Literature Survey on *TransMLA: Multi-Head Latent Attention Is All You Need[1]*

Pengli Shao
Student ID: 002295825
CS6120.MERGED.202530 - Natural Language Processing
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02115
shao.pe@northeastern.edu

April 16, 2025

Abstract

As the size of the model increase, the increase in K and V cache of LLM has become one of the main bottleneck in terms of the development of LLM. `Multi-Head Latent Attention` is an architectural innovation based on MHA(Multi-Head Attention) and is applied to the Deepseek V2 236B model that effectively reduces the KV cache by 93.3% compare to DeepSeek 67B model according to the paper *DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model*[2].

1 Introduction

1.1 Background

KV cache is one of the main bottleneck of LLM model, as the sequence length and model size increase, the KV cache increases. An example is that LLaMA-65B with 8-bit key-value quantization needs over 86GB of GPU memory to store 512K tokens(Touvron et al., 2023)[3]. In comparison, a single NVIDIA H100 PCIe GPU provides only 80GB memory size, which is insufficient for such contect lengths. Therefore, to run inference with very long contects on LLaMA-65B model locally with only one GPU, one would need to buy the newest NVIDIA H100 NVL GPU that has 94 GB memory size.
The paper(Meng et al., 2025)[1] stated that most mainstream models are still using GQA(Group Query Attention) or MQA(Multi-Query Attention) technique to reduce KV cache. However, these two techniques reduced KV cache requirements by sacrificing performance, and both techniques require fine-tuning to recover the model's accuracy.

1.2 Objective

The purpose of the paper(Meng et al., 2025)[1] is to prove that under the same KV cache condition, MLA outperforms GQA. Moreover, it introduces a technique called TransMLA that can efficiently convert any GQA-based model(such as LLaMA-3, Qwen-2.5, Mistral, Mixtral, Gemma-2, and Phi-4) into a MLA-base at low cost.

The purpose of this literature review is to help readers understand the difference between MLA, GQA, and MQA, understanding it's math behind and basic logics of TransMLA technique.

1.3 Scope

Deepseek V2 is well known for it's extremly low cost, about 1% of GPT-4-Turbo-1106 model. In addition, it is the first model that used MLA architectural to reduce KV cache and allow it to output with such low cost.

Model	API Pirce / 1M Tokens	
	Input $	Output $
DeepSeek-V2	0.14	0.28
GPT-4-Turbo-1106	10.00	30.00
GPT-4-0613	30.00	60.00
GPT-3.5	1.50	2.00
Gemini 1.5 Pro	7.00	21.00
Claude 3 Opus	15.00	75.00
Claude 3 Sonnet	3.00	15.00
Claude 3 Haiku	0.25	1.25
abab-6.5（MiniMax）	4.14	4.14
abab-6.5s（MiniMax）	1.38	1.38
ERNIE-4.0（文心一言）	16.56	16.56
GLM-4（智谱清言）	13.80	13.80
Moonshot-v1（月之暗面）	3.32	3.32
Qwen1.5 72B（通义千问）	2.76	2.76
LLaMA 3 70B	3.78	11.34
Mixtral 8x22B	2.00	6.00

Figure 1: Model Price Table[2]

Figure 1 illustrates the API price of the Deepseek V2 input price is only $0.14 and its output price is only $0.28, the API price of Deepseek V2 is relatively much lower than other models, especially GPT-4-Turbo and GPT-4.

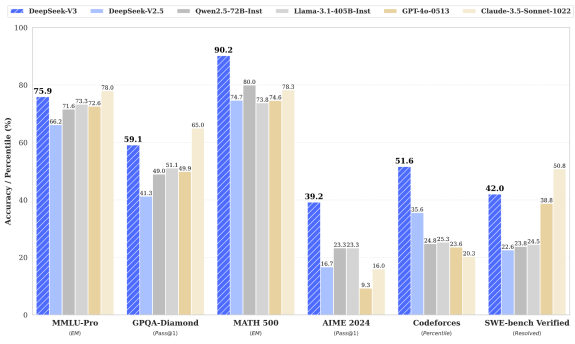


Figure 2: Performance Chart[4]

From figure 2, the next-generation model of Deepseek V2, which is Deepseek v3, has achieved performance compatible with GPT-4o while continuing to apply the MLA technique. Moreover, its API pricing remains significantly lower than that of GPT-4o.

