EXPLORING SPEED/QUALITY TRADE-OFFS IN DIMENSIONALITY OF ATTENTION MECHANISM

Optimization with Grouped Query Attention and Diverse Key-Query-Value Dimensionalities

Authors

Khalit Gulamov
K.Gulamov@student.tudelft.nl

Arie van Deursen
Responsible Professor

Maliheh Izadi
Responsible Professor

Aral de Moor
Supervisor

Faculty of EEMCS
Delft University of Technology
The Netherlands

References

[1] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, "GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints." arXiv, Dec. 23, 2023. Accessed: Apr. 02, 2024. [Online]. Available: http://arxiv.org/abs/2305.13245

[2] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021.

[3] R. Eldan and Y. Li, "TinyStories: How Small Can Language Models Be and Still Speak Coherent English?" arXiv, May 24, 2023. Accessed: Nov. 01, 2023. [Online]. Available: http://arxiv.org/abs/2305.07759

[4] N. Shazeer, "Fast Transformer Decoding: One Write-Head is All You Need." arXiv, Nov. 05, 2019. Accessed: Apr. 02, 2024. [Online]. Available: http://arxiv.org/abs/1911.02150

1 INTRODUCTION

Transformers have led to the rise of various language models, including those that generate text. A prominent example is the GPT models used in ChatGPT.

One of the challenges researchers have encountered is optimizing the speed of text generation. This challenge arises from autoregressive decoding, where each token is generated sequentially, with each new token contextually dependent on all preceding tokens.

The inference is important to optimize whether:

- Saving on operational costs;
- Allowing the model to operate in a resource-constraint environment (e.g., locally);
- Providing good user experience in real-time applications.

Notable speedup techniques include altering attention dimensionality, namely - Multi and Group Query Attention (MQA and GQA) [1] and shrinking key and value vectors [4].

2 CONTRIBUTIONS

Research Goal: To explore possible trade-offs and better understand the properties of existing approaches. This will effectively help select an optimal architecture based on needs in future applications.

To achieve it, we:

1. Assess previously non-addressed properties of GQA and introduce a modified GQA that allows for more possible configurations;
2. Compare GQA models to reduced keys/queries and values (KQV) models in terms of speed and quality;
3. Try a combined approach.

3 BACKGROUND

Autoregressive decoding forces to store all the previous token representations (called key-value cache). This causes a memory bottleneck, a bigger issue than parallelizable computation [4]. Thus, the techniques are aimed at decreasing the cache size.

Figure 1 shows a comparison of **MHA vs. GQA vs. MHA.**

As for the **MHA vs. GQA vs. KQV:**

- MHA has keys and values for all attention heads;
- GQA has them for fewer heads (but of the same size);
- KQV models have them for all heads, but each vector is smaller (including queries).

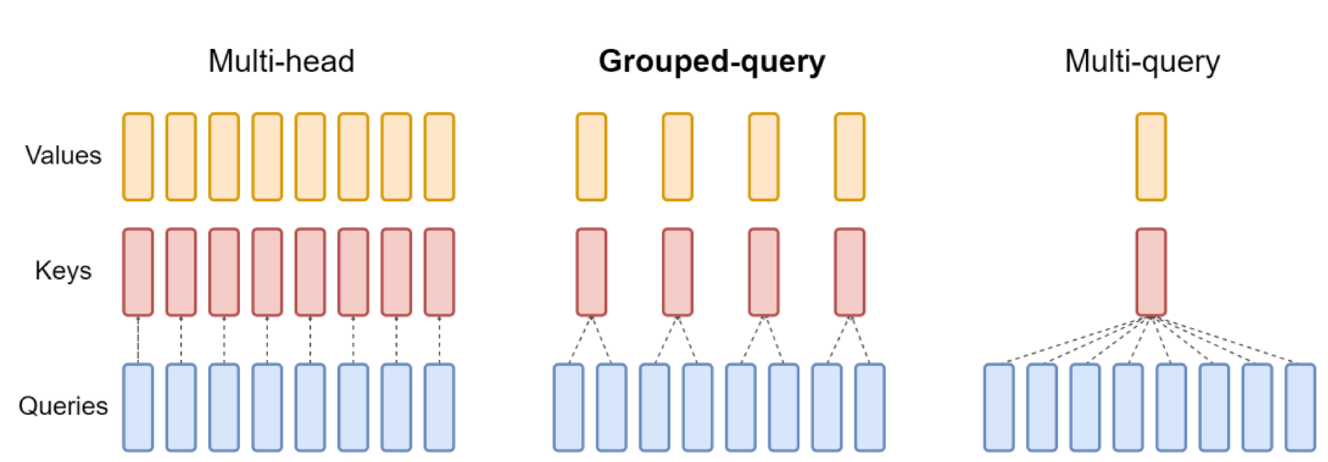

Figure 1: High-level Comparison of Standard Attention, GQA, and MQA Taken from Ainslie et al. [1]

4 RESEARCH QUESTIONS AND METHOD

RQ1.1 Relationships between speed and quality for different group ratios (GQA)?
RQ1.2 Effect of the total number of attention heads on GQA performance?
RQ2 Speed and quality difference between GQA and KQV models?
RQ3 Combined approach performance?

