Virtual Head Attention

Xibo Ding[1,2]*, **Guoxia Wang**[1]*, **Jinle Zeng**[1],
Jiabin Yang[1], **Sirui Tao**[2], **Chao Yang**[2], **Dianhai Yu**[1,2]†
[1]Baidu Inc. [2]Peking University
{dingxibo, wangguoxia, yudianhai}@baidu.com

Abstract

The design of attention mechanisms in long-context large language models faces an "impossible triangle" among KV cache size, training and prefill FLOPs, and model quality. Grouped-Query Attention (GQA) achieves strong quality but incurs high KV cache overhead; alternative approaches such as Multi-head Latent Attention (MLA) and Multi-matrix Factorization Attention (MFA) can compress the cache, but MLA suffers from quality loss while MFA substantially increases training and prefill compute. We propose Virtual Head Attention (VHA), which simultaneously improves all three vertices through two lightweight linear operations: Q Premix applies a near-identity transformation to queries within each KV group to recover virtual head diversity from halved physical query heads, and Linear Postmix fuses inter-head features via a low-rank $I + AB^\top$ residual structure. Since VHA introduces no nonlinear operations, Premix can be folded into the query projection and Postmix into the output projection at inference time, reducing the model to a standard GQA-2 architecture that directly reuses all existing GQA inference optimizations. With only two KV heads, the group size in large models readily reaches 64, fully exploiting Tensor Core parallelism during decoding. Across four scales (0.6B, 1.7B, 4B dense, and 8B-A1B MoE) with 4K-context 50B-token pretraining, VHA consistently surpasses the standard GQA baseline at every scale ($+0.05\%$ to $+0.81\%$), while reducing KV cache by $4\times$ vs. GQA-8 at dense scales and $2\times$ vs. GQA-4 at the MoE scale, and lowering training and prefill FLOPs by 7.9% (1.7B, S=4096). On an industrial 30B-A3B MoE model trained on 500B tokens at 8K context, VHA outperforms both MLA and GQA-4 by $+0.70$ / $+2.60$ % on a 12-benchmark base-model evaluation suite; in a separate matched 10B-A1B MoE comparison after long-context mid-training, VHA attains an average RULER score of 84.4 over 8K–128K versus 73.0 for GQA, while using two rather than four KV heads in the full-attention layers. The source code is available at `https://github.com/PaddlePaddle/VHA`.

1 Introduction

Large language models are rapidly scaling toward longer contexts, with context windows expanding from 2K tokens to 128K and beyond [Grattafiori et al., 2024, Yang et al., 2024]. As sequence lengths grow, the attention mechanism [Vaswani et al., 2017] becomes a central efficiency bottleneck in both inference and training. During inference, the key-value (KV) cache grows linearly with sequence length, consuming substantial GPU memory and constraining both serving concurrency and generation latency. During training and prefill, the FLOPs of attention computation scale quadratically with sequence length. Since the design of the attention mechanism jointly determines KV cache size, computational cost, and model expressiveness, it is a critical factor governing the overall efficiency and quality of large language models.

*Equal contribution. Work done at Baidu Inc.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

The design of existing attention mechanisms faces three mutually constraining objectives, forming an "impossible triangle": KV cache size, training and prefill FLOPs, and model quality. Consider the widely adopted Grouped-Query Attention (GQA) [Ainslie et al., 2023]: GQA with 8 KV heads (GQA-8) achieves strong quality, but incurs a KV cache of 4 KB per layer per token (in BF16). Directly reducing the number of KV heads to 2 (GQA-2) compresses the cache to 1 KB, but dramatically enlarges the group size so that far more query heads must share the same key-value pairs, causing attention diversity to drop sharply and resulting in a 1.00% quality degradation at the 1.7B scale. Multi-head Latent Attention (MLA) [Liu et al., 2024] applies low-rank compression to reduce KV cache to 1.12 KB, but yields the lowest quality across multiple model scales. Multi-matrix Factorization Attention (MFA) [Hu et al., 2025] reduces the cache to 1 KB by enlarging the head dimension, achieving quality close to GQA-8, but at the cost of a 49.8% increase in training and prefill FLOPs (1.7B, S=4096). No existing approach achieves a satisfactory balance across all three dimensions.

We propose Virtual Head Attention (VHA), which simultaneously improves all three vertices of the impossible triangle through two lightweight linear operations. The core idea of VHA is to halve the number of physical query heads while setting the number of KV heads to only 2. Q Premix then applies a near-identity linear transformation to queries within each KV group, expanding the reduced physical query heads into virtual query heads to recover attention diversity. Linear Postmix applies a low-rank identity-residual transformation ($I + AB^\top$, r=$H'_q/2$, e.g., r=4 at 1.7B) on the attention output, fusing cross-group inter-head features. Since VHA introduces no nonlinear operations, Premix can be directly folded into the query projection matrix and Postmix into the output projection matrix at inference time, reducing the model to a standard GQA-2 architecture that reuses all existing GQA inference optimizations in mainstream serving frameworks such as vLLM [Kwon et al., 2023] and SGLang [Zheng et al., 2024]. Moreover, with only 2 KV heads, the group size in large models readily reaches 64, fully exploiting Tensor Core parallelism during the decoding phase.

Across four model scales (0.6B, 1.7B, 4B dense, and 8B-A1B MoE) with 4K-context 50B-token pretraining, VHA consistently surpasses the standard GQA baseline at every scale ($+0.05\%$ to $+0.81\%$), while reducing KV cache by $4\times$ vs. GQA-8 at dense scales and $2\times$ vs. GQA-4 at the MoE scale, and lowering training and prefill FLOPs by 7.9% (1.7B, S=4096). On an industrial 30B-A3B MoE model trained on 500B tokens at 8K context, VHA outperforms both MLA and GQA-4 by $+0.70$ / $+2.60$ % on a 12-benchmark base-model evaluation suite, demonstrating scalability to industrial model scale. A separate matched 10B-A1B MoE comparison further extends this advantage to 128K context under more aggressive KV compression. Our main contributions are as follows:

1. We propose VHA, employing Q Premix and Linear Postmix, two lightweight linear operations that simultaneously reduce KV cache and training/prefill FLOPs while improving model quality.
2. VHA uses only linear operations and can be fully absorbed into a standard GQA-2 model at inference time, reusing existing serving frameworks such as vLLM and SGLang with zero additional overhead.
3. Direct inspection of the trained Premix and Postmix matrices reveals two complementary structures (group-conditioned query specialization and structured low-rank cross-head communication) that jointly close the gap of the doubly-compressed configuration; component ablations confirm that neither module is sufficient on its own.
4. Comprehensive evaluations across five scales (0.6B to 30B-A3B MoE) against MHA, GQA, MFA, and MLA establish VHA as the best attention mechanism that is Pareto-optimal on KV cache, training/prefill FLOPs, and downstream quality simultaneously; matched long-context evaluations through 128K further validate this advantage under aggressive KV compression.

2 Related Work

2.1 KV Cache Efficient Attention

Standard Multi-Head Attention (MHA) [Vaswani et al., 2017] maintains independent key and value projections per head, causing the KV cache to grow linearly with the number of heads. Multi-Query Attention (MQA) [Shazeer, 2019] shares a single KV head across all query heads, drastically reducing the cache at the cost of quality. Grouped-Query Attention (GQA) [Ainslie et al., 2023] interpolates

between MHA and MQA by allowing multiple query heads to share a group of KV heads, and has been widely adopted in LLaMA 2/3 [Touvron et al., 2023, Grattafiori et al., 2024] and Qwen2 [Yang et al., 2024].

Several recent designs further compress the KV cache. Multi-head Latent Attention (MLA) [Liu et al., 2024] projects KV into a shared low-rank latent space and absorbs the up-projection matrices into Q and O at inference, but requires custom kernels and a non-standard inference pipeline. Multi-matrix Factorization Attention (MFA) [Hu et al., 2025] enlarges the head dimension while reducing the number of KV heads, preserving model capacity under a tight cache budget but substantially increasing training and prefill FLOPs. Tensor Product Attention (TPA) [Zhang et al., 2025] decomposes QKV into context-dependent tensor products. Grouped-head Latent Attention (GTA) [Sun et al., 2025] extends MLA's global latent space to a group-wise design. Simulated Attention Score (SAS) [Zheng et al., 2025] projects a small number of low-dimensional KV heads into higher-dimensional spaces to simulate more attention heads, an idea related to VHA's virtual head expansion; however, SAS operates on the KV side whereas VHA virtualizes on the query side. Low-Rank Key-Value Attention (LRKV) [O'Neill et al., 2026] shares a full-rank KV basis across heads and restores head-specific flexibility through low-rank residuals, a form reminiscent of VHA's Postmix ($I + AB^\top$); yet LRKV acts in KV space and its additional structure remains at inference time. Cross-Layer Attention (CLA) [Brandon et al., 2024] and YOCO [Sun et al., 2024] reduce memory by sharing or reusing KV caches across layers.

Unlike all of the above, VHA employs a richer training-time parameterization that merges exactly back into standard GQA at deployment, requiring no custom kernels or specialized inference procedures.

2.2 Inter-Head Communication and Virtual Heads

Talking-Heads Attention [Shazeer et al., 2020] applies learned linear projections across heads before and after softmax, pioneering inter-head communication. Cordonnier et al. [2020] analyze multi-head attention from a collaborative perspective, revealing substantial redundancy among heads. More recently, Explicit Multi-head Attention (MEA) [Peng et al., 2026] introduces a head-level linear composition module for explicit cross-head interaction, and Interleaved Head Attention [Duvvuri et al., 2026] interleaves head representations to enable information exchange. In contrast to these methods, VHA does not mix across heads in the attention weight space; instead, it virtualizes in the query space via Premix and fuses inter-head features in the output space via a low-rank Postmix, with all operations being strictly linear and fully absorbable at inference.