In an era where LLMs are increasingly driving up inference and deployment costs in pursuit of better performance, few techniques existed to significantly reduce costs of KV memory useage before MLA was introduced by Deepseek V2. This led to the consistently high API pricing across most mainstream models. However, the MLA technique has indeed enabled Deepseek's models to achieve performance on par with most contemporary models, while keeping their pricing substantially lower than other mainstream models.
Introducing the MLA technique allows more companies and individuals to deploy long-context LLMs more affordably. The TransMLA technique also makes it possible to convert any GQA-based model into MLA-based.

2 Methodology

2.1 Search Strategy

This survey mainly used Google Scholar, GitHub, and arxiv.org. And, I also used YouTube to help me better understand the Deepseek model's architectural.

2.2 Keywords

- MLA
- Deepseek
- GQA
- XXX Technical Report
  XXX: Model name
- GPT
- MQA
- KV cache

The above keywords are not all, most related paper were found in other paper. For example, TransMLA paper(Meng et al., 2025)[1] introduced GQA, MQA, and Deepseek V2 technical report, Deepseek V3 technical report, etc.

2.3 Selection Criteria

The report mainly focuses on techniques mentioned in TransMLA paper(Meng et al., 2025)[1], and some relative works about reducing KV caches such as GQA and MQA.

3 Literature Review

3.1 Thematic Analysis

This section will introduce each subtopics of the paper, discuss the key findings, and highlight how paper relate to each other.

3.1.1 Basic Information

The paper(Meng et al., 2025)[1] introduced some basic informations, including MHA(Multi-Head Attention), GQA(Group Query Attention), and MLA(Multi-Head Latent Attention.

a. MHA(Multi-Head Attention)
MHA computes:

$$Q = XW_Q$$
$$K = XW_K$$
$$V = XW_V$$

where X is the input sequence, $X \in \mathbb{R}^{T \cdot D}$, $W_{i \in Q,K,V} \in \mathbb{R}^{D \cdot (n_h \cdot d_h)}$, $Q, K, V \in \mathbb{R}^{T \cdot (n_h \cdot d_h)}$
Then, split Q,K, and V into n_h attention head:

$$[Q_1; Q_2; ...; Q_{n_h}] = Q$$
$$[K_1; K_2; ...; K_{n_h}] = K$$
$$[V_1; V_2; ...; V_{n_h}] = V$$

where $Q_i, K_i, V_i \in \mathbb{R}^{T \cdot d_h}$
Then, calculate the attention output of each attention head:

$$O_i = softmax(\frac{Q_i K_i^T}{\sqrt{d_h}}) V_i W_{O_i} \in \mathbb{R}^{T \cdot D}$$

Finally, the final output is the sum of all attention head outputs:

$$O = \sum_{i=1}^{n_h} O_i$$

Therefore, each token in MHA will have n_h number of Q, K, and V

b. GQA(Group Query Attention)
GQA is a technique to reduce the KV cache, it divides the Q heads into g_h groups instead of having n_h heads($g_h < n_h$).
Each group sharing a single K and V. For example, if $n_h = 32, g_h = 4$, then each group will have $\frac{n_h}{g_h} = 8$ Q, sharing 1 group of K and V. Therefore, at the same condition of having 32 heads, instead of having $n_h = 32$ K and V, we only need 4 K and 4 V for GQA, such that:

$$[Q_1; Q_2; ...; Q_{n_h}] = Q$$
$$[K_1; K_2; ...; K_{g_h}] = K$$
$$[V_1; V_2; ...; V_{g_h}] = V$$

