μ-MoE: Test-Time Pruning as Micro-Grained Mixture-of-Experts

Toshiaki Koike-Akino [1] Jing Liu [1] Ye Wang [1]

Abstract

To tackle the huge computational demand of large foundation models, activation-aware compression techniques without retraining have been introduced. However, since these rely on calibration data, domain shift may arise for unseen downstream tasks. With an efficient calibration, activation-aware pruning can be executed for every prompt adaptively, yet achieving reduced complexity at inference. We formulate it as a mixture of micro-experts, called μ-MoE. Several experiments demonstrate that μ-MoE can dynamically adapt to prompt-dependent structured sparsity.

1. Introduction

Large foundation models (Touvron et al., 2023; Achiam et al., 2023; Liu et al., 2023a) have shown excellent performance across a variety of tasks (Wei et al., 2022; Katz et al., 2024; Bubeck et al., 2023). Nonetheless, these models, with billions of parameters, demand significant computational resources (Schwartz et al., 2020). Towards increasing the accessibility of large language models (LLMs), a number of compression methods (Xu & McAuley, 2023; Zhu et al., 2024; Bai et al., 2024a) have been introduced: e.g., partial activation (Jiang et al., 2024; Lin et al., 2024a), pruning (Frantar & Alistarh, 2023; Sun et al., 2023; Bai et al., 2024b; Hassibi et al., 1993), quantization (Frantar et al., 2022; Lin et al., 2024b; Wang et al., 2024), knowledge distillation (Hsieh et al., 2023; DeepSeek-AI et al., 2025; Hwang et al., 2024), and rank reduction (Yuan et al., 2023; Liu et al., 2024; Hwang et al., 2024; Saxena et al., 2024).

Test-time scaling (Chen et al., 2024b; Muennighoff et al., 2025) is a paradigm to improve LLM performance by increasing inference computation. We instead consider extra test-time computing to reduce the total cost of inference on the fly. Specifically, we use a pruning operation that dynamically selects important weights depending on each prompt. We view it as a mixture of micro-experts, namely μ-MoE, where, instead of a few massive experts, we may have a massive number of single-parameter weight multiplier experts. We show that activation-aware pruning makes this concept feasible. The contributions of this paper are summarized below:

- We propose a mixture of micro-experts μ-MoE concept to realize the finest-grained adaptation.
- We adopt low-complexity activation-aware pruning to realize test-time LLM compression as μ-MoE.
- We tackle the domain shift issue caused by offline calibration required for baseline static pruning.
- We demonstrate the benefit of μ-MoE over state-of-the-art methods for several LLM benchmarks.

2. Micro-Grained Mixture of Experts (MoE)

Coarse to Micro-Grained MoE Figure 1 illustrates the MoE framework from coarse-grained to micro-grained scales. A coarse-grained MoE may involve multiple LLM modules that activate depending on provided prompts. Most MoEs use mid- to fine-grained architectures. For example, Mixtral-8x7B (Jiang et al., 2024) has 8 multi-layer perceptrons (MLPs) per layer, but select only 2 of them, realizing a six-fold speedup at inference. Finest-grained experts would be a single-parameter weight multiplier within the linear modules of LLMs. We consider such a mixture of micro-experts, referred to as μ-MoE.

μ-MoE employs test-time adaptation to reduce the total test-time compute, i.e., online dynamic pruning to reduce the number of active weights for inference computation. Besides the computational efficiency, the online dynamic pruning may potentially solve the domain shift issue caused by mismatched calibration data used for activation-aware offline static pruning as illustrated in Figure 2.

Activation-Aware Pruning As an alternative to magnitude-based pruning, activation-aware pruning (Williams & Aletras, 2023) leverages the statistics of the activation features. Let $X \in \mathbb{R}^{d\times T}$ be input activation of embedding dimension d for token length T. The aim is

[1] Mitsubishi Electric Research Laboratories (MERL), 201 Broadway, Cambridge, MA 02139, USA.. Correspondence to: Toshiaki Koike-Akino <koike@merl.com>.

Accepted to the 42^{nd} International Conference on Machine Learning Workshop, Vancouver, Canada. PMLR 267, 2025.

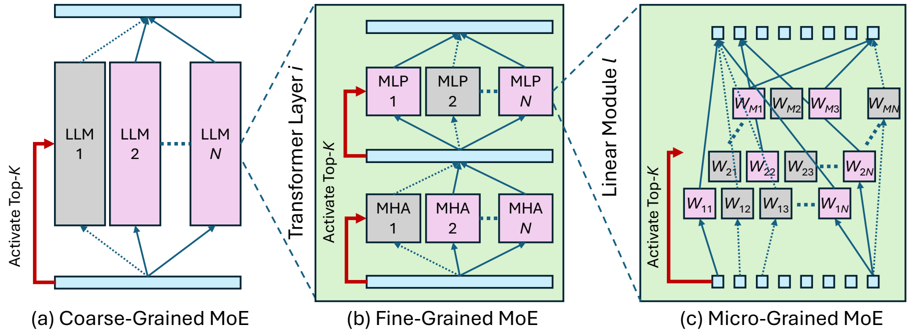


Figure 1. Coarse to micro-grained MoE.

