There is an I in Attention

Akshay Sivaraman & Natjanan Mongkolsupawan & Abhishek Sankar
10-623 Generative AI Course Project

December 10, 2025

1 Introduction

Multi-head self-attention (Vaswani et al. [2023]) relies on query and key projections $W_Q, W_K \in \mathbb{R}^{d\times r}$ whose product determines the attention scores. A key observation is that the score matrix is invariant under the joint transformation

$$W_Q \leftarrow W_Q R, \qquad W_K \leftarrow W_K R^{-\top},$$

for any invertible $R \in \mathbb{R}^{r\times r}$. This implies that each head contains r^2 redundant degrees of freedom: parameters that do not affect the computed attention.

We address this redundancy by reparameterizing the projections in a fixed canonical form. For each head, we replace the usual learned (W_Q, W_K) with a structured pair in which part of W_K is held to a fixed orthonormal basis and only the remaining components are learned. A symmetric construction similarly applies to the value projection W_V and the output projection W_O, whose product also contains an analogous r^2 degrees of freedom. The orthonormal basis chosen for both W_K and W_O is I_r ($r \times r$ identity matrix). By constraining these matrices to a shared canonical structure, we eliminate all redundant parameters while preserving the representational capacity of the original layer.

In total, this yields a reduction of $2r^2$ parameters per attention head in multi-headed attention. Unlike low-rank or efficient-attention variants, this approach maintains the full mathematical expressivity of standard attention. We refer to this formulation as I-Attention.

Our experiments on GPT-2 Small demonstrate that this reduction comes at no cost to model performance. In fact, we observe that removing these symmetries accelerates training convergence, as the optimizer no longer needs to navigate flat directions in the loss landscape caused by the redundancy.

2 Dataset / Task

To simulate a research diary environment with rapid iteration cycles, we utilized WikiText-2 (Merity et al. [2017]) for the majority of our ablations. WikiText-2 contains approximately 2 million tokens of high-quality Wikipedia articles. While smaller than modern pre-training corpora, it is sufficient to observe the convergence dynamics and stability of attention mechanisms.

2.1 Parameter Reduction for Attention

The advent of multi-head attention in transformers (Vaswani et al. [2023]) has brought with it several variants. The parameter count can be reduced via multi-query attention (Shazeer [2019]) and grouped-query attention (Ainslie et al. [2023]), both of which provide considerable speed up with a tradeoff of some level of diminished performance.

Efficiency in the transformer architecture is an active field of research, with approaches ranging from high-level architecture pruning to low-level matrix operation optimization. Bermeitinger et al. [2024], for example, seeks to simplify the attention mechanism in both single-head attention, by combining the query and key parameters, and multi-head attention, using Cholesky decomposition in symmetric cases. Linformer (Wang et al. [2020]), utilizes the low-rank matrices to approximate the attention mechanism, thus allowing attention to be run in linear time.

Recent work by Karbevski and Mijoski [2025] demonstrates that W_Q is redundant in simplified decoder-only settings and can be absorbed into W_K and W_V. We instead exploit the inherent $GL(r)$ symmetry of $W_Q^\top W_K$ and provide a canonical parameterization that removes the precise redundant r^2 degrees of freedom per head.

Other approaches to parameter reduction include strategies such as weight sharing between different blocks. Zhussip et al. [2025] utilizes dictionary learning to separate the learning of vector atoms and their coefficients, a useful technique in alternative representations of weight matrices.

2.2 Rank of Projection Matrices

The low-rank nature of many deep learning methods, including transformers, is widely noted in the literature (Balzano et al. [2025]), and this over-parameterization may even contribute positively to gradient descent-based learning methods (Allen-Zhu et al. [2019]). Perhaps the most notable application of low-rank qualities lies in the intrinsic dimensionality within transformer models (Aghajanyan et al. [2020]), enabling fine-tuning methods such as LoRA (Hu et al. [2021]), which can be applied to projection matrices within the self-attention modules.

