THE SURPRISING EFFECTIVENESS OF RANDOMNESS IN LLM PRUNING

Shuyao Xu[1,2], **Jiayao Liu**[1], **Zhenfeng He**[1], **Cheng Peng**[2] , **Weidi Xu**[2]
[1] National University of Singapore, [2] INFLY TECH (Shanghai) Co., Ltd
shuyao@u.nus.edu

ABSTRACT

This paper investigates the structured pruning of large language models (LLMs). We find that random pruning, despite its simplicity, is a surprisingly effective baseline, particularly at lower pruning ratios. We further propose a simple and efficient method that combines randomness with existing pruning heuristics. Specifically, our method combines random neuron clustering with activation magnitude pruning, exhibiting performance comparable to gradient-based methods while being significantly more efficient (up to 50x faster). Our code is available at https://github.com/Tim-Siu/llm-random-prune.

1 INTRODUCTION AND RELATED WORK

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks (Brown et al., 2020; Kaplan et al., 2020). However, their immense size poses significant challenges for deployment in resource-constrained environments. Model compression techniques (Zhu et al., 2024), particularly structured pruning (Cheng et al., 2024), which aims to remove entire neurons or coupled weights, offer a promising avenue for reducing computational footprint. In this work, we focus specifically on structured pruning of Multi-Layer Perceptron (MLP) layers. MLP layers constitute around 70% of model parameters, and have great potential for compression. Additionally, this narrowed focus facilitates controlled ablation studies of different pruning strategies.

Existing one-shot pruning methods for LLMs typically rely on importance metrics such as weight magnitude (Han et al., 2016), activation magnitude (Muralidharan et al., 2024), or gradients (Ma et al., 2023). While effective, gradient-based methods incur significant computational overhead, requiring around three times the memory compared to methods that require only forward passes.

Prior work has explored randomness in the context of training neural networks from scratch. The Lottery Ticket Hypothesis (Frankle & Carbin, 2019) demonstrated that randomly-initialized networks contain capable subnetworks that can reach full accuracy when trained in isolation. Liu et al. (2022) showed random pruning is effective in sparse training. However, the effectiveness of randomness in pruning already trained neural networks and LLMs in particular receives less attention.

This paper makes three contributions: (1) We demonstrate that random pruning can be surprisingly effective at moderate pruning ratios (e.g., 25%), outperforming other methods in task accuracy. We suggest that this may be due to the preservation of representational diversity. (2) We propose a novel and efficient method combining random clustering with activation-based pruning. This method achieves lower perplexity than gradient-based approaches at a 25% pruning ratio, competitive average accuracy at a 50% pruning ratio, and is significantly faster (up to 50x, see Appendix B.7). (3) We find that similarity-based clustering consistently underperforms random clustering, suggesting the importance of randomness for maintaining representational diversity.

2 PRELIMINARIES: MLP LAYERS IN MODERN LLMS

In recent LLMs, such as Llama and Qwen2 (Touvron et al., 2023; Qwen et al., 2025), the MLP layer typically employs Gated Linear Units (Shazeer, 2020). Given an input vector $x \in \mathbb{R}^d$, the MLP layer is defined as:

$$\text{MLP}(x) = \text{Down}\big(\text{Act}(\text{Gate}(x)) \odot \text{Up}(x)\big),$$

where Gate : $\mathbb{R}^d \to \mathbb{R}^m$, Up : $\mathbb{R}^d \to \mathbb{R}^m$, and Down : $\mathbb{R}^m \to \mathbb{R}^d$ are linear projections, Act denotes a nonlinear activation function (e.g., SiLU), and $\odot$ represents element-wise multiplication.

In this work, we prune individual neurons within MLP layers. In our context, a *neuron* corresponds to a single dimension of the intermediate representation, indexed within the m-dimensional outputs of the Gate and Up projections. When a neuron is pruned, the corresponding row in the Gate and Up matrices, and the matching column in the Down matrix are removed.

For pruning methods that rely on activation information, we define three types of activations for a neuron: *Pre-act:* $(\text{Gate}(x))$, *Post-act:* $(\text{Act}(\text{Gate}(x)))$, and *Modulated-act:* $(\text{Act}(\text{Gate}(x)) \odot \text{Up}(x))$. Unless otherwise specified, we use *Post-act*, as it yielded the best performance in our ablation studies (see C.4). After processing the calibration data (see A.1), the activation representation for a neuron is a vector of length $n \times sl$, where n is the number of samples and sl is the sequence length.

