NOWAG: A UNIFIED FRAMEWORK FOR SHAPE PRESERVING COMPRESSION OF LARGE LANGUAGE MODELS

Lawrence Liu[1] **Inesh Chakrabarti**[1] **Yixiao Li**[2] **Mengdi Wang**[3] **Tuo Zhao**[2]
Lin F. Yang[1]
[1]*University of California, Los Angeles* [2]*Georgia Institute of Technology* [3]*Princeton University*
{lawrencerliu, inesh33}@ucla.edu, yixiaoli@gatech.edu
mengdiw@princeton.edu, tourzhao@gatech.edu, linyang@ee.ucla.edu

ABSTRACT

Large language models (LLMs) exhibit remarkable performance across various natural language processing tasks but suffer from immense computational and memory demands, limiting their deployment in resource-constrained environments. To address this challenge, we propose NoWag (**No**rmalized **W**eight and **A**ctivation **G**uided Compression), a unified framework for zero-shot shape preserving compression algorithms. We compressed Llama-2 7B/13B/70B and Llama-3 8B/70B models, using two popular forms of shape-preserving compression, vector quantization NoWag-VQ (**NoWag** for **V**ector **Q**uantization), and unstructured/structured pruning NoWag-P (**NoWag** for **P**runing). We found that NoWag-VQ significantly outperforms state-of-the-art zero shot VQ, and that NoWag-P performs competitively against state-of-the-art methods. Our code is available at https://github.com/LawrenceRLiu/NoWag

1 INTRODUCTION

Large language models (LLMs) (Brown et al., 2020) have demonstrated remarkable capabilities across a wide range of natural language processing tasks, but their immense computational and memory requirements during inference pose significant challenges for deployment. Consequently, post-training compression techniques have emerged as a promising tool to reduce model size and computational overhead while maintaining accuracy. Two promising families of methods for post-training compression are Pruning (Lecun et al., 1989; Hassibi et al., 1993; Han et al., 2015) and Quantization (Yao et al., 2022; Dettmers et al., 2022b; Ahmadian et al., 2023).

Pruning aims to remove redundant parameters from LLMs while preserving performance. We will focus on two forms of pruning, unstructured pruning (Liao et al., 2023), which removes zeroed out, and N:M semi-structured pruning (Huang et al., 2024), where N of every M elements are zeroed out. SparseGPT (Frantar & Alistarh, 2023) introduced an efficient, unstructured and semi-structured pruning method that leverages Hessian-based weight updates to minimize accuracy loss. More recently, Wanda (Sun et al., 2024) demonstrated a simple yet effective unstructured and semi-structured pruning method that requires no weight updates or hessian computation, making it significantly faster and easier to apply than SparseGPT. However current hardware only supports 2:4 semi-structured sparsity, which results in significant post compression performance loss.

A more effective compression method is quantization, which reduces the number of bits used to store each weight (Kuzmin et al., 2023). For the scope of this paper we focus on a common form of quantization, Weight Only Post Training Quantization (PTQ). Pioneering works (Frantar et al., 2023; Lin et al., 2024; Kim et al., 2024) focused on scalar quantization. For extreme compression (e.g., ≤ 4 bits per weight), Vector Quantization (VQ), where groups of d consecutive weights are quantized together, has demonstrated superior performance because the codebook to be shaped to the distribution of weights (Egiazarian et al., 2024; van Baalen et al., 2024; Tseng et al., 2024a; Liu et al., 2024). However, most current algorithms all share at least one of the following two drawbacks: First, an expensive weight update process necessitating matrix inversion, similar to

SparseGPT. Second, sampling an adequately accurate hessian for quantization requires as much as 25 million tokens and $\sim$ 1TB of CPU memory for Llama-3 70B, creating a new compute bottleneck for quantization.

In this work, we address these issues by formulating a unifying framework for shape preserving compression algorithms, where the compressed weight matrix has the same shape as the original uncompressed counterpart but can be stored with less memory. This method is weight update free, less dependent on calibration data, and has a novel normalization that is beneficial to both pruning and quantization. We term this family of compression methods NoWag (**No**rmalized **W**eight and **A**ctivation **G**uided Compression). We show that the VQ variation of NoWag, NoWag-VQ (**NoWag** for **V**ector **Q**uantization), outperforms the SOTA one-shot VQs QuIP# (Tseng et al., 2024a), at bits per value, while using 48x less calibration data. Furthermore, we show that the pruning variation of NoWag, NoWag-P (**NoWag** for **P**runing), offers comparable performance to recent pruning algorithms, Wanda and SparseGPT, and results in greater preservation of Language Modeling abilities.

2 METHODS

In this work, we focus on "one-shot" compression methods for large language models (LLMs). Here, "one-shot" refers to directly compressing the model based on the calibration data without fine-tuning to adjust the compressed model parameters. Given a trained LLM, our goal is to obtain a compressed model that significantly reduces the computational and memory requirements while retaining as much general performance as possible. Due to the large number of parameters, using global optimization for compression is computationally infeasible. As a result, one-shot compression methods commonly optimize each linear layer independently (Nagel et al., 2020). Following this principle, our method compresses each linear layer by minimizing a data-aware loss function, which we define below.

Problem Formulation Consider a linear layer in an LLM with weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. Given input activations $x \in \mathbb{R}^{d_{\text{in}}}$, the output is computed as $y = Wx$, where $y \in \mathbb{R}^{d_{\text{out}}}$.

Our objective is to find a compressed weight matrix $\hat{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ that retains the same dimensions but requires less memory while minimizing the deviation from the original model's behavior. To incorporate data awareness, we sample n sequences of length l from a calibration dataset and collect the corresponding activation samples $X^T \in \mathbb{R}^{m \times d_{\text{in}}}$ where $m = n \times l$. Given these activation samples, we define a data-weighted loss function for compression.