Many results in the literature favor increasing the dimensions of the heads to increase the rank and representative capabilities of the attention mechanism. Formal proofs by Bhojanapalli et al. [2020] shows that multi-head attentions suffer from a low-rank bottleneck when the number of head increases beyond a certain point, suggesting increasing the sizes of the heads. The tradeoff between the number of heads and their ranks is also explored in formal proofs by Amsel et al. [2024], suggesting that long context tasks may benefit from either increasing the rank or the depth of the networks.

The utilization the inherently low rank properties of matrices, as we plan to do in this project, has been less explored. One example is Cordonnier et al. [2021], which shows via empirical evidence the inherently low rank of multi-head attention due to redundancy among heads, and proposes a decomposition-based method that reduces the parameter count of the attention score computation using a custom compression ratio.

In a similar vein, we seek to make the most out of existing low-rank architectures without adding parameters. To our knowledge, we are the first to explore parameter reduction by fixing the projection matrices to their canonical bases.

3 Methodology

3.1 Baseline

Since we attempt to make a modification to the parameterization of attention by Vaswani et al. [2023], we simply use standard multi-headed self attention as the baseline. As detailed in Section 4, we use standard attention layers in GPT2, trained from scratch on next-token prediction, as our baseline.

3.2 I-Attention

Eliminating Redundant Degrees of Freedom in Self-Attention

Let d denote the model dimension and let r denote the per-head dimension in multi-head self-attention. In nearly all modern architectures, d is chosen so that $d = n_h \times r$, where n_h is the number of attention heads. For clarity and consistency with common practice, we follow this convention. However, the mathematical arguments below do not explicitly rely on d being divisible by n_h. Similar redundancies hold even when r is chosen independently.

In a standard multi-head self-attention layer, each head is parameterized by projection matrices $W^Q, W^K, W^V \in \mathbb{R}^{d\times r}$, where r is the embedding dimension of the head, and d is the embedding dimension across the entire attention layer. An output projection $W^O \in \mathbb{R}^{d\times d}$ is also present for the entire attention layer, that operates on the concatenation of the outputs of each head.

The attention scores (before softmax) for head i is computed as follows:

$$S_i = Q_i K_i^\top/\sqrt{r} = X W_i^Q (W_i^K)^\top X^\top/\sqrt{r}$$

A fundamental observation is that the attention score matrix S_i depends on W_i^Q and W_i^K only through the product $W_i^Q (W_i^K)^\top$. This implies that the parameterization of (W_i^Q, W_i^K) contains an inherent r^2-dimensional redundancy. Specifically, for any invertible matrix $R \in \mathbb{R}^{r\times r}$, the transformation

$$W_i^Q \leftarrow W_i^Q R, \qquad W_i^K \leftarrow W_i^K R^{-\top}$$

leaves the score matrix unchanged.

Thus, the parameterization of (W_i^Q, W_i^K) contains an inherent r^2-dimensional redundancy per head: any internal change of basis within the head subspace yields identical attention scores.

Canonical Parameterization of Queries and Keys

To remove this redundancy, we introduce a canonical parameterization that fixes a portion of the key projection matrix. For each head i, we write:

$$W_i^Q = C_i^Q, \qquad W_i^K = \begin{bmatrix} I_r \\ A_i^K \end{bmatrix},$$

where $A_i^K \in \mathbb{R}^{(d-r)\times r}$ and $C_i^Q \in \mathbb{R}^{d\times r}$ contain all learnable parameters. Any original pair (W_i^Q, W_i^K) can be mapped to this canonical form by selecting an appropriate invertible change of basis R (with the assumption that there are at least r linearly independent rows in W_i^K, which is essentially guaranteed unless degenerate matrices are chosen). Because the invariance is exact, this reparameterization preserves the full functional capacity of the attention mechanism.