To answer the questions, we train small GPT-Neo [2] models* on the TinyStories dataset [3] and evaluate quality and speed (when applicable). We introduce a modified GQA approach (Figure 2) for more possible group ratios to account for smaller total number of heads.

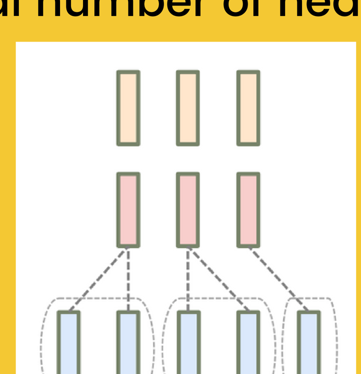

Figure 2: Modified GQA

*To eliminate the number of parameters from the equation, FFN width is adjusted to have the same number of parameters for each model.

5 RESULTS

Terminology: Group Proportion - the ratio of key-value heads to total heads in GQA models; KQV proportion - the ratio of new key/query and value vectors' sizes to the original ones; batch size - number of inputs processed for inference in parallel.

We train on the TinyStories dataset, evaluate quality on BLiMP and GLUE/SuperGLUE to test natural language understanding and measure inference speed for small and large batch sizes in Python.

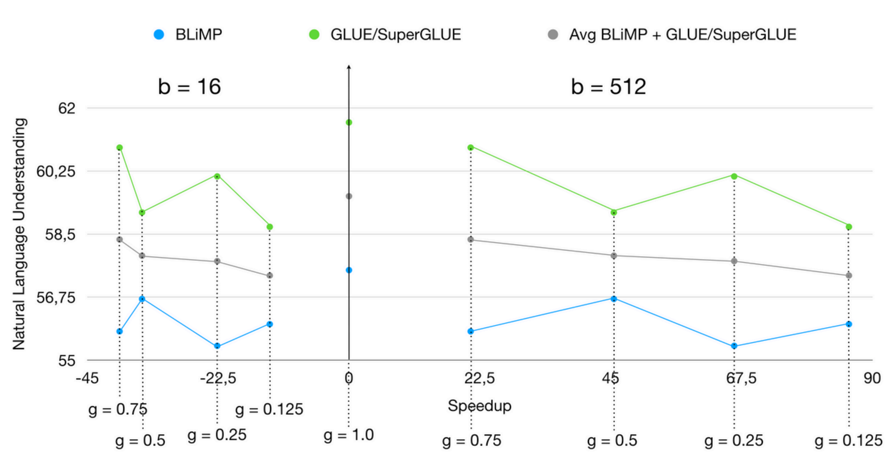

Figure 3: RQ1.1 - Speed to Quality Proportions for Various Group Proportions

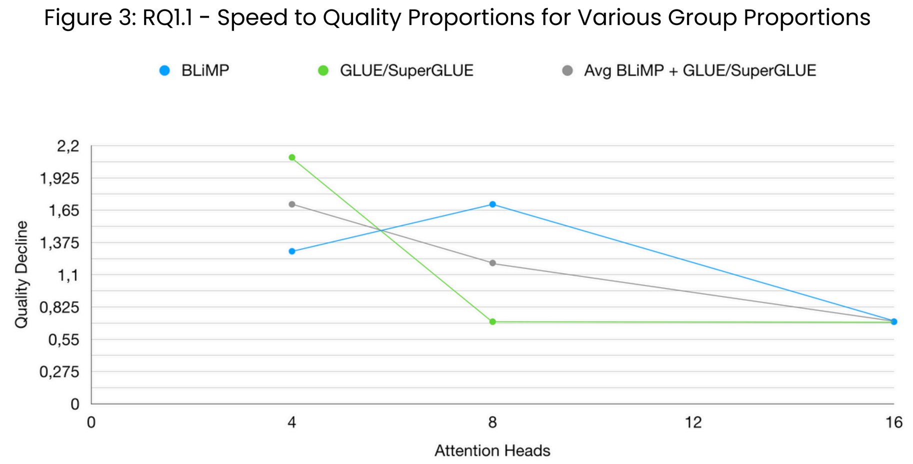

Figure 4: RQ1.2 - Quality Decline Relative to Corresponding Baselines with Fixed Group Proportion for Various Numbers of Attention heads