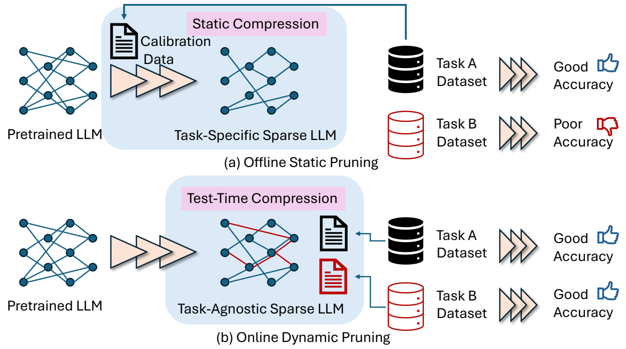


Figure 2. Offline vs online pruning: dynamic pruning finds prompt-dependent sparse structure at test time, preventing domain shift.

to minimize the approximation loss:

$$\mathcal{L} = \mathbb{E}_X\big[\big\|(W - \hat{W})X\big\|^2\big], \qquad (1)$$

where $W \in \mathbb{R}^{d' \times d}$ is a weight matrix and $\hat{W}$ is its pruned version such that only a fraction ρ of the weights are active: $\|\hat{W}\|_0 = \rho \cdot dd'$. SparseGPT (Frantar & Alistarh, 2023) is such an activation-aware LLM pruning inspired by optimal brain surgeon (Hassibi et al., 1993). It uses a scoring metric for pruning:

$$S_{i,j} = |W_{i,j}|^2 / \big[\mathsf{Chol}[(\bar{X}\bar{X}^\top + \lambda I)^{-1}]\big]^2_{j,j}, \qquad (2)$$

where $\mathsf{Chol}[\cdot]$ denotes the Cholesky factorization, $\bar{X} \in \mathbb{R}^{d \times T_\mathrm{c}}$ are T_c tokens of input activation features sampled from calibration data. Here λ is a small damping factor. Although SparseGPT achieves good pruning performance, the computational cost to obtain the sparse matrix is at least of cubic order due to the inverse Hessian calculation: $\mathcal{O}[d^3 + dd'T_\mathrm{c}]$. In addition, it needs extra computations to update the non-zero weights with the Gaussian elimination.

Wanda (Sun et al., 2023) simplifies the metric by approximating with a diagonal correlation: $\bar{X}\bar{X}^\top + \lambda I \simeq \mathsf{diag}[\bar{X}\bar{X}^\top]$. The modified score is written as:

$$S'_{i,j} = |W_{i,j}| \cdot \|\bar{X}_{j,:}\|_2. \qquad (3)$$

It only activates weights within the top-ρ fraction in this metric. The pseudo code in PyTorch is given below:

```
# W:(d',d), X:(d,Tc), kc=int((1-rho) * d)
S = W.abs() * X.norm(p=2, dim=-1)
val, _ = torch.kthvalue(S, dim=-1, k=kc)
W = torch.where(S > val[:,None], W, 0)
```

Wanda requires only quadratic complexity $\mathcal{O}[3dd' + dT_\mathrm{c}]$ including norm calculation, metric product, top-k search, and comparators, yet achieves performance competitive with SparseGPT. It yields semi-structured sparsity with a constant number of active weights per row.

Remark 2.1. While the original Wanda uses torch.sort, this sorting complexity of $\mathcal{O}[d'd\log(d)]$ can be reduced by torch.topk or torch.kthvalue. We note that torch.kthvalue has a linear theoretical complexity. See Appendix B.

Instant Wanda Pruning as μ-MoE To realize μ-MoE, we use test-time tokens X as an online calibration to prune weights, rather than offline calibration tokens $\bar{X}$. When the number of active weights is reduced, the inference complexity of linear modules will be reduced from $\mathcal{O}[dd'T]$ to $\mathcal{O}[\rho dd'T]$. However, it is meaningless if the cost to find the top-ρ weights is higher than the reduced complexity. Hence, SparseGPT is not suitable due to its cubic complexity, while Wanda is a viable candidate. The total complexity with online Wanda pruning is $\mathcal{O}[3dd' + dT + \rho dd'T]$. The complexity ratio (compared to the original $\mathcal{O}[dd'T]$) is on the order of:

$$\frac{3dd' + dT + \rho dd'T}{dd'T} = \rho + \frac{3}{T} + \frac{1}{d'} \simeq \rho, \quad (T, d' \gg 1).$$

This suggests that instant Wanda pruning for every prompt has almost no additional complexity compared to the original full-weight operations for a large enough token length ($T \gg 1$) and embedding dimension ($d' \gg 1$). Moreover, Wanda is known to be robust even with a single calibration sample (Williams & Aletras, 2023), motivating us to use test-time Wanda pruning for our μ-MoE.

Table 1. Perplexity (↓) of OPT models with different pruning methods at 60–40% active weights. Red-highlighted cells indicate that Wanda uses a matched calibration-test dataset. Bold-face letters indicate the best cases.