3 Random Pruning: A Surprisingly Strong Baseline

3.1 Methodology

Random pruning removes neurons without considering any importance metric. Given a total of N neurons and a target pruning ratio r, we select a random subset of $N \times (1 - r)$ neurons to retain. This simple approach assigns an equal probability of retention to each neuron.

3.2 Experimental Setup

We compare random pruning with several established methods:

Taylor Pruning (Ma et al., 2023): A gradient-based method that estimates neuron importance using the first-order Taylor expansion of the loss function (See B.1).

Activation Magnitude Pruning (Muralidharan et al., 2024): Prunes neurons with low average activation magnitudes (See B.2).

Weight Magnitude Pruning (Han et al., 2016): Prunes neurons with small weight norms (See B.3).

We evaluate all methods on the Qwen-2.5-7B-Instruct model (Qwen et al., 2025). We prune 25% and 50% of neurons in layers 5-26, following (Ma et al., 2023). We report perplexity on WikiText2 (Merity et al., 2017), zero-shot accuracy on MMLU (Hendrycks et al., 2021), and average accuracy across six tasks: ARC-c, ARC-e, HellaSwag, MMLU, OpenBookQA, and Winogrande (Gao et al., 2023; Chang et al., 2024). Llama-3.1-8B-Instruct results are in Appendix C.3.

3.3 Results and Discussion

Table 1 reveals a surprising result for pruning the Qwen model: at 25% pruning, simple random pruning outperforms other methods on downstream tasks. Random pruning also exhibits lower perplexity than other non-gradient-based techniques (except the one we proposed later in 4) and remains competitive with Taylor pruning. This suggests that, for moderate parameter removal, random pruning can be as effective as more sophisticated importance metrics.

However, this trend reverses at 50% pruning. Random pruning suffers significant performance degradation, with perplexity nearly doubling compared to Taylor pruning. This reveals a shift in what constitutes effective pruning as model capacity becomes severely constrained.

3.4 Hypothesis: Diversity vs. Importance

The effectiveness of random pruning at lower pruning ratios suggests that successful pruning may be more complex than simply preserving individually ”important” neurons. We hypothesize that random pruning maintains representational diversity by giving each neuron an equal chance of retention. At low pruning ratios, this diversity might be more beneficial than solely preserving neurons deemed important by specific metrics. Importance-based methods, conversely, may inadvertently homogenize features, where only neurons with similar characteristics (e.g., large activation magnitudes) are preserved, potentially limiting the model’s representational capacity.

At higher pruning ratios, we speculate a paradigm shift from preserving diversity to safeguarding essential functionality. When removing half of the neurons, the model can no longer afford to retain non-essential components. Random pruning, lacking the ability to distinguish between dispensable and critical neurons, exhibits significant performance degradation in this regime. Our experimental results suggest that activation magnitude serves as a reliable indicator of essential neurons, as evidenced by the relatively strong performance of activation-based methods at 50% pruning.

4 Combining Randomness with Heuristics

Motivated by the complementary strengths speculated in random and activation-based pruning, we propose a novel method called **Random Clustering + Activation L2 Pruning (RC+A)**. This approach is designed to balance two objectives: preserving representational diversity through random clustering and retaining functionally important neurons via activation magnitude.

4.1 Methodology

The RC+A algorithm operates in two main stages:

1. **Diversity Preservation**: All neurons are randomly partitioned into $k = N \times (1 - r)$ clusters [1], where N is the total number of neurons and r is the target pruning ratio. Each cluster will have one neuron in the final pruned network. This random partitioning is intended to enforce some degree of randomness and proportional representation across the parameter space.

2. **Importance Selection**: Within each cluster, we retain only the neuron that exhibits the highest L2 norm of its Post-act vector. This local selection mechanism is intended to reintroduce our heuristics that neurons with higher activation magnitude are more likely to be essential.

4.2 Results and Discussion

Table 1: Results on Qwen-2.5-7B-Instruct. RC+A: Random Clustering + Activation L2.

