Grouped-Query Latent Attention: A More Device-Friendly Attention Structure

Yingkai Tang
Yali High School International Department, Changsha, Hunan, China
tangyingkai08908@outlook.com

1. Abstract

In recent years, the Multi-Head Attention(MHA) mechanism has gained widespread use due to its exceptional language modeling capabilities. However, its high time complexity and memory usage limit its potential for deployment on edge devices. While existing improvements, such as Grouped-Query Attention (GQA), have mitigated these issues, they can still pose challenges for resource-constrained devices. Therefore, this paper proposes an improved GQA structure: Grouped-Query Latent Attention (GQLA). GQLA reduces resource consumption further based on GQA by projecting the input into a lower-dimensional latent space (using a latent denominator of 2) and performing GQA within this space. Evaluation results showed that, compared to GQA, GQLA is 9.11% faster, and 72.19% lower standard deviation indicates a more stable generation. Also, GQLA uses 0.17 MB less memory on average with negligible variance across runs. Meanwhile, the degradation of expressive capabilities of GQLA-based model remains acceptable. This work provides a new, practical approach to developing and deploying large-scale models.

2. Introduction

Transformers have revolutionized Natural Language Processing(NLP). While Multi-Head Attention(MHA) is foundational due to its capability of capturing diverse dependencies across different subspaces within the input sequence[1], the community has shifted towards more efficient variants for deployment, since the quadratic growth of its computational and memory complexity[2] can be challenging for most edge devices. The overall time complexity for computing attention scores in MHA is:

$$O(L^2 d)$$

where
L is the length of the sequence

d is the total hidden dimension

Grouped-Query Attention(GQA) has emerged as a dominant standard. It efficiently balances the performance and memory usage by grouping multiple query heads to share a single key and value head[3]. This reduces the size of the KV cache during autoregressive generation, leading to lower memory usage at inference time. However, GQA does not alter the original hidden dimension, meaning that the process of high-dimension vectors is still required. This constitutes the primary computational and memory bottleneck. It is worth exploring methods to maintain the advantages of grouping in GQA while further reducing the dimensionality of internal representations to achieve ultimate efficiency.

This paper proposes one possible solution, Grouped-Query Latent Attention(GQLA). Based on GQA, GQLA introduces a lower-dimension latent space where the inputs are projected. The attention calculations are performed in this latent space. At last, the results are projected to the original high-dimension space.

The advantage of GQLA is that while retaining the grouping feature of GQA, it further reduces the time consumption and memory usage of attention calculation.

3. Methods

3.1 Preliminaries: Grouped Query Attention (GQA)

GQA significantly reduces memory consumption and computational consumption while preserving model expressiveness by grouping multiple query heads to share the same set of keys and values.

- Generate Q, K, V: For the input sequence $X \in \mathbb{R}^{L\times d}$ where L is the length of the sequence length and d is the hidden dimension,

$$Q = XW_Q \in \mathbb{R}^{L\times H\times d_h}$$
$$K = XW_K \in \mathbb{R}^{L\times \mathrm{G}\times d_h}$$
$$V = XW_V \in \mathbb{R}^{L\times \mathrm{G}\times d_h}$$

where
H is the number of heads
G is the number of groups
$d_h = d/H$ is the dimension of each head

- Reshape to support grouping: Q is reshaped into (L, H, d_h), and K, V are reshaped into (L, G, d_h)

- Copy K and V to corresponding heads of Q: For each group $g \in [0, G)$, the index of its corresponding Q head is from $g \cdot (H/G)$ to $(g+1)\cdot(H/G) - 1$. Copy the K and V in each group for the corresponding H/G Q heads.

- Calculate attention: for every Q head h:

$$O_h = softmax(Q_h K_g^T / \sqrt{d_h}) V_g$$

where
O_h is the output

- Concatenate all head outputs and output: Concatenate all O_h to $O \in \mathbb{R}^{L\times Hd_h} = \mathbb{R}^{L\times d}$, and the final output is:

$$Output = OW_o$$

where
W_o is the output projection matrix

3.2 Grouped Query Latent Attention (GQLA)

Grouped Query Latent Attention(GQLA) builds on the basis of GQA, further reducing computational overhead by introducing a low-dimensional latent space.

The dimension of the latent space d_{lat} is:

$$d_{lat} = \mathrm{d/r}$$

where
d is the original dimension
r is the latent denominator, set by the developer

-Latent projection: Given the input sequence $X \in \mathbb{R}^{L\times d}$, the latent projections are:

$$q_{lat} = XW_q \in \mathbb{R}^{L\times H\times d_{h-lat}}$$
$$\mathrm{k}_{lat} = XW_\mathrm{k} \in \mathbb{R}^{L\times G\times d_{h-lat}}$$
$$\mathrm{v}_{lat} = XW_\mathrm{v} \in \mathbb{R}^{L\times G\times d_{h-lat}}$$

where
H is the number of heads
G is the number of groups
$d_{h-lat} = d_{lat}/H$ is the dimension of each head

- Grouping and copying: q_{lat} is reshaped to (L, H, d_{lat}). k_{lat} and v_{lat} is reshaped to (L, G, d_{lat}). Copy the K and V in each group to pair with the Q heads.

