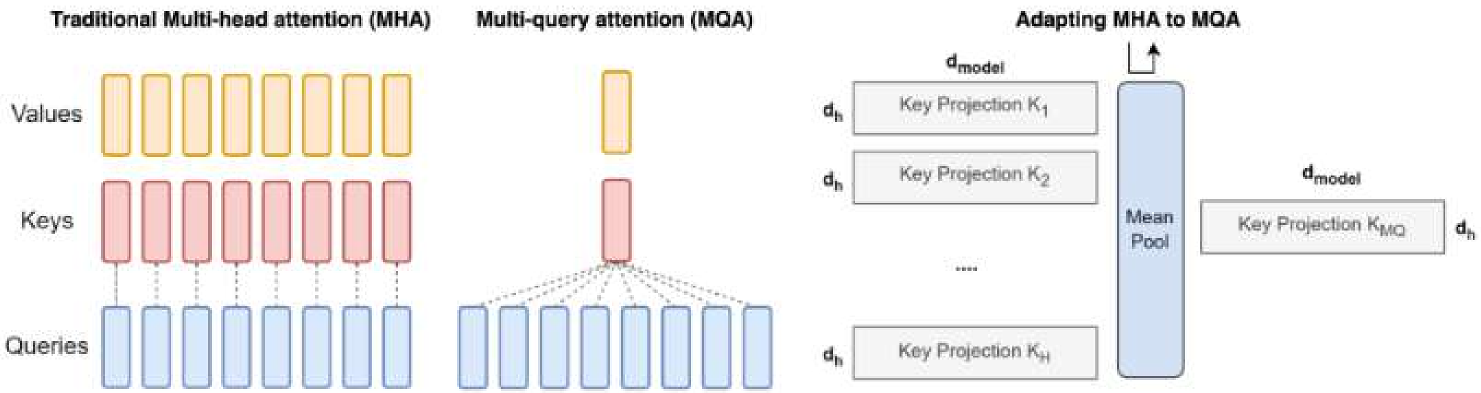
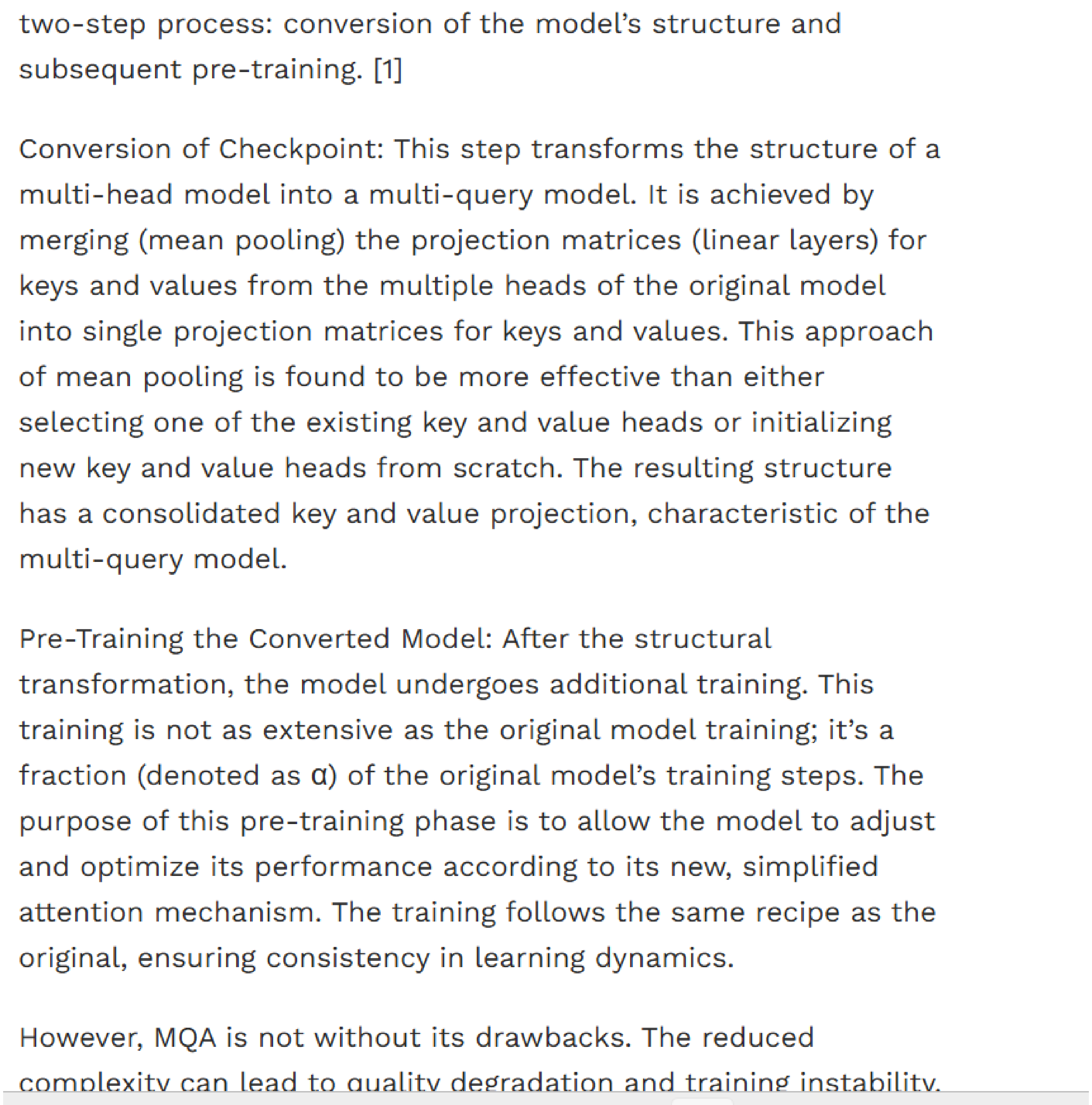
Emergence of Multi-Query Attention (MQA)