	25% Pruning			50% Pruning		
Method	Perplexity ↓	MMLU	Average	Perplexity ↓	MMLU	Average
Original	9.589	0.718	0.622	9.589	0.718	0.622
Taylor	15.244	0.607	0.504	**31.839**	**0.476**	0.394
Random Prune	15.382	0.601	**0.529**	71.295	0.442	0.382
RC+A (Ours)	**14.944**	**0.618**	0.524	34.204	0.455	**0.396**
Activation Mean	15.995	0.587	0.495	56.585	0.401	0.362
Activation L2	17.159	0.614	0.511	42.404	0.462	0.391
Weight L2	485.878	0.435	0.427	190214.389	0.251	0.287

As shown in Table 1, RC+A surpasses both pure random pruning and standalone activation-based pruning at both 25% and 50% pruning levels. Notably, at 25% pruning, RC+A achieves **lower perplexity** than Taylor pruning, and at 50% pruning, it maintains **competitive average accuracy**. Further experiments (Appendices C.2 and C.3) demonstrate the robustness of RC+A across different random seeds and its effectiveness on a different model (Llama-3.1-8B-Instruct).

Furthermore, our method is substantially more efficient than gradient-based techniques; it yields up to 50-fold speedup relative to Taylor pruning (see Appendix B.7), though this speedup is hardware and implementation dependent. This efficiency is particularly beneficial for large-scale models where the computational cost (GPU memory requirement) of backpropagation is prohibitive.

[1] We leave the exploration of different cluster counts to future work.

5 Unsuccessful Attempts: Similarity-Based Clustering

Drawing inspiration from recent advances in pruning Mixture-of-Experts (MoE) models (He et al., 2023; Li et al., 2024; Chen et al., 2024) and techniques for merging neurons in convolutional neural networks (Kim et al., 2020), we explored an alternative strategy based on similarity-based clustering. The underlying hypothesis was that neurons with similar activation patterns likely perform redundant functions and could, therefore, be merged or pruned more aggressively.

5.1 Methodology

In this approach, we applied agglomerative clustering (Müllner, 2011) with complete linkage to group neurons based on the cosine distance between their Modulated-act vectors (as defined in Section 2). Following the clustering step, we investigated several strategies:

Activation L2 Pruning: Within each cluster, retain the neuron with the highest L2 norm of its Modulated-act vector. We used Modulated-act for compatibility with the merging algorithms.

Mean Merge: Average the weight matrices of all neurons within a cluster to form a single, merged neuron, in a manner analogous to model averaging (Wortsman et al., 2022).

Dominant Merge: Retain the Gate and Up projections of the neuron with the highest Modulated-act norm while combining the Down projections of all neurons in the cluster using a scaled sum, where each neuron's contribution is weighted by its Modulated-act norm (See B.6 for details).

5.2 Results and Discussion

Table 2 shows the performance of these similarity-based clustering approaches, compared with our RC+A method. For a direct comparison with RC+A, we also evaluated a variant using Post-act for both similarity clustering and activation L2 pruning; these results are presented in Appendix C.5.

Table 2: Results of Similarity Clustering vs. RC+A on Qwen-2.5-7B-Instruct

	25% Pruning			50% Pruning		
Method	Perplexity ↓	MMLU	Average	Perplexity ↓	MMLU	Average
RC+A (Ours)	**14.944**	**0.618**	**0.524**	**34.204**	**0.455**	**0.396**
Similarity Cluster + Activation L2	16.930	0.607	0.516	49.134	0.424	0.395
Similarity Cluster + Mean Merge	91.430	0.497	0.454	15838.9	0.266	0.312
Similarity Cluster + Dominant Merge	20.241	0.562	0.485	72.560	0.346	0.335

Contrary to our expectations, similarity-based clustering followed by activation pruning **performs worse than our random clustering approach (RC+A)**. This surprising result highlights the critical role of randomness, rather than similarity, in effective pruning. We hypothesize that enforcing similarity among neurons reduces representational diversity, which is crucial for maintaining performance. Moreover, the neuron merging strategies we tested within similarity clusters (Mean and Dominant Merge) significantly decreased performance. This contrasts sharply with the success of model-level merging, suggesting that fine-grained neuron merging poses unique challenges.