Compression Objective To ensure numerical stability and enhance compression efficiency, we first normalize W to obtain $\bar{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ using normalization vectors $r^{(1)} \in \mathbb{R}^{d_{\text{in}}}$ and $r^{(2)} \in \mathbb{R}^{d_{\text{out}}}$:

$$\bar{W}_{ij} = \frac{1}{r_i^{(2)}} \left(\frac{W_{ij}}{r_j^{(1)}} \right), \quad r_j^{(1)} = \sqrt{\sum_{i=1}^{d_{\text{out}}} W_{ij}^2}, \quad \forall j \in [d_{\text{in}}], \quad r_i^{(2)} = \sqrt{\sum_{j=1}^{d_{\text{in}}} \left(\frac{W_{ij}}{r_j^{(1)}} \right)^2}, \quad \forall i \in [d_{\text{out}}].$$

The compressed weight matrix $\hat{\mathrm{W}}$ is obtained by minimizing the following weighted Frobenius norm:

$$\tilde{\ell}(\hat{\mathrm{W}}) = \|\bar{W} - \hat{\mathrm{W}}\|_{F,\mathrm{diag}(XX^T)}^2 = \sum_i \sum_j (\bar{W}_{ij} - \hat{\mathrm{W}}_{ij})^2 \|X_j\|_2^2. \tag{1}$$

Here, $X_j \in \mathbb{R}^m$ represents the calibration activations for the jth input channel, and $\mathrm{diag}(XX^T)$ acts as a weighting term that prioritizes important elements of W.

Paradigms of Compression The above formulation unifies the following two paradigms of shape preserving compression.

1. **Quantization (NoWag-VQ):** When using vector quantization, this corresponds to a weighted K-means clustering problem, where the weights are determined by $\mathrm{diag}(XX^T)$. A detailed formulation is provided in Appendix C.
2. **Unstructured/Semi-Structured Pruning (NoWag-P):** For an $x\%$-unstructured pruning pattern, where $x\%$ of the weight matrix entries (i, j) are zeroed out, our method selects

the $x\%$ of entries with the smallest $\bar{W}_{ij}^2 \|X_j\|_2^2$, thereby minimizing Equation 1. For N:M Semi-Structured Pruning, our method selects the N entries in each group of M with the smallest $\bar{W}_{ij}^2 \|X_j\|_2^2$.

The average computational cost of both NoWag-P and NoWag-VQ scales linearly with the size, $d_{\text{in}} d_{\text{out}}$ of the weight matrices.

Why this works. The critical step is our approach is the normalization of W, which rescales the outliers rows and columns with large magnitudes. Without normalization, even after scaling by activations, the location of preserved elements of unstructured pruning would be largely localized to rows with large magnitudes, especially in the Multi Head Attention layers. This results in entire output channels effectively being removed, dramatically reducing the performance of the compressed LLM. Furthermore, normalization makes $\bar{W}$ more VQ friendly, by bounding all elements to $[-1, 1]$, and rescaling outliers to make the d dimensional distribution of consecutive weights "ball shaped." A visualization is provided in figure 2 in appendix D.

3 Related Works

Pruning: A popular pruning algorithm for LLMs is Wanda (Sun et al., 2024), which prunes based on a score metric $S_{ij} = |W_{ij}| \|X_j\|_2$, furthermore, for unstructured pruning, pruning is performed independently in per-output groups. Several parallels can be drawn to our approach. First, without normalization NoWag-P is equivalent to Wanda without output grouped pruning. Second, normalization performs a similar process to output grouped pruning by distributing preserved entries more evenly, pruned masks are emperically very similar, please see Appendix D for a more detailed discussion.

Quantization: Kmeans has been explored for LLM PTQ in several works. In many VQ algorithms, it is used to initialize before optimizing the quantization (van Baalen et al., 2024; Liu et al., 2024; Egiazarian et al., 2024). For scalar quantization, SqueezeLLM has employed weighted K-means using the diagonal of the fisher information as weights. Our algorithm has several key differences to those aforementioned. First, we use K-means *only* without any computationally expensive optimization procedures required by previous VQ algorithms. Second, our weights are simply the second moment of the sample activations, which can be calculated without a backwards pass.

Weight Update Compression Methods For both Pruning and Quantization, many compression methods use linear feedback updates during compression (Chee et al., 2023; Tseng et al., 2024a; Liu et al., 2024; van Baalen et al., 2024; Frantar & Alistarh, 2023). This method requires calculating the inverse of a sample activation's outer product, which costs $\mathcal{O}(d_{\text{in}}^3)$. Since d_{in} and d_{out} are of roughly the same magnitude in a modern LLM, NoWag-P and NoWag-VQ offers a significant speedup for compression over linear feedback based pruning methods, whose computational complexity scales cubically with d_{in}.

4 Experiments

Models and Evaluations. We evaluate NoWag on two popular families of models Llama 2 (Llama-2 7B/13B/70B) (Touvron et al., 2023) as well as Llama-3 8B (Grattafiori et al., 2024) for VQ and Llama-3 8B/70B (Grattafiori et al., 2024) for Pruning.

Baselines We compared our results against the SOTA one-shot VQ algorithm at 2 bits per value, QuIP# (Tseng et al., 2024a). This algorithm incorporates VQ with Hammard incoherence matrices and an E_8 structured codebook. We did not compare against QTIP (Tseng et al., 2024b) as our focus was on VQ rounding methods, and because Trellis coding can be extended to any VQ rounding method. For pruning, we compare NoWag-P against Wanda (Sun et al., 2024). As discussed previously, the key difference between Wanda and NoWag-P is our normalization method. As such, a comparison between NoWag-P and Wanda serves to highlight the impact of our normalization scheme in both a unstructured and semi-structured pruning scheme.

