HOW CAN REPRESENTATION DIMENSION DOMINATE STRUCTURALLY PRUNED LLMS?

Mingxue Xu, Lisa Alazraki & Danilo P. Mandic
Imperial College London
London SW7 2AZ, United Kingdom
{m.xu21, lisa.alazraki20, d.mandic}@imperial.ac.uk

ABSTRACT

Pruning assumes a subnetwork exists in the original deep neural network (Frankle & Carbin, 2019), which can achieve comparative model performance with less computation than the original. However, it is unclear how the model performance varies with the different subnetwork extractions. In this paper, we choose the representation dimension (or embedding dimension, model dimension, the dimension of the residual stream in the relevant literature) as the entry point to this issue. We investigate the linear transformations in the LLM transformer blocks and consider a specific structured pruning approach, SliceGPT (Ashkboos et al., 2024), to extract the subnetworks of different representation dimensions. We mechanistically analyse the activation flow during the model forward passes, and find the representation dimension dominates the linear transformations, model predictions, and, finally, the model performance. Explicit analytical relations are given to calculate the pruned model performance (perplexity and accuracy) without actual evaluation, and are empirically validated with Llama-3-8B-Instruct and Phi-3-mini-4k-Instruct.

1 INTRODUCTION

Recent progress in pruning for LLMs has proved that freezing or deleting unnecessary LLM model weights can retain similar language task performances with less computations (Frantar & Alistarh, 2023; Men et al., 2025). However, current research has not fully addressed the functionality shifts after pruning, which may cause safety issues (e.g. model collapse (Yang et al., 2024) and unknown backdoor features (Wang et al., 2024)), and makes it difficult to set pruning hyperparameters (e.g., sparsity).

To investigate this functionality shifting issue, we take the LLM as a self-contained system. We analyse the mappings from the input space to the output space, how pruning transforms these mappings, and then transforms the model predictions, as shown in Fig. 1. This analysis compiles well with structured pruning on LLM transformers since ① structured pruning works on structural components, which means it edits the linear representations and linear transformations in LLMs (Park et al., 2024), then rewrites the transmitted signals throughout the network; ② the extracted subnetworks are also the subgraphs in the model computational graph with distinct functionality, so they are human-understandable "circuits" (Wang et al., 2023).

Focusing on the representation dimension in LLMs, our contributions are as follows:

1. We carry out a mechanistic investigation and find that the representation dimension dominates the transformations (linear and non-linear) in the LLM forward passes. This dominance includes global input-output mappings and how the weight matrices of different dimensions interact with the activation flow locally.
2. With the definition of the model performance metrics (i.e. perplexity and multiple-choice accuracy), we give the analytical relations to quantify the pruned model performance with sparsity. We verify our analytical relations empirically with two LLMs, LlaMa-3 and Phi-3, using SliceGPT (Ashkboos et al., 2024) for dimension reduction.

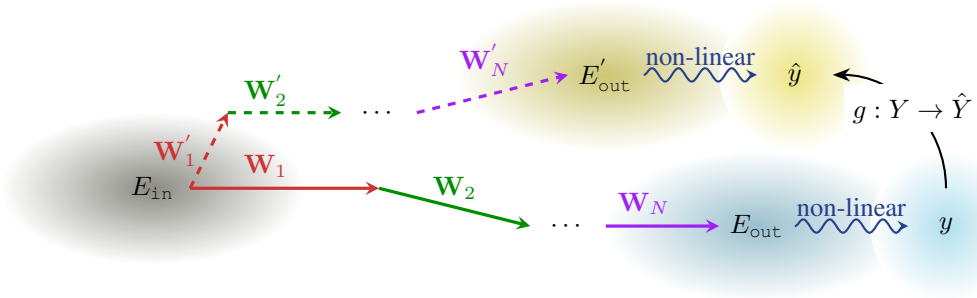


Figure 1: The linear transformations before (solid arrows) and after (dashed arrows) pruning in the model. The original model maps the input embeddings E_{in} from the input space (illustrated with the shading grey ellipse) to the output embedding E_{out} in the output space, through a series of linear transformations (i.e. those defined by the weight matrices $\mathbf{W}_1, \ldots, \mathbf{W}_N$). After pruning, $\mathbf{W}_i$ is converted to $\mathbf{W}'_i$ ($i \in \{1, 2, \ldots, N\}$), and the original output E_{out} is shifted to E'_{out}. The final predictions (y and $\hat{y}$) are generated (normally non-linearly) from E_{out} and E'_{out}. Denote the prediction domain of the pruned model as $\hat{Y}$ and that of the unpruned model Y, the mapping $g: Y \to \hat{Y}$ shifts the model performance.