6 Conclusion

This work highlights the effectiveness of randomness in post-training pruning of LLMs. At moderate pruning ratios (25%) and when pruning certain models (Qwen-2.5-7B-Instruct), simple random pruning can rival or outperform established methods. We posit that maintaining representational diversity may be more important than solely focusing on individual neuron importance. Our proposed RC+A method, combining random clustering with activation-based pruning, achieves performance comparable to gradient-based approaches with significantly greater efficiency. Based on the success of this randomized approach, we speculate that the space of functionally viable LLM subnetworks may be broader than what is typically explored by standard importance-based pruning techniques. Future work could build on search-based approaches (Elsken et al., 2019; Klein et al., 2024) to navigate this space efficiently. We discuss our limitations in D.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3), March 2024. ISSN 2157-6904. doi: 10.1145/3641289. URL https://doi.org/10.1145/3641289.

I-Chun Chen, Hsu-Shen Liu, Wei-Fang Sun, Chen-Hao Chao, Yen-Chang Hsu, and Chun-Yi Lee. Retraining-free merging of sparse mixture-of-experts via hierarchical clustering, 2024. URL https://arxiv.org/abs/2410.08589.

Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10558–10578, 2024. doi: 10.1109/TPAMI.2024.3447085.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: a survey. *J. Mach. Learn. Res.*, 20(1):1997–2017, January 2019. ISSN 1532-4435.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

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, 12 2023. URL https://zenodo.org/records/10256836.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate, 2022.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1510.00149.

Shwai He, Run-Ze Fan, Liang Ding, Li Shen, Tianyi Zhou, and Dacheng Tao. Merging experts into one: Improving computational efficiency of mixture of experts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14685–14691, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.907. URL https://aclanthology.org/2023.emnlp-main.907/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Woojeong Kim, Suhyun Kim, Mincheol Park, and Geunseok Jeon. Neuron merging: Compensating for pruned neurons. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 585–595. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0678ca2eae02d542cc931e81b74de122-Paper.pdf.

Aaron Klein, Jacek Golebiowski, Xingchen Ma, Valerio Perrone, and Cedric Archambeau. Structural pruning of pre-trained language models via neural architecture search. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=XiK8tHDQNX. Survey Certification, Expert Certification.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. Merge, then compress: Demystify efficient SMoe with hints from its routing policy. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=eFWG9Cy3WK.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Li Shen, Decebal Constantin Mocanu, Zhangyang Wang, and Mykola Pechenizkiy. The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=VBZJ_3tz-t.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*, 2023.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation, 2024. URL https://arxiv.org/abs/2407.14679.

Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms, 2011. URL https://arxiv.org/abs/1109.2378.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=F7aAhfitX6.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577, 11 2024. ISSN 2307-387X. doi: 10.1162/tacl_a_00704. URL https://doi.org/10.1162/tacl_a_00704.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

A Experimental Setup

A.1 Calibration Data

Following LLM-Pruner (Ma et al., 2023), our primary calibration data comprises 10 samples of length 128 from BookCorpus (Zhu et al., 2015). For ablation studies (Table 3) and comparison with Minitron (Muralidharan et al., 2024), we also use 1024 samples of length 2048 from C4 (Raffel et al., 2019). Unless otherwise specified, we use BookCorpus in our experiments. We acknowledge the small primary calibration set size, chosen for computational efficiency and comparability with LLM-Pruner.

B Implementation Details

B.1 Taylor Pruning

Taylor pruning estimates the importance of a neuron based on the first-order Taylor expansion of the loss function (Ma et al., 2023). For a given neuron i and its corresponding weight w_i within an MLP layer, the importance score s_i is approximated as:

$$s_i \approx |w_i \cdot \nabla_{w_i} L| \tag{1}$$

where L is the loss function and $\nabla_{w_i} L$ is the gradient of the loss with respect to the weight. We sum the absolute values of the products of the weights and their corresponding gradients across the gate, up, and down projections. The neurons with the lowest scores are pruned. Gradients are computed based on calibration data.

B.2 Activation-based Pruning

Activation-based pruning uses the magnitudes of neuron activations to determine importance. We consider three types of activations within an MLP layer:

- **Pre-act** $a_{pre}(x) = \text{Gate}(x)$
- **Post-act**: $a_{post}(x) = \text{Act}(\text{Gate}(x))$
- **Modulated-act**: $a_{mod}(x) = \text{Act}(\text{Gate}(x)) \odot \text{Up}(x)$

where x is the input to the MLP layer. We consider two types of reduction: mean and L2-norm. For a neuron's activation vector a_i, the importance score s_i is:

- **Mean**: $s_i = \frac{1}{N} \sum_j a_{i,j}$ where N is the number of calibration samples and $a_{i,j}$ is the activation of neuron i at the j-th sample.
- **L2-Norm**: $s_i = \sqrt{\sum_j a_{i,j}^2}$ where $a_{i,j}$ is the activation of neuron i at the j-th sample, aggregated across samples and sequence positions.