Active Weights	60%				50%				40%			
Test Dataset	WT2	PTB	C4	Avg	WT2	PTB	C4	Avg	WT2	PTB	C4	Avg
OPT-125M (WT2: 27.7, PTB: 39.0, C4: 26.6, Avg: 31.1)												
Magnitude Prune	43.9	71.9	39.8	51.9	90.9	168.6	71.9	110.4	533.2	906.3	349.9	596.5
Wanda (WT2 Calib)	30.8	44.2	30.3	35.1	37.1	56.3	37.5	43.6	65.4	106.5	37.5	81.6
Wanda (PTB Calib)	32.4	43.8	31.7	36.0	44.0	52.8	42.1	46.3	89.7	**86.5**	87.6	87.9
Wanda (C4 Calib)	30.7	44.4	29.3	34.8	39.1	57.1	34.8	43.7	75.1	104.3	60.4	80.0
μ-MoE	**30.3**	**43.3**	**28.6**	**34.1**	**35.8**	**51.8**	**32.5**	**40.1**	**61.0**	87.5	**52.3**	**66.9**
OPT-1.3B (WT2: 14.6, PTB: 20.3, C4: 16.1, Avg: 17.0)												
Magnitude Prune	150.0	306.1	103.2	186.4	799.5	1438.1	298.5	845.4	6218.7	7303.5	2385.8	5302.7
Wanda (WT2 Calib)	16.6	23.6	18.8	19.6	18.9	28.3	22.4	23.2	25.6	43.2	34.1	34.3
Wanda (PTB Calib)	16.4	22.5	19.6	19.5	20.4	25.3	25.9	23.9	34.6	34.0	48.6	39.1
Wanda (C4 Calib)	**16.0**	23.4	18.0	19.1	18.9	27.7	20.8	22.5	27.5	43.4	28.7	33.2
μ-MoE	16.4	**22.3**	**17.6**	**18.8**	**18.0**	**24.9**	**19.1**	**20.7**	**23.1**	**33.2**	**23.9**	**26.7**
OPT-2.7B (WT2: 12.5, PTB: 18.0, C4: 14.3, Avg: 14.9)												
Magnitude Prune	21.8	33.9	19.8	25.2	119.8	172.0	57.3	116.4	4282.4	4667.9	2263.1	3737.8
Wanda (WT2 Calib)	13.0	19.6	16.0	16.2	14.0	22.5	18.6	18.4	**18.4**	34.0	27.2	26.6
Wanda (PTB Calib)	13.2	18.8	16.6	16.2	15.4	20.4	19.7	18.5	26.1	**27.2**	34.3	29.2
Wanda (C4 Calib)	**12.9**	19.1	15.1	15.7	14.5	21.9	16.6	17.7	20.3	33.9	22.2	25.5
μ-MoE	13.1	**18.6**	**14.8**	**15.5**	**13.8**	**19.9**	**15.6**	**16.4**	18.5	31.6	**20.9**	**23.6**
OPT-6.7B (WT2: 10.9, PTB: 15.8, C4: 12.7, Avg: 13.1)												
Magnitude Prune	16.3	23.9	17.0	19.1	532.2	281.6	257.4	357.1	9490.4	6743.4	6169.1	7467.6
Wanda (WT2 Calib)	11.0	17.2	14.2	14.2	12.0	19.0	16.3	15.8	15.1	25.0	22.8	21.0
Wanda (PTB Calib)	11.2	16.3	14.6	14.0	13.6	17.1	17.6	16.1	19.4	20.6	25.8	21.9
Wanda (C4 Calib)	**10.9**	16.4	13.3	13.5	11.9	17.9	14.3	14.7	15.3	23.6	18.2	19.0
μ-MoE	11.1	**16.1**	**13.0**	**13.4**	**11.7**	**16.7**	**13.5**	**14.0**	**13.7**	**19.7**	**15.7**	**16.4**
OPT-13B (WT2: 10.1, PTB: 14.5, C4: 12.1, Avg: 12.2)												
Magnitude Prune	59.8	78.4	44.5	60.9	2960.9	5406.3	3432.5	3933.2	112900.6	28381.4	13734.1	51672.0
Wanda (WT2 Calib)	10.7	15.8	13.6	13.3	12.0	18.7	15.7	15.5	15.5	25.3	20.7	20.5
Wanda (PTB Calib)	10.9	15.2	14.2	13.4	13.4	16.8	17.4	15.9	20.6	20.5	24.6	21.9
Wanda (C4 Calib)	10.9	15.2	14.2	13.4	13.4	16.8	17.4	15.9	20.6	20.5	24.6	21.9
μ-MoE	**10.6**	**15.0**	**12.3**	**12.7**	**11.5**	**16.4**	**12.9**	**13.6**	**14.3**	**20.2**	**14.6**	**16.4**

3. Expriments

Experiments Setup We conduct experiments for LLM benchmarks to evaluate the effectiveness of our method. Our experiments are based on the same setting of SparseLLM (Bai et al., 2024b) and their code base[1]. Following existing work (Sun et al., 2023), we compress all linear layers in LLM transformers to the target compression ratio.

For LLM experiments, we first consider the OPT model family (Zhang et al., 2022) as it provides a wide range of model scales from 125M to 175B. We measure perplexity score for three popular benchmarks: raw-WikiText2 (WT2) (Merity et al., 2016); the Penn Treebank (PTB) (Marcus et al., 1994); and C4 (Raffel et al., 2020).

We also analyze visual tasks for the LLaVA-7B model (Liu et al., 2023a), which consists of a language transformer based on Vicuna and a vision transformer tower. We use the official code base[2] to evaluate the capability of the multi-modal answer reasoning for two benchmarks: ScienceQA (Lu et al., 2022); and TextVQA (Singh et al., 2019). ScienceQA contains 21K vision-language multi-choice questions for three subjects: natural, social, and language science. Some fractions of questions have image and/or text contexts, and the problem levels range from grades 1 to 12. TextVQA makes LLMs to read and reason about text in images to answer visual reasoning questions for 28K images.