Calibration dataset We use 128 samples at the model's native sequence length (4096 for the Llama 2 family and 8192 for the Llama 3 family) of the RedPajama 1T dataset (Weber et al., 2024) as our

	Bits	Wino (↑)	RTE (↑)	PiQA (↑)	ArcE (↑)	ArcC (↑)	Avg Acc (↑)
FP16 (2-7B)	16	67.3	63.2	78.5	69.3	40.0	63.66
FP16 (2-13B)	16	69.5	61.7	78.8	73.2	45.6	65.76
FP16 (2-70B)	16	77.0	67.9	81.1	77.7	51.1	70.96
FP16 (3-8B)	16	73.5	68.6	79.7	80.1	50.2	70.42
QuIP# (2-7B)	2	61.7	**57.8**	69.6	61.2	29.9	56.04
NoWag-VQ (2-7B)	2.02	**64.4**	54.5	**73.6**	**60.7**	**31.7**	**56.99**
QuIP # (2-13B)	2	63.6	54.5	74.2	**68.7**	36.2	59.44
NoWag-VQ (2-13B)	2.01	**68.1**	**62.5**	**75.9**	67.3	**37.9**	**62.34**
QuIP # (2-70B)	2	74.2	**70.0**	78.8	**77.9**	**48.6**	**69.9**
NoWag-VQ (2-70B)	2.02	**74.5**	69.0	**79.4**	75.4	46.2	68.9
QuIP # (3-8B)	2	63.2	52.7	67.6	57.6	28.2	53.86
NoWag-VQ (3-8B)	2.02	**67.7**	**53.0**	**72.3**	**68.4**	**33.2**	**58.93**

Table 1: Zeroshot sequence classification accuracies (%) across 5 tasks and the average accuracies of Quantized Models without finetuning. [1]

calibration data for both pruning and quantization. This is the same dataset used by QuIP#, which uses 6144 samples, or 48x times more data.

Evaluation To evaluate NoWag-VQ and NoWag-P, we follow standard evaluation metrics for quantized models of measuring the zero shot perplexity on the test splits of the C4 (Dodge et al., 2021) and Wikitext2 (Merity et al., 2016), and task specific sequence classification zero shot accuracy through the Eleuther AI LM Harness (Gao et al., 2024), the exact tasks are listed in the appendix E. Because Wanda did not report C4 perplexities, we modified the code to compute them, furthermore we added support for pruning the Llama-3 family of models. Because of this, several libraries had to be upgraded from what the original Wanda paper used, resulting perplexities in Wikitext2 that are slightly different to those reported in Wanda.

4.1 Quantization Evaluation

For Llama-2 7B/13B and Llama-3 8B, we performed VQ with groups of $d = 6$ elements together, this allows for the codebook to fit inside the L1 cache of an Nvidia A6000 GPU, enabling fast decoding. For Llama-2 70B we performed VQ with groups of $d = 7$ elements, since the relative overhead of the codebook is smaller with the larger model size. While larger, this codebook is still able to fit inside the L1 cache of a Nvidia H100 GPU. In table 1 we compare the Zero Shot accuracies of NoWag-VQ against QuIP# and in table 2 we compare the perplexities of NoWag-VQ against QuIP#, both at ~ 2 bits per value. NoWag-VQ outperforms QuIP# in perplexity for Llama-2 7B/13B and Llama 3-8B, and outperforms QuIP# for almost every zero shot task for all models, while using 48x less calibration data.

4.2 Pruning Evaluation

Table 3, we compare the Wikitext2 Perplexity of NoWag-P with Wanda pruning for 50% unstructured pruning and 4:8 semi-structured pruning and 2:4 semi-structured results. In the interest of space, C4 and zeroshot results are reported in are reported in the Appendix. NoWag-P uniformly produces lower perplexity than Wanda at 50% and 4:8 semi-structured. This empirically demonstrates the benefits of the NoWag normalizer. However at 2:4 semi-structured pruning, NoWag-P only roughly matches the performance of Wanda. We believe that this is due to the more structured pattern, which negates the impact of the normalizer. A detailed analysis is provided in the appendix.

5 Conclusion

In this work, we introduced NoWag, a novel framework unifying pruning and quantization under a common normalization-based approach. Our experimental results demonstrate that NoWag-P improves upon existing pruning techniques in maintaining language modeling accuracy, while

[1] QuIP# accuracies are taken from CALDERA (Saha et al., 2024).

Method	Bits	Wiki2 (↓)	C4 (↓)
fp16 (2-7B)	16	5.12	6.63
fp16 (2-13B)	16	4.57	6.05
fp16 (2-70B)	16	3.12	4.97
fp16 (3-8B)	16	5.54	7.01
QUIP # (2-7B)	2	8.23	10.8
NoWag-VQ (2-7B)	2.02	**7.07**	**9.02**
QuIP # (2-13B)	2	6.06	8.07
NoWag-VQ (2-13B)	2.01	**5.93**	**7.94**
QuIP # (2-70B)	2	4.16	6.01
NoWag-VQ (2-70B)	2.02	**4.15**	**5.94**
QuIP# (3-8B)	2	13.8	15.6
NoWag-VQ (3-8B)	2.02	**10.68**	**11.92**

Table 2: Perplexities for WikiText2 and C4 without finetuning for 2-bit Quantized Llama-2 7B/13B/70B, and Llama-3 8B

		Wikitext2 PPL (↓)				
Method	Sparsity	2-7B	2-13B	2-70B	3-8B	3-70B
Dense	0%	5.12	4.57	3.12	5.54	2.58
Wanda	50%	6.46	5.58	3.97	9.06	5.34
NoWag-P	50%	**6.37**	**5.49**	**3.89**	**8.32**	**4.95**
Wanda	4:8	8.07	6.55	4.49	13.39	6.50
NoWag-P	4:8	**8.04**	**6.47**	**4.45**	**12.66**	**6.24**
Wanda	2:4	11.35	8.36	5.20	**22.42**	8.29
NoWag-P	2:4	**11.14**	**8.28**	**5.17**	24.0	**7.52**

Table 3: Wikitext2 for NoWag-P and Wanda at 50% unstructured, and 4:8 and 2:4 semistructured pruning for Llama-2 7B/13B/70B and Llama-3 8B/70B. Context length was at the model's native context length, 4096 for Llama-2 and 8192 for Llama-3.