- Calculate the latent attention:

$$O_h = softmax(q_{lat} k_{lat}^T / \sqrt{d_{h-lat}}) v_{lat}$$

where
O_h is the output of the latent attention calculation

- Concatenate all head outputs: Concatenate all O_h to $O \in \mathbb{R}^{L\times d}$. Therefore, the output is:

$$O_{lat} = OW_o$$

where
W_o is the output projection matrix

- Project to the original dimension:

$$Output = Linear(O_{\mathrm{lat}})$$

There are improvements in time complexity of attention calculation in GQLA compared to GQA. For GQA, the time complexity of the calculation of attention score is:

$$O(L^2 d)$$

where:
L is the length of the sequence
d is the total hidden dimension

With the latent space, GQLA reduced the time complexity to:

$$O(L^2 \cdot \frac{d}{r})$$

where:
r is the latent denominator

GQLA also reduces the QKV activation memory. In GQA:

$$Q_{activation} = Ld$$
$$K_{activation} = V_{activation} = L(d \cdot h_{kv}/h_q)$$

where:
h_{kv} is the number of K/V heads
h_q is the number of Q heads

In GQLA:

$$Q_{activation} = L(d/r)$$
$$K_{activation} = V_{activation} = L(d \cdot h_{kv}/(rh_q))$$

3.3 Experimental Setup

My experiment aims to compare two attention mechanisms: GQLA and GQA. The experiments were conducted using two simple models with token and position embeddings, one attention layer, and one linear output layer. Both models used the same seed and parameters and ran on one Nvidia Tesla T4 GPU. One model used GQLA, and the other used GQA. The parameters are listed below.

Table 1: Parameters of the Models

Vocabulary Size	1000
Hidden Dimension	512
Number of Attention Heads	8
Number of Groups	2

For GQLA model, the latent denominator r is set to 2.
The experimental factors and explanations are shown in the following figure.

Table 2: Factors and Explanations

Factor	Explanation
t	The average time required for the model to complete a single inference.
Mem	Peak GPU memory consumed during model inference.
AttnSim	Cosine similarity between attention patterns of the two models, measuring alignment in learned attention behaviors.
$\lvert\Delta_{diag}\rvert$	Absolute value of difference of mean diagonal attention of the two models.

Experiment 1: This test aims to evaluate the difference in inference speed between GQA and GQLA under identical input conditions to determine whether GQLA introduces additional computational overhead or possesses acceleration potential. The two models perform 100 forward propagations. The time required for each inference is recorded. I applied Welch's t-test to compare the two time distributions, setting the significance level(α) to 0.05.

Table 3: H_0 and H_a for Time Test

Null Hypothesis(H_0)	There are no differences between two time distributions
Alternative Hypothesis(H_a)	The inference time of GQLA model is less than that of GQA model. $t_{GQLA}<t_{GQA}$

Experiment 2: This test aims to evaluate whether GQLA reduces GPU/memory consumption under identical input conditions. The two models perform 30 forward propagations. The peak graphics memory usage is recorded. We applied Welch's t-test to compare the two memory distributions, setting the significance level(α) to 0.05.

Table 4: H_0 and H_a for Memory Test

Null Hypothesis(H_0)	There are no differences between two memory distributions
Alternative Hypothesis(H_a)	The memory usage of GQLA model is less than that of GQA model. $Mem_{GQLA} < Mem_{GQA}$

Experiment 3: This test analyzes whether the latent space projection introduced by the GQLA model alters the distribution characteristics of attention weights. We assume that GQA serves as a strong reference for expressive attention modeling. Thus, high similarity between attention distributions between GQLA model and GQA model would suggest that the latent projection preserves the model's ability despite operating in a reduced-dimensional space.

The attention heat map of the two models are compared by calculating the cosine similarity, and difference of mean diagonal attention.

4. Results

The result of Experiment 1 is shown below:

Table 5: Time Test Results

Mean Time(seconds)	t_{GQA}=0.000812±0.000151 t_{GQLA}=0.000738±0.000042
T-statistic	4.7400
P-value	0.000003
Mean Difference(seconds)	0.000075

The P-value is 0.000003, which is smaller than α. Therefore, we can reject H_0 and conclude that $t_{GQLA} < t_{GQA}$. Also, the standard deviation of GQLA model 0.000042 is 72.19% less than that of GQA model, which is 0.000151, indicating a more stable generation.

The result of Experiment 2 is shown below:

Table 6: Memory Test Results

Mean Memory Usage(MB)	Mem_{GQA}=23.15±0.00 Mem_{GQLA}=22.98±0.00
T-statistic	inf
P-value	$\ll$0.05
Mean Difference(MB)	0.17

GQLA demonstrated lower peak memory consumption, with an average reduction of 0.17 MB. Although this value may seem small, it is noteworthy that the models applied in this experiment only contain 1 attention layer. In deeper models composed of multiple such layers, the per-layer memory savings are expected to accumulate. Thus, it is reasonable to conclude that the design of GQLA has indeed resulted in improved resource efficiency.