In the ablation studies in (Muralidharan et al., 2024), Mean performs better. We find L2-Norm reduction works better in our case and adopt it in our clustering-based algorithms.

B.3 Weight-based Pruning

Weight-based pruning uses the magnitude of the weights to determine neuron importance. It is discussed in (Han et al., 2016) and compared as a baseline in (Ma et al., 2023). We compute the L2 norm of the concatenated weight matrices $(\text{W}_{\text{gate}}|\text{W}_{\text{up}}|\text{W}_{\text{down}})$. Specifically for a given neuron i, the importance score is given by

$$s_i = ||(\text{W}_{\text{gate},i}|\text{W}_{\text{up},i}|\text{W}_{\text{down},i}^T)||_2 \tag{2}$$

Where $W_{gate,i}$, $W_{up,i}$ represent the i-th row of the gate and up weights respectively, and $W_{down,i}$ represents the i-th column of the down projection. The neurons with the lowest scores are pruned.

B.4 Random Pruning

In random pruning, each neuron has an equal chance of being removed. A predefined ratio of neurons are selected uniformly at random. Let N be the total number of neurons and r be the target pruning ratio. Then, we generate a set of $N \times (1 - r)$ random indices uniformly from the range $[0, N - 1]$. These indices represent the neurons to be kept after pruning.

B.5 Random Clustering + Activation L2 Pruning

In this approach, we first randomly cluster the neurons into groups and then prune based on activation magnitudes. Let N be the total number of neurons, k be the number of clusters (or the target number of neurons to retain after pruning) and r be the pruning ratio, then $k = N \times (1 - r)$.

1. We randomly and uniformly partition the N neurons into k clusters, each denoted by C_i.
2. For each cluster C_i, we calculate the Post-act norm activation vector a_j for each neuron j in C_i: $s_j = ||a_j||_2$.
3. We select the neuron with the highest activation norm in each cluster.

All other neurons are pruned.

B.6 Similarity Clustering

In similarity clustering, we cluster neurons based on the cosine similarity of their Modulated-act activation vectors. Let a_i be the Modulated-act activation vector for neuron i and let N be the total number of neurons, and k be the target number of clusters.

1. Compute the activation vector a_i for each neuron $i \in [1, N]$.
2. Define the distance between neurons as the cosine distance between the activation vector of every pair of neurons: $d(a_i, a_j) = 1 - \frac{a_i \cdot a_j}{||a_i||_2 ||a_j||_2}$.
3. Perform agglomerative clustering using the complete linkage method to group neurons into k clusters.

After clustering, we apply one of three different merging strategies for each cluster:

1. **Activation L2 Prune**: Within each cluster, the neuron with the highest L2-norm of its Modulated-act is selected. All other neurons in the cluster are pruned.
2. **Mean Merge**: All weight matrices corresponding to the clustered neurons are averaged, resulting in a merged neuron.
3. **Dominant Merge**: Within each cluster, we retain the Gate and Up projections of the neuron with the highest L2 norm of Modulated-act and perform a scaled sum of the Down projections. Let W_{down} be the down projection and let s_i be the L2 norm of Modulated-act of neuron i within cluster C. Then the weight of the merged neuron is given by

$$\bar{W}_{down} = \sum_{i \in C} \frac{s_i}{s_{dominant}} W_{down,i} \tag{3}$$

where $s_{dominant}$ is the L2 norm of Modulated-act of the neuron in C with the highest Modulated-act norm.