NoWag-VQ achieves superior quantization performance using substantially less calibration data. By leveraging a structured normalization strategy, NoWag reduces the sensitivity of compression to outlier weights and enhances the efficiency of both pruning and quantization. These findings suggest that NoWag provides a scalable and adaptable compression paradigm for LLMs, facilitating their deployment in real-world applications with reduced computational costs.

REFERENCES

Arash Ahmadian, Saurabh Dash, Hongyu Chen, Bharat Venkitesh, Zhen Stephen Gou, Phil Blunsom, Ahmet Üstün, and Sara Hooker. Intriguing properties of quantization at scale. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 34278–34294. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6c0ff499edc529c7d8c9f05c7c0ccb82-Paper-Conference.pdf.

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pp. 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 9780898716245.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The Fifth PASCAL Recognizing Textual Entailment Challenge, 2009.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-Bit Quantization of Large Language Models With Guarantees. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022a. Curran Associates Inc. ISBN 9781713871088.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30318–30332. Curran Associates, Inc., 2022b. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c3ba4962c05c49636d4c6206a97e9c8a-Paper-Conference.pdf.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless llm weight compression, 2023. URL https://arxiv.org/abs/2306.03078.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus, 2021. URL https://arxiv.org/abs/2104.08758.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization, 2024. URL https://arxiv.org/abs/2401.06118.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot, 2023. URL https://arxiv.org/abs/2301.00774.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024. URL https://zenodo.org/records/12608602.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew

Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha,

Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks, 2015. URL https://arxiv.org/abs/1506.02626.

B. Hassibi, D.G. Stork, and G.J. Wolff. Optimal brain surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, pp. 293–299 vol.1, 1993. doi: 10.1109/ICNN.1993.298572.

Weiyu Huang, Yuezhou Hu, Guohao Jian, Jun Zhu, and Jianfei Chen. Pruning large language models with semi-structural adaptive sparse training, 2024. URL https://arxiv.org/abs/2407.20584.

Sakaguchi Keisuke, Le Bras Ronan, Bhagavatula Chandra, and Choi Yejin. WinoGrande: An Adversarial Winograd Schema Challenge at Scale, 2019.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization, 2024. URL https://arxiv.org/abs/2306.07629.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Andrey Kuzmin, Markus Nagel, Mart van Baalen, Arash Behboodi, and Tijmen Blankevoort. Pruning vs quantization: Which is better? In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 62414–62427. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/c48bc80aa5d3cbbdd712d1cc107b8319-Paper-Conference.pdf.

Yann Lecun, John Denker, and Sara Solla. Optimal brain damage. volume 2, pp. 598–605, 01 1989.

Zhu Liao, Victor Quétu, Van-Tam Nguyen, and Enzo Tartaglione. Can unstructured pruning reduce the depth in deep neural networks? In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 1394–1398. IEEE, October 2023. doi: 10.1109/iccvw60793.2023.00151. URL http://dx.doi.org/10.1109/ICCVW60793.2023.00151.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. In *MLSys*, 2024.

Yifei Liu, Jicheng Wen, Yang Wang, Shengyu Ye, Li Lyna Zhang, Ting Cao, Cheng Li, and Mao Yang. Vptq: Extreme low-bit vector post-training quantization for large language models. In *The 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Vladimir Malinovskii, Denis Mazur, Ivan Ilin, Denis Kuznedelev, Konstantin Burlachenko, Kai Yi, Dan Alistarh, and Peter Richtarik. Pv-tuning: Beyond straight-through estimation for extreme llm compression, 2024. URL https://arxiv.org/abs/2405.14852.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization, 2020. URL https://arxiv.org/abs/2004.10568.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

Rajarshi Saha, Naomi Sagan, Varun Srivastava, Andrea J. Goldsmith, and Mert Pilanci. Compressing large language models using low rank and low precision decomposition, 2024. URL https://arxiv.org/abs/2405.18886.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL https://www.sciencedirect.com/science/article/pii/S0925231223011864.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models, 2024. URL https://arxiv.org/abs/2306.11695.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, 2023.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks, 2024a. URL https://arxiv.org/abs/2402.04396.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De Sa. Qtip: Quantization with trellises and incoherence processing, 2024b. URL https://arxiv.org/abs/2406.11235.

Mart van Baalen, Andrey Kuzmin, Markus Nagel, Peter Couperus, Cedric Bastoul, Eric Mahurin, Tijmen Blankevoort, and Paul Whatmough. Gptvq: The blessing of dimensionality for llm quantization, 2024. URL https://arxiv.org/abs/2402.15319.

Jesse Vig. A multiscale visualization of attention in the transformer model, 2019. URL https://arxiv.org/abs/1906.05714.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*, 2024.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers, 2022. URL https://arxiv.org/abs/2206.01861.