2 Activation Flow in LLM Transformers

This section clarifies how the linear representations are transmitted throughout the network, so that we can analyse the impacts of dimension reduction on this transmission in Sec. 3. We use the transformer architecture structure in Vaswani et al. (2017), denoting the LLM transformer module as $\mathcal{M}$. $\mathcal{M}$ contains groups of linear and non-linear transformations in multi-head attention implementation, as shown in Fig. 2.

Proposition 1. *The mapping $M: E_{in} \to \bigoplus_{i=1}^{h} A_i$ defined by the transformer $\mathcal{M}$, consists of h groups of the following transformations:*

- **linear transformations** *defined by* $\{\mathbf{W}_{norm}, \mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v, \mathbf{W}_o, \mathbf{W}_{gate}, \mathbf{W}_{up}, \mathbf{W}_{down}\}$;
- **non-linear transformations** *involved in* $\{\texttt{RMSNorm}(\cdot), \texttt{softmax}(\cdot), \sigma(\cdot)\}$.

With the transformations in Propos. 1, we have the following analysis of the activation flow.

Let d denote the representation dimension. We input a text of l tokens to the embedding layer and obtain the encoded texts (embedding matrix) $E_{\text{in}} \in \mathbb{R}^{l \times d}$. E_{in} is then processed by a series of linear and non-linear transformations, as clarified in Appx. A.1.

Suppose there are h parallel attention layers in $\mathcal{M}$, the attention outputs of the ith attention layer is A_i, so the final output of the transformer module $\mathcal{M}$ is

$$\bigoplus_{i=1}^{h} A_i = A_1 \oplus A_2 \oplus \cdots \oplus A_h \in \mathbb{R}^{h \times l \times h_{\text{attn}} h_{\text{dim}}}. \tag{1}$$

Here, $\oplus$ denotes the direct sum, which is equivalent to concatenation in the actual implementation (Xu et al., 2024; Barbero et al., 2024). Since in the default setting, $h_{\text{attn}} h_{\text{dim}} = d$, we have the final hidden states output $\bigoplus_{i=1}^{h} A_i \in \mathbb{R}^{h \times l \times d}$.

We can observe that most of this transmission implemented by the transformer is linear transformations, which are also the investigated objects in the scaling law (Kaplan et al., 2020). **The representation dimension d dominates the linear transformation throughout the whole network**.

3 Functionality Shifting Caused by Dimension Reduction

In this section, we discuss the functionality shifting caused by the structured pruning, i.e., the mapping from the unpruned model prediction domain to that of the pruned model ($g: Y \to \hat{Y}$ in Fig. 1), and then the shift in the model performance. We adopt SliceGPT as the structure pruning approach, since this approach unifiedly reduces the representation dimension in the attention and feedforward

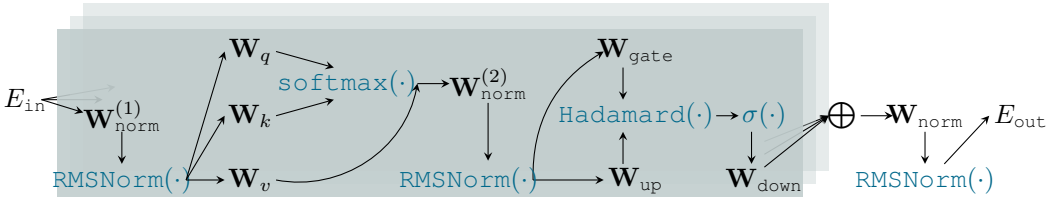


Figure 2: Activation flow in the LLM transformer. Linear transformations are defined by weight matrices like $\mathbf{W}_i$, and non-linear transformations are represented with `teletype font`. The detailed shapes of the weight matrices are clarified in Tab. 2.

layers. We denote sparsity as s ($0 < s < 1$), $(1-s)d \in \mathbb{Z}^+$ the representation dimension after pruning.