Multi-query attention (MQA) emerged as a solution to mitigate this bottleneck. The idea is simple yet effective: use multiple query heads but only a single key and value head. This approach significantly reduces the memory load, enhancing inference speed. It has been employed in multiple large-scale models such as PaLM, StarCoder, and Falcon.

In multi-query attention, we average the heads for keys and values so that all query heads share the same key and value head. This is achieved by replicating the mean-pooled "head" H times, where H is the number of query heads.



Left – Multihead attention, Middle – MultiQuery Attention, Right – Converting existing MHA checkpoint to MQA (Source – https://arxiv.org/pdf/2305.13245.pdf)

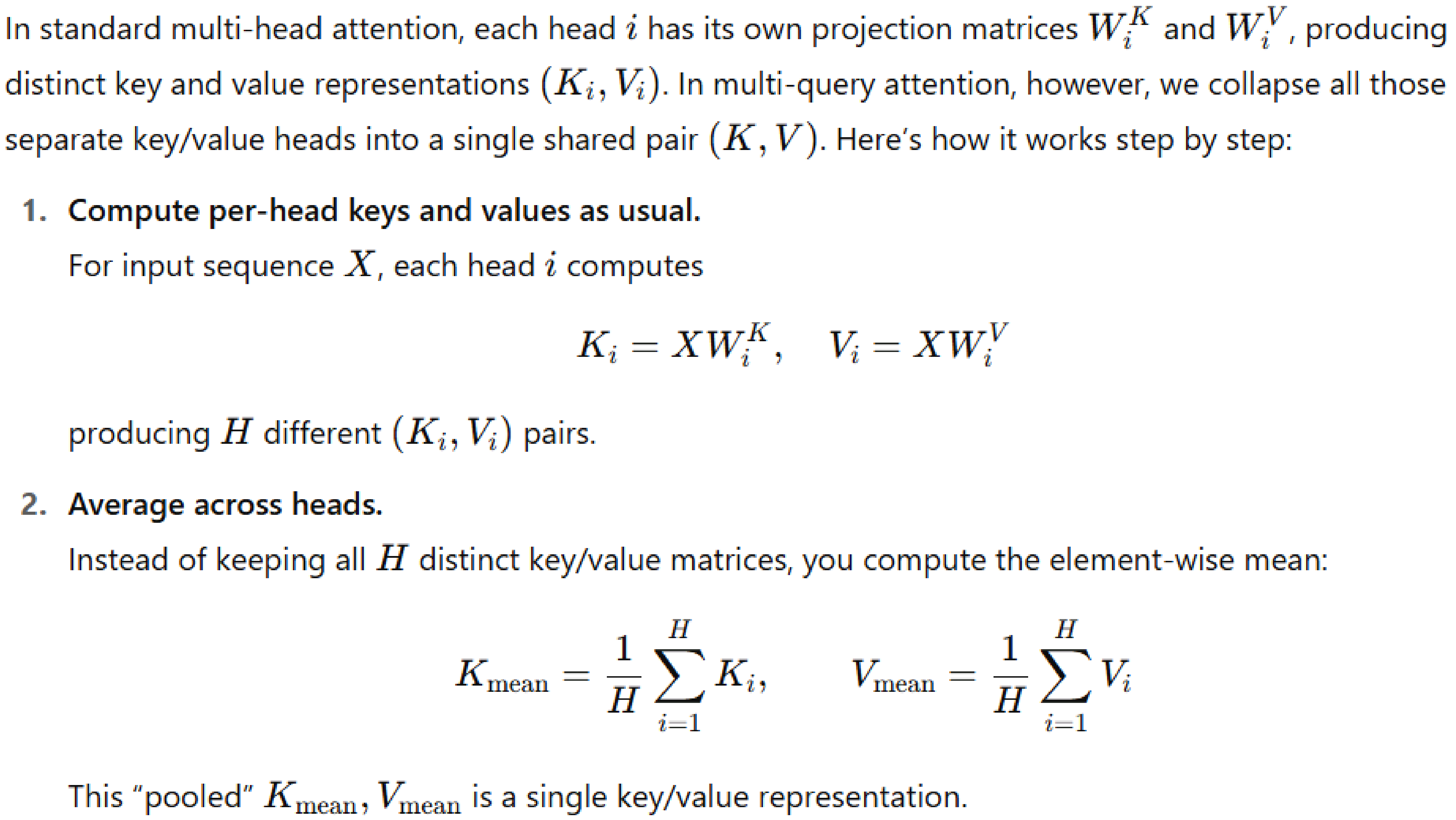


two-step process: conversion of the model’s structure and subsequent pre-training. [1]

Conversion of Checkpoint: This step transforms the structure of a multi-head model into a multi-query model. It is achieved by merging (mean pooling) the projection matrices (linear layers) for keys and values from the multiple heads of the original model into single projection matrices for keys and values. This approach of mean pooling is found to be more effective than either selecting one of the existing key and value heads or initializing new key and value heads from scratch. The resulting structure has a consolidated key and value projection, characteristic of the multi-query model.