Impact of Model Size We first look into the impact of LLM model sizes in Table 1, where perplexity of OPT models at active weight ratios of 60–40% are listed over 125M through 13B scales. The perplexity results of the original full-parameter LLM models are reported next to the names

[1] https://github.com/BaiTheBest/SparseLLM

[2] https://github.com/haotian-liu/LLaVA

Table 2. Accuracy in percent (↑) on ScienceQA dataset of LLaVA-7B model with different compression methods for 40%–60% active weights. Question subjects: natural science (NAT); social science (SOC); language science (LAN). Context modality: text (TXT); image (IMG); or no context (NO). Grades: 1–6 (G1-6); 7–12 (G7-12). Wanda and SpargeGPT use TextVQA for calibration.

		Subject			Context Modality			Grades		
Method	Active Weights	NAT	SOC	LAN	TXT	IMG	NO	G1-6	G7-12	Avg
Original full	100%	72.47	69.18	65.73	73.51	68.82	65.99	72.72	65.19	70.03
Magnitude Prune	60%	65.80	63.33	55.82	66.96	64.15	56.03	66.34	56.16	62.70
SparseGPT	60%	67.05	**65.47**	55.91	68.52	**66.73**	57.21	66.12	59.72	63.83
Wanda	60%	67.79	63.10	63.54	67.94	63.31	62.79	67.66	60.71	65.17
μ-MoE	60%	**68.56**	65.35	**65.73**	**69.94**	64.35	**65.09**	**69.35**	**63.22**	**67.15**
Magnitude Prune	50%	40.63	46.01	39.91	37.05	36.94	43.90	45.70	34.15	41.57
SparseGPT	50%	55.02	48.14	53.55	55.43	54.19	52.96	54.92	50.10	53.20
Wanda	50%	59.33	56.81	**56.09**	60.07	56.32	**56.03**	60.54	53.33	57.96
μ-MoE	50%	**63.23**	**59.17**	53.45	**64.37**	**59.69**	54.43	**63.36**	**53.53**	**59.84**
Magnitude Prune	40%	0.31	0.22	0.00	0.24	0.25	0.21	0.18	0.26	0.21
SparseGPT	40%	42.81	27.90	40.36	43.60	37.08	38.40	40.16	37.05	39.05
Wanda	40%	32.99	28.23	34.73	30.16	31.68	35.47	33.22	31.05	32.45
μ-MoE	40%	**45.16**	**35.21**	**37.64**	**44.62**	**37.43**	**40.07**	**43.28**	**37.24**	**41.12**

Table 3. Accuracy in percent (↑) on TextVQA dataset of LLaVA-7B model with different compression methods at 40–60% active weights. Full-weight accuracy is 61.32%. Wanda and SparseGPT use ScienceQA for calibration.

Active Weights	60%	50%	40%
Magnitude Prune	54.12	45.56	24.62
SparseGPT	53.37	47.42	28.27
Wanda	55.80	52.36	39.27
μ-MoE	**57.16**	**54.65**	**46.97**

Table 4. Complexity of OPT-17B models with μ-MoE.

Active Weights	FLOPs	MACs
100%	3.29T	1.64T
80%	3.21T	1.33T
60%	2.55T	999G
40%	1.90T	671G
20%	1.24T	342G

of the models in the table. Magnitude-based pruning is poor compared to offline Wanda pruning. While Wanda is relatively robust over different calibration and test dataset, mismatched calibration often suffers a marginal loss. Online Wanda pruning at μ-MoE is found to be best for most cases across LLM sizes and compression ratios.

Multi-Modal Reasoning Capability We next show the accuracy of the LLaVA-7B model for the ScienceQA multi-modal reasoning benchmark in Table 2. SparseGPT and Wanda use TextVQA as the offline calibration data. It is verified that our μ-MoE can outperform offline pruning methods across diverse reasoning problems over most subjects, contexts, and grades. Similar trends can be seen in another visual reasoning benchmark in Table 3, where accuracy results for TextVQA are listed. Here, Wanda and SparseGPT use ScienceQA as the calibration dataset. In all experiments, μ-MoE achieves better average performance over state-of-the-art baselines, especially for cases with fewer active weights. The results suggest that online dynamic pruning can realize a task-agnostic MoE by adapting to every prompt given at test time.

Computational Complexity We finally show the complexity analysis in Table 4 for the OPT-17B models using our μ-MoE dynamic pruning, based on the `calflops`[3] library. We included counts of floating point operations (FLOPs) and multiply-accumulate operations (MACs) for ℓ_2-norm, top-ρ value search, and comparators for instant Wanda pruning. We use the token length of 128 for the analysis. We found that the runtime complexity, especially in MACs, is almost proportional to the number of active weights.

4. Conclusion

We proposed a test-time pruning to realize mixture of micro-experts: μ-MoE. With the low complexity of Wanda pruning, online dynamic activation of massive single-parameter micro-experts became feasible for every prompt. We demonstrated that μ-MoE outperforms offline static pruning over several LLM benchmarks. Studying the effect of fine-tuning for μ-MoE is an interesting direction for future work.

[3] https://pypi.org/project/calflops/

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Ba, J. and Frey, B. Adaptive dropout for training deep neural networks. *Advances in neural information processing systems*, 26, 2013.

Bai, G., Chai, Z., Ling, C., Wang, S., Lu, J., Zhang, N., Shi, T., Yu, Z., Zhu, M., Zhang, Y., et al. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625*, 2024a.

Bai, G., Li, Y., Ling, C., Kim, K., and Zhao, L. SparseLLM: Towards global pruning for pre-trained language models. *arXiv preprint arXiv:2402.17946*, 2024b.