B.7 Computational Efficiency

We compare the runtime of our Random Clustering + Activation L2 pruning method (RC+A) with Taylor pruning on the Qwen-2.5-7B-Instruct model. Experiments were conducted on a single NVIDIA L40 GPU using Hugging Face Transformers (Wolf et al., 2020) (version 4.45.2), Accelerate (Gugger et al., 2022) (version 1.3.0), and gradient checkpointing. Note that for Taylor pruning, we performed back-propagation on the GPU but offloaded the final computation of importance (which involves multiplication of gradient with weight magnitude) to the CPU to accommodate GPU memory constraints. The batch size was automatically set by the Accelerate library to maximize GPU utilization. Our RC+A method completes pruning in approximately 4 seconds for all 22 MLP layers, while Taylor pruning requires ∼215 seconds due to the gradient computation. This represents an approximately 50x speedup while maintaining competitive performance (see Table 1). This significant speedup is primarily attributed to the avoidance of backpropagation in our method, which is a computationally expensive operation. We acknowledge that the exact speedup is highly dependent on hardware and implementation.

C Ablation Studies

C.1 Effect of Calibration Data

Table 3: Effect of calibration data on Qwen-2.5-7B-Instruct (25% pruning).

Method	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande
Activation Mean (BookCorpus)	15.995	0.357	0.654	0.504	0.587	0.266	0.601
Activation Mean (C4)	15.944	0.352	0.657	0.505	0.567	0.244	0.610

We evaluate the effect of different calibration datasets on the performance of activation-based pruning. As shown in Table 3, using a larger and more diverse but arguably lower-quality dataset (C4) improves perplexity but degrades accuracy on knowledge-based tasks for the Qwen-2.5-7B-Instruct model.

C.2 Effect of random seed on RC+A performance

We evaluated Random Cluster + Activation L2 pruning using different random seeds and report the results in Table 4 and 5 for 25% and 50% pruning respectively, using the Qwen-2.5-7B-Instruct model. These results demonstrate the stability of our method across different random seeds.

Table 4: Effect of different random seeds for Random Cluster + Activation L2 Pruning (25% pruning, Qwen-2.5-7B-Instruct).

Method	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande	Average
Original model	9.589	0.526	0.819	0.620	0.718	0.348	0.702	0.622
Seed 42 (reported)	14.944	0.397	0.681	0.510	0.618	0.288	0.650	0.524
Seed 0	14.867	0.391	0.674	0.508	0.623	0.290	0.657	0.524
Seed 1	14.913	0.408	0.686	0.503	0.610	0.292	0.642	0.524
Seed 7	14.948	0.371	0.660	0.501	0.617	0.290	0.658	0.516

Table 5: Effect of different random seeds for Random Cluster + Activation L2 Pruning (50% pruning, Qwen-2.5-7B-Instruct).

Method	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande	Average
Original model	9.589	0.526	0.819	0.620	0.718	0.348	0.702	0.622
Seed 42 (reported)	34.204	0.265	0.488	0.381	0.455	0.22	0.567	0.396
Seed 0	35.790	0.271	0.519	0.382	0.479	0.230	0.573	0.409
Seed 1	34.286	0.271	0.548	0.378	0.454	0.200	0.582	0.406
Seed 7	34.394	0.279	0.540	0.379	0.423	0.224	0.556	0.400

The performance of our proposed method, Random Cluster + Activation L2 pruning, is relatively stable across different random seeds. The average downstream task accuracy varies only slightly (within 0.01 for both 25% and 50% pruning) across different seeds, as shown in Tables 4 and 5.

C.3 Different Model

We also evaluate our method on the Meta-Llama-3.1-8B-Instruct model. The results for 25% and 50% pruning are presented in Tables 6 and 7 respectively.

Table 6: Pruning results on Llama-3.1-8B-Instruct (25% pruning).

Method	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande	Average
Original model	8.641	0.520	0.819	0.591	0.680	0.336	0.743	0.615
Taylor	14.332	0.409	0.718	0.495	0.561	0.268	0.679	0.522
Random Prune	24.224	0.340	0.594	0.443	0.382	0.206	0.632	0.433
RC+A (Ours)	14.126	0.414	0.712	0.502	0.555	0.272	0.706	0.527
Activation L2	13.176	0.388	0.722	0.517	0.556	0.284	0.701	0.528
Activation Mean	22218.949	0.212	0.301	0.298	0.251	0.152	0.515	0.288
Weight L2	5973.610	0.238	0.407	0.283	0.273	0.168	0.541	0.318

Table 7: Pruning results on Llama-3.1-8B-Instruct (50% pruning).