Pre-Training the Converted Model: After the structural transformation, the model undergoes additional training. This training is not as extensive as the original model training; it’s a fraction (denoted as α) of the original model’s training steps. The purpose of this pre-training phase is to allow the model to adjust and optimize its performance according to its new, simplified attention mechanism. The training follows the same recipe as the original, ensuring consistency in learning dynamics.

However, MQA is not without its drawbacks. The reduced complexity can lead to quality degradation and training instability.



In standard multi-head attention, each head i has its own projection matrices W_i^K and W_i^V, producing distinct key and value representations (K_i, V_i). In multi-query attention, however, we collapse all those separate key/value heads into a single shared pair (K, V). Here's how it works step by step:

1. **Compute per-head keys and values as usual.**

   For input sequence X, each head i computes

   $$K_i = XW_i^K, \quad V_i = XW_i^V$$

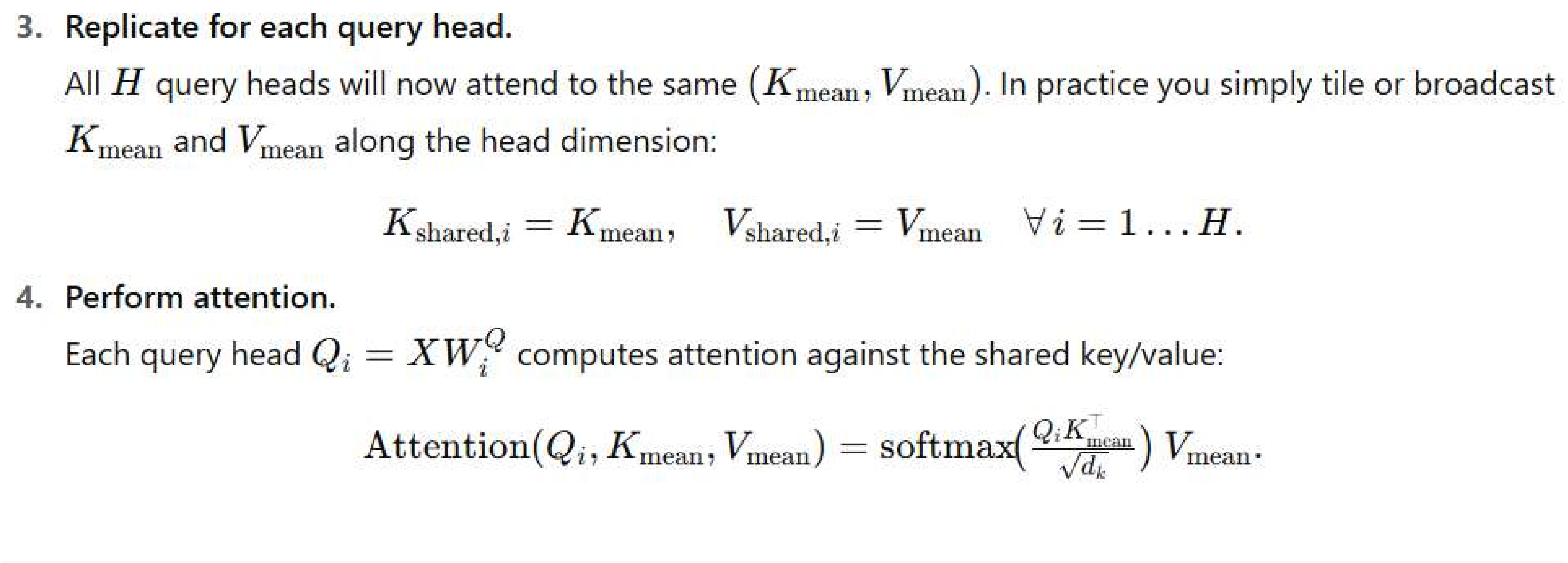
   producing H different (K_i, V_i) pairs.