Bansal, H., Gopalakrishnan, K., Dingliwal, S., Bodapati, S., Kirchhoff, K., and Roth, D. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. *arXiv preprint arXiv:2212.09095*, 2022.

Bershatsky, D., Cherniuk, D., Daulbaev, T., Mikhalev, A., and Oseledets, I. LoTR: Low tensor rank weight adaptation. *arXiv preprint arXiv:2402.01376*, 2024.

Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., and Guttag, J. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146, 2020.

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., et al. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*, 2023.

Buehler, E. L. and Buehler, M. J. X-LoRA: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design. *APL Machine Learning*, 2(2), 2024.

Chen, J., Zhu, Z., Li, C., and Zhao, Y. Self-adaptive network pruning. In *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part I 26*, pp. 175–186. Springer, 2019.

Chen, X., Liu, J., Wang, Y., Wang, P., Brand, M., Wang, G., and Koike-Akino, T. SuperLoRA: Parameter-efficient unified adaptation for large vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8050–8055, 2024a.

Chen, Z., Wang, W., Cao, Y., Liu, Y., Gao, Z., Cui, E., Zhu, J., Ye, S., Tian, H., Liu, Z., et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024b.

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Dong, X., Chen, S., and Pan, S. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *Advances in neural information processing systems*, 30, 2017.

Edalati, A., Tahaei, M., Kobyzev, I., Nia, V. P., Clark, J. J., and Rezagholizadeh, M. KronA: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*, 2022.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Frantar, E. and Alistarh, D. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Gao, X., Zhao, Y., Dudziak, Ł., Mullins, R., and Xu, C.-z. Dynamic channel pruning: Feature boosting and suppression. *arXiv preprint arXiv:1810.05331*, 2018.

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

Hassibi, B., Stork, D., and Wolff, G. Optimal brain surgeon: Extensions and performance comparisons. *Advances in neural information processing systems*, 6, 1993.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. LoRA: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Hua, W., Zhou, Y., De Sa, C. M., Zhang, Z., and Suh, G. E. Channel gating neural networks. *Advances in neural information processing systems*, 32, 2019.

Hwang, I., Park, H., Lee, Y., Yang, J., and Maeng, S. PC-LoRA: Low-rank adaptation for progressive model compression with knowledge distillation. *arXiv preprint arXiv:2406.09117*, 2024.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Katz, D. M., Bommarito, M. J., Gao, S., and Arredondo, P. GPT-4 passes the bar exam. *Philosophical Transactions of the Royal Society A*, 382(2270):20230254, 2024.

Koike-Akino, T., Tonin, F., Wu, Y., Wu, F. Z., Candogan, L. N., and Cevher, V. Quantum-PEFT: Ultra parameter-efficient fine-tuning. *arXiv preprint arXiv:2503.05431*, 2025.

Krajewski, J., Ludziejewski, J., Adamczewski, K., Pióro, M., Krutul, M., Antoniak, S., Ciebiera, K., Król, K., Odrzygóźdź, T., Sankowski, P., et al. Scaling laws for fine-grained mixture of experts. *arXiv preprint arXiv:2402.07871*, 2024.

LeCun, Y., Denker, J., and Solla, S. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Lin, B., Tang, Z., Ye, Y., Cui, J., Zhu, B., Jin, P., Zhang, J., Ning, M., and Yuan, L. MoE-LlaVa: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024a.

Lin, J., Rao, Y., Lu, J., and Zhou, J. Runtime neural pruning. *Advances in neural information processing systems*, 30, 2017.

Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang, W.-C., Xiao, G., Dang, X., Gan, C., and Han, S. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024b.

Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., et al. DeepSeek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning, 2023a.

Liu, J., Koike-Akino, T., Wang, P., Brand, M., Wang, Y., and Parsons, K. LoDA: Low-dimensional adaptation of large language models. In *NeurIPS'23 Workshop on on Efficient Natural Language and Speech Processing*, 2023b.

Liu, L. and Deng, J. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Liu, Z., Wang, J., Dao, T., Zhou, T., Yuan, B., Song, Z., Shrivastava, A., Zhang, C., Tian, Y., Re, C., et al. Deja vu: Contextual sparsity for efficient LLMs at inference time. In *International Conference on Machine Learning*, pp. 22137–22176. PMLR, 2023c.

Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Ma, X., Fang, G., and Wang, X. LLM-Pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21 (140):1–67, 2020.

Saha, R., Sagan, N., Srivastava, V., Goldsmith, A., and Pilanci, M. Compressing large language models using low rank and low precision decomposition. *Advances in Neural Information Processing Systems*, 37:88981–89018, 2024.

Saxena, U., Saha, G., Choudhary, S., and Roy, K. Eigen attention: Attention in low-rank space for KV cache compression. *arXiv preprint arXiv:2408.05646*, 2024.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. Green AI. *Communications of the ACM*, 63(12):54–63, 2020.

Singh, A., Natarjan, V., Shah, M., Jiang, Y., Chen, X., Parikh, D., and Rohrbach, M. Towards VQA models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8317–8326, 2019.

Sun, M., Liu, Z., Bair, A., and Kolter, J. Z. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Voita, E., Ferrando, J., and Nalmpantis, C. Neurons in large language models: Dead, n-gram, positional. *arXiv preprint arXiv:2309.04827*, 2023.