2.3 Reparameterization

RepVGG [Ding et al., 2021] uses a multi-branch training architecture ($3\times3 + 1\times1 +$ identity) that merges into a single convolution at inference, demonstrating that decoupled training-time parameterization provides implicit regularization at zero inference cost. LoRA [Hu et al., 2022] performs parameter-efficient fine-tuning via low-rank decomposition ($W + BA$). VHA extends this reparameterization paradigm from a single weight matrix to the head dimension of attention: Premix structurally factorizes the per-virtual-head query projections ($\widetilde{W}_Q^{(i,g)} = W_Q^{(i)} W_g^{\text{premix}}$), while Postmix introduces a low-rank identity-residual mixing on the head axis ($I + AB^\top$). Both modules are mathematically equivalent in the forward pass to a standard GQA configuration, yet the decoupled parameterization yields distinct optimization dynamics during training and merges back into a standard GQA-2 architecture at inference with zero overhead.

3 Method: Virtual Head Attention

3.1 Preliminaries

Grouped-Query Attention (GQA) [Ainslie et al., 2023] partitions H_q query heads into H_{kv} groups, where heads within each group share a common key-value pair. Given input $X \in \mathbb{R}^{S\times h}$, GQA computes $Q = XW_Q$, $K = XW_K$, $V = XW_V$, where $W_Q \in \mathbb{R}^{h\times H_q d}$, $W_K \in \mathbb{R}^{h\times H_{kv} d}$, $W_V \in \mathbb{R}^{h\times H_{kv} d}$. Each query head attends to its group's key-value pair via scaled dot-product attention [Vaswani et al., 2017], and all head outputs are concatenated and projected by $W_O \in \mathbb{R}^{H_q d\times h}$. During inference, K and V are cached, consuming $4H_{kv}d$ bytes per layer per token in BF16. Reducing H_{kv} compresses the cache but enlarges the group size $G = H_q/H_{kv}$, forcing more query heads to share the same key-value pair and degrading attention diversity.

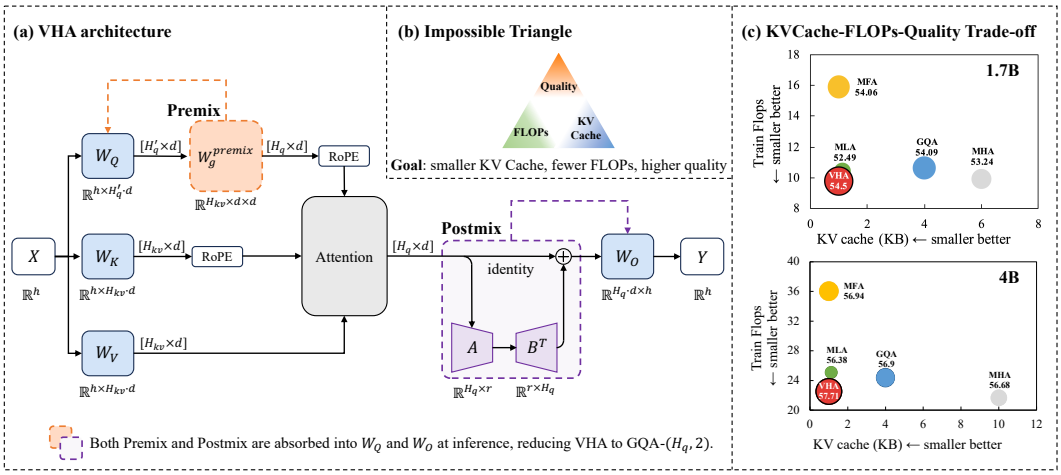

Figure 1: **Overview of Virtual Head Attention.** (a) VHA improves the trade-off among KV cache size, training/prefill FLOPs, and model quality by combining Q Premix and Linear Postmix during training, while absorbing both linear modules into standard GQA projections at inference. (b) Schematic of the KV cache–FLOPs–quality impossible triangle faced by existing attention designs. (c) Method comparison at 1.7B and 4B scales, plotting KV cache size against training FLOPs, with bubble area proportional to 8-task average accuracy.

3.2 Overall Design

VHA halves the number of physical query heads while setting $H_{kv} = 2$, then recovers model expressiveness through two lightweight linear operations. Specifically, $H'_q = H_q/2$ physical query heads and $H_{kv} = 2$ KV heads yield $H_v = H_{kv} \times H'_q = H_q$ virtual heads (matching the original GQA query head count), with $W_O \in \mathbb{R}^{H_v d \times h}$ identical in size to the original GQA output projection. The resulting KV cache is $4 \times 2 \times d$ bytes per layer per token, $4\times$ smaller than GQA-8 at dense scales and $2\times$ smaller than GQA-4 at MoE scales.

On top of this configuration, VHA introduces **Q Premix**, which expands H'_q physical query heads into H_v virtual query heads to restore attention diversity, and **Linear Postmix**, which applies a low-rank identity-residual transformation on the attention output to fuse inter-head features.

Inference absorption principle. VHA uses only linear operations (no nonlinear activations), so that at inference time Premix can be folded into W_Q and Postmix into W_O, reducing the entire model to a standard GQA-2 architecture that directly reuses all existing GQA optimizations in frameworks such as vLLM [Kwon et al., 2023] and SGLang [Zheng et al., 2024].

3.3 Q Premix: Virtual Query Head Expansion

Q Premix applies a near-identity linear transformation to physical query heads within each KV group, expanding them into virtual query heads. For KV group $g \in \{1, \ldots, H_{kv}\}$, each physical query head $Q_i \in \mathbb{R}^{S\times d}$, $i \in \{1, \ldots, H'_q\}$, is transformed as:

$$\tilde{Q}_{i,g} = Q_i \, W_g^{\text{premix}}, \quad W_g^{\text{premix}} \in \mathbb{R}^{d\times d} \tag{1}$$

where W_g^{premix} is a per-group transformation matrix. We initialize $W_g^{\text{premix}} = I + \alpha \cdot \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, with a small scaling factor α. This near-identity initialization ensures that virtual query heads start as slight perturbations of the physical query heads, preserving training stability at the early stage. As training progresses, each W_g^{premix} gradually deviates from the identity to produce increasingly diverse attention patterns across virtual heads. Through this operation, H'_q physical query heads are expanded into $H_v = H_{kv} \times H'_q$ virtual query heads.

Structured factorization view. Premix is mathematically equivalent in the forward pass to a standard H_v-head GQA whose query projections are structurally factorized. Specifically, defining

$$\widetilde{W}_Q^{(i,g)} = W_Q^{(i)} \, W_g^{\text{premix}} \in \mathbb{R}^{h\times d}, \tag{2}$$

the entire Premix-then-attend computation reduces, for any input X and any parameter values, to

$$\mathrm{softmax}\Big(\tilde{Q}_{i,g}K_g^\top/\sqrt{d}\Big)\,V_g \;=\; \mathrm{softmax}\Big(X\widetilde{W}_Q^{(i,g)}K_g^\top/\sqrt{d}\Big)\,V_g. \tag{3}$$

Thus VHA is a standard GQA with $H_v = H_{kv}H'_q$ virtual query heads and H_{kv} KV heads, where the per-virtual-head query projections are constrained to lie on the structured low-dimensional manifold $\{W_Q^{(i)}W_g^{\mathrm{premix}}\}$.

Why this factorization is useful. The structured parameterization brings two benefits over an unconstrained H_v-head GQA. First, it is parameter- and compute-efficient: the Q path uses $H'_q hd + H_{kv}d^2$ parameters and a comparable number of FLOPs, versus $H_v hd$ for a fully decoupled H_v-head Q projection. At the 1.7B scale, this yields a roughly 50% reduction in Q-projection parameters and FLOPs (the primary source of VHA's training-FLOPs reduction). Second, although the forward pass is algebraically equivalent to training a single $\widetilde{W}_Q^{(i,g)}$, the gradients of $W_Q^{(i)}$ and W_g^{premix} are coupled across all groups and physical heads, yielding a different optimization landscape and providing implicit regularization analogous to LoRA [Hu et al., 2022] and RepVGG [Ding et al., 2021] (see Appendix G for a formal analysis).

Inference-time absorption. Once training is complete and parameters are frozen, for each virtual head (i, g) we precompute $\widetilde{W}_Q^{(i,g)} = W_Q^{(i)} \cdot W_g^{\mathrm{premix}}$, after which VHA reduces to a standard GQA-2 with H_v independent query projections and $H_{kv} = 2$ KV heads.

3.4 Linear Postmix: Low-Rank Identity-Residual Mixing

While Premix restores virtual head diversity within each KV group, the outputs across different groups still lack cross-group information exchange. Linear Postmix addresses this through a low-rank identity-residual structure.

Let $Y \in \mathbb{R}^{S\times H_v\times d}$ denote the attention output. Postmix computes:

$$Y' = Y + BA^\top Y = \big(I + BA^\top\big)\,Y \tag{4}$$

where $A, B \in \mathbb{R}^{H_v\times r}$ and r is the bottleneck rank (default $r = H'_q/2$, e.g., r=4 at 1.7B). The operation acts along the head dimension independently for each token and head-dimension channel, applying the transformation $I + BA^\top \in \mathbb{R}^{H_v\times H_v}$ to the head axis.

Initialization. A is initialized with small random values and B with zeros, ensuring $BA^\top = 0$ at the start of training so that Postmix acts as the identity and does not disturb the initial training dynamics. As training progresses, B gradually learns effective cross-group mixing patterns.