3.1 Perplexity

Perplexity, denoted as $\texttt{PPL}(\cdot)$, represents the uncertainty of a discrete probability distribution. Let us start with a single token sequence $X = (x_1, x_2, \ldots, x_l)$. We assign each token in X to an embedding vector $\mathbf{v}_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,d}) \in \mathbb{R}^{1\times d}$, and obtain the stacked embedding vectors (embedding matrix) $E = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_l) \in \mathbb{R}^{l\times d}$. The differential entropy of the stacked embedding vectors is $H(E_{\texttt{in}}) = H(E_{\texttt{out}}) = ld \cdot \kappa(t)$, where $\kappa(t) = -\int p(t)\log(p(t))\ \ dt$. After pruning, the entropy of the output embeddings shifts from $ld \cdot \kappa(t)$ to $(1-s)ld \cdot \kappa(t)$, such that $\frac{H(E'_{\texttt{out}})}{H(E_{\texttt{out}})} = 1 - s$. Given the definition of perplexity correlation is $\texttt{PPL}(E) = 2^{H(E)}$, we have the following theorem about the perplexity before and after pruning:

Proposition 2. *For structured pruning with representation dimension sparsity s, the perplexity of the pruned model, denoted as $\texttt{PPL}(D)$, and the perplexity of the original model, denoted as $\texttt{PPL}_0(D)$, satisfy*

$$\frac{\ln \texttt{PPL}_0(D)}{\ln \texttt{PPL}(D)} = 1 - s, \tag{2}$$

where D is the test dataset from the same distribution as the dataset used for pruning.

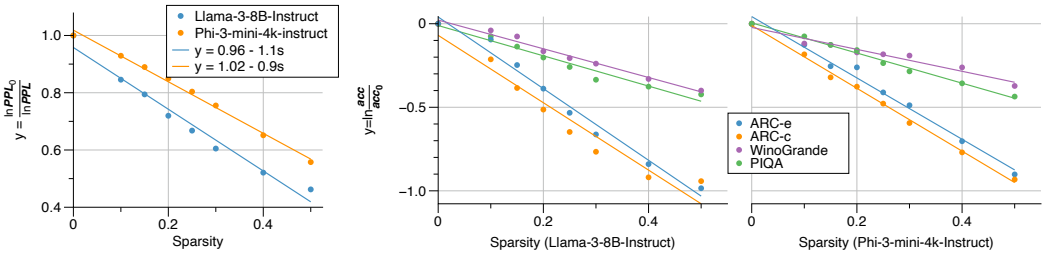
To verify Propos. 2, we pruned Llama-3-8B-Instruct (Dubey et al., 2024) and Phi-3-mini-4k-Instruct (Abdin et al., 2024) with different dimension sparsity with SliceGPT, then evaluated the pruned and unpruned models on WikiText2, as shown in Fig. 3a. We can observe that the perplexities fit well with the linear function, which justifies our analytical expression Eq. (2).

3.2 Accuracy

Accuracy of generative language models is bonded to the sequence probability, while perplexity is defined as the multiplicative inverse of the sequence probability. A well-generalized model would assign a high probability to the correct sequence (high accuracy but low perplexity). Therefore, intuitively, accuracy is inversely relevant to perplexity.

We postulate that a logarithmic form like Eq. (2) might also exist in the correlation between accuracy and representation dimension sparsity. Note that here we consider the accuracy of short, multiple-choice answers (e.g. "Yes"/"No", or "A"/"B"/"C"), since this accuracy can be computed via exact match without ambiguity. We tried possible logarithmic expressions similar to $\frac{\ln \texttt{PPL}_0(D)}{\ln \texttt{PPL}(D)}$ with the same LLMs used for perplexity evaluation. We replaced the perplexities with multiple-choice accuracies on ARC-e, ARC-c, WinoGrande and PIQA. Among these logarithmic expressions, the best-fitted form is $\ln \frac{\texttt{acc}}{\texttt{acc}_0}$, as shown in Fig. 3b. Thus we have the following empirical theorem with similar analytical expressions as Propos. 2:

Proposition 3. *For structured pruning with representation dimension sparsity s, the multiple-choice accuracy of the pruned model, denoted as $\texttt{acc}(D)$, and that of the original model, denoted as*



(a) Perplexity on WikiText2

(b) Accuracy in binary/multiple-choice Tasks

Figure 3: Fitted evaluation results. The fitting coefficients and errors are in Appx. A.2.