Wang, C., Wang, Z., Xu, X., Tang, Y., Zhou, J., and Lu, J. Q-VLM: Post-training quantization for large vision-language models. *arXiv preprint arXiv:2410.08119*, 2024.

Wang, Y., Agarwal, S., Mukherjee, S., Liu, X., Gao, J., Awadallah, A. H., and Gao, J. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. *arXiv preprint arXiv:2205.12410*, 2022.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Williams, M. and Aletras, N. On the impact of calibration data in post-training quantization and pruning. *arXiv preprint arXiv:2311.09755*, 2023.

Wu, X., Huang, S., and Wei, F. Mixture of LoRA experts. *arXiv preprint arXiv:2404.13628*, 2024.

Xie, Z., Ma, Y., Zheng, X., Chao, F., and Ji, R. Automated fine-grained mixture-of-experts quantization.

Xu, C. and McAuley, J. A survey on model compression and acceleration for pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 10566–10575, 2023.

Yang, Y.-C. and Chen, H.-H. Dynamic DropConnect: Enhancing neural network robustness through adaptive edge dropping strategies. *arXiv preprint arXiv:2502.19948*, 2025.

Yeh, S.-Y., Hsieh, Y.-G., Gao, Z., Yang, B. B., Oh, G., and Gong, Y. Navigating text-to-image customization: From lycoris fine-tuning to model evaluation. In *The Twelfth International Conference on Learning Representations*, 2023.

Yuan, Z., Shang, Y., Song, Y., Wu, Q., Yan, Y., and Sun, G. ASVD: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*, 2023.

Yuan, Z., Shang, Y., Zhou, Y., Dong, Z., Zhou, Z., Xue, C., Wu, B., Li, Z., Gu, Q., Lee, Y. J., et al. LLM inference unveiled: Survey and roofline model insights. *arXiv preprint arXiv:2402.16363*, 2024.

Zhang, J., Zhao, Y., Chen, D., Tian, X., Zheng, H., and Zhu, W. MiLoRA: Efficient mixture of low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2410.18035*, 2024.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhu, X., Li, J., Liu, Y., Ma, C., and Wang, W. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577, 2024.

A. Related Work

Model Compression The field of model compression for LLMs has aimed at mitigating the substantial computation and memory requirements (Zhu et al., 2024; Yuan et al., 2024). Such methods primarily fall into four categories: weight quantization (Lin et al., 2024b; Frantar et al., 2022; Wang et al., 2024), network pruning (LeCun et al., 1989; Hassibi et al., 1993; Frantar & Alistarh, 2023; Bai et al., 2024b), knowledge distillation (Hsieh et al., 2023; DeepSeek-AI et al., 2025; Hwang et al., 2024), and rank reduction (Yuan et al., 2023; Liu et al., 2024; Hwang et al., 2024; Saxena et al., 2024; Saha et al., 2024).

Static Pruning Model pruning (Blalock et al., 2020) methods generate a fixed reduced-parameter network. For example, weight pruning includes magnitude pruning (Han et al., 2015), pruning-aware retraining (Frankle & Carbin, 2018) and activation-aware pruning (Williams & Aletras, 2023). SparseGPT (Frantar & Alistarh, 2023) uses layer-wise optimal brain surgeon (Dong et al., 2017; Hassibi et al., 1993; LeCun et al., 1989), and SparseLLM (Bai et al., 2024b) extends with joint multilayer perceptron (MLP) compression. Wanda (Sun et al., 2023) (as further discussed in Appendix B) greatly simplifies the pruning mechanism, and has been extensively adopted for LLM post-training compression. LLM pruner (Ma et al., 2023) studied task-agnostic structured pruning. (Bansal et al., 2022; Liu et al., 2023c; Voita et al., 2023) have demonstrated the existence of prompt-dependent and task-specific sparsity in LLMs.

Dynamic Pruning Dynamic networks (Lin et al., 2017; Liu & Deng, 2018; Hua et al., 2019; Gao et al., 2018; Chen et al., 2019) selectively execute a subset of modules at inference time based on input samples. Typically, module selections are based on reinforcement learning or gating networks. Adaptive dropout (Ba & Frey, 2013; Yang & Chen, 2025) is regarded as a fine-grained dynamic network.

Mixture of Experts (MoE) MoE (Jiang et al., 2024; Lin et al., 2024a; Liu et al., 2024) dynamically selects a subset of experts from a large pool. Instead of using LLM experts, fine-grained MoE (Krajewski et al., 2024; Xie et al.) uses relatively small experts.. The success and widespread use of parameter-efficient fine-tuning (PEFT) (Hu et al., 2022; Chen et al., 2024a; Edalati et al., 2022; Yeh et al., 2023; Bershatsky et al., 2024; Liu et al., 2023b; Koike-Akino et al., 2025), has enabled mixture of adapters (Wu et al., 2024; Buehler & Buehler, 2024; Wang et al., 2022; Zhang et al., 2024) to become a viable solution for task-agnostic MoE.

B. Wanda: Efficient Activation-Aware Pruning

Sorting Operation Wanda (Sun et al., 2023) is a light-weight yet effective pruning method, suitable for online dynamic pruning. The original algorithm uses a sorting operation to find the top-k weights with high scores $S'_{i,j}$ per row:

```
# W: (d’, d), X: (d, Tc), kc=int((1-rho) * d)
S = W.abs() * X.norm(p=2, dim=-1)     # Score metric
_, idx = torch.sort(S, dim=-1)        # Sorting scores
pruned = idx[..., :kc]                # Select index having the kc smallest scores
W = torch.scatter(W, index=pruned, value=0)  # Zero-out weights
```

Here, $k_\mathrm{c} := (1-\rho)d$ is the complement of $k := \rho d$, which means that k corresponds to the number of active weights (micro-experts) per output neuron, whereas k_c corresponds to the number of inactive weights per output neuron.

