STLM ENGINEERING REPORT: WEIGHT TYING

Dylan Hillier
A*STAR and Singapore Management University
das.hillier.2023@phdcs.smu.edu.sg

Leon Guertler
A*STAR and Nanyang Technological University
leon002@e.ntu.edu.sg

Bobby Cheng
A*STAR, Institute of Infocomm Research (I2R)
bobby_cheng@i2r.a-star.edu.sg

Cheston Tan
A*STAR, Institute of High Performance Computing (IHPC)
cheston-tan@i2r.a-star.edu.sg

ABSTRACT

In this work we explore parameter reduction through the use of weight tying and its impact on the development of tiny language models. We propose to use a combination of tying the FFNs across a subset of transformer layers, and then use LoRA to enable each layer to continue specialising.

***K*eywords** weight tying · language models · efficiency

1 Motivation

In our earlier work, titled 'Super Tiny Language Models'[1], we outlined our ambitions and approach for developing Super Tiny Language Models (STLMs) that are high performing in spite of having significantly smaller parameter counts. One of our listed strategies is experimentation over the effects of weight tying.

In principle the idea of weight tying is that you can reduce the number of parameters of a model without necessarily harming the performance of it, although it does have the downside of not necessarily reducing the computational load in terms of FLOPs which may be the effective limit for organizations with a limited computational budget.

2 Method

We consider a transformer with depth D hidden dimension H, and feed-forward dimension F (where $F \sim 4 \cdot H$ and vocab size V. We consider several potential schemes of weight tying:

- **Input/Output Embedding tying**: The most common form of weight tying is between the input embedding layer (which maps from token-indices to the latent embedding space) and the final output embedding layer (which maps from the latent embedding space to a distribution over tokens). This is seen in a wide variety of models including models like GPT2 [2], although not in some more modern models like the Llama family [3]. These saves $V \cdot H$ parameters. For our baseline this constitutes 25M parameters with the GPT2 tokenizer and is thus necessary for our baseline model.
- **FFN-Tying**: MobiLlama [4] utilise parameter sharing among the FFN layers of a transformer network, achieving a 60% reduction in the parameter size of existing models without a significant drop in performance. In addition to tying all the layers as performend by Thawaker et al, we propose to only share the k-interior layers (i.e. layers $1 + k$ through $D - k$) since these layers are thought to be largely redundant [5] in existing models. This saves roughly $3 \cdot (D - 2k) \cdot H \cdot F$ parameters for e.g. SwiGLU [6] FFNs. At our scale this constitutes 8 Million parameters

- **FFN-Tying + LoRA**: Following on from this we propose to augment these shared layers with LoRA [7], with the idea of enabling each such layer to specialise with a minimal number of parameters. These add back in $64 \cdot 6 \cdot (H + F) \cdot (D - 2k)$ parameters or roughly 2 Million parameters.
- **Attention Projection Sharing**: Attention layers typically include an output projection layer that mix the heads together back into the residual stream. These parameters are of size H^2 for each layer or 1 Million parameters if applied along the same layers as the previous techniques.

In order to meet the requirements of keeping our model around 50M parameters, we accordingly scale the models over depth, hidden dimension, and FFN dimension. This is with the exception of untying the input/output embeddings due to the fact that our prior belief is that it would damage modelling performance at our scale.

3 Experiments

We plan to train a series of models using these different techniques in various configurations with a means to investigating:

- Whether shared FFN layers with LoRA can be used to scale the depth of a model
- Whether there is a significant performance loss from tying the input and output embeddings at all
- Whether attention can be shared without damaging the model performance.

These runs are summarized in Table 1.

Technique	Number of Layers	Embedding Dimension	FFN Dimension	Number of Parameters
Baseline	8	528	1361	49M
Untied Input/Output Embedding	8	528	1361	76M
FFN Tying				
All, Wide	8	672	1733	50M
1-interior, Wide	8	608	1568	49M
2-interior, Wide	8	560	1444	49M
All, Long	14	608	1568	49M
FFN-Tying; LoRA				
All, Deep; $r = 64$	12	592	1527	50M
All, Wide; $r = 64$	8	640	1650	50M
1-interior, Wide; $r = 64$	8	592	1527	50M
2-interior, Deep; $r = 64$	12	512	1440	50M
Attention Projection Sharing				
Long	10	512	1320	50M

Table 1: Planned runs for this experiment

4 Pre-Registration Disclosure

This paper is intended to act as a preregistered copy of our experimentation into weight tying, before we actually perform our experimentation. For this report we have tried to establish exactly which runs we will do before running them. Nevertheless we note that these may change slightly. Furthermore we have some existing experimentation over using shared FFN weights, which were relatively inconclusive. As such we have a weak prior towards these techniques working well enough to justify being integrated into standard runs.

References

[1] Dylan Hillier, Leon Guertler, Cheston Tan, Palaash Agrawal, Chen Ruirui, and Bobby Cheng. Super tiny language models, 2024.

[2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[3] 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.

[4] Omkar Thawakar, Ashmal Vayani, Salman Khan, Hisham Cholakal, Rao M Anwer, Michael Felsberg, Tim Baldwin, Eric P Xing, and Fahad Shahbaz Khan. Mobillama: Towards accurate and lightweight fully transparent gpt. *arXiv preprint arXiv:2402.16840*, 2024.

[5] 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. *arXiv preprint arXiv:2403.03853*, 2024.

[6] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[7] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.