$\texttt{acc}_0(D)$*, satisfy*

$$\ln \frac{\texttt{acc}(D)}{\texttt{acc}_0(D)} \propto 1 - s, \tag{3}$$

where D is the test dataset for evaluating multiple-choice accuracy.

We can observe that the left-hand side of Eq. (3) is almost the multiplicative inverse of the left-hand side of Eq. (2). This is consistent with the fact that $\frac{1}{\texttt{PPL}}$ and $\texttt{acc}$ are positively correlated when the model generalizes well, and they have (nearly) the same range ($\frac{1}{\texttt{PPL}} \in (0, 1]$ and $\texttt{acc} \in [0, 1]$). The right-hand sides of Eq. (2) and (3) are exactly the same, indicating that the representation dimension dominates the model predictions. Though the multiple-choice accuracies fit Eq. (3) albeit less accurately than the perplexities, this formula consistency further empirically justifies what we claimed in Sec. 2.

4 Related Work

Sparsity and Scaling Law in LLM. Sparsifying LLMs at the matrix level (Frantar & Alistarh, 2023), module level (Men et al., 2025) or representation dimension level (Ashkboos et al., 2024), has drawn a lot of attention in recent years. The correlation between sparsity and model performance has remained underexplored. In this work, we analytically investigate this correlation. Frantar et al. (2024) put sparsity into the original scaling law (Kaplan et al., 2020), but consider other factors, such as the size of the dataset and training steps. In the context of scaling law, we solely focus on the entropy loss shifting with sparsity, and our Eq. (2) and (3) are expressed in standard evaluation metrics. Broadly speaking, our work is a special case of scaling law, but finer-grained, more precise and more analytical than the original scaling law.

Circuits in Transformers. In the field of mechanistic interpretability, circuits are the subgraphs in the neural networks that have distinct computation mechanisms (Jain et al.; Conmy et al., 2023). In the context of the circuit analysis in transformers, Elhage et al. (2021) discusses the contributions of the weight matrices to the final residual stream from a mathematical perspective. However, they do not fully address the impacts of the representation dimension on the model predictions. We dive into these impacts and give quantitative conclusions on two common evaluation metrics (i.e. perplexity and accuracy).

5 Conclusion

This work mechanistically investigates the impacts of representation dimension on the model predictions and, herein, model performance. It introduces analytical relations to estimate the pruned model performance, which is empirically valid in real-world LLM settings. Limitations and future work are discussed in Appx. A.3.

REFERENCES

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*, 2024.

Federico Barbero, Alex Vitvitskyi, Christos Perivolaropoulos, Razvan Pascanu, and Petar Veličković. Round and round we go! what makes rotary positional encodings useful? *arXiv preprint arXiv:2410.06205*, 2024.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 16318–16352. Curran Associates, Inc., 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

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

Elias Frantar, Carlos Riquelme Ruiz, Neil Houlsby, Dan Alistarh, and Utku Evci. Scaling laws for sparsely-connected foundation models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=i9K2ZWkYIP.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P Dick, Hidenori Tanaka, Tim Rocktäschel, Edward Grefenstette, and David Krueger. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. In *The Twelfth International Conference on Learning Representations*.

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. *arXiv preprint arXiv:2001.08361*, 2020.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and weipeng chen. ShortGPT: Layers in large language models are more redundant than you expect, 2025. URL https://openreview.net/forum?id=JMNht3SmcG.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Hao Wang, Tao Xiang, Shangwei Guo, Jialing He, Hangcheng Liu, and Tianwei Zhang. Transtroj: Transferable backdoor attacks to pre-trained models via embedding indistinguishability. *arXiv preprint arXiv:2401.15883*, 2024.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. In *The Eleventh International Conference on Learning Representations*, 2023.

Mingxue Xu, Sadia Sharmin, and Danilo P Mandic. Geometry is all you need: A unified taxonomy of matrix and tensor factorization for compression of generative language models. *arXiv preprint arXiv:2410.03040*, 2024.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Du Su, Dawei Yin, and Huawei Shen. The fall of rome: Understanding the collapse of llms in model editing. *arXiv preprint arXiv:2406.11263*, 2024.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

A APPENDIX

Table 1: Notation in this paper.