Analogous Redundancy in Value and Output Projections

A similar degree-of-freedom redundancy arises in the value and output projections. In standard attention,

$$\mathrm{head}_i(X) = A_i V_i$$

where A_i is the attention matrix computed for that head, and $V_i = XW_i^V$

$$\text{MHA}(X) = [A_1V_1 \,|\, A_2V_2 \,|\, ... \,|\, A_{n_h}V_{n_h}]\, W^O,$$

where $W^O \in \mathbb{R}^{d\times d}$.

Since $W_i^V \in \mathbb{R}^{d\times r}$ and W^O acts on the concatenation of all heads, this is equivalent to:

$$\text{MHA}(X) = \sum_{i=1}^{n_h} A_i V_i W_i^O,$$

Where $W^O = \begin{bmatrix} W_1^O \\ W_2^O \\ \cdots \\ W_{n_h}^O \end{bmatrix}$ and $W_i^O \in \mathbb{R}^{r\times d}$

Note that $V_iW_i^O = XW_i^VW_i^O$.

Once again, $W_i^VW_i^O$ forms a $d \times d$ matrix of rank r, indicating that we can save r^2 parameters.

More specifically, $P_i = W_i^VW_i^O$ is invariant under

$$W_i^V \leftarrow W_i^V R, \qquad W_i^O \leftarrow R^{-1}W_i^O,$$

for any invertible R. To remove the associated r^2 redundant parameters, we again fix a canonical basis:

$$W_i^V = C_i^V, \qquad W_i^O = \begin{bmatrix} I_r & A_i^O \end{bmatrix},$$

and learn only A_i^O, with W_i^V remaining unconstrained.

Parameter Reduction Analysis

Each of the two invariances (W_Q, W_K) and (W_V, W^O) contributes r^2 redundant parameters per attention head in MHA. I-Attention removes both redundancies while preserving the full expressive power of the layer, saving exactly $2r^2$ parameters per head.

In an MHA block, we have a total of $4d^2$ parameters. In I-Attention, we save $2n_hr^2$ parameters. Therefore, in total, the number of parameters in I-Attention is $4d^2 - 2n_hr^2$.

Given that typically, $r = d/n_h$, the total parameters in an I-Attention block is $4d^2 - \frac{2d^2}{n_h}$. In other words, we save $\frac{1}{2n_h}$ of the total parameters per attention head.

4 Experiments

We evaluate our proposed attention parameterization by training GPT-2 models on the WikiText-2. Our primary goal is to verify that our method maintains the expressivity and convergence properties of standard multi-head attention while reducing the number of parameters.

The code for these experiments can be found here.

4.1 Experimental Setup

We use the GPT-2 Small architecture configuration for all experiments: 12 layers, an embedding dimension of 768, and a varying number of attention heads. Table 1 lists the different experiment configurations for the number of heads. The models are trained on WikiText-2 for 40 epochs with a batch size of 32. These hyperparameters are chosen due to limited compute.

Hyperparameters for the experiments follow the standard hyperparameters for GPT-3 training (Brown et al. [2020]), which provides a decent approximation since we were not able to identify the GPT-2 training hyperparams. The number of epochs and batch sizes are adjusted to fit our own compute budget. We use the AdamW optimizer with a learning rate of 6×10^{-4}. A learning rate scheduler is used to warm up the LR from 0 to the maximum value over 1 epoch, followed by cosine decay to 10% over 9 epochs and constant LR thereafter.

Each experiment includes identical trianing processes for two attention modules:

- **Standard Attention (Baseline)**: The standard multi-head attention mechanism as defined in the original Transformer and GPT-2 architectures.
- **I-Attention (Ours)**: Our reparameterized attention mechanism where the query projection is fixed to an identity-like basis, removing redundant degrees of freedom.

4.2 Hardware and Timing

All experiments were conducted on NVIDIA A100 GPUs via Modal and Google Colab. Each experiment takes approximately 1 hour per 10 epoch, totaling to about 4 hours per experiment. Validation at the end of each epoch on the evaluation set takes another 2-5 minutes per epoch.