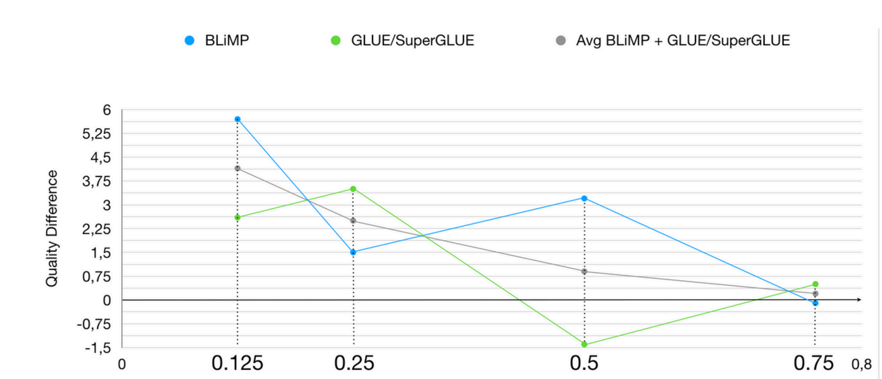

Figure 5: RQ2 - Quality Difference Between GQA and KQV models, where the KQV and GQA proportions are equal. A negative difference means KQV performs better.

Attention Type		Baseline	GQA	KQV	GQA	KQV	GQA	KQV	GQA	KQV
Group Proportion		–	0.75	–	0.5	–	0.25	–	0.125	–
KQV Proportion		–	–	0.75	–	0.5	–	0.25	–	0.125
FFN Width		1024	1120	1216	1216	1408	1312	1600	1360	1696
BLiMP		57.5%	55.8%	**55.9%**	**56.7%**	53.5%	**55.4%**	53.9%	**56.0%**	50.3%
GLUE/ SuperGLUE		61.6%	**60.9%**	60.4%	59.1%	**60.5%**	**60.1%**	56.6%	**58.7%**	56.1%
Average of BLiMP and GLUE/ SuperGLUE		59.55%	**58.35%**	58.15 %	**57.9%**	57.0%	**57.75%**	55.25%	**57.35%**	53.2%
Average Time per Token Generation (μs)	b = 16	136.67	225.66	135.19	212.16	133.51	176.69	134.44	158.22	134.15
	b = 512	17.31	14.32	11.82	11.90	7.48	10.42	6.47	9.31	5.99
Speedup Over the GPT-Neo Baseline	b = 16	–	−39.44%	**1.09%**	−35.58%	**2.36%**	−22.65%	**1.66%**	−13.62%	**1.87%**
	b = 512	–	20.92%	**46.50%**	45.54%	**131.36%**	66.23%	**167.54%**	85.96%	**188.84%**

Table 2: RQ2 - Full Speed and Quality Pair-wise Comparison of GQA and KQV Models Leading to an Equivalent KV-Cache Cut.

Attention Type	MHA	GQA	MHA	GQA	MHA	GQA
Attention Heads	4	4	8	8	16	16
FFN width	1024	1120	1024	1120	1024	1120
BLiMP	58.1%	56.8%	57.5 %	55.8%	54.1%	53.4%
GLUE/ SuperGLUE	61.2%	59.1%	61.6%	60.9%	59.5%	58.8%
Average of BLiMP and GLUE/ SuperGLUE	59.65%	57.95%	59.55%	58.35%	56.8%	56.1%

Table 1: RQ1.2 - Performance of GQA Models with a Fixed Group Proportion for Different Numbers of Attention Heads Next to Corresponding Baselines

Attention Type	GQA	COMBINED	KQV
Group Proportion	0.25	0.5	–
KQV Proportion	–	0.5	0.25
FFN width	1312	1504	1600
BLiMP	55.4%	54.5%	53.9%
GLUE/ SuperGLUE	60.1%	58.5%	56.6%
Average of BLiMP and GLUE/SuperGLUE	57.75%	**56.5%**	55.25%
Time per Token (μs) (b = 16)	176.69	207.79	134.44
Time per Token (μs) (b = 512)	10.42	9.01	6.47
Speedup (b = 16)	−22.65%	-34.23%	1.66%
Speedup (b = 512)	66.23%	**92.12%**	167.54%

Table 3: RQ3 - Performance of the Combined Approach Next to Individual Ones, resulting in the Equivalent Key-Value Cache Cut

Observations:

- Speed and quality have an inverse relation in GQA; with the large memory consumption, it becomes close to being linearly proportional;
- The more attention heads there is, the less GQA degrades the quality given the same group proportion;
- KQV models tend to be faster but of lower quality. With the lower group and KQV proportions, however, the difference in quality vanishes;
- The combined model's metrics lie in between those that use individual approaches.

6 DISCUSSION & FUTURE WORK

Implications:

- The lack of proportionality in Figure 3 could be due to the modified GQA overhead;
- The decrease in Figure 4 could be due to removing excessive attention heads;
- KQV models could be faster due to reducing the number of floating point operations or due to modified GQA overhead
- The combined model could be a valid way to expand the range of options available for choosing the desired trade-offs.

Main Limitations:

- Overhead induced by modified GQA;
- The models are trained on one epoch

Future Work:

- Addressing main limitations and trying to train on a better dataset and evaluate with other quality metrics;
- Trying post-training optimization with various techniques.