Top-k Search Operation Note that selecting the top-k experts does not require a full sort operation. Because the sorting operation is known to have a log-linear complexity order $\mathcal{O}[d'd\log(d)]$, an immediate alternative is to use the top-k search operation instead of sorting as below:

```
# W: (d’, d), X: (d, Tc), kc=int((1-rho) * d)
S = W.abs() * X.norm(p=2, dim=-1)
_, idx = torch.topk(S, dim=-1, k=kc, largest=False, sorted=False) # Select index having
    the kc smallest scores without sorting
W = torch.scatter(W, index=idx, value=0)
```

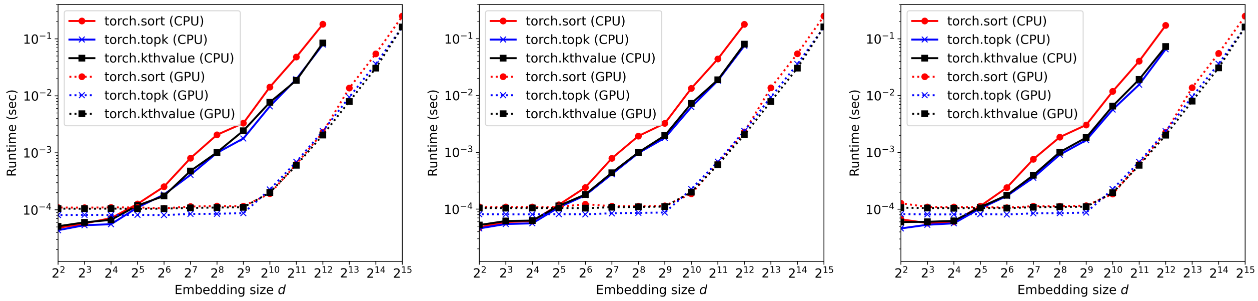
This should have a reduced theoretical complexity of $\mathcal{O}[d'd\log(d_\mathrm{c})]$, using the heap-based method.

The kth Value Search Operation Note that top-k search can be also accomplished based on QuickSelect method searching for the kth largest value. Hence, another option is to find the kth largest value to threshold scores:

```
# W: (d', d), X: (d, Tc), kc=int((1-rho) * d)
S = W.abs() * X.norm(p=2, dim=-1)
val, _ = torch.kthvalue(S, dim=-1, k=kc)    # Find the kc-th smallest score
W = torch.where(S > val[:,None], W, 0)      # Activate weights whose scores are above it
```

Note that torch.kthvalue returns the k_cth smallest value, not the largest value. This should have the lowest theoretical complexity of $\mathcal{O}[d'd]$ on average.

Runtime Analysis The practical complexity highly depends on its implementation on hardware. An empirical experiment is shown in Figure 3, where the average runtime for Wanda pruning over different embedding size d is measured on Apple M1 CPU and NVIDIA A100 GPU. It does not include the computation of linear affine transforms after weight pruning. We see that torch.topk and torch.kthvalue can be moderately faster than torch.sort on the CPU, while there is no significant difference on the GPU. Nevertheless, torch.topk and torch.kthvalue are found to be slightly advantageous for large weights on the GPU. We also observe that the top-k search computation is insensitive to the active weight ratio ρ.



(a) 25% Active Weights (b) 50% Active Weights (c) 75% Active Weights

Figure 3. Wanda pruning complexity based on torch.sort/topk/kthvalue on CPU and GPU at $\rho = 0.25, 0.50, 0.75$.