c. MQA(Multi-Query Attention MQA is a more extreme approach to save KV cache, the key idea of this technique is to further reduce the number of K and V. The n_h number of Q will share only 1 K and 1 V, K and V will not be per-head, it will be the only vector that all heads are sharing,

such that:

$$[Q_1; Q_2; ...; Q_{n_h}] = Q$$
$$K_1 = K$$
$$V_1 = V$$

d. MLA(Multi-Head Latent Attention There are two assumptions before explaining MLA. First, the explanation does not consider RoPE(Rotary Position Embedding), and it does not apply compression to the Query.
The basic idea of MLA is the low-rank joint compression for K and V to reduce KV cache(Liu et al., 2024)[2], such that::

$$W_Q \in \mathbb{R}^{D \cdot (n_h \cdot d_h)}$$
$$W_K^a, W_V^a \in \mathbb{R}^{D \cdot r}$$
$$W_K^b, W_V^b \in \mathbb{R}^{r \cdot (n_h \cdot d_h)}$$
$$W_O \in \mathbb{R}^{(n_h \cdot d_h) \cdot D}$$

where r represents the KV compression dimension.
Then, MLA computes:

$$Q = XW_Q \in \mathbb{R}^{T \cdot (n_h \cdot d_h)}$$
$$K = XW_K^a W_K^b \in \mathbb{R}^{T \cdot (n_h \cdot d_h)}$$
$$V = XW_V^a W_V^b \in \mathbb{R}^{T \cdot (n_h \cdot d_h)}$$

Finally, compute the attention output:

$$O = \sum_{i=1}^{n_h} softmax(\frac{Q_i K_i^T}{\sqrt{d_h}}) V_i W_{O_i} \in \mathbb{R}^{T \cdot D}$$

MLA only requires the intermediate latent representations to be stored:

$$K^c = XW_K^a \in \mathbb{R}^{T \cdot r}$$
$$V^c = XW_V^a \in \mathbb{R}^{T \cdot r}$$

where $r < n_h \cdot d_h$. During inference stage, MLA will absorb W_K^b into W_Q and W_V^b into W_O, the attention output for each head is:

$$O = softmax(\frac{XW_Q \cdot (W_K^b)^T \cdot (K^c)^T}{\sqrt{d_h}}) V^c \cdot W_V^b \cdot W_O \in \mathbb{R}^{T \cdot D}$$

At last, its final attention output will just be the sum of all O of each head.

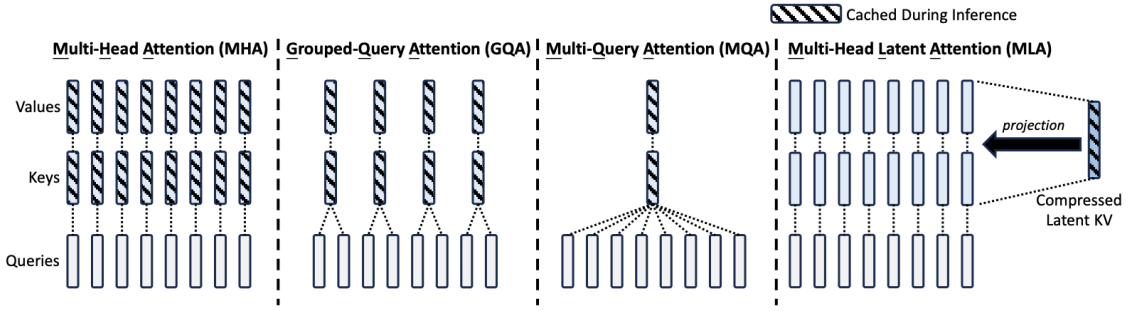


Figure 3: KV cache saving techniques(Liu et al., 2024)[2]