Linear redundancy and reparameterization benefit. Mathematically, Postmix can be absorbed into W_O. Using the row-major Kronecker identity $\mathrm{vec}_{\mathrm{row}}(MY)^\top = \mathrm{vec}_{\mathrm{row}}(Y)^\top(M^\top \otimes I_d)$ [Roger and Charles, 1994]:

$$\mathrm{vec}_{\mathrm{row}}(Y)^\top\big((I_{H_v} + AB^\top)\otimes I_d\big)\,W_O = \mathrm{vec}_{\mathrm{row}}(Y)^\top W'_O \tag{5}$$

where $W'_O = \big((I_{H_v} + AB^\top)\otimes I_d\big)\,W_O \in \mathbb{R}^{H_v d\times h}$. Thus, linear Postmix is theoretically redundant (see Appendix H for the complete derivation). However, ablation experiments (Section 4.4) show that adding Postmix on top of Premix yields a +1.19 % improvement at 1.7B, analogous to the implicit regularization observed in RepVGG [Ding et al., 2021] and the low-rank decomposition in LoRA [Hu et al., 2022]: the separated gradient paths of A and B provide a different optimization landscape, while the low-rank constraint ($r \ll H_v$) acts as an additional regularizer preventing cross-head mixing from overfitting.

Inference-time absorption. At inference, we precompute $W'_O = \big((I_{H_v} + AB^\top)\otimes I_d\big)W_O$, after which VHA reduces to standard GQA-2. Algorithm 1 (Appendix F) summarizes the complete training forward pass and the one-time inference-absorption procedure.

3.5 Efficiency Analysis

Training and prefill FLOPs. By halving the physical H_q, VHA directly reduces Q projection FLOPs. The additional cost of Premix ($2H_{kv}H'_q d^2$) and Postmix ($4H_v rd$) is negligible compared to the FLOPs saved by halving the Q projection. Table 5 quantifies this trade-off in detail across all four model scales (S=4096); VHA emerges as the best Pareto-optimal point on KV cache and FLOPs at every scale (Section 4.5).

Table 1: **Pretraining accuracy across four model scales.** We report zero-shot accuracy (%) on eight standard multiple-choice benchmarks. CACHE is KV cache per layer per token in BF16 (KB). **Bold** marks the best per scale per column; ties are bolded jointly. VHA achieves the highest average accuracy at every scale while using the smallest KV cache. Per-method full configurations and benchmark citations are in Appendix D; per-method training-FLOPs comparison is in Table 5.

Scale	Attn	Q-K-V	Cache	ARC-C	BoolQ	HSwag	OBQA	PIQA	Wino	SciQ	SIQA	**Avg**
0.6B	MHA	12-12-12	6.00	29.10	60.40	44.88	33.20	68.66	51.85	74.50	39.15	50.22
	GQA	16-8-8	4.00	29.86	58.01	44.77	**35.80**	68.17	52.33	75.70	38.69	50.42
	MLA	14-14-14	1.12	29.27	55.96	43.20	33.80	68.23	50.67	**76.40**	38.59	49.52
	MFA	17-1-1	**1.00**	30.80	60.18	**45.47**	32.80	**69.04**	**54.70**	73.20	39.05	50.66
	VHA	8-2-2	**1.00**	**31.91**	**61.19**	45.26	35.60	68.44	50.99	75.60	**39.36**	**51.04**
1.7B	MHA	12-12-12	6.00	35.24	58.53	51.86	35.20	71.33	54.06	80.30	39.36	53.24
	GQA	16-8-8	4.00	35.15	**62.11**	51.98	35.60	71.71	**55.25**	81.10	39.82	54.09
	MLA	12-12-12	1.12	31.57	61.31	49.22	36.20	70.51	54.22	77.60	39.30	52.49
	MFA	19-1-1	**1.00**	34.47	59.45	**52.16**	37.80	**71.82**	54.85	81.50	40.43	54.06
	VHA	8-2-2	**1.00**	**35.92**	60.64	51.71	**39.00**	71.38	54.70	**81.70**	**40.94**	**54.50**
4B	MHA	20-20-20	10.00	38.31	60.21	57.52	39.60	73.88	56.99	**85.40**	41.56	56.68
	GQA	32-8-8	4.00	**39.16**	62.32	58.39	39.40	73.83	57.06	83.80	41.20	56.90
	MLA	28-28-28	1.12	37.20	61.65	56.55	38.80	73.29	56.75	84.90	41.86	56.38
	MFA	33-1-1	**1.00**	38.40	62.20	57.67	**39.80**	73.88	57.70	83.60	42.27	56.94
	VHA	16-2-2	**1.00**	38.91	**64.74**	**58.67**	39.20	**74.43**	**58.80**	84.60	**42.34**	**57.71**
8B-A1B	MHA	20-20-20	10.00	35.67	59.63	53.98	37.40	72.25	53.04	**82.70**	42.12	54.60
	GQA	32-4-4	2.00	34.73	61.53	54.03	**38.60**	71.55	55.09	82.60	40.74	54.86
	MLA	28-28-28	1.12	33.11	60.86	52.29	36.60	72.63	53.28	82.00	39.56	53.79
	MFA	33-1-1	**1.00**	35.07	**62.48**	**54.08**	36.60	**72.80**	**55.17**	81.80	40.48	54.81
	VHA	16-2-2	**1.00**	**36.69**	62.17	53.78	37.00	71.71	54.06	82.40	**41.45**	**54.91**

Decoding inference. Autoregressive decoding is memory-bound [Pope et al., 2023], so KV cache size directly governs latency and serving concurrency. With KV cache reduced by $4\times$ vs. GQA-8 (dense) or $2\times$ vs. GQA-4 (MoE), VHA achieves a corresponding 2–$4\times$ theoretical speedup in memory-bound decoding. Since Premix and Postmix are fully absorbed at inference, VHA's decoding cost is identical to standard GQA-2.

4 Experiments

4.1 Experimental Setup

Baselines. We compare VHA with MHA, GQA, MFA, and MLA across all scales. Following common practice, GQA uses 8 KV heads at dense scales (0.6B/1.7B/4B) and 4 KV heads at MoE scales (8B-A1B and 30B-A3B). VHA halves the physical query heads relative to GQA and uses H_{kv}=2, yielding $4\times$ KV cache compression at dense scales and $2\times$ at MoE scales.

Small-scale pretraining. We pretrain Qwen3-architecture models at four scales (dense at 0.6B, 1.7B, and 4B parameters, plus an 8B-A1B MoE) on 50B tokens of FineWeb-Edu with BF16 mixed precision and a 4K context length. We report zero-shot accuracy on eight standard benchmarks.

Industrial-scale pretraining. We pretrain an ERNIE-style 30B-A3B MoE on an internal multi-domain pretraining corpus (web text, books, code, and multilingual data) with FP8 mixed precision on 500B tokens at 8K context. We evaluate on 12 standard base-model benchmarks spanning general knowledge, math & STEM, coding, and multilingual capabilities. Full model configurations, benchmark details, and training hyperparameters are in Appendix D.

Long-context mid-training. We additionally compare VHA and GQA on a matched MiMo-2.5-Flash-style 10B-A1B MoE backbone that interleaves full and sliding-window attention at a 1:3 ratio. The two models share the backbone, data, optimization, and sliding-window layers; only the full-attention layers differ, using two KV heads for VHA and four for GQA. We evaluate through 128K, with the stage-wise recipe and benchmark definitions reported in Appendix A.

Table 2: **Recovering from a doubly-compressed shape (1.7B).** GQA 16-8-8 is the standard GQA-8 baseline; GQA 16-2-2 reduces only H_{kv} to obtain a 1 KB KV cache; GQA 8-2-2 additionally halves the physical query heads to match VHA's shape but lacks Premix and Postmix. Δ is the gap to the GQA 16-8-8 baseline.

Method	Cache	ARC-C	BoolQ	HSwag	OBQA	PIQA	Wino	SciQ	SIQA	**Avg**	Δ
GQA 16-8-8	4.00	35.15	**62.11**	**51.98**	35.60	**71.71**	**55.25**	81.10	39.82	54.09	—
GQA 16-2-2	**1.00**	33.19	58.96	50.92	36.00	71.27	54.22	79.60	40.53	53.09	-1.00
GQA 8-2-2	**1.00**	33.02	59.11	49.98	34.00	71.44	53.28	77.80	40.43	52.38	-1.71
VHA 8-2-2	**1.00**	**35.92**	60.64	51.71	**39.00**	71.38	54.70	**81.70**	**40.94**	**54.50**	$+0.41$

Table 3: **Pretraining accuracy at the 30B-A3B MoE scale.** We report accuracy (%) averaged within four task categories (General, Math & STEM, Coding, Multilingual) over 12 base-model benchmarks; AVG is the macro-average over the four task categories. CACHE is KV cache per layer per token in BF16 (KB). **Bold** marks the best per column. Per-benchmark scores and benchmark citations are in Appendix C.

Method	Q-K-V	Cache	General	Math & STEM	Coding	Multilingual	**Avg**
GQA	48-4-4	2.00	52.86	43.09	51.85	43.26	47.76
MLA	28-28-28	1.12	55.57	43.49	**54.37**	45.20	49.66
VHA	24-2-2	**1.00**	**56.25**	**46.40**	53.23	**45.56**	**50.36**

4.2 Main Results

Cross-scale consistency. VHA achieves the highest average accuracy at every scale, surpassing the GQA baseline by $+0.05\%$ to $+0.81\%$ while using the smallest KV cache: 1.00 KB per layer per token, tied with MFA, $4\times$ smaller than GQA-8 at dense scales and $2\times$ smaller than GQA-4 at the MoE scale. The advantage holds across both dense and MoE architectures. A separate matched-harness study at 0.6B confirms the same ordering across repeated runs, with VHA reaching 50.40 ± 0.07 (n=4) versus GQA's 50.00 ± 0.34 (n=3); full results are in Appendix B.

Trade-offs against each baseline. At equal KV cache (1.00 KB), VHA dominates MFA at every scale ($+0.10\%$ to $+0.77\%$) while reducing training and prefill FLOPs by 7.9%, compared with MFA's $+49.8\%$ increase at the 1.7B scale (S=4096; Table 5). Full-cache MHA falls below GQA-8 at every dense scale despite using a $\geq 1.5\times$ larger cache, confirming that grouping the queries is essentially free. MLA underperforms at every scale, suggesting that latent KV compression is less effective than head-level virtualization at the 50B-token training budget used here. Per-method training loss curves at all four scales are reported in Appendix E.