2. **Average across heads.**

   Instead of keeping all H distinct key/value matrices, you compute the element-wise mean:

   $$K_{\text{mean}} = \frac{1}{H}\sum_{i=1}^{H} K_i, \qquad V_{\text{mean}} = \frac{1}{H}\sum_{i=1}^{H} V_i$$

   This "pooled" $K_{\text{mean}}, V_{\text{mean}}$ is a single key/value representation.



3. **Replicate for each query head.**

   All H query heads will now attend to the same $(K_{\text{mean}}, V_{\text{mean}})$. In practice you simply tile or broadcast K_{mean} and V_{mean} along the head dimension:

$$K_{\text{shared},i} = K_{\text{mean}}, \quad V_{\text{shared},i} = V_{\text{mean}} \quad \forall\, i = 1 \dots H.$$

4. **Perform attention.**

   Each query head $Q_i = XW_i^Q$ computes attention against the shared key/value:

$$\text{Attention}(Q_i, K_{\text{mean}}, V_{\text{mean}}) = \text{softmax}\!\Big(\frac{Q_i K_{\text{mean}}^\top}{\sqrt{d_k}}\Big)\, V_{\text{mean}}.$$

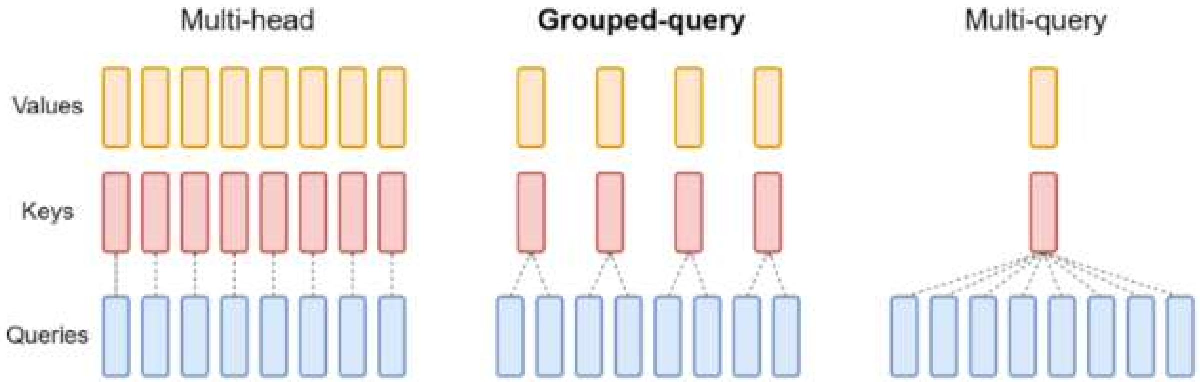
Grouped Query Attention

Grouped-query attention (GQA) is a simple approach that blends elements of multi-head attention (MHA) and multi-query attention (MQA) to create a more efficient attention mechanism. The mathematical framework of GQA can be understood as follows:

Division into Groups: In GQA, the query heads (Q) from a traditional multi-head model are divided into G groups. Each group is assigned a single key (K) and value (V) head. This configuration is denoted as GQA-G, where G represents the number of groups.

Special Cases of GQA:

- GQA-1 = MQA: With only one group (G = 1), GQA becomes equivalent to MQA, as there’s only a single key and value head for all query heads.
- GQA-H = MHA: When the number of groups equals the number of heads (G = H), GQA behaves like traditional MHA, with each query head having its unique key and value head.



Difference between MHA, GQA, and MQA (Source – https://arxiv.org/pdf/2305.13245.pdf)

We mean-pool the key and value projection matrices of the original heads within each group to convert a multi-head model into a GQA model. This technique averages the projection matrices of each head in a group, resulting in a single key and value projection for that group.

By utilizing GQA, the model maintains a balance between MHA quality and MQA speed. Because there are fewer key-value pairs, memory bandwidth and data loading needs are minimized. The choice of G presents a trade-off: more groups (closer to MHA) result in higher quality but slower performance, whereas fewer groups (near to MQA) boost speed at the risk of sacrificing quality. Furthermore, as the model size grows, GQA allows for a proportional decrease in memory bandwidth and model capacity, corresponding with the model's scale. In contrast, for bigger models, the reduction to a single key and value head can be unduly severe in MQA.