Also, the P-value is much smaller than α. Therefore, we can reject H_0 and conclude that $\mathrm{Mem}_{GQLA} < \mathrm{Mem}_{GQA}$. Although the T-statistic approaches infinity due to minimal measurement fluctuations, considering the figure below showing GQLA's overall distribution slightly below GQA's, we can reasonably infer that GQLA effectively reduces memory occupation by decreasing intermediate dimensions in attention computations.

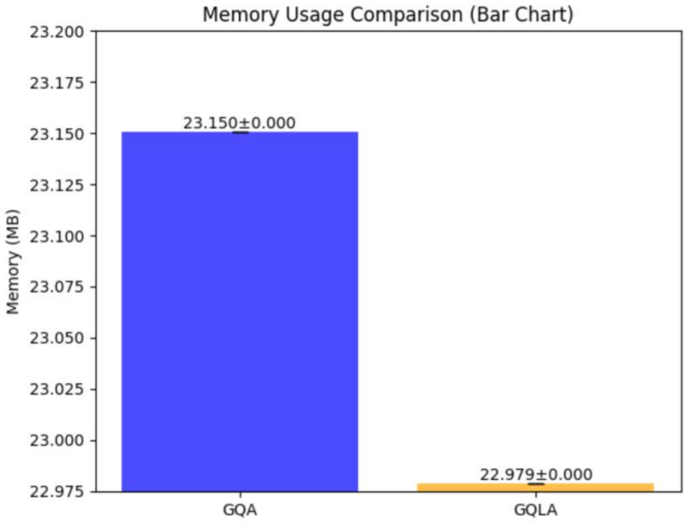

Figure 1: Memory Usage Distribution

The result of Experiment 3 is shown below:

Table 7: Attention Test Result

$AttnSim$	0.9961
$|\Delta_{diag}|$	0.0001

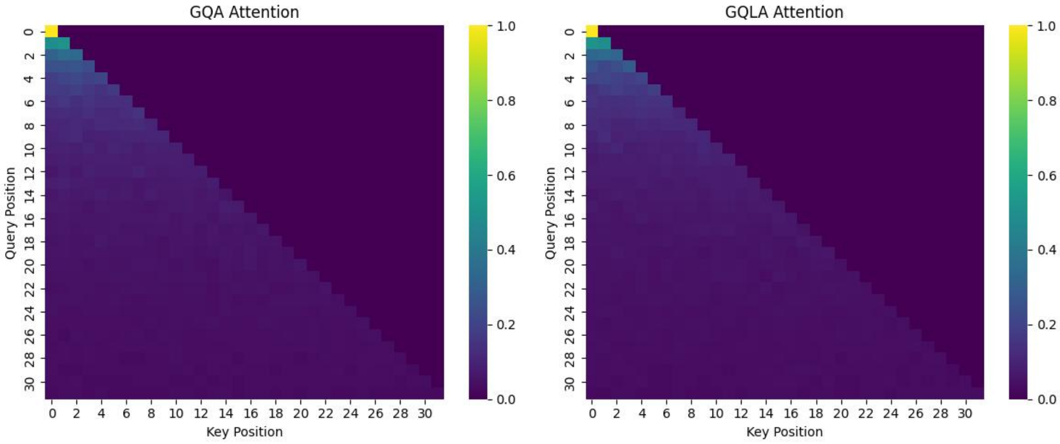

Figure 2: the Attention Heat Map of the Two Models

The cosine similarity(0.9961) is close to 1. This means though GQLA operate in latent space, its attention distribution remains largely consistent with that of GQA. Combining with the result of the difference in mean diagonal attention is small(0.0001), we can conclude that GQLA does not introduce significant structural bias and preserves the attention patterns of the original GQA model.

The results have revealed that GQLA is capable of using less time and memory while retaining most expressive capabilities compared to GQA.

5. Conclusion

In an effort to reduce the computational costs associated with the Grouped-Query Attention (GQA) mechanism, this paper presents a new architecture: Grouped-Query Latent Attention (GQLA). By projecting the inputs into a lower-dimensional latent space and performing attention calculations there, GQLA is expected to improve calculation efficiency while maintaining low loss of expressiveness. Evaluations have shown that GQLA reduces inference time by 9.11% and peak GPU memory usage by 0.17 MB. Additionally, the 72.19% reduction in standard deviation indicates enhanced generative stability. GQLA offers a potential solution for optimizing transformer efficiency. However, more research is needed to evaluate GQLA's performance on edge devices in real-world situations.

6. References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*. https://arxiv.org/abs/1706.03762

[2] Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., & Metzler, D. (2021). Long Range Arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*. https://arxiv.org/abs/2011.04006

[3] Ainslie, J., Lee, S., Pham, M.-T., Zhai, S., Li, Z., & Wang, Y. (2023). GQA: Training generalized multi-query transformer models from multi-head checkpoints. arXiv preprint arXiv:2305.13245. https://arxiv.org/abs/2305.13245