Figure 3 is a graph that clearly shows the difference in MHA, GQA, MQA, and MLA. A very obvious feature of MLA is that it saves KV cache compare to MHA without sacrificing the number of K and V in different head like GQA and MQA.

e. TransMLA

GQA has $n_q = \frac{D}{d_h}$ number of heads(e.g. 32 heads) for Q, and saving KV cache by decreasing the number of heads of K and V to n_k(e.g. 4 heads) number of heads where $n_k < n_q$, such that:

$$[\underbrace{K_1, K_2, ..., K_{n_k}}_{n_k=4}] = K$$

To match the number of heads for Q and K, GQA need to copy these 4 Ks in 4 heads $s = \frac{n_q}{n_k} = \frac{32}{4} = 8$ times, such that:

$$[\underbrace{K_1, ..., K_1}_{s=8}, ..., \underbrace{K_{n_k}, ..., K_{n_k}}_{s=8}] = XW'_K = K'$$

Thus, in our example where Q has 32 heads, K has 4 heads, we need to copy each K 8 times to match 32 Qs.

Then, apply SVD decomposition to W'_K(W'_K has low rank because most columns and rows are linearly dependent) where:

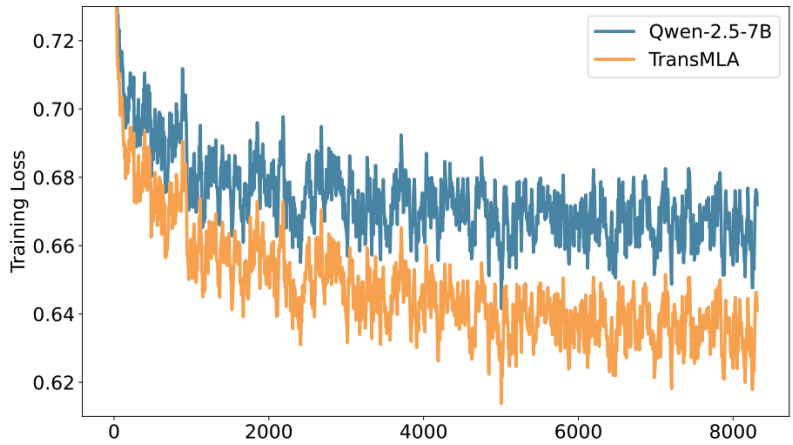
$$W'_K = [\underbrace{W_K^1, ..., W_K^1}_{s=8}, ..., \underbrace{W_K^{n_k}, ..., W_K^{n_k}}_{s=8}] = U \cdot S \cdot V^T = (U \cdot S^{\frac{1}{2}}) \cdot (S^{\frac{1}{2}} \cdot V^T) \equiv W_K^a \cdot W_K^b$$

Finally, we get the MLA form of K in GQA, and we can apply the same steps for V in GQA.