A Additional Pruning Evaluations

Method	Sparsity	C4 PPL (↓) 2-7B	2-13B	2-70B	3-8B	3-70B
Dense	0%	6.63	6.05	4.97	7.01	5.78
Wanda	50%	8.39	7.47	5.77	10.19	7.00
NoWag-P	50%	**8.27**	**7.35**	**5.71**	**9.67**	**6.81**
Wanda	4:8	10.19	8.68	6.39	13.95	7.95
NoWag-P	4:8	**10.17**	**8.67**	**6.38**	**13.86**	**7.69**
Wanda	2:4	**13.80**	**10.96**	7.19	**21.63**	9.63
NoWag-P	2:4	13.91	11.05	**7.23**	23.5	**9.18**

Table 4: C4 Perplexities

Method	Sparsity	Avg Zero Shot Accuracy (↑) 2-7B	2-13B	2-70B	3-8B	3-70B
Dense	0%	63.66	65.76	70.96	70.42	75.89
Wanda	50%	60.24	**63.66**	70.16	**63.49**	**73.45**
NoWag-P	50%	**60.48**	63.57	**70.28**	62.93	72.3
Wanda	4:8	**58.27**	**61.32**	**68.7**	**57.84**	70.42
NoWag-P	4:8	56.71	60.6	68.43	57.46	**70.76**
Wanda	2:4	**55.37**	**58.24**	66.73	**52.59**	**68.08**
NoWag-P	2:4	54.3	58.14	**66.95**	51.21	67.71

Table 5: Average Zeroshot Accuracies

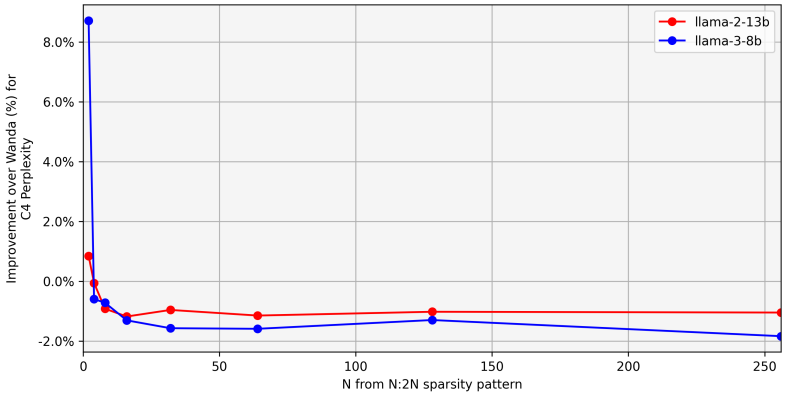

Figure 1: Relative difference in C4 perplexity NoWag-P between Wanda: $(\text{NoWag Perplexity})/(\text{Wanda Perplexity}) - 1$. Calculated for a range of semi structured patterns for Llama-2-13B and Llama-3-8B. The improvements provided by NoWag-P over Wanda dimishes for more structured patterns (smaller N)

A.1 Pruning Zeroshot Results Taskwise

	Sparsity	Wino (↑)	RTE (↑)	PiQA (↑)	ArcE (↑)	ArcC (↑)	Avg Acc (↑)
FP16 (2-7B)	0%	67.3	63.2	78.5	69.3	40.0	63.66
FP16 (2-13B)	0%	69.5	61.7	78.8	73.2	45.6	65.76
FP16 (2-70B)	0%	77.0	67.9	81.1	77.7	51.1	70.96
FP16 (3-8B)	0%	73.5	68.6	79.7	80.1	50.2	70.42
FP16 (3-70B)	0%	80.7	69.0	82.5	86.8	60.4	75.89
Wanda (2-7B)	50%	**66.9**	55.6	75.6	**66.2**	37.0	60.24
NoWag-P (2-7B)	50%	65.7	**60.7**	75.7	65.5	34.9	**60.48**
Wanda (2-13B)	50%	68.9	58.5	**78.4**	**71.6**	**41.0**	63.66
NoWag-P (2-13B)	50%	68.9	59.6	77.8	71.3	**41.0**	63.57
Wanda (2-70B)	50%	**76.9**	69.3	80.5	**75.9**	**48.2**	70.16
NoWag-P (2-70B)	50%	76.6	**71.1**	**80.7**	75.5	47.4	**70.28**
Wanda (3-8B)	50%	**71.0**	**59.9**	74.9	71.4	40.3	**63.49**
NoWag-P (3-8B)	50%	70.0	56.7	**75.8**	**71.7**	**40.4**	62.93
Wanda (3-70B)	50%	**78.0**	**70.0**	**81.3**	**83.0**	**55.0**	**73.5**
NoWag-P (3-70B)	50%	76.7	67.9	81.2	82.8	52.9	72.30

Table 6: Zeroshot accuracies for each task for 50% Pruning

	Sparsity	Wino (↑)	RTE (↑)	PiQA (↑)	ArcE (↑)	ArcC (↑)	Avg Acc (↑)
FP16 (2-7B)	0%	67.3	63.2	78.5	69.3	40.0	63.66
FP16 (2-13B)	0%	69.5	61.7	78.8	73.2	45.6	65.76
FP16 (2-70B)	0%	77.0	67.9	81.1	77.7	51.1	70.96
FP16 (3-8B)	0%	73.5	68.6	79.7	80.1	50.2	70.42
FP16 (3-70B)	0%	80.7	69.0	82.5	86.8	60.4	75.89
Wanda (2-7B)	4:8	**65.35**	**58.12**	**73.61**	**62.46**	**31.83**	**58.27**
NoWag-P (2-7B)	4:8	64.7	54.2	72.7	61.7	30.2	56.71
Wanda (2-13B)	4:8	68.98	55.96	**75.79**	**67.47**	**38.4**	**61.32**
NoWag-P (2-13B)	4:8	**69.0**	**57.0**	75.0	65.5	36.5	60.60
Wanda (2-70B)	4:8	74.9	**67.87**	**79.54**	**74.71**	**46.5**	**68.7**
NoWag-P (2-70B)	4:8	**75.6**	67.2	79.3	74.0	46.2	68.43
Wanda (3-8B)	4:8	**66.69**	53.07	**71.0**	**64.31**	**34.13**	**57.84**
NoWag-P (3-8B)	4:8	65.0	**54.5**	70.7	63.6	33.4	57.46
Wanda (3-70B)	4:8	73.8	**66.06**	**80.09**	80.89	**51.28**	70.42
NoWag-P (3-70B)	4:8	**76.1**	65.7	79.7	**81.4**	50.9	**70.76**