C. LLM Models

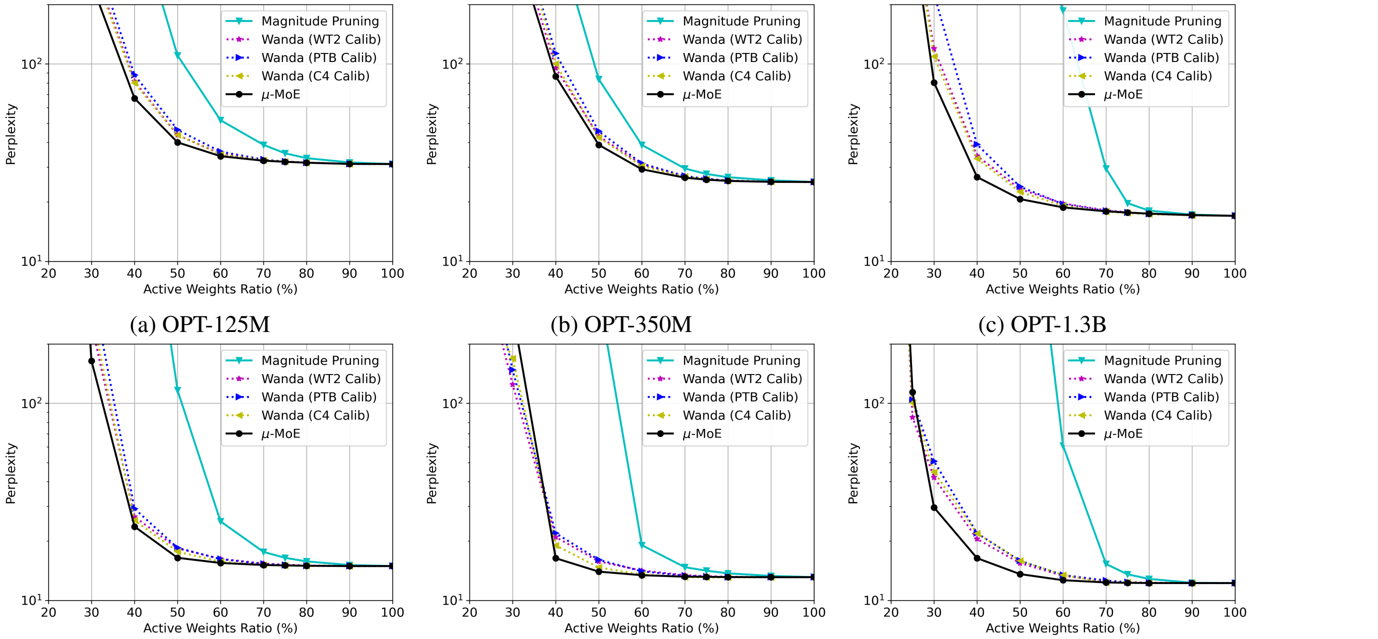
The Open Pre-trained Transformers (OPT) (Zhang et al., 2022) is a suite of decoder-only pre-trained transformers ranging from 125M to 175B parameters. It was claimed that OPT-175B is comparable to GPT-3, while requiring only 1/7th the carbon footprint to develop. Table 5 shows model parameters for the OPT open LLM family.

Table 5. OPT model parameters (Zhang et al., 2022)

Models	# layers L	# heads h	hidden size d	head dim d_h	$d_\mathrm{i} = 4d$	Huggingface ID
125M	12	12	768	64	3072	facebook/opt-125m
350M	24	16	1024	64	4096	facebook/opt-350m
1.3B	24	32	2048	64	8192	facebook/opt-1.3b
2.7B	32	32	2560	80	10240	facebook/opt-2.7b
6.7B	32	32	4096	128	16384	facebook/opt-6.7b
13B	40	40	5120	128	20480	facebook/opt-13b
30B	48	56	7168	128	28672	facebook/opt-30b
66B	64	72	9216	128	36864	facebook/opt-66b
175B	96	96	12288	128	49152	—

D. LLM Experiment Results

Impact of Model Size Figure 4 plots the perplexity results averaged over the WT2, PTB, and C4 datasets for compressed OPT models of 125M, 1.3B, and 13B scales. This partly corresponds to Table 1, while including a wider range of compression ratios. We can see that the magnitude pruning is poor and activation-aware pruning works well. Online pruning with μ-MoE can further improve the perplexity through prompt-wise adaptation, especially around 40%.



(a) OPT-125M (b) OPT-350M (c) OPT-1.3B

(d) OPT-2.7B (e) OPT-6.7B (f) OPT-13B

Figure 4. Perplexity results averaged over WT2, PTB, and C4 datasets for compressed OPT models.

E. Datasets

Wikitext-2 (WT2) The WikiText language modeling dataset (Merity et al., 2016) is a collection of over 100 million tokens extracted from the set of verified good and featured articles on Wikipedia. The dataset is available under the CC BY-SA-4.0 license. The wikitext-2-raw-v1 contains 36,718, 3,760, and 4,358 samples for train, validation, and test splits, respectively. We use https://huggingface.co/datasets/mindchain/wikitext2.

Penn Treebank (PTB) The English Penn Treebank (PTB) corpus (Marcus et al., 1994) is one of the most known and used corpus for the evaluation of models for sequence labeling. The dataset features a million words of 1989 Wall Street Journal material. We use https://huggingface.co/datasets/ptb-text-only/ptb_text_only.

C4 C4 (Raffel et al., 2020) is based on a colossal, cleaned version of Common Crawl's web crawl corpus. This is release under the OCD-By license. We consider a subset "en", containing 364,868,892 and 364,608 samples for train and validation splits, respectively, while we use the first shard for each split in https://huggingface.co/datasets/allenai/c4.

ScienceQA ScienceQA (Lu et al., 2022) is collected from elementary and high school science curricula (i.e., grades 1 through 12), and contains 21,208 multimodal multiple-choice science questions. Out of the questions in ScienceQA, 10,332 (48.7%) have an image context, 10,220 (48.2%) have a text context, and 6,532 (30.8%) have both. Most questions are annotated with grounded lectures (83.9%) and detailed explanations (90.5%). The lecture and explanation provide general external knowledge and specific reasons, respectively, for arriving at the correct answer. ScienceQA has rich domain diversity from three subjects: natural science, language science, and social science. ScienceQA features 26 topics, 127 categories, and 379 skills that cover a wide range of domains. It contains 12,726, 4,241, and 4,241 samples for train, validation, and test splits in https://huggingface.co/datasets/derek-thomas/ScienceQA. This is released under the CC BY-NC-SA 4.0 license.

TextVQA TextVQA (Singh et al., 2019) requires VLM models to read and reason about text in images to answer questions about them. Specifically, models need to incorporate the new modality of text present in the images and reason over it to answer TextVQA questions. TextVQA dataset contains 45,336 questions over 28,408 images from the OpenImages dataset. We use https://huggingface.co/datasets/facebook/textvqa. This is licensed under CC-BY-4.0.