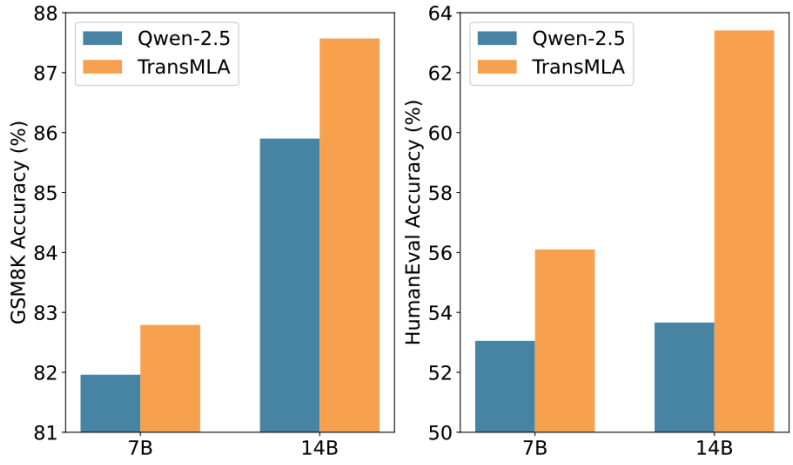
3.1.2 Key Findings and Trends

The paper(Meng et al., 2025)[1] indicates that although GQA can be converted to MLA by SVD decomposition of W_k, but MLA can't be converted into GQA because most columns in W_K^b is linearly independent, even orthogonal. The fact that W_K^b has high rank will lead to $K = XW_K^aW_K^b$ can generate completely different Key heads unlike GQA. As a result, MLA is not representable in GQA, MLA has more diversed K and V heads compare to GQA.

In the experiment part of the paper, a Qwen-2.5 7B and 14B model was converted into MLA form, and the gourp tested the difference in performance and training loss after fine-tuning both model.



(a) Training Loss over Steps



(b) Test Accuracy

Figure 4: Comparison of Training Loss and Accuracy between TransMLA and GQA-based Qwen Model(Meng et al., 2025)[1]

Figure 4 shows a clear trend of decreasing in training loss after Qwen-2.5 model was converted into MLA-based. After 8000 train iterations, MLA-based Qwen model's training loss decreased to about 0.63, and GQA-based Qwen model's training loss is about 0.66. The GSM8K accuracy increased from about 81.9% of GQA-based Qwen-2.5 7B to about 82.7% of MLA-based. And, about 85.9% of GQA-based Qwen-2.5 14B to about 87.6%.

MLA-based Qwen 2.5 performs even better compare to GQA-based in HumanEval test, the accuracy increased from about 53.1% to about 56% for 7B, and the 14B model's accuracy incresed from about 53.5% to 63.5%.

In theory and practice, the paper proves MLA outperforms GQA when KV caches are the same. Moreover, it provides a way to convert GQA-based model into MLA-based model. The paper clearified the principles of all the existing KV caches saving techniques and its own technique, showing the math-behind the methodology.

The DeepSeek V2 paper(Liu et al., 2024)[2] introduced MLA architecture but it did not analyzed the difference in mainstream GQA and the new architecture MLA from math, structure, experiment perspectives. This is the first time the DeepSeek group introduced the reason they developed MLA

and deployed MLA on their own model.
The reason that DeepSeek group didn't compare the performance in MLA and MQA is that in GQA paper(Ainslie et al., 2023)[5] GQA is already proved its performance is better than MQA.

3.2 Comparative Analysis

The primary objective of both GQA and MLA is to break the KV cache bottleneck in LLMs during inference.
GQA divided Q heads into different groups and letting each group share a single K and V head. In contrast, MLA introduced a more complex way that decomposes K and V projection matrix into two small matrices W^a and W^b for both K and V.
In experiment part, GQA was evaluated by summarization, translation, and Q&A tasks. On the other hand, MLA focused on tuning tasks such as math and coding by comparing the difference in Qwen-2.5-7B&14B in GQA-based and MLA-based.
Both approaches successfully reduced inferece latency and KV cache size, GQA outperforms MQA while remaining its performance score close to MHA. However, MLA-based model shows superiority in training loss and test accuracy in math and coding tasks compare to GQA-based model with same condition.