Method	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande	Average
Original model	8.641	0.520	0.819	0.591	0.680	0.336	0.743	0.615
Taylor	33.552	0.277	0.533	0.378	0.359	0.176	0.595	0.386
Random Prune	412660.209	0.212	0.262	0.257	0.248	0.164	0.508	0.275
RC+A (Ours)	30.528	0.307	0.540	0.392	0.414	0.212	0.618	0.414
Activation L2	30.420	0.302	0.551	0.402	0.404	0.218	0.617	0.416
Activation Mean	647082.039	0.220	0.271	0.262	0.242	0.182	0.528	0.284
Weight L2	27588.505	0.225	0.285	0.264	0.229	0.140	0.523	0.278

For Llama-3.1-8B-Instruct, our method outperforms Taylor pruning but is slightly behind pure Activation L2 pruning. We hypothesize that this is due to the prevalence of very large activations in Llama models (Sun et al., 2024), which makes pure activation-based pruning more effective.

C.4 Effects of Activation Types

Table 8: Ablation of activation types and calibration data on Qwen-2.5-7B-Instruct.

Methods	Perplexity	ARC-c	ARC-e	HellaSwag	MMLU	OpenBookQA	Winogrande	Average
Qwen-2.5-7B-Instruct 25%								
Original model	9.589	0.526	0.819	0.620	0.718	0.348	0.702	0.622
Post-act Mean	15.995	0.357	0.654	0.504	0.587	0.266	0.601	0.495
Post-act Mean (c4)	15.944	0.352	0.657	0.505	0.567	0.244	0.610	0.489
Pre-act Mean (c4)	26.323	0.337	0.617	0.494	0.547	0.278	0.603	0.479
Modulated-act Mean (c4)	30.163	0.437	0.696	0.500	0.553	0.296	0.631	0.519
Qwen-2.5-7B-Instruct 50%								
Original model	9.589	0.526	0.819	0.629	0.718	0.348	0.702	0.622
Post-act Mean (BookCorpus)	56.585	0.235	0.460	0.364	0.401	0.172	0.539	0.362
Post-act Mean (c4)	50.750	0.229	0.494	0.376	0.310	0.174	0.551	0.356
Pre-act Mean (c4)	81.373	0.246	0.458	0.375	0.265	0.204	0.542	0.348
Modulated-act Mean (c4)	82.340	0.322	0.549	0.385	0.330	0.210	0.564	0.393

We ablate the choice of activation types used for pruning. As shown in Table 8, Post-act generally performs better than Pre-act and Modulated-act for both 25% and 50% pruning ratios on the Qwen-2.5-7B-Instruct model. This suggests that the activation function plays a crucial role in determining neuron importance.

C.5 Similarity Clustering with Post-act

Table 9: Results of Similarity Clustering using Post-act vs. RC+A on Qwen-2.5-7B-Instruct

	25% Pruning			50% Pruning		
Method	Perplexity ↓	MMLU	Average	Perplexity ↓	MMLU	Average
RC+A (Ours)	**14.944**	**0.618**	**0.524**	**34.204**	**0.455**	**0.396**
Similarity Cluster (Post-act) + Activation L2	16.606	0.588	0.506	46.703	0.412	0.375

Table 9 presents the results of similarity-based clustering when **Post-act** is used for **both** the clustering and the subsequent L2 pruning. This provides a more direct comparison with our RC+A method, which also utilizes Post-act. As the table shows, even with this modification, similarity-based clustering still underperforms RC+A.

D LIMITATIONS

Our study has several limitations:

Limited Focus on One-shot Pruning: We focused on one-shot pruning without considering iterative pruning or subsequent retraining. Pruning is often followed by retraining to recover performance, and the interaction between our random pruning approach and retraining remains unexplored. It's possible that networks pruned with randomness may exhibit different recovery behaviors compared to those pruned using gradient-based methods. However, retraining requires significant computational resources that we have limited access to.

Layer Type Restriction: We only studied MLP layers. While our method could theoretically extend to attention layers (by pruning of attention intermediate neurons or entire attention heads), we did not evaluate this scenario. The effectiveness of random pruning in attention layers may differ significantly given their distinct functional roles.

Model Variability : Our experiments were limited to two decoder-only models (Qwen and Llama). The effectiveness of random pruning may vary across architectures (e.g., encoder-decoder models) or models with substantially different activation distributions (Sun et al., 2024), as evidenced by Llama's stronger performance with activation-based methods compared to random pruning.