Table 7: Zeroshot accuracies for each task for 4:8 Pruning

	Sparsity	Wino (↑)	RTE (↑)	PiQA (↑)	ArcE (↑)	ArcC (↑)	Avg Acc (↑)
FP16 (2-7B)	0%	67.3	63.2	78.5	69.3	40.0	63.66
FP16 (2-13B)	0%	69.5	61.7	78.8	73.2	45.6	65.76
FP16 (2-70B)	0%	77.0	67.9	81.1	77.7	51.1	70.96
FP16 (3-8B)	0%	73.5	68.6	79.7	80.1	50.2	70.42
FP16 (3-70B)	0%	80.7	69.0	82.5	86.8	60.4	75.89
Wanda (2-7B)	2:4	**60.5**	**58.5**	**70.1**	**57.6**	**30.2**	**55.37**
NoWag-P (2-7B)	2:4	**60.5**	58.1	69.3	55.8	27.9	54.30
Wanda (2-13B)	2:4	**65.8**	54.5	**73.1**	**63.8**	34.0	**58.2**
NoWag-P (2-13B)	2:4	65.6	**58.1**	72.4	62.9	32.7	58.14
Wanda (2-70B)	2:4	73.6	64.6	**78.9**	**72.9**	**43.2**	66.70
NoWag-P (2-70B)	2:4	**75.1**	**66.8**	77.6	72.5	42.7	**66.95**
Wanda (3-8B)	2:4	**59.9**	**52.7**	**67.5**	**56.9**	**25.9**	**52.59**
NoWag-P (3-8B)	2:4	58.1	**52.7**	66.6	54.4	24.2	51.21
Wanda (3-70B)	2:4	71.7	**63.9**	78.1	**78.5**	**48.2**	**68.08**
NoWag-P (3-70B)	2:4	**72.9**	62.5	**78.5**	77.8	46.9	67.71

Table 8: Zeroshot accuracies for each task for 2:4 Pruning

B QUANTIZED FINETUNING

Method	Bits	Wiki2 (↓)	C4 (↓)
fp16 (2-7B)	16	5.12	6.63
fp16 (2-13B)	16	4.57	6.05
fp16 (2-70B)	16	3.12	4.97
AQLM (2-7B)	2.02	6.59	8.54
NoWag-VQ (2-7B)	2.02	**6.51**	**8.50**
AQLM (2-13B)	1.97	5.60	7.49
NoWag-VQ (2-13B)	2.01	**5.53**	**7.39**
AQLM (2-70B)	2.07	**3.94**	**5.72**
NoWag-VQ (2-70B)	2.02	3.99	5.77

Table 9: Perplexities for WikiText2 and C4 with blockwise finetuning for 2-bit Quantized Llama-2 7B/13B/70B compared with AQLM.

We also examine the performance of NoWag-VQ beyond the "one-shot" compression regime. Existing literature has proposed several methods for post quantization finetuning. One popular method is finetuning the remaining continuous parameters of each transformer block to minimize the block output errors (Egiazarian et al., 2024). Another is model-wise finetuning to minimize the overall Kullback–Leibler divergence with the original model, optimizing over the continuous (Tseng et al., 2024a), and the discrete parameters (Malinovskii et al., 2024). Because of our limited computational resources, we were only able to perform block-wise finetuning. We compare the perplexities of those models against those of AQLM (Egiazarian et al., 2024) in table 9. NoWag-VQ outperforms AQLM for Llama 2 7B and 13B, but falls short for Llama-2 70B. We suspect this is due to AQLM using $d = 8$ VQ rather than $d = 7$, which allows for more than 4x the parameters. However, our codebook fits into the L1 cache of an H100 and AQLM's does not.

Finetuning was performed in a blockwise fashion. For each transformer block, our objective was minimizing the l2 norm between the outputs of the original block and those of the quantized blocks. The parameters to optimize over were the codebooks, and the normalization vectors of each quantized layers, and the RMS norm parameters. In addition we initalized a bias vector, set to all zeros, for each linear layer in the block.

For Llama-2 7B/13B, finetuning was done using 128 samples of Red Pajamas (Weber et al., 2024), with 32 held out as a validation set. Optimization was done through Adam Kingma & Ba (2017),

without any weight decay and a learning rate of 10^{-4}. For Llama-2 70B, 256 samples of Red Pajamas was used with a learning rate 10^{-6}.

C Detailed Formulation of Quantization

To quantize a model, we map each weight entry or a contiguous vector of weight entries to a codebook. Without loss of generality, we assume that use a vector quantization algorithm where we quantize every d parameters, $w_{i,j:j+d}$ together. Then quantization results in the following:

- A **codebook** $\mathcal{C} = \{c_1 ... c_k | c_k \in \mathbb{R}^d\}$
- A **mapping** $\mathcal{M}(w_{i,j:j+d}) = c_l \in \mathcal{C}$.

when we represent the quantized weights for inference, these mappings become a string of bits of length $\lceil |\mathcal{C}| \rceil$. Therefore the resulting in a bits per value of $\left(\log_2 \left(\lceil |\mathcal{C}| \rceil\right) + \epsilon\right) \frac{1}{d}$, where ϵ is the bits needed to represent the overhead of normalization parameters, codebooks, etc. Note that we can inverse this relationship to find that if we have a target bits per values $\sim n_{bpv}$, the size of the codebook should be $|\mathcal{C}| = 2^{n_{bpv} d}$. The main benefit of vector quantization is that it allows for the quantization codebook to be better shaped to the weights. However, the size of the codebook increases exponentially with the dimension d, which leads to ϵ/d no longer becoming a negligible quantity compared with the bits used to encode each value. Furthermore, for fast inference, $\mathcal{C}$ must fit inside the L1 cache of a GPU. This has lead to a line of work on more efficient encoding schemes, such as trellis encoding schemes pioneered by (Tseng et al., 2024b). We focus on only "vanilla" VQ, as the general construction of QTIP can be used as a drop-in replacement for VQ in any rounding framework. (Tseng et al., 2024b)

C.1 Weighted Vector K-Means Formulation

As discussed in section 2, for quantization we used weighted vector K-Means, this consists of two steps, an assignment step and an update step, below we explicitly write each step. However, in the interests of runtime, for each layer, we only initialized once using the K-means++ algorithm (Arthur & Vassilvitskii, 2007), and only performed 100 assignment update step pairs. We did observe increasing performance scaling from 20 to 100 assignment steps, therefore we believe that the results reported in tables 2 1 do not demonstrate the full performance of NoWag-VQ.