4 Critical Analysis

4.1 Evaluation of Research Quality & Identification of Gaps

In my perspective, both MLA(Meng et al., 2025)[1] and GQA(Ainslie et al., 2023)[5] papers follow a same structure, they introduced other methods for reducing KV caches, explain the math-behind, and provide detailed descriptions of their own methodologies.
I found the MLA paper to be more straightforward. However, it only compare MLA and GQA, without including other methods in the experiment section. Although the GQA paper already showed that MQA performs worse than GQA in its experiment part, I believe it is stll important for the MLA paper to include comparisons with MHA to clearly demonstrate the performance differences. Moreover, just like what they mentioned in future work part in MLA paper, Qwen-2.5-7B&14B are relatively small models, it would be more comprehensive if they provide some experiments on big models such as LLaMA and Mistral(Meng et al., 2025)[1] .
On the ohter hand, the GQA is more comprehensive. It presents a broader overview of the methods available at the time they published the paper for reducing KV cache and includes experimental comparisons between these methods including MHA, making the analysis more complete.
In terms of the impact and significance of the research, GQA becomes the mainstream method for saving KV cache within 2 years they published the paper. However, the potential of MLA architecture is unpredictable. The fact that it allows KV heads to obtain more diversity is a remarkable improvement compare to GQA. The most important impact of MLA paper is that it provides a technique TransMLA to convert GQA-based model to MLA while bring least influence to the model's performance, even enhance the model's performance in terms of math and coding tasks. But, because the MLA paper was published in year 2025, very few models decided to convert their architecture from GQA-based to MLA-based yet.

4.2 Implications

In fact, most inference model mainly working on math and coding tasks, GQA's experiment of Q&A, translation, and summarization does not need inference that much. Therefore, MLA paper showing its capability of working on math and coding tasks bring more options for those of models want to develop inference model for specific math and coding tasks that requires more logic understanding. With the emergence of GPT-o1 and GPT-o3, competitors of OpenAI are likely to shift their research focus more on inferece models. As sequence lengths continue to grow, overcoming the KV cache

bottleneck will become increasingly important. In the future, once this bottleneck is resolved, AI systems may be able to input and process entire coding projects with thousands of lines of code, or even more. Given that MLA has shown superiority in performance in math and coding tasks, it is likely that more inferece models will favor MLA over GQA.

4.3 Limitations

Although this literature review servey provides a complete understand of GQA and MLA, but I did not conduct any experiment that can prove the performance of MLA architecture and the effectiveness of TransMLA techniques on large models. Yet we do not know if TransMLA is repeatable on models other than Qwen-2.5 7B&14B.
Other than that, there are lots of other KV cache saving techniques such as MLKV, PagedAttention, and xKV. This literature review primarily focuses on the mainstream GQA architecture and MLA, but not covering many other KV cache saving techniques with high potentials.

5 Conclusion

This literature survey introduced the math-behind of GQA, MQA, MHA, and MLA, mainly focusing on comparing GQA and MLA, the two techniques that allow models to save KV cache during inference step. We found that GQA outperforms MQA in summarization, translation, and Q&A tasks while keeping its accuracy level close to MHA architecture model. And, MLA-based Qwen-2.5 7B and 14B model outperforms GQA-based Qwen-2.5 7B and 14B in model in math and coding tasks after fine-tuning both models. The technique called TransMLA allows GQA-based model such as Qwen to convert into MLA-based model by using SVD decomposition of K and V heads weight parameter while bringing least influence to the original model.
However, we also observe that the MLA paper lacks of experimental comparisons with MHA and only applied TransMLA on one model. It is still unclear if TransMLA can still enhance model's performance for larger model.
For future research, it would be beneficial to:

- Conduct a comparison analysis between MLA and MHA and other cache-saving techniques in more downstream tasks.
- Explore the capability of TransMLA technique on large models besides Qwen.

6 References

References

[1] Meng, F., Yao, Z., & Zhang, M. (2025). TransMLA: Multi-head Latent Attention Is All You Need. arXiv preprint arXiv:2502.07864.

[2] Liu, A., Feng, B., Wang, B., Wang, B., Liu, B., Zhao, C., ... & Xu, Z. (2024). Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. arXiv preprint arXiv:2405.04434.

[3] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.

[4] Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., ... & Piao, Y. (2024). Deepseek-v3 technical report. arXiv preprint arXiv:2412.19437.

[5] Ainslie, J., Lee-Thorp, J., De Jong, M., Zemlyanskiy, Y., Lebrón, F., & Sanghai, S. (2023). Gqa: Training generalized multi-query transformer models from multi-head checkpoints. arXiv preprint arXiv:2305.13245.