Recovering from a doubly-compressed shape. Reducing GQA's KV head count from 8 to 2 (GQA 16-2-2) costs -1.00% in average accuracy at the 1.7B scale; further halving the physical query heads to match VHA's shape (GQA 8-2-2) costs another -0.71%, totaling -1.71% vs. the GQA-8 baseline. With the same (8, 2, 2) shape, VHA recovers all of this loss and exceeds the original baseline by $+0.41\%$, despite using only 25% of GQA-8's KV cache and 50% of its physical query heads. The doubly-compressed (8, 2, 2) shape that on its own forfeits -1.71% thus becomes a winning configuration once Premix and Postmix are introduced.

4.3 Large-scale and long-context validation

At the 30B-A3B MoE scale (500B tokens; FP8 mixed precision; full setup in Appendix D), Table 3 shows that VHA 24-2-2 attains the highest 12-benchmark average while using only 50% of GQA-4's KV cache and 89% of MLA's, confirming that the trends observed at 0.6B∼8B-A1B carry over to the production-scale MoE regime. The category-level breakdown shows that gains are broadly distributed: General $+3.39$ / $+0.68\%$, Math & STEM $+3.31$ / $+2.91\%$, Coding $+1.38$ / -1.14%, Multilingual $+2.30$ / $+0.36\%$ (VHA vs. GQA / MLA), indicating head-level virtualization scales gracefully across heterogeneous downstream tasks. Per-benchmark scores for all 12 evaluations are in Appendix C.

Long-context validation. At the final 128K checkpoint, VHA outperforms the matched GQA baseline on RULER at every evaluated length from 8K to 128K, with a length-average of 84.4 versus 73.0 ($+11.4$ points). The advantage extends to complementary evaluations at 128K: VHA outperforms GQA on LongBench-v2 (24.2 vs. 18.9) and BABILong (38.2 vs. 31.1), while remaining on par on InfiniteBench (31.5 vs. 31.8); averaged across lengths, VHA wins, ties, and loses 3/1/1

Table 4: **VHA ablations.** *Left, middle:* component contribution at 1.7B and 4B (each row uses VHA's shape, 8-2-2 or 16-2-2, with KV cache 1.00 KB / layer / token vs 4.00 KB for the GQA-8 baseline; Premix and Postmix are toggled individually). *Right:* activation ablation at 1.7B (replacing the linear formulation with SiLU on either or both modules). Avg (%) is over the eight benchmarks of Table 1; Δ is the average delta vs the GQA-8 baseline (component) or the linear default (activation); the linear VHA default is **bold**. Each module on its own only partially recovers the doubly-compressed loss from Table 2; the full VHA configuration exceeds the GQA-8 baseline at both scales, and SiLU consistently underperforms the linear default.

Component @ 1.7B

Variant	Avg	Δ
GQA	54.09	—
Premix only	53.31	-0.78
Postmix only	53.95	-0.14
VHA	**54.50**	$\mathbf{+0.41}$

Component @ 4B

Variant	Avg	Δ
GQA	56.90	—
Premix only	56.62	-0.28
Postmix only	57.22	$+0.32$
VHA	**57.71**	$\mathbf{+0.81}$

Activation @ 1.7B

Variant	Avg	Δ
+ SiLU on Premix	53.83	-0.67
+ SiLU on Postmix	53.79	-0.71
+ SiLU on both	53.67	-0.83
Linear (default)	**54.50**	—

Table 5: **Computational efficiency across four model scales** (S=4096). For each scale we list per-method query/key/value head counts, KV cache per layer per token (BF16, KB), training FLOPs per token aggregated over all transformer layers (G), and the relative change vs. the per-scale GQA baseline (GQA-8 dense, GQA-4 MoE). **Bold** highlights VHA, the method that achieves the best joint KV cache+FLOPs Pareto optimum at every scale. MHA's lower FLOPs at three scales reflect its smaller query head count under our parameter-matched configurations (H_q=12 at 0.6B/1.7B, H_q=20 at 4B/8B-A1B versus GQA's H_q=16 or H_q=32), at the cost of a 6–10$\times$ larger KV cache.

	0.6B				**1.7B**				**4B**				**8B-A1B**			
Method	Q-K-V	Cache	FLOPs	Δ %	Q-K-V	Cache	FLOPs	Δ %	Q-K-V	Cache	FLOPs	Δ %	Q-K-V	Cache	FLOPs	Δ %
MHA	12-12-12	6.00	6.87	-9.3	12-12-12	6.00	9.91	-6.6	20-20-20	10.00	21.66	-11.1	20-20-20	10.00	8.76	-14.7
GQA	16-8-8	4.00	7.57	0	16-8-8	4.00	10.61	0	32-8-8	4.00	24.38	0	32-4-4	2.00	10.27	0
MLA	14-14-14	1.12	8.25	$+8.9$	12-12-12	1.12	10.42	-1.8	28-28-28	1.12	25.09	$+2.9$	28-28-28	1.12	11.39	$+10.9$
MFA	17-1-1	1.00	11.54	$+52.3$	19-1-1	1.00	15.90	$+49.8$	33-1-1	1.00	36.04	$+47.9$	33-1-1	1.00	17.82	$+73.5$
VHA	**8-2-2**	**1.00**	**6.96**	$\mathbf{-8.1}$	**8-2-2**	**1.00**	**9.78**	$\mathbf{-7.9}$	**16-2-2**	**1.00**	**22.52**	$\mathbf{-7.6}$	**16-2-2**	**1.00**	**9.60**	$\mathbf{-6.5}$

HELMET categories. Together, these results demonstrate that VHA preserves its quality advantage through 128K while halving the KV heads and KV cache of the full-attention layers relative to GQA-4; complete results are in Appendix A.

4.4 Ablations

We ablate VHA's design choices at the 1.7B scale (additionally at 4B for the component contribution), isolating (i) the contribution of Premix and Postmix, (ii) the Postmix rank, and (iii) whether non-linearity helps either module.

Reading the Δ column of Table 4 (left and middle), the two modules address complementary deficits of the doubly-compressed shape: only their joint VHA configuration consistently achieves the largest gains at both scales at 25% of GQA-8's KV cache; we provide a mechanistic interpretation of this complementarity in Section 5.

Postmix rank. With Premix fixed at the 1.7B scale, sweeping the Postmix rank gives Avg $54.50/54.24/54.15$ for $r \in \{4, 32, 128\}$: larger r monotonically lowers training loss but degrades downstream evaluation, indicating that the low-rank constraint provides useful implicit regularization. We therefore fix $r = H'_q/2$ across all scales (Section 3.4 / Appendix D).

Activation. Replacing the linear formulation with SiLU on either or both modules consistently degrades quality at 1.7B (Table 4, right), regardless of where the non-linearity is inserted. Beyond the empirical loss, every non-linear variant forfeits inference-time absorption: with both modules linear the entire VHA computation collapses to standard GQA-2 at serving time, whereas any non-linearity must remain in the runtime kernel. We therefore keep both modules strictly linear, simultaneously achieving the highest empirical quality and lossless absorption into standard GQA-2 at inference.

4.5 Computational Efficiency

Table 5 quantifies the joint KV cache–FLOPs trade-off across the four model scales of Table 1. VHA is the only method that simultaneously matches the smallest KV cache (1.00 KB) and reduces training/prefill FLOPs below the per-scale GQA baseline (-8.1 / -7.9 / -7.6 / -6.5 % across scales).

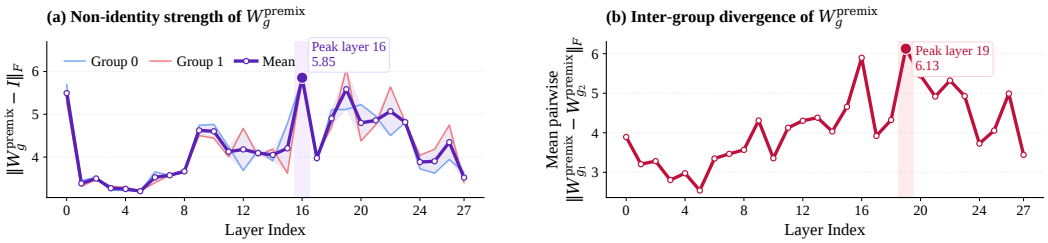

Figure 2: **Q Premix weight analysis on a trained Qwen-1.7B VHA model.** (a) Per-layer Frobenius deviation $\|W_g^{\text{premix}} - I\|_F$ per KV group (group mean overlaid). (b) Per-layer mean pairwise distance $\|W_{g_1}^{\text{premix}} - W_{g_2}^{\text{premix}}\|_F$ between KV groups.

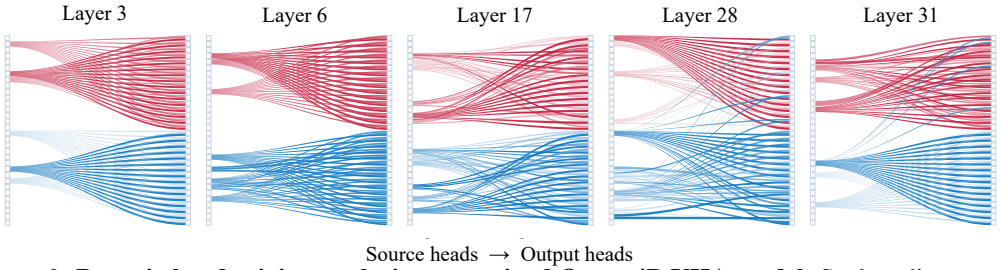

Figure 3: **Postmix head-mixing analysis on a trained Qwen-4B VHA model.** Sankey diagrams of $|AB^\top|$ across five representative layers, with ribbons flowing from source heads (left, colored by KV group) to recombined output heads (right); width and color saturation proportional to $|AB^\top|_{j,i}$.