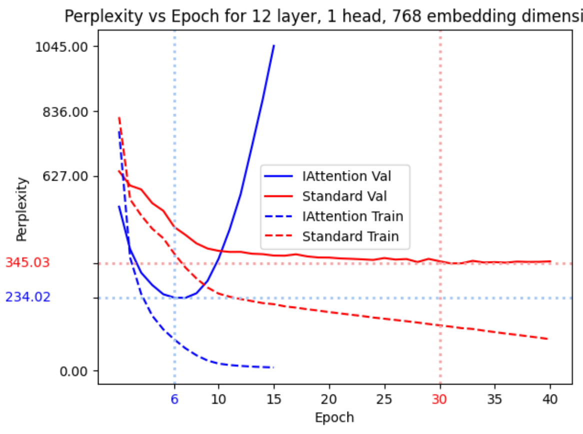
Due to compute limitations, we are not conducting other experiments that require extensive computation, memory, or storage. For instance, GPT-2 small remains overparameterized compared to the WikiText-2 dataset, but we are unable to fit the WikiText-103 dataset into the processors available to us.

4.3 Results and Analysis

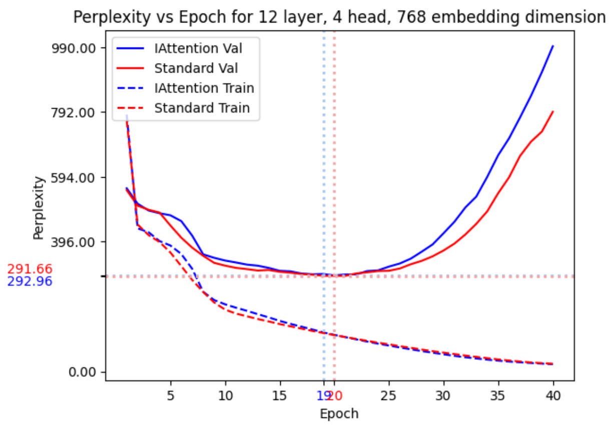
Parity in Performance. Across all configurations, I-Attention achieved perplexity scores comparable to the baseline. In the standard 12-head configuration, the Baseline achieved a best validation PPL of **295.0**, while I-Attention achieved **248.85**. This seems consistent with the fact that the $2r^2$ parameters removed were redundant.

Dataset	Layers	Heads	Embedding	Attention	Param Saving	Best Train PPL	Best Val PPL
WikiText-2	12	12	768	Standard	-	25.20	295.00
WikiText-2	12	12	768	I-Attention	1.18 M	6.89	248.85
WikiText-2	12	8	768	Standard	-	25.24	297.43
WikiText-2	12	8	768	I-Attention	1.77M	8.72	258.57
WikiText-2	12	4	768	Standard	-	15.59	291.66
WikiText-2	12	4	768	I-Attention	3.54 M	94.02	292.96
WikiText-2	12	1	768	Standard	-	100.89	345.03
WikiText-2	12	1	768	I-Attention	14.16M	10.03	234.02

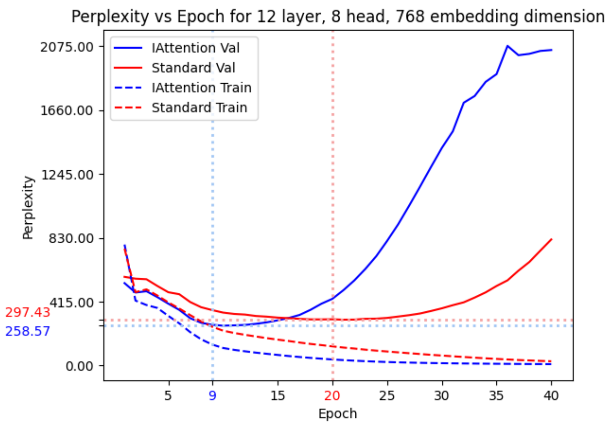
Table 1: Experiments conducted evaluating the reparameterization across hyperparameters. The top row is the standard GPT-2 Small.



