Outliers by Helping Attention Heads Do Nothing, November 2023. URL http://arxiv.org/abs/2306.12929. arXiv:2306.12929 [cs].

Botev, A., De, S., Smith, S. L., Fernando, A., Muraru, G.-C., Haroun, R., Berrada, L., Pascanu, R., Sessa, P. G., Dadashi, R., Hussenot, L., Ferret, J., Girgin, S., Bachem, O., Andreev, A., Kenealy, K., Mesnard, T., Hardin, C., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M. S., Love, J., Tafti, P., Joulin, A., Fiedel, N., Senter, E., Chen, Y., Srinivasan, S., Desjardins, G., Budden, D., Doucet, A., Vikram, S., Paszke, A., Gale, T., Borgeaud, S., Chen, C., Brock, A., Paterson, A., Brennan, J., Risdal, M., Gundluru, R., Devanathan, N., Mooney, P., Chauhan, N., Culliton, P., Martins, L. G., Bandy, E., Huntsperger, D., Cameron, G., Zucker, A., Warkentin, T., Peran, L., Giang, M., Ghahramani, Z., Farabet, C., Kavukcuoglu, K., Hassabis, D., Hadsell, R., Teh, Y. W., and de Frietas, N. RecurrentGemma: Moving Past Transformers for Efficient Open Language Models, April 2024. URL http://arxiv.org/abs/2404.07839. arXiv:2404.07839 [cs].

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language Models are Few-Shot Learners, July 2020. URL http://arxiv.org/abs/2005.14165. arXiv:2005.14165 [cs].

Dao, T. and Gu, A. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024.

De, S., Smith, S. L., Fernando, A., Botev, A., Cristian-Muraru, G., Gu, A., Haroun, R., Berrada, L., Chen, Y., Srinivasan, S., Desjardins, G., Doucet, A., Budden, D., Teh, Y. W., Pascanu, R., De Freitas, N., and Gulcehre, C. Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models, February 2024. URL http://arxiv.org/abs/2402.19427. arXiv:2402.19427 [cs].

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale, November 2022. URL http://arxiv.org/abs/2208.07339. arXiv:2208.07339 [cs].

Gu, A. and Dao, T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, December 2023. URL http://arxiv.org/abs/2312.00752. arXiv:2312.00752 [cs].

Gu, A., Dao, T., Ermon, S., Rudra, A., and Re, C. HiPPO: Recurrent Memory with Optimal Polynomial Projections, October 2020. URL http://arxiv.org/abs/2008.07669. arXiv:2008.07669 [cs, stat].

Gu, A., Goel, K., and Ré, C. Efficiently Modeling Long Sequences with Structured State Spaces, August 2022. URL http://arxiv.org/abs/2111.00396. arXiv:2111.00396 [cs].

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference, December 2017. URL http://arxiv.org/abs/1712.05877. arXiv:1712.05877 [cs, stat].

Lieber, O., Lenz, B., Bata, H., Cohen, G., Osin, J., Dalmedigos, I., Safahi, E., Meirom, S., Belinkov, Y., Shalev-Shwartz, S., Abend, O., Alon, R., Asida, T., Bergman, A., Glozman, R., Gokhman, M., Manevich, A., Ratner, N., Rozen, N., Shwartz, E., Zusman, M., and Shoham, Y. Jamba: A Hybrid Transformer-Mamba Language Model, March 2024. URL http://arxiv.org/abs/2403.19887. arXiv:2403.19887 [cs].

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer Sentinel Mixture Models, September 2016. URL http://arxiv.org/abs/1609.07843. arXiv:1609.07843 [cs].

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q. N., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context, June 2016. URL http://arxiv.org/abs/1606.06031. arXiv:1606.06031 [cs].

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Grella, M., GV, K. K., He, X., Hou, H., Lin, J., Kazienko, P., Kocon, J., Kong, J., Koptyra, B., Lau, H., Mantri, K. S. I., Mom, F., Saito, A., Song, G., Tang, X., Wang, B., Wind, J. S., Wozniak, S., Zhang, R., Zhang, Z., Zhao, Q., Zhou, P., Zhou, Q., Zhu, J., and Zhu, R.-J. RWKV: Reinventing RNNs for the Transformer Era, December 2023. URL http://arxiv.org/abs/2305.13048. arXiv:2305.13048 [cs].

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S., and Ré, C. Hyena Hierarchy: Towards Larger Convolutional Language Models, April 2023. URL http://arxiv.org/abs/2302.10866. arXiv:2302.10866 [cs].

Roemmele, M., Bejan, C. A., and Gordon, A. S. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. WinoGrande: An Adversarial Winograd Schema Challenge at Scale, November 2019. URL http://arxiv.org/abs/1907.10641. arXiv:1907.10641 [cs].

Shrestha, S. B., Timcheck, J., Frady, P., Campos-Macias, L., and Davies, M. Efficient Video and Audio Processing with Loihi 2. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 13481–13485, April 2024. doi: 10.1109/ICASSP48485.2024.10448003. URL https://ieeexplore.ieee.org/abstract/document/10448003.

Sun, S., Krishna, K., Mattarella-Micke, A., and Iyyer, M. Do long-range language models actually use long-range context?, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention Is All You Need, August 2023. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs].