Symbol	Meaning
$\oplus$	Direct sum (concatenation in practice).
d	Embedding dimension, or model dimension (Kaplan et al., 2020), the dimension of the residual stream (Elhage et al., 2021; Kaplan et al., 2020)
m	Intermediate size of the MLPs.
$h_{\texttt{dim}}$	Dimension of the attention heads.
$h_{\texttt{attn}}$	Number of the attention heads.
v	Number of the key-value heads.
h	Number of the attention layers.
l	Token sequence length.
s	Embedding dimension sparsity.
γ	Scaling factor.
$\mathbf{v}$, $\mathbf{v}_i$	Embedding vector, ith row the embedding matrix E.
$\mathbf{W}$	Weight matrix.
$\mathcal{M}$	Transformer module.
D	Dataset.
N	The number of the layers to be compressed in $\mathcal{M}$.
E, $E_{\texttt{in}}$, $E_{\texttt{out}}$	Embedding matrix, input embedding matrix, output embedding matrix.
A, A_i	Output of the attention layer, output of the ith attention layer.
H	Entropy.
κ	Unit entropy of a real scalar variable.
σ	Activation function.

Table 2: The weight matrices in the transformer modules and their sizes, ordered by their appearances during the forwarding pass. The detailed notation is in Tab. 1.

$\mathbf{W}_{\texttt{RMSNorm}}$	d	$\mathbf{W}_v$	$d \times vh_{\texttt{dim}}$	$\mathbf{W}_{\texttt{up}}$	$d \times m$
$\mathbf{W}_q$	$d \times h_{\texttt{attn}}h_{\texttt{dim}}$ *	$\mathbf{W}_o$	$h_{\texttt{attn}}h_{\texttt{dim}} \times d$	$\mathbf{W}_{\texttt{down}}$	$m \times d$
$\mathbf{W}_k$	$d \times vh_{\texttt{dim}}$	$\mathbf{W}_{\texttt{gate}}$	$d \times m$		

* $h_{\texttt{attn}}h_{\texttt{dim}} = d$

A.1 ACTIVATION FLOW ALYSIS

Let d be the embedding dimension. Input a text of l tokens to the embedding layer first, we then have the encoded texts (embeddings) $E_{\texttt{in}} \in \mathbb{R}^{l \times d}$. $E_{\texttt{in}}$ is then processed by a layer norm operation. Take the current widely-used RMSNorm (Zhang & Sennrich, 2019) as our case, the normalized embeddings after the layer norm operation is

$$\bar{E}_{\texttt{in}} = \mathbf{W}_{\texttt{norm}} E_{\texttt{in}} \cdot \texttt{RMSNorm}(E_{\texttt{in}}) \in \mathbb{R}^{l \times d}, \tag{4}$$

$$\texttt{RMSNorm}(E) = \sqrt{\frac{1}{d}\sum_{i=1}^{d} E^\top[:,i]E[:,i]} \in \mathbb{R}, \tag{5}$$

where $\mathbf{W}_{\texttt{RMSNorm}} \in \mathbb{R}^d$ is the weight of the RMSNorm layer, $E_{\texttt{in}}[:,i]$ is the ith column of $E_{\texttt{in}}$.

The attention layer typically consists of four linear layers with weight matrices $\mathbf{W}_q \in \mathbb{R}^{d \times h_{\texttt{attn}}h_{\texttt{dim}}}$, $\mathbf{W}_k \in \mathbb{R}^{d \times vh_{\texttt{dim}}}$, $\mathbf{W}_v \in \mathbb{R}^{d \times vh_{\texttt{dim}}}$ and $\mathbf{W}_o \in \mathbb{R}^{h_{\texttt{attn}}h_{\texttt{dim}} \times d}$. $h_{\texttt{attn}}$ is the number of the attention heads, $h_{\texttt{dim}}$ is the head dimension, and v is the number of key-value heads[1]. In the default setting of

[1] We use the same notation names as listed in Llama-3-8B Configuration.