MFA matches the cache but pays a +48–74 % FLOPs penalty; MLA's latent compression yields a slightly larger cache (1.12 KB) and raises FLOPs at three of four scales (up to +10.9 % at 8B-A1B). MHA appears competitive on FLOPs only because its smaller query head count under our parameter-matched configurations comes with a 6–10× larger KV cache. Combined with VHA's complete inference absorption to standard GQA-2 (Section 3.4), this makes VHA the best Pareto-optimal point on the cache–FLOPs–quality triangle in our experiments.

5 Analysis: How VHA Reparameterization Works

Beyond task accuracy, VHA should exploit the limited information available under aggressive KV sharing. We therefore inspect the Premix and Postmix matrices of trained VHA models (Premix on the 1.7B, Postmix on the 4B model), asking whether (i) Premix learns a group-conditioned query geometry, (ii) Postmix learns structured low-rank cross-head communication, and (iii) the two jointly close the doubly-compressed gap of Section 4.4.

Premix learns non-trivial, group-differentiated query transforms. Premix introduces a learnable per-group query transform W_g^{premix} before RoPE, initialized near identity. Two trivial outcomes would falsify our hypothesis: (a) W_g^{premix} remains near identity after training, rendering the module a no-op; or (b) all KV groups converge to the same matrix, degenerating Premix into a group-agnostic shared adapter without group-conditioned specialization. Figure 2 refutes both: panel (a) shows that $\|W_g^{\text{premix}} - I\|_F$ is substantially nonzero and most pronounced in mid-to-late layers (peak 5.85), and panel (b) shows that the inter-group distance $\|W_{g_1}^{\text{premix}} - W_{g_2}^{\text{premix}}\|_F$ is of comparable magnitude (peak 6.13), yielding a ratio well below the $\sqrt{2}$ value expected under fully independent groups and far above the zero expected under a shared deviation. This quantifies that the two KV groups learn substantially but not independently different query transforms. Premix therefore learns differentiated query probing directions per KV group under a fixed minimal KV cache, realizing genuine group-conditioned virtual-head expansion rather than mere parameter augmentation.

Postmix learns structured low-rank cross-head communication. Postmix recombines the H_v attention output heads via the low-rank residual $I + BA^\top$, where entry (j, i) of $|AB^\top|$ encodes the contribution of source head j to recombined head i. The central question is whether $AB^\top$ merely performs a per-head rescaling (a strictly diagonal pattern) or genuinely redistributes information across heads (off-diagonal flow). Figure 3 visualizes $|AB^\top|$ for the trained 4B model and reveals pronounced off-diagonal flow, ruling out the trivial per-head-rescaling outcome. The dominant

pattern is *within-group* redistribution: Postmix primarily rebalances information among heads sharing the same KV cache, with a sparse but consistent set of cross-group bridges providing inter-group connectivity without additional KV heads.

Joint reparameterization closes the doubly-compressed gap. Tying these analyses back to the component ablation in Table 4 clarifies the three numbers reported there. Premix-only retains the group-conditioned query geometry of Figure 2 but lacks Postmix's cross-group repair pathway, hence the -0.78% gap at 1.7B. Postmix-only owns the cross-head mixing channels of Figure 3 but operates over collapsed, unspecialized virtual heads, hence the -0.14% gap. Only the joint configuration realizes both structures simultaneously, yielding a $+0.41\%$ advantage over GQA-8 at 25% of its KV cache. VHA's effectiveness therefore arises from the functional complementarity of its two modules: Premix and Postmix are not interchangeable, and neither suffices alone.

6 Conclusion

We have presented Virtual Head Attention (VHA), which mitigates the tension among KV cache size, training/prefill FLOPs, and model quality through two lightweight linear operations: Q Premix and Linear Postmix. By halving physical query heads and recovering expressiveness via virtual head expansion and low-rank cross-group mixing, VHA reduces the KV cache to its architectural minimum while simultaneously lowering training FLOPs and improving model quality. Since all added operations are strictly linear, VHA fully absorbs into standard GQA-2 at inference, reusing existing serving optimizations with zero additional overhead. Analysis of trained models reveals that Premix and Postmix are functionally complementary and neither suffices alone, explaining why the joint configuration consistently surpasses the GQA baseline across all five evaluated scales (0.6B to 30B-A3B MoE). A matched 10B-A1B evaluation further shows that VHA maintains this advantage over GQA through 128K context.

Acknowledgments

We thank Wenbin Chang for valuable input and helpful discussions. We also thank the anonymous reviewers, area chairs, and senior area chairs for their insightful comments and constructive suggestions.

This work was supported by the National Engineering Research Center of Deep Learning Technology and Application and the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (Grant No. JYB2025XDXM202).

References

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. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, 2023.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan-Kelley. Reducing transformer key-value cache size with cross-layer attention. *Advances in Neural Information Processing Systems*, 37:86927–86957, 2024.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2924–2936, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. Multi-head attention: Collaborate instead of concatenate. *arXiv preprint arXiv:2006.16362*, 2020.

Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021.

Sai Surya Duvvuri, Chanakya Ekbote, Rachit Bansal, Rishabh Tiwari, Devvrit Khatri, David Brandfonbrener, Paul Liang, Inderjit Dhillon, and Manzil Zaheer. Interleaved head attention. *arXiv preprint arXiv:2602.21371*, 2026.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, et al. Are we done with mmlu? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5069–5096, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Jingcheng Hu, Houyi Li, Yinmin Zhang, Zili Wang, Shuigeng Zhou, Xiangyu Zhang, and Heung Yeung Shum. Multi-matrix factorization attention. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25114–25126, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391, 2018.

James O'Neill, Robert Clancy, Mariia Matskevichus, and Fergal Reid. Low-rank key value attention. *arXiv preprint arXiv:2601.11471*, 2026.

OpenAI. MMMLU: Multilingual massive multitask language understanding, 2024. URL `https://huggingface.co/datasets/openai/mmmlu`.

Runyu Peng, Yunhua Zhou, Demin Song, Kai Lv, Bo Wang, Qipeng Guo, and Xipeng Qiu. Explicit multi-head attention for inter-head interaction in large language models. *arXiv preprint arXiv:2601.19611*, 2026.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624, 2023.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Horn Roger and R Johnson Charles. Topics in matrix analysis, 1994.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A Haggag, Alfonso Amayuelas, et al. Include: Evaluating multilingual language understanding with regional knowledge. *arXiv preprint arXiv:2411.19799*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 4463–4473, 2019.

Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Noam Shazeer, Zhenzhong Lan, Youlong Cheng, Nan Ding, and Le Hou. Talking-heads attention. *arXiv preprint arXiv:2003.02436*, 2020.

Luoyang Sun, Cheng Deng, Jiwen Jiang, Xinjian Wu, Haifeng Zhang, Lei Chen, Lionel Ni, and Jun Wang. Gta: Grouped-head latent attention. *arXiv preprint arXiv:2506.17286*, 2025.

Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models. *Advances in Neural Information Processing Systems*, 37:7339–7361, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

Johannes Welbl, Nelson F Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, 2017.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL `https://arxiv.org/abs/2407.10671`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 4791–4800, 2019.

Yifan Zhang, Yifeng Liu, Huizhuo Yuan, Zhen Qin, Yang Yuan, Quanquan Gu, and Andrew Chi-Chih Yao. Tensor product attention is all you need. *arXiv preprint arXiv:2501.06425*, 2025.

Chuanyang Zheng, Jiankai Sun, Yihang Gao, Yuehao Wang, Peihao Wang, Jing Xiong, Liliang Ren, Hao Cheng, Janardhan Kulkarni, Yelong Shen, et al. Sas: Simulated attention score. *arXiv preprint arXiv:2507.07694*, 2025.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024.

Limitations and Future Work

Our largest validation is at the 30B-A3B MoE scale trained on 500B tokens; scaling VHA to larger MoE models and substantially more training data (e.g., 20T tokens) remains an important direction. We further validate VHA through 128K on a 10B-A1B MoE, while broader long-context architectures and training recipes remain to be explored. VHA's behavior under reinforcement learning training paradigms is also worth exploring. Finally, VHA is orthogonal to other KV cache compression techniques such as quantization and sparsification, and combining them may yield further inference savings.

A Long-Context Evaluation

A.1 Model and Stage-wise Training

We conduct a controlled long-context comparison on a MiMo-2.5-Flash-style 10B-A1B MoE backbone that interleaves full and sliding-window attention at a 1:3 ratio. The GQA and VHA models share the same backbone, data, optimization settings, and sliding-window layers. They differ only in the full-attention layers: GQA uses four KV heads, whereas VHA uses two; all sliding-window layers use eight KV heads in both models. Thus, VHA halves the KV cache of the full-attention layers without changing the sliding-window component.

The 8K pretraining stage uses 500B tokens and ends at 60K steps. Long-context mid-training then uses 100B tokens at 32K followed by another 100B tokens at 128K. These stages add 12K and 24K steps, yielding cumulative checkpoints at 72K and 96K steps, respectively. The RoPE base is increased from 10^4 at 8K to 6.4×10^5 at 32K and 5×10^6 at 128K. We report intermediate results from the 32K checkpoint and final results from the 128K checkpoint.

A.2 RULER and HELMET

We evaluate RULER and HELMET from 8K to 128K. RULER Overall is the mean over 13 tasks: eight Retrieval, two Aggregation, two QA, and one Variable Tracking task. HELMET is reported by category because its categories measure distinct capabilities.

At the final checkpoint, VHA leads RULER Overall at every evaluated length, with an average of 84.4 across lengths versus 73.0 for GQA. The largest gap occurs in Variable Tracking, where VHA scores 92.6/87.2/83.0 at 32K/64K/128K versus 3.2/4.2/3.0 for GQA. HELMET is more heterogeneous: averaged across lengths, VHA wins RAG, LongQA, and Summarization, ties Reranking, and trails on ICL.