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding, February 2019. URL http://arxiv.org/abs/1804.07461. arXiv:1804.07461 [cs].

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models, March 2024. URL http://arxiv.org/abs/2211.10438. arXiv:2211.10438 [cs].

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a Machine Really Finish Your Sentence?, May 2019. URL http://arxiv.org/abs/1905.07830. arXiv:1905.07830 [cs].

Zhang, Y., Yang, F., Peng, S., Wang, F., and Pan, A. FlattenQuant: Breaking Through the Inference Compute-bound for Large Language Models with Per-tensor Quantization, February 2024. URL http://arxiv.org/abs/2402.17985. arXiv:2402.17985 [cs].

A. Additional results

Herein we present all additional experimental results from the experiments presented in this paper.

A.1. Impact of removing outlier channels

Table 2 shows the accuracy on all evaluated tasks for the Mamba-130m model and Mamba-2.8B model, with different rows indicating the outlier removal specific to particular layers, or across the entire model.

Table 2: Impact of removing outlier channels on downstream task accuracy.

Model	LAMBADA	HellaSwag	PIQA	WinoGrande	RTE	COPA	Avg.
Mamba-130m							
Baseline	44.25%	35.25%	64.47%	52%	54.87%	65%	52.64%
Linear(in) only	0%	25.61%	53.75%	51.14%	52.71%	53%	39.37%
Linear(x) only	26.88%	26.73%	58.49%	49.57%	55.6%	63%	46.71%
Linear(dt) only	44.25%	30.80%	64.47%	52.09%	54.87%	65%	51.91%
Linear(out) only	32.45%	30.42%	65.34%	53.35%	53.43%	70%	50.83%
All	0%	25.83%	54.52%	53.12%	52.71%	54%	40.03%
Mamba-2.8B							
Baseline	69.24%	66.16%	75, 24%	63.46%	52.71%	84%	68.46%
All	7.45%	49.43%	66.92%	58.25%	53.79%	70%	50.97%

A.2. Impact of quantization on downstream task accuracy

Table 3 shows the accuracy on all evaluated tasks for all Mamba models and all quantization configurations.

Table 3: One-shot accuracy on downstream tasks for the Mamba model family across different quantization configurations.

Model	LAMBADA	HellaSwag	PIQA	WinoGrande	RTE	COPA
Mamba-130m						
Baseline	44.25%	35.25%	64.47%	52%	54.87%	65%
W8 (mlp)	42.48%	30.71%	64.09%	52.88%	52.71%	63.00%
W8 (all)	5.53%	28.03%	58.11%	50.59%	47.29%	64.00%
W4 (mlp)	0.00%	25.80%	51.74%	50.75%	49.82%	57.00%
W4 (all)	0.00%	25.34%	53.43%	50.36%	52.35%	53.00%
W8A8 (mlp)	5.72%	28.09%	57.83%	51.3%	47.65%	65%
W8A8 (all)	4.31%	27.72%	56.47%	51.07%	50.18%	65%
Mamba-370m						
Baseline	55.62%	46.48%	69.48%	55.49%	53.07%	70%
W8 (mlp)	54.86%	37.17%	69.04%	55.80%	53.79%	70.00%
W8 (all)	16.61%	31.58%	61.37%	51.38%	49.10%	72.00%
W4 (mlp)	0.00%	25.23%	53.81%	50.28%	52.35%	55.00%
W4 (all)	0.00%	25.72%	53.10%	51.38%	52.71%	55.00%
W8A8 (mlp)	16.61%	36.87%	61.53%	51.22%	48.74%	70%
W8A8 (all)	10.29%	31.04%	58.76%	50.99%	45.49%	63%
Mamba-790m						
Baseline	61.71%	55.07%	72.14%	55.96%	55.23%	72%
W8 (mlp)	2.64%	25.30%	54.13%	50.83%	53.07%	52.00%
W8 (all)	1.20%	25.84%	54.08%	51.07%	55.96%	53.00%
W4 (mlp)	0.00%	25.93%	53.05%	48.22%	46.21%	60.00%
W4 (all)	0.00%	25.29%	50.98%	47.67%	46.57%	56.00%
W8A8 (mlp)	1.44%	25.37%	54.30%	50.2%	53.43%	53%
W8A8 (all)	0.95%	25.77%	54.73%	50.99%	55.23%	57%
Mamba-1.4B						
Baseline	64.95%	59.11%	74.16%	61.4%	48.01%	79%
W8 (mlp)	64.43%	44.91%	74.32%	60.06%	48.01%	77.00%
W8 (all)	63.01%	44.71%	73.07%	60.06%	51.62%	76.00%
W4 (mlp)	0.02%	25.70%	52.29%	51.54%	52.35%	56.00%
W4 (all)	0.00%	25.72%	52.39%	50.99%	55.23%	64.00%
W8A8 (mlp)	63.11%	44.42%	73.01%	60.06%	51.62%	76%
W8A8 (all)	55.35%	43.84%	70.24%	54.3%	52.71%	75%
Mamba-2.8B						
Baseline	69.24%	66.16%	75.24%	63.46%	52.71%	84%
W8A8 (mlp)	64.64%	48.74%	73.72%	64.33%	56.32%	78%
W8A8 (all)	51.39%	47.64%	70.24%	57.62%	54.51%	79%