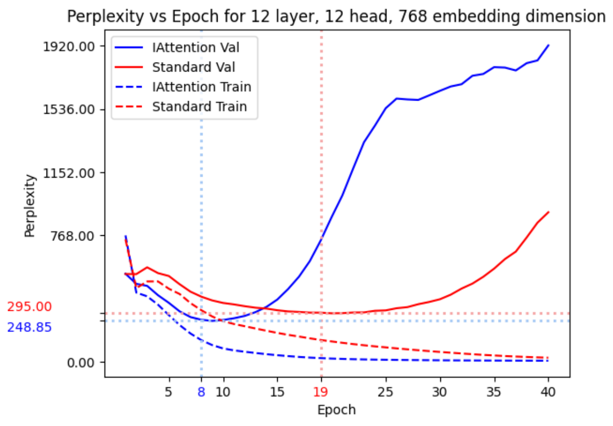
(a) Single-Head Attention



(b) 4-Head Multi-Head Attention



(c) 8-Head Multi-Head Attention



(d) 12-Head Multi-Head Attention

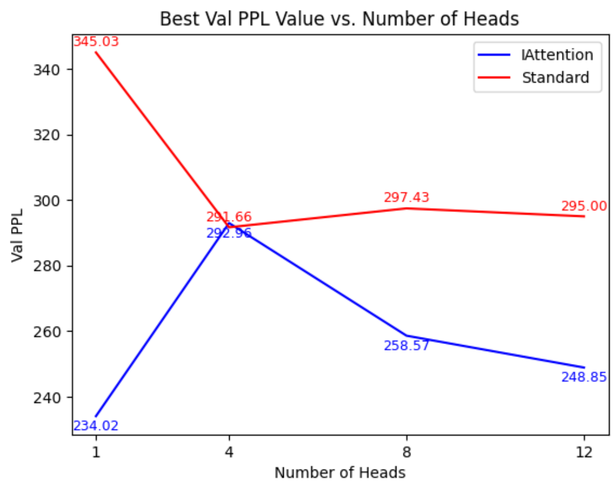
Figure 1: Perplexity Comparisons across Train and Validation Runs

Convergence Velocity. A distinct finding was the speed of optimization. In the 1, 8, and 12-head ablation, I-Attention reached its optimal validation PPL more than 10 epochs earlier as compared to standard attention, and reaches a lower validation perplexity.

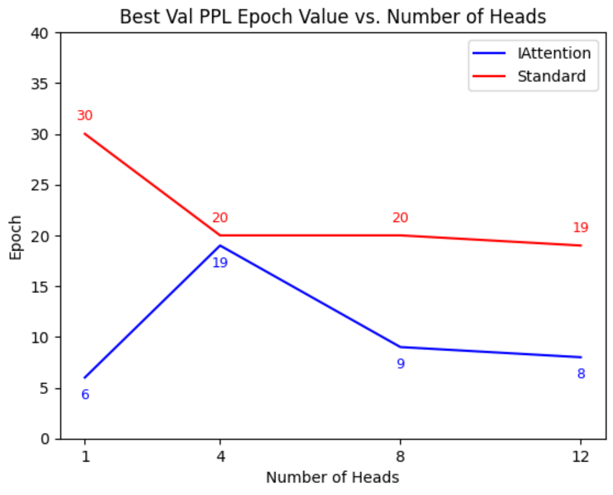
We hypothesize that removing the $GL(r)$ symmetry eliminates "flat" directions in the loss landscape. The optimizer no longer wastes updates traversing equivalent parameter configurations $(W_Q R, W_K R^{-T})$, leading to a steeper, more direct descent.