Assignment step: For each vector $\bar{W}_{i,j:j+d}$ we select the mappings to such that the weighted l2 norm is minimized.

$$\mathcal{M}(\bar{W}_{i,j:j+d}) = \arg\min_{c_l \in \mathcal{C}} \left(\bar{W}_{i,j:j+d} - c_l\right)^T \left(\text{diag}(XX^T)_{j:j+d} \odot \left(\bar{W}_{i,j:j+d} - c_l\right)\right) \tag{2}$$

Update step: For each vector in the codebook, we take the weighted average of the assignments:

$$c_l = \left(\sum_{i,j \forall \mathcal{M}(\bar{W}_{i,j:j+d}) = c_l} \text{diag}(XX^T)_{j:j+d} \odot \bar{W}_{i,j:j+d}\right) \oslash \left(\sum_{i,j \forall \mathcal{M}(\bar{W}_{i,j:j+d}) = c_l} \text{diag}(XX^T)_{j:j+d}\right) \tag{3}$$

D WHY NORMALIZATION WORKS

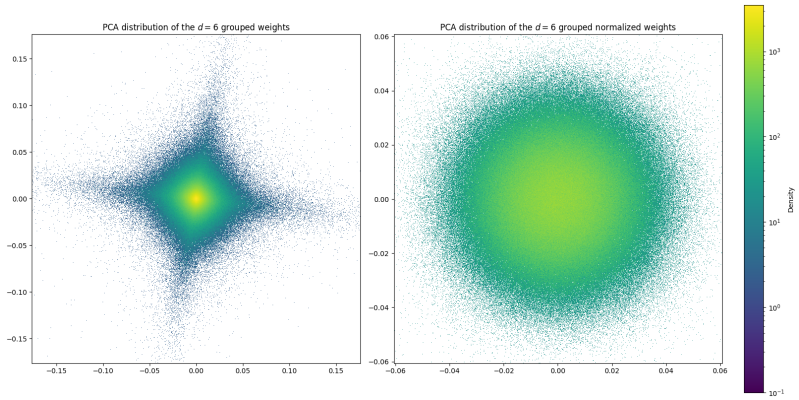

Figure 2: 2d PCA visualization of the distribution of $d = 6$ grouped entries from W and $\bar{\mathrm{W}}$. Densities are plotted at log scale. Normalization reshapes the distribution into a more "ball-shaped distribution.

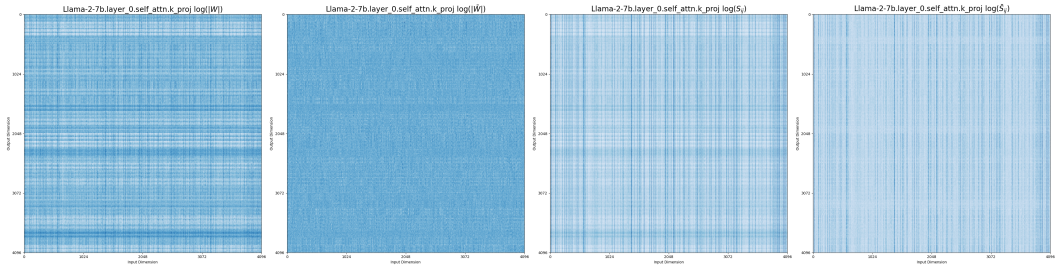

Figure 3: A sample weight from the first attention layer of Llama-2-7B. From left to right: visualization of the absolute values of the weights, normalized weights, importance scores, and normalized importance scores all down-sampled to 1:4 scale by max pooling. Individual elements are visualized in log scale, with blue implying larger value.

The critical step in our approach is the normalization of W. Our normalization method effectively normalizes W_{ij} by both the input and output group. This removes the biases on the compression algorithm to focus on smaller-magnitude rows/columns, leading to a better retention of the overall performance of an LLM.

To illustrate why, we visualize W and $\bar{\mathrm{W}}$ for an example weight in figure 3. In addition, to understand the effects of data awareness, we also visualized the element wise importance scores $\bar{S}_{ij} = \| \bar{\mathrm{W}}_{ij} \| \|X_j\|_2$ derived from equation 1. Likewise, for comparison, we also visualized the naive scores $S_{ij} = \| \mathrm{W}_{ij} \| \|X_j\|_2$ without considering normalization.

We observe that non-normalized weights exhibit a structured pattern, with specific outlier rows and columns, with larger magnitudes. These structures can be attributed to several phenomenons, such as sensitive attention heads, rotatory embedding pattern, and outlier features (Dettmers et al., 2023; Su et al., 2024; Dettmers et al., 2022a; Vig, 2019; Olsson et al., 2022). In comparison, the normalized weights do not exhibit this patterns. This is highly beneficial for vector quantization, as it projects the d dimensional distribution of consecutive weights into a bounded $[0, 1]$ "ball shaped" distribution, visualized in Figure 2

The importance visualizations in Figure 3 once again exhibits these row and column wise structures. Thus, when pruning is applied, the removed elements will be concentrated away from these rows and columns on the non-outlier columns. In turn, this effectively removes entire input/output channels,

reducing the performance of the compressed LLM. Normalization largely removes the rowise outlier structures from the importance scores. In addition some of the columnwise structure is removed, while some still remains. The remaining structure is due to the $\|X_j\|_2$ component of the scores $\bar{S}_{ij} = \| \bar{\mathrm{W}}_{ij} \| \|X_j\|_2$.