Llama[2], $h_{\texttt{attn}}h_{\texttt{dim}} = d$. Then we have the hidden states of query Q, key K and value V as follows:

$$Q = \mathbf{W}_q \bar{E}_{\texttt{in}}, \quad K = \mathbf{W}_k \bar{E}_{\texttt{in}}, \quad V = \mathbf{W}_v \bar{E}_{\texttt{in}}, \tag{6}$$

and then we get the attention weights of parallel attention layers in $\mathcal{M}$, $\mathbf{W}_a$, and attention outputs A as

$$\mathbf{W}_a = \texttt{softmax}(QK^\top)\gamma \in \mathbb{R}^{h_{\texttt{attn}} \times h_{\texttt{attn}} \times l}, \quad A^{'} = \mathbf{W}_o(\mathbf{W}_a V) \in \mathbb{R}^{l \times h_{\texttt{attn}} h_{\texttt{dim}}}, \tag{7}$$

where γ is the scaling factor. The attention outputs are then normalized again as $\bar{A}^{'} = \texttt{RMSNorm}(A^{'})$, then input to an MLP $\mathcal{M}_{\texttt{MLP}} = \{\mathbf{W}_{\texttt{gate}}, \mathbf{W}_{\texttt{up}}, \mathbf{W}_{\texttt{down}}\}$, such that we have the hidden state

$$A = \texttt{RMSNorm}(\mathbf{W}_{\texttt{down}}(\sigma((\mathbf{W}_{\texttt{gate}}\bar{A}^{'})(\mathbf{W}_{\texttt{up}}\bar{A}^{'})))) \in \mathbb{R}^{l \times h_{\texttt{attn}} h_{\texttt{dim}}}, \tag{8}$$

where σ is the activation function. Suppose there are h parallel attention layers in $\mathcal{M}$, the attention outputs of the ith attention layer is H_i, so the final output of the transformer module $\mathcal{M}$ is

$$\bigoplus_{i=1}^{h} A_i = A_1 \oplus A_2 \oplus \cdots \oplus A_h \in \mathbb{R}^{h \times l \times h_{\texttt{attn}} h_{\texttt{dim}}}. \tag{9}$$

Here, $\oplus$ denotes the direct sum, which means concatenation in the actual implementation. Since in the default setting, $h_{\texttt{attn}} h_{\texttt{dim}} = d$, we have the final hidden states output $\bigoplus_{i=1}^{h} A_i \in \mathbb{R}^{h \times l \times d}$.

A.2 Additional Experiment Results

The fitting details in Fig. 3 are clarified in Tab. 3.

Table 3: Fitting perplexities in Fig. 3a and accuracies in Fig. 3b.

Perplexity (Fig. 3a)	**Dataset**(D)	$y = as + b$ $(y = \frac{\ln \texttt{PPL}_0(D)}{\ln \texttt{PPL}(D)}$		
		a	b	RMSE
Llama-3-8B-Instruct	WikiText2	-1.08	0.96	0.03
Phi-3-mini-4k-Instruct		-0.90	1.02	0.01

Accuracy (Fig. 3b)	**Dataset**(D)	$y = as + b$ $(y = \ln \frac{\texttt{acc}(D)}{\texttt{acc}_0(D)})$		
		a	b	RMSE
Llama-3-8B-Instruct	ARC-e	-2.14	0.04	0.05
Phi-3-mini-4k-Instruct		-1.84	0.04	0.04
Llama-3-8B-Instruct	ARC-c	-2.02	-0.07	0.09
Phi-3-mini-4k-Instruct		-1.88	-0.01	0.02
Llama-3-8B-Instruct	WinoGrande	-0.86	-0.02	0.02
Phi-3-mini-4k-Instruct		-0.66	-0.02	0.02
Llama-3-8B-Instruct	PIQA	-0.91	-0.01	0.03
Phi-3-mini-4k-Instruct		-0.90	0.01	0.01

A.3 Limitations and Future Work

This paper gives explicit analytical relations between the pruned model performance and sparsity. However, these relations are derived from the direct representation dimension reduction, which is implemented by SliceGPT. The representation dimension may play similar roles in other pruning approaches (explicitly or implicitly), and it is worth investigating how it impacts their pruned model performance.

[2]For simplicity, we do not consider position encoding, dropout, attention mask, and MLP bias in LLMs.