Overfitting Risks. As a consequence of faster learning, I-Attention exhibited earlier overfitting. Once the model extracted the signal, it began fitting noise several epochs earlier than the baseline. This suggests that efficient parameterizations may require more aggressive early stopping.

5 Code Overview



(a) Best Val Perplexity vs. Number of Heads



(b) Best Val Epoch vs. Number of Heads

Figure 2: Scaling of the Best Validation Runs Across Numbers of heads

All the code in the repository (submitted to Gradescope) is written by us. The most notable file is `i_attention_modules/mha.py`, which consists of our multihead I-Attention layer. In `orig_attn/mha.py`, we implement standard attention. Other important files are `models/gpt2.py` which is the implementation of our GPT2 model (adapted from HW3 this semester). We create a config that allows us to easily swap out regular attention layers for I-Attention layers within the model.

`data_loaders/wikitext.py` contains our dataloader for next-token prediction for both `wikitext2` and `wikitext103` datasets. `wikitext103` is roughly 50 times larger than `wikitext2`. However, due to computational constraints, we never performed training on this dataset.

`scheduler.py` is for our learning rate scheduler. This is a similar scheduler used for the training of GPT-3. Lastly, `train_gpt2.py` is our training script for kicking off experiments. `modal_train.py` runs the same script, but is used for kicking off training jobs on Modal (an online platform for running scripts on GPUs).

These are all the core files for our implementation.

6 Timeline

Please refer to Table 2 for a succinct summary of our time distribution on this project.

7 Research Log

Our initial idea in the proposal revolved around a larger scope: multiple attention variants, measured in both memory and FLOPs. After discussing with the course staff, we decided to narrow the scope down to simply comparing the parameter requirements and training performance, as those have less dependencies on trying to benchmark our runtime or FLOPs against state-of-the-art optimized PyTorch code. We further decided to focus on only single-head and multi-head attention in order to fully flesh out the implications of exact pairs of projection matrices. This also served the purpose of experimentally verifying our hypothesis around their mathematical equivalence.

During the midpoint report, there were several bugs (such as alignment issues, memory and runtime issues, etc.) that we fixed over time in order to arrive at the current code. Further, we encountered significant limitations from our limited compute. We experimented with several different GPUs (Modal ones as well as Colab ones, like T4) and eventually decided to spend all of our Colab budget on A100 processors. As a result, we could not complete planned experiments targeting the number of layers, embedding dimensions, or hyperparameters.

8 Conclusion

8.1 Key Findings

In this project, we introduce I-Attention, a novel attention parameterization for single-head and multi-head attention that tends to converge faster to lower perplexities, while maintaining mathematical equivalence and expressivity to standard attention mechanisms. This supports our claim that $2r^2$ parameters per attention head are redundant.

Hours Taken	Focus / Activity	Notes / Challenges
10	Math and proofs	We did a lot of scratch math and proofs to understand if this could be a viable method to save parameters and to find theoretical justification behind this method.
6	Literature review	It was difficult to find multiple relevant papers. We delved deep into related research areas to check if similar optimizations have been done.
18	Implementation	We took great care to implement I-Attention correctly, as vectorization optimizations are error-prone. We tackled many bugs and built boilerplate code for model definitions, training, and evaluation, among others.
6	Designing experiments	We decided to focus on ablating over the number of heads, since this directly affects the number of parameters in I-Attention. We also looked up relevant datasets and experiment tasks for this kind of project.
20	Running Experiments	We babysat many different experiment types and versions.

Table 2: Summary of project trajectory and timeline.

8.2 Future Work

As we plan to dive deeper into the capabilities of I-Attention, our future work spans multiple areas:

Time-Efficient Implementation: This project utilizes higher-level PyTorch layers such as `nn.Linear` inside the attention computations, which should be replaced by low-level C++ implementations for CPU and CUDA. We plan to draw inspiration from efficient attention implementations such as FlashAttention (Dao et al. [2022]).