A.3 Complementary Long-Context Evaluations

We additionally report LongBench-v2, InfiniteBench, BABILong, a short-context eight-task average, and language-modeling bits per byte on the four ProLong domains. At the final checkpoint, VHA wins seven of the eight reported metrics; InfiniteBench is nearly tied.

B Additional Robustness Experiments

The following experiments use a more efficient training implementation than the main 50B-token runs. Absolute scores therefore need not match the main tables, but all methods within each comparison use the same implementation and training recipe.

B.1 Repeated-run Robustness at 0.6B

We repeat the 0.6B experiment four times for VHA and three times for GQA. We report every run using zero-shot BF16 evaluation, with normalized accuracy used where available. Avg is the unweighted mean over eight tasks, and $\pm$ denotes the sample standard deviation across run-level averages. VHA achieves a higher mean with lower observed dispersion.

B.2 Sensitivity to Premix Initialization

We train the 1.7B model from scratch with the same recipe while changing only the Premix initialization: the default near-identity initialization (α=0.1), a larger identity-anchored perturbation (α=1.0), and fully random initialization without the identity anchor. All three runs remain stable, with no

Table 6: Long-context results on RULER and HELMET (%). RULER Overall averages all 13 tasks. Bold marks the better result within each matched checkpoint and evaluation length; exact ties are unbolded.

			RULER					HELMET				
Checkpoint	Length	Model	Overall	Retr.	Aggr.	QA	VT	RAG	LongQA	Summ.	ICL	Rerank
32K	8K	GQA	80.9	92.4	73.1	56.5	53.6	24.5	9.0	**7.2**	**60.7**	0.3
		VHA	**85.7**	**95.9**	**78.0**	**60.1**	**69.8**	**29.6**	**11.0**	6.6	57.1	**0.4**
	16K	GQA	73.7	90.0	51.8	53.7	27.6	22.6	**11.0**	**8.0**	**70.4**	0.0
		VHA	**83.2**	**93.3**	**71.5**	**55.4**	**80.8**	**32.3**	10.5	7.2	68.6	0.0
	32K	GQA	69.7	81.9	51.9	49.0	49.4	22.3	**17.0**	**8.1**	**78.3**	0.0
		VHA	**73.5**	**83.5**	**56.8**	**51.8**	**71.4**	**33.4**	11.0	7.6	75.1	**0.5**
128K	8K	GQA	79.1	89.5	87.9	46.0	44.6	27.4	12.5	**5.9**	**58.7**	**0.6**
		VHA	**90.1**	**96.7**	**92.8**	**59.4**	**94.0**	**34.2**	12.5	5.6	46.1	0.1
	16K	GQA	78.2	88.1	95.1	46.7	28.4	28.4	13.0	**6.4**	**66.4**	0.5
		VHA	**88.5**	**93.5**	**98.9**	**55.9**	**93.4**	**34.3**	**14.5**	6.3	61.0	**0.7**
	32K	GQA	72.6	83.3	93.6	43.2	3.2	26.7	**21.0**	6.7	**75.6**	0.4
		VHA	**85.3**	**89.6**	**96.0**	**53.7**	**92.6**	**36.9**	18.5	**6.9**	72.1	0.4
	64K	GQA	69.5	81.0	86.4	39.5	4.2	27.8	19.0	**7.9**	**80.1**	**0.7**
		VHA	**81.3**	**85.6**	**90.2**	**52.2**	**87.2**	**39.5**	**20.0**	7.7	79.6	0.4
	128K	GQA	65.7	75.3	92.5	31.5	3.0	30.4	13.0	7.8	**82.6**	2.0
		VHA	**76.8**	**79.9**	92.5	**45.5**	**83.0**	**42.5**	**15.0**	**9.5**	82.2	**2.6**

Table 7: Complementary evaluation results. LB-v2, InfBench, and BABI denote LongBench-v2, InfiniteBench, and BABILong; Short is the short-context eight-task average. The remaining columns report bits per byte (BPB) on ProLong domains. Bold marks the better matched result.

Checkpoint	Model	LB-v2 ↑	InfBench ↑	BABI ↑	Short ↑	arXiv ↓	Books ↓	Code ↓	Web ↓
32K	GQA	**28.0**	**29.9**	36.5	62.7	0.568	0.812	**0.294**	**0.763**
	VHA	22.3	27.3	**37.4**	**63.7**	0.568	0.812	0.295	0.765
128K	GQA	18.9	**31.8**	31.1	62.6	0.582	0.819	0.311	0.768
	VHA	**24.2**	31.5	**38.2**	**64.1**	**0.564**	**0.804**	**0.296**	**0.760**

gradient explosion, divergence, NaN, or Inf. Their early loss and pre-clipping gradient norms are similar (Table 9), while the default near-identity initialization gives the highest downstream average (Table 10).

C Per-benchmark Scores for the 30B-A3B MoE Evaluation

Table 11 reports per-benchmark scores for all evaluated models on the 30B-A3B MoE suite. Category-averaged numbers are reproduced in Table 3 in the main text. Model configurations: GQA 48-4-4, MLA 28-28-28, VHA 24-2-2. **Bold** marks the best per row.

D Training Configuration and Benchmark Citations

D.1 Model Architecture and Training Hyperparameters

Table 12 lists the backbone architecture and training hyperparameters for all five scales. Small-scale models follow the Qwen3 architecture; the 30B-A3B model uses an ERNIE-style MoE architecture with auxiliary-free routing (`noaux_tc`) and z-loss (λ=4×10^{-4}). We use AdamW [Loshchilov and Hutter, 2017] with β_1=0.9, β_2=0.95, weight decay 0.1, and gradient clipping at 1.0.

D.2 Evaluation Benchmarks

Small-scale benchmarks (8 tasks). We report zero-shot accuracy on the following benchmarks: ARC-Challenge [Clark et al., 2018], BoolQ [Clark et al., 2019], HellaSwag [Zellers et al., 2019],

Table 8: Repeated-run results at 0.6B (%). Every run is reported. Bold in the summary rows marks the higher across-run mean.

Model	Run	ARC-C	BoolQ	HSwag	OBQA	PIQA	Wino	SciQ	SIQA	Avg
VHA	01	30.12	60.18	44.46	33.80	68.77	52.25	74.20	39.56	50.42
VHA	02	29.35	59.27	44.36	32.80	68.55	55.96	74.00	39.56	50.48
VHA	03	29.27	59.36	44.79	33.80	67.14	52.25	76.00	40.23	50.35
VHA	04	30.12	56.09	44.68	34.80	68.55	52.64	76.10	39.76	50.34
VHA mean		29.72	58.73	**44.57**	**33.80**	68.25	**53.28**	**75.08**	**39.78**	50.40 ± 0.07
GQA	01	29.44	58.59	44.49	32.40	68.28	51.14	75.50	38.79	49.83
GQA	02	30.12	57.86	44.55	32.60	68.34	50.91	74.10	39.87	49.79
GQA	03	30.20	60.55	44.61	35.00	68.99	51.93	72.50	39.30	50.39
GQA mean		**29.92**	**59.00**	44.55	33.33	**68.54**	51.33	74.03	39.32	50.00 ± 0.34

Table 9: Early training trajectory under different Premix initializations. Each entry is loss / pre-clipping gradient norm.

Step	α=0.1	α=1.0	Random
10	12.2976 / 14.74	12.3072 / 14.80	12.3132 / 14.98
20	11.1501 / 15.97	11.1581 / 16.26	11.1576 / 16.25
30	9.9346 / 4.98	9.9389 / 5.01	9.9400 / 5.07
50	9.1168 / 2.67	9.1243 / 3.76	9.1255 / 2.03
100	7.9670 / 2.96	7.9788 / 2.93	7.9669 / 4.02
200	6.4777 / 2.99	6.4878 / 2.21	6.4898 / 1.65
500	4.9822 / 1.70	4.9990 / 2.47	5.0198 / 2.85
1000	3.8647 / 1.17	3.8851 / 0.94	3.8888 / 0.89
2000	3.1699 / 0.27	3.1842 / 0.33	3.1967 / 0.54

OpenBookQA [Mihaylov et al., 2018], PIQA [Bisk et al., 2020], Winogrande [Sakaguchi et al., 2021], SciQ [Welbl et al., 2017], Social IQA [Sap et al., 2019].

Large-scale benchmarks (12 tasks). We report accuracy with the shot counts listed in Table 11 on the following benchmarks: MMLU [Hendrycks et al., 2020], MMLU-Redux [Gema et al., 2025], MMLU-Pro [Wang et al., 2024], BBH [Suzgun et al., 2023], GPQA [Rein et al., 2023], GSM8K [Cobbe et al., 2021], MATH [Hendrycks et al., 2021], HumanEval+ [Liu et al., 2023], MBPP [Austin et al., 2021], MBPP+ [Liu et al., 2023], MMMLU [OpenAI, 2024], INCLUDE [Romanou et al., 2024].

E Training Loss Curves

Figure 4 shows per-method training loss curves at four model scales (50B tokens, 4K context). VHA achieves the lowest convergence loss at the 4B scale and remains competitive at 0.6B, 1.7B, and 8B-A1B, where its gap from the best-converging method is within 0.01 nats.

F VHA Algorithm

Algorithm 1 provides a self-contained reference for VHA's training forward pass and the one-time inference-absorption procedure that reduces the trained model to a standard GQA-2 deployment.

G Premix as Structured Factorization of Virtual-Head Query Projections

We show that, in the forward pass, Q Premix is mathematically equivalent to a standard multi-head attention with $H_v = H_{kv}H'_q$ virtual query heads, whose per-virtual-head query projections are constrained to a structured low-dimensional manifold. The training-time benefit of the separated parameterization arises from the coupled gradient structure induced by this factorization.