E ZERO SHOT DISCRIPTIONS

NoWag-P was evaluated on zero-shot accuracy as noted Tables 8, 7, and 6. The classification sequence classification tasks are as follows:

1. **RTE** (Bentivogli et al., 2009) — Recognizing Text Entailment, a task in the GLUE benchmark, requires a model to determine if one statement logically follows from another.
2. **WinoGrande** (Keisuke et al., 2019) — WINOGRANDE is a large-scale dataset of 44k problems based on the Winograd Schema Challenge, specifically crafted to minimize biases in training data. It features a two-choice fill-in-the-blank format that requires deep commonsense reasoning.
3. **ARC-e** (Clark et al., 2018) — A subset of the AI2 Reasoning Challenge (ARC), ArcE consists of multiple-choice questions designed to assess grade-school level knowledge and represents the "Easy" portion of the dataset.
4. **ARC-c** (Clark et al., 2018) — The ARC-Challenge subset follows the same format as ARC-Easy but includes only questions that baseline algorithms previously failed to answer correctly.
5. **PIQA** (Bisk et al., 2020) — PIQA is a benchmark dataset for physical commonsense reasoning, having AI answer questions about everyday interactions without direct physical experience.

F QUANTIZATION ADDITIONAL EVALUATIONS

We include additional comparisons with three more VQ algorithms, AQLM (Egiazarian et al., 2024), VPTQ (Liu et al., 2024) and CALDERA (Saha et al., 2024). AQLM (Additive Quantiztation for Language Models) performs quantization through additive multi codebook VQ. VPTQ combindeds are vector quantization extension of GPTQ (Frantar et al., 2023) with residual quantization. CALDERA builds ontop of QuIP# by adding additional quantized low rank matrices, and as a result, requires higher bits per value.

For AQLM and VPTQ, their appendices included zero shot ablation results for perplexity of Wikitext2 and C4. For VPTQ, because a very detailed ablation was chosen, we simply chose the best performing zero-shot quantization. For CALDERA, we chose the compression with the least equivalent bits per value. Perplexities are shown in table 10, and zero shot results with CALDERA are show in table 11. We can see that our algorithm performs competitively to these modern VQ algorithms as well.

Method	Bits	Wiki2 (↓)	C4 (↓)
fp16 (2-7B)	16	5.12	6.63
fp16 (2-13B)	16	4.57	6.05
fp16 (3-8B)	16	5.54	7.01
fp16 (2-70B)	16	3.12	4.97
AQLM (2-7B)	2.02	8.18	10.59
QUIP # (2-7B)	2	8.23	10.8
CALDERA (7B)	2.1	7.37	9.74
NoWag-VQ (2-7B)	2.02	**7.07**	**9.02**
QuIP # (2-13B)	2	6.06	8.07
CALDERA (2-13B)	2.08	6.04	7.98
VPTQ (2-13B)	2.07	6.02	7.96
NoWag-VQ (2-13B)	2.01	**5.93**	**7.94**
QuIP# (3-8B)	2	13.8	15.6
CALDERA	2.1	**10.6**	**11.8**
NoWag-VQ (3-8B)	2.02	10.68	11.92
QuIP# (2-70B)	2	4.16	6.01
CALDERA (2-70B)	2.1	**4.11**	5.95
NoWag-VQ (2-70B)	2.01	4.15	**5.94**

Table 10: Performance comparison of different methods on Wiki2 and C4 datasets.

	Bits	Wino (↑)	RTE (↑)	PiQA (↑)	ArcE (↑)	ArcC (↑)	Avg Acc (↑)
FP16 (2-7B)	16	67.3	63.2	78.5	69.3	40.0	63.66
FP16 (2-13B)	16	69.5	61.7	78.8	73.2	45.6	65.76
FP16 (3-8B)	16	73.5	68.6	79.7	80.1	50.2	70.42
FP16 (2-70B)	16	77.0	67.9	81.1	77.7	51.1	70.96
QuIP# (2-7B)	2	61.7	**57.8**	69.6	61.2	29.9	56.04
Caldera (2-7B)	2.1	63.7	**62.1**	72.3	**60.9**	**31.7**	**58.14**
NoWag-VQ (2-7B)	2.02	**64.4**	54.5	**73.6**	60.7	**31.7**	56.99
QuIP # (2-13B)	2	63.6	54.5	74.2	68.7	36.2	59.44
Caldera (2-13B)	2.08	66.9	61.0	**76.0**	**69.5**	37.2	62.12
NoWag-VQ (2-13B)	2.01	**68.1**	**62.5**	75.9	67.3	**37.9**	**62.34**
QuIP # (3-8B)	2	63.2	52.7	67.6	57.6	28.2	53.86
Caldera (3-8B)	2.1	66.9	**58.5**	71.8	68.2	**34.3**	**59.94**
NoWag-VQ (3-8B)	2.02	**67.7**	53.0	**72.3**	**68.4**	33.2	58.93
QuIP # (2-70B)	2	74.2	**70.0**	78.8	77.9	**48.6**	**69.9**
Caldera (2-70B)	2.1	**75.5**	69.3	**79.8**	**76.9**	47.7	69.84
NoWag-VQ (2-70B)	2.02	74.5	69.0	79.4	75.4	46.2	68.9

Table 11: Zeroshot accuracies (%) across 5 tasks and the average accuracies of Quantized Models without finetuning.