Extensively Benchmark Performance: If more compute is available, we wish to evaluate the capabilities of I-Attention more rigorously using larger datasets (WikiText-103 (Merity et al. [2016]), OpenWebText (Gokaslan and Cohen), etc.), larger model types with more heads (GPT-2 Medium and Large), and comprehensive hyperparameter grid searches.

Attention Variants: We plan to investigate extensions of this method to other types of attention layers, such as Grouped Query Attention (GQA) (Ainslie et al. [2023]).

FLOPs Analysis for Reparameterizing Existing Models: Since matrix operations in typical attention layers can be reparameterized into I-Attention forms as in 3.2, we hope to evaluate the impact of this reparameterization on FLOPs, memory, and performance.

Fine-Tuning Experiments on Existing Models: Since I-Attention seems to converge faster with fewer parameters, we plan to investigate the usage of I-Attention during fine-tuning of existing models.

9 Thought-Experiment on Compute

1. Actual compute use. We utilized NVIDIA T4 GPUs via the Modal platform for our experiments, as well as A100s on Google Colab. We conducted 8 primary training runs (Standard attention vs I attention, over 4 different number of heads), each training a GPT-2 Small model on WikiText-2 for 40 epochs. The total training time was approximately 48 hours.

2. Additions if given more compute. We expect this to extend to about 2400 to 3600 hours of GPU use to run on a scaled down version of OpenWebText or WikiText103, which would be a much more realistic dataset for pretraining models this large.

Additionally, we wish to convert existing models to I-attention models and perform finetuning. We estimate another 2000 hours of GPU usage for doing experiemnts on varied model architectures and sizes, and for different common datasets and tasks.

- **Total GPU Hours:** $\approx$ 50 hours (NVIDIA A100).
- **Cost:** At an estimated rate of \$4.09/node/hour for A100 instances, the total cost was approximately \$200, but a major chunk of this came via colab pro student subscriptions and Modal credits.

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning, 2020. URL `https://arxiv.org/abs/2012.13255`.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints,

2023. URL https://arxiv.org/abs/2305.13245.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization, 2019. URL https://arxiv.org/abs/1811.03962.

Noah Amsel, Gilad Yehudai, and Joan Bruna. On the benefits of rank in attention layers, 2024. URL https://arxiv.org/abs/2407.16153.

Laura Balzano, Tianjiao Ding, Benjamin D. Haeffele, Soo Min Kwon, Qing Qu, Peng Wang, Zhangyang Wang, and Can Yaras. An overview of low-rank structures in the training and adaptation of large models, 2025. URL https://arxiv.org/abs/2503.19859.

Bernhard Bermeitinger, Tomas Hrycej, Massimo Pavone, Julianus Kath, and Siegfried Handschuh. Reducing the transformer architecture to a minimum. In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, page 234–241. SCITEPRESS - Science and Technology Publications, 2024. doi: 10.5220/0012891000003838. URL http://dx.doi.org/10.5220/0012891000003838.

Srinadh Bhojanapalli, Chulhee Yun, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Low-rank bottleneck in multi-head attention models, 2020. URL https://arxiv.org/abs/2002.07028.

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

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. Multi-head attention: Collaborate instead of concatenate, 2021. URL https://arxiv.org/abs/2006.16362.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022. URL https://arxiv.org/abs/2205.14135.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Marko Karbevski and Antonij Mijoski. Key and value weights are probably all you need: On the necessity of the query, key, value weight triplet in decoder-only transformers, 2025. URL https://arxiv.org/abs/2510.23912.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2017.

Noam Shazeer. Fast transformer decoding: One write-head is all you need, 2019. URL https://arxiv.org/abs/1911.02150.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity, 2020. URL https://arxiv.org/abs/2006.04768.

Magauiya Zhussip, Dmitriy Shopkhoev, Ammar Ali, and Stamatios Lefkimmiatis. Share your attention: Transformer weight sharing via matrix-based dictionary learning, 2025. URL https://arxiv.org/abs/2508.04581.