Setup. Let $W_Q^{\text{phys}} \in \mathbb{R}^{h \times H'_q d}$ be the physical query projection (with H'_q heads, each of dimension d), and let $W_g^{\text{premix}} \in \mathbb{R}^{d \times d}$ be the Premix matrix for KV group $g \in \{1, \ldots, H_{kv}\}$. For physical

Table 10: Downstream accuracy under different Premix initializations (%). Bold marks the best result per row.

Task	α=0.1	α=1.0	Random
ARC-C	**34.81**	34.13	34.39
BoolQ	**60.06**	58.04	57.03
HellaSwag	**51.80**	51.72	51.76
OpenBookQA	**37.60**	37.00	36.00
PIQA	71.06	71.33	**72.14**
Winogrande	**55.72**	54.85	53.28
SciQ	**80.70**	80.30	80.30
Social IQA	**40.58**	40.23	39.97
Avg	**54.04**	53.45	53.11

Table 11: Per-benchmark scores on the 30B-A3B MoE evaluation suite (accuracy %).

Category	Benchmark	Shot	GQA	MLA	**VHA**
General	MMLU	5-shot	55.97	59.34	**60.24**
	MMLU-Redux	5-shot	63.05	65.25	**66.10**
	MMLU-Pro	5-shot	36.20	**41.12**	40.03
	BBH	3-shot CoT	56.23	56.57	**58.61**
Math & STEM	GPQA	10-shot CoT	**34.62**	31.62	34.11
	GSM8K	4-shot CoT	**56.48**	55.42	56.25
	MATH	4-shot CoT	38.18	43.42	**48.84**
Coding	HumanEval+	0-shot	50.00	**52.44**	47.56
	MBPP	3-shot	45.00	48.60	**49.20**
	MBPP+	3-shot	60.54	62.07	**62.93**
Multilingual	MMMLU	5-shot	39.82	42.69	**42.95**
	INCLUDE	5-shot	46.69	47.71	**48.16**

head $i \in \{1, \ldots, H'_q\}$ within group g, the virtual query is

$$\tilde{Q}_{i,g} \;=\; Q_i\, W_g^{\text{premix}} \;=\; X\, W_Q^{(i)}\, W_g^{\text{premix}}, \tag{6}$$

where $W_Q^{(i)} \in \mathbb{R}^{h\times d}$ is the i-th slice of W_Q^{phys}. The attention output for virtual head (i, g) is

$$\text{head}_{i,g} \;=\; \text{softmax}\!\left(\frac{\tilde{Q}_{i,g} K_g^\top}{\sqrt{d}}\right) V_g. \tag{7}$$

Forward equivalence to standard multi-head attention. Define

$$\widetilde{W}_Q^{(i,g)} \;\triangleq\; W_Q^{(i)}\, W_g^{\text{premix}} \;\in\; \mathbb{R}^{h\times d}. \tag{8}$$

By the associativity of matrix multiplication, $\tilde{Q}_{i,g} = X\widetilde{W}_Q^{(i,g)}$, and therefore

$$\text{head}_{i,g} \;=\; \text{softmax}\!\left(\frac{X\widetilde{W}_Q^{(i,g)} K_g^\top}{\sqrt{d}}\right) V_g \tag{9}$$

holds as a pointwise identity for every input X and every parameter assignment. Hence VHA is, at the level of input-output behavior, a standard multi-head attention with H_v query heads and H_{kv} KV heads, in which the i-th virtual head of group g uses the projection matrix $\widetilde{W}_Q^{(i,g)}$.

The factorization defines a low-dimensional parameter manifold. The standard H_v-head Q projection has $H_v hd = H_{kv} H'_q hd$ free parameters. The Premix factorization uses

$$H'_q hd \;+\; H_{kv} d^2 \tag{10}$$

parameters, parameterized by $(W_Q^{\text{phys}}, \{W_g^{\text{premix}}\}_{g=1}^{H_{kv}})$. The set of attainable $\{\widetilde{W}_Q^{(i,g)}\}$ tuples is therefore a structured submanifold of the unconstrained parameter space:

$$\mathcal{M} \;=\; \left\{ \left\{ W_Q^{(i)} W_g^{\text{premix}} \right\}_{i,g} \;:\; W_Q^{(i)} \in \mathbb{R}^{h\times d},\; W_g^{\text{premix}} \in \mathbb{R}^{d\times d} \right\}. \tag{11}$$

Table 12: Model architecture and training hyperparameters across all five scales. "–" denotes not applicable.

	0.6B	**1.7B**	**4B**	**8B-A1B (MoE)**	**30B-A3B (MoE)**
Architecture					
Precision	BF16	BF16	BF16	BF16	FP8
Num layers	28	28	36	24	46
Hidden size	1024	2048	2560	2048	1536
FFN dim	3072	6144	9728	768×64	1024×128
Num experts	–	–	–	64	128 (+1 shared)
Experts per token	–	–	–	4	8
Vocab size	151936	151936	151936	151936	102400
RoPE base	10^6	10^6	10^6	10^6	10^6
Tied embeddings	✓	✓	✓	×	×
Training					
Training tokens	50B	50B	50B	50B	500B
Context length	4096	4096	4096	4096	8192
Peak learning rate	2.5×10^{-4}	2×10^{-4}	2×10^{-4}	2×10^{-4}	1×10^{-4}
Min learning rate	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
LR schedule	cosine	cosine	cosine	cosine	wsd
Warmup steps	2000	2000	2000	2000	2000
Total steps	24000	24000	24000	24000	40000
Batch size (tokens)	2M	2M	2M	2M	13.1M
Adam β_1	0.9	0.9	0.9	0.9	0.9
Adam β_2	0.95	0.95	0.95	0.95	0.95
Weight decay	0.1	0.1	0.1	0.1	0.1
Gradient clipping	1.0	1.0	1.0	1.0	1.0
VHA-specific					
Premix init scale α	0.1	0.1	0.1	0.1	0.1
Premix init noise σ	1	1	1	1	1
Postmix rank r	4	4	8	8	12
Baseline-specific					
MLA latent dim c_{kv}	512	512	512	512	512
MLA latent dim c_q	512	1536	1536	1536	1536
MFA head dim d'	256	256	256	256	–

At the 1.7B scale (h=2048, d=128, H'_q=8, H_{kv}=2, H_v=16), the factorized parameterization uses ∼2.13M parameters, compared to ∼4.19M for the unconstrained H_v-head Q projection, a ∼50% reduction. The corresponding Q-projection FLOPs are reduced by a similar factor, as the additional cost $2H_{kv}H'_q d^2$ of applying Premix is much smaller than the savings $2hH'_q(H_{kv}-1)d$ achieved by halving the physical head count. This is the principal source of VHA's reduction in training and prefill FLOPs.

Reparameterization benefit via gradient coupling. Although Eq. 9 shows that the forward pass is algebraically equivalent to training a single matrix $\widetilde{W}_Q^{(i,g)}$ per virtual head, the gradients under the factorized parameterization are coupled across groups and physical heads:

$$\frac{\partial \mathcal{L}}{\partial W_Q^{(i)}} = \sum_{g=1}^{H_{kv}} X^\top \frac{\partial \mathcal{L}}{\partial \tilde{Q}_{i,g}} \left(W_g^{\text{premix}}\right)^\top, \tag{12}$$

$$\frac{\partial \mathcal{L}}{\partial W_g^{\text{premix}}} = \sum_{i=1}^{H'_q} \left(X W_Q^{(i)}\right)^\top \frac{\partial \mathcal{L}}{\partial \tilde{Q}_{i,g}}. \tag{13}$$

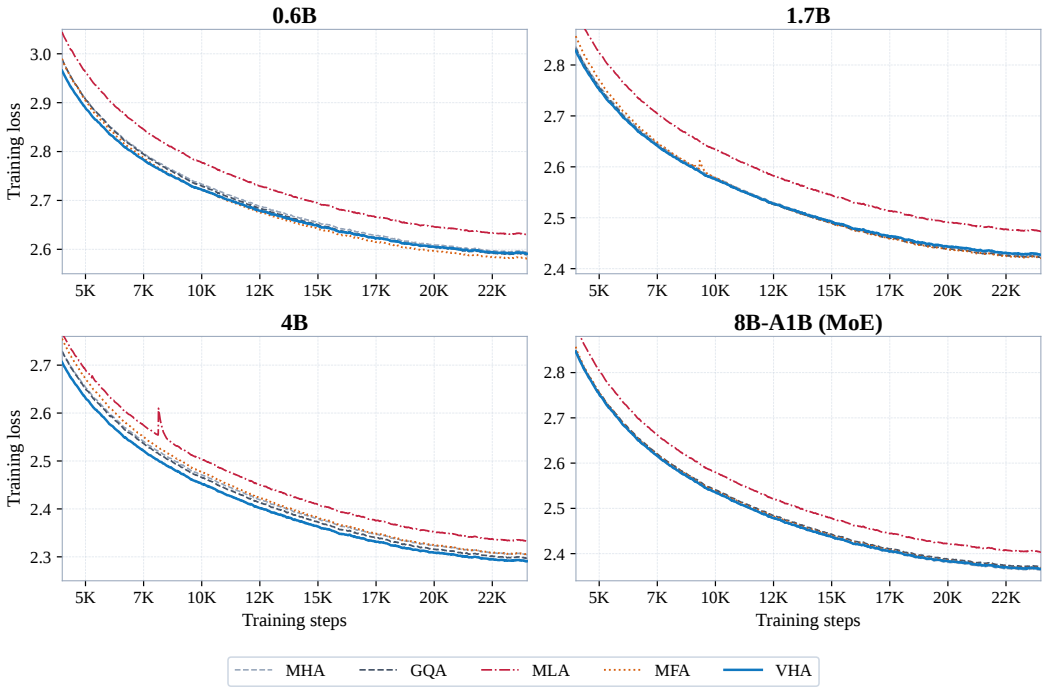

Figure 4: **Training loss curves across four model scales.** Each panel compares MHA, GQA, MLA, MFA, and VHA on FineWeb-Edu (50B tokens, 4K context). All curves are plotted from step 4K onward after the learning-rate warm-up; smoothing is applied with an exponential moving average (α=0.93). VHA (blue, solid) achieves the lowest loss at convergence at the 4B scale.

Each $W_Q^{(i)}$ aggregates contributions from all H_{kv} groups, modulated by the corresponding Premix matrices, while each W_g^{premix} receives gradient signal through all H'_q physical heads. SGD on the factorized parameters therefore traverses the manifold $\mathcal{M}$ along directions induced by this coupled gradient structure, which differs from the trajectories obtained by independently optimizing H_v unconstrained virtual-head matrices. This is the same source of implicit regularization that underlies LoRA [Hu et al., 2022] and RepVGG [Ding et al., 2021]: a forward-equivalent factorization of a large weight tensor yields different optimization dynamics and, empirically, different (and often better) final solutions.

Inference-time absorption. Since the forward equivalence in Eq. 9 holds for any frozen parameter values, at inference we simply precompute $\widetilde{W}_Q^{(i,g)} = W_Q^{(i)} W_g^{\text{premix}}$ for all (i, g) pairs and discard W_Q^{phys} and $\{W_g^{\text{premix}}\}$. The model then runs as a standard GQA-2 with H_v independent query projections, $H_{kv} = 2$ KV heads, and the same output projection W_O, fully compatible with existing GQA serving kernels.

Unified view with Postmix. Premix and Postmix are two instantiations of the same head-axis reparameterization principle. Premix factorizes the per-virtual-head query projection on the Q side ($\widetilde{W}_Q^{(i,g)} = W_Q^{(i)} W_g^{\text{premix}}$); Postmix factorizes the merged output projection on the O side ($W'_O = \big((I_{H_v} + AB^\top) \otimes I_d\big) W_O$, see Appendix H). Both are forward-equivalent to a standard GQA in their merged form, both gain training-time benefits from coupled gradients of the separated parameters, and both fold back into a standard GQA-2 architecture at inference with zero overhead.

H Proof of Postmix Linear Redundancy

We prove that Linear Postmix can be mathematically absorbed into the output projection W_O, and discuss why the separated parameterization nonetheless improves training.

Algorithm 1 Virtual Head Attention (VHA)

Training Forward Pass

Require: Input $X \in \mathbb{R}^{S\times h}$; parameters $W_Q, W_K, W_V, W_O, \{W_g^{\text{premix}}\}_{g=1}^{H_{kv}}, A, B$

$Q \leftarrow XW_Q \in \mathbb{R}^{S\times H'_q\times d}$ ▷ Physical query heads ($H'_q{=}H_q/2$)

$K \leftarrow XW_K \in \mathbb{R}^{S\times H_{kv}\times d}, \quad V \leftarrow XW_V \in \mathbb{R}^{S\times H_{kv}\times d}$

for each KV group $g = 1, \ldots, H_{kv}$ **do**

  $\tilde{Q}_g \leftarrow Q\, W_g^{\text{premix}}$ ▷ Q Premix: virtual head expansion

end for

$\tilde{Q} \leftarrow \text{concat}(\tilde{Q}_1, \ldots, \tilde{Q}_{H_{kv}}) \in \mathbb{R}^{S\times H_v\times d}$ ▷ $H_v{=}H_{kv}{\times}H'_q$ virtual heads

$Y_g \leftarrow \text{softmax}\Big(\tilde{Q}_g K_g^\top/\sqrt{d}\Big)\, V_g$ for each group g ▷ Grouped attention

$Y \leftarrow \text{concat}(Y_1, \ldots, Y_{H_{kv}})$

$Y' \leftarrow (I_{H_v} + BA^\top)\, Y$ ▷ Linear Postmix: cross-group mixing

$\text{output} \leftarrow \text{reshape}(Y', [S, H_v d])\, W_O$

Inference Absorption (one-time precomputation)

Require: $W_Q \in \mathbb{R}^{h\times H'_q d}$ (sliced into per-head $W_Q^{(i)} \in \mathbb{R}^{h\times d}$); $\{W_g^{\text{premix}}\}_{g=1}^{H_{kv}}$ each $\in \mathbb{R}^{d\times d}$; $W_O \in \mathbb{R}^{H_v d\times h}$; $A, B \in \mathbb{R}^{H_v\times r}$

for each $(i, g) \in \{1, \ldots, H'_q\} \times \{1, \ldots, H_{kv}\}$ **do**

  $\widetilde{W}_Q^{(i,g)} \leftarrow W_Q^{(i)}\, W_g^{\text{premix}} \in \mathbb{R}^{h\times d}$ ▷ Fold Premix into the (i, g)-th virtual query projection

end for

$W_Q^{\text{abs}} \leftarrow \text{concat}\Big(\big\{\widetilde{W}_Q^{(i,g)}\big\}_{i,g}\Big) \in \mathbb{R}^{h\times H_v d}$ ▷ $H_v{=}H_{kv}{\times}H'_q$

$W_O^{\text{abs}} \leftarrow \big((I_{H_v} + AB^\top) \otimes I_d\big)\, W_O \in \mathbb{R}^{H_v d\times h}$ ▷ Fold Postmix into W_O

Discard $W_Q, \{W_g^{\text{premix}}\}, A, B$; run standard GQA-2 with $W_Q^{\text{abs}}, W_K, W_V, W_O^{\text{abs}}$

Setup. For a single token, let $Y \in \mathbb{R}^{H_v\times d}$ be the concatenated attention output across all H_v virtual heads. Postmix computes:

$$Y' = (I_{H_v} + BA^\top)\, Y \tag{14}$$

where $A, B \in \mathbb{R}^{H_v\times r}$. The final output is obtained by applying the output projection:

$$\text{output}^\top = \text{vec}_{\text{row}}(Y')^\top W_O \in \mathbb{R}^{1\times h} \tag{15}$$

where $W_O \in \mathbb{R}^{H_v d\times h}$ and $\text{vec}_{\text{row}}(\cdot)$ flattens a matrix in row-major order (i.e., $Y.\text{reshape}(H_v d)$ in PyTorch, concatenating head outputs).

Absorption via Kronecker product. Using the row-major identity: for $M \in \mathbb{R}^{H_v\times H_v}$ and $Y \in \mathbb{R}^{H_v\times d}$,

$$\text{vec}_{\text{row}}(MY)^\top = \text{vec}_{\text{row}}(Y)^\top \big(M^\top \otimes I_d\big) \tag{16}$$

Setting $M = I_{H_v} + BA^\top$ (so $M^\top = I_{H_v} + AB^\top$) [Roger and Charles, 1994]:

$$\begin{aligned}\text{vec}_{\text{row}}(Y')^\top &= \text{vec}_{\text{row}}\big((I_{H_v} + BA^\top)\, Y\big)^\top \\ &= \text{vec}_{\text{row}}(Y)^\top \big((I_{H_v} + AB^\top) \otimes I_d\big)\end{aligned} \tag{17}$$

Substituting into the output computation:

$$\begin{aligned}\text{output}^\top &= \text{vec}_{\text{row}}(Y')^\top W_O \\ &= \text{vec}_{\text{row}}(Y)^\top \big((I_{H_v} + AB^\top) \otimes I_d\big) W_O \\ &= \text{vec}_{\text{row}}(Y)^\top W'_O\end{aligned} \tag{18}$$

where $W'_O = \big((I_{H_v} + AB^\top) \otimes I_d\big) W_O \in \mathbb{R}^{H_v d\times h}$.

Since $I_{H_v} + BA^\top$ is generically invertible (it equals the identity at initialization and remains near-identity during training), W'_O has the same number of free parameters as W_O. Therefore, for any choice of W_O, A, B, there exists a single matrix W'_O that produces the identical output, confirming that Postmix is theoretically redundant.

Reparameterization benefit. Despite this redundancy, the separated parameterization $\{W_O, A, B\}$ provides training advantages analogous to those observed in RepVGG [Ding et al., 2021]:

1. **Factored gradient paths.** The gradients with respect to A, B, and W_O follow different paths through the computation graph. This factorization provides a form of implicit regularization, as the effective weight update to W'_O at each step is constrained by the low-rank structure of $AB^\top$.
2. **Identity initialization.** With B initialized to zero, $BA^\top = 0$ and $Y' = Y$ at the start of training. Postmix begins as a no-op and gradually learns useful cross-group mixing patterns, avoiding disruption to the early training dynamics.
3. **Low-rank constraint.** The bottleneck rank $r \ll H_v$ limits the expressiveness of the cross-head mixing, acting as a structural regularizer. This prevents Postmix from overfitting, consistent with the ablation finding that larger ranks (r=$32, 128$) degrade evaluation performance despite lower training loss.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Refer to the Abstract and Introduction sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Refer to the Limitations and Future Work section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Appendix G provides a formal proof of Q Premix's forward equivalence to standard multi-head attention and derives the gradient coupling structure. Appendix H provides a complete proof of Linear Postmix's theoretical redundancy via the Kronecker product identity. All assumptions are stated in the setup paragraphs of each appendix section.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Refer to the Method section and Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Our code will be open-sourced.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Refer to the Experiments section and Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [N/A]

   Justification: Error bars are not standard practice in LLM pretraining research; reliability is demonstrated through consistent trends across five model scales and multi-benchmark averaging.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments are conducted on NVIDIA H100 80GB GPUs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors have reviewed and complied with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This is foundational infrastructure research with no direct path to specific harmful applications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No model weights, datasets, or other assets are released with this submission, so safeguard measures are not applicable at this stage.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All evaluation benchmarks are cited with their original publications; see Appendix D.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper involves no crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper involves no human subjects; IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLM assistance was used only for writing and editing and does not impact the core methodology, scientific rigor, or originality of the research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.