Decoupled Bottleneck Attention:

Scaling Efficient Transformers via Low-Rank Semantic Routing

Daniel Owen van Dommelen
Independent Research
theapemachine@gmail.com

December 2025

Abstract

The Key-Value cache in Transformer models scales linearly with sequence length and model dimension, creating a critical memory bottleneck for long-context inference. While techniques like Grouped-Query Attention (GQA) reduce cache size by sharing key-value heads, they preserve the full computational cost of attention scoring in high-dimensional space.

We propose **Decoupled Bottleneck Attention**, an architectural modification that exploits the empirical observation that *semantic routing*—deciding which tokens attend to which—operates in a low-rank subspace ($r \approx 32$), while *positional geometry* requires higher fidelity ($r \approx 64$). By decoupling these concerns into separate projection paths, we achieve:

- **Up to 168× KV-cache reduction** in a fixed-rank long-context scenario via dimension reduction plus 4-bit quantization (Section 2.6)
- **Evidence at two scales:** v21 WikiText-2 ablations show that bottlenecking attention can outperform a full-rank baseline; v29 FineWeb-Edu scaling runs show that decoupling (48/96) outperforms a standard baseline, GQA (kv=2), and a matched-rank bottleneck baseline (Table 2)
- **Superior data efficiency compared to Grouped-Query Attention (GQA).** On FineWeb-Edu, our Decoupled Bottleneck model (d_{sem}=48, d_{geo}=96; total $d_{\text{attn}} = 144$) achieves substantially lower validation loss (5.862 vs 6.302) than a GQA baseline with shared KV heads (kv=2) under the same training recipe, demonstrating that reducing the *interaction rank* is a more effective compression strategy than head sharing.

In our v21 WikiText-2 suite, a simple rank-96 bottleneck *outperforms* the full rank-512 baseline (val loss 5.33 vs 5.37), suggesting that standard Transformers can over-allocate capacity to attention. In our v29 FineWeb-Edu suite, the Decoupled Bottleneck model achieves the best validation loss among baseline, GQA, and a matched-rank Bottleneck baseline (Table 2).

Keywords: Transformer, attention mechanism, low-rank, KV-cache, memory efficiency, quantization, long context, rotary position embeddings

1 Introduction

Modern Transformer architectures [15] achieve remarkable performance across language modeling, translation, and reasoning tasks. However, their quadratic attention complexity and linear KV-cache growth present fundamental scalability challenges for long-context applications.

1.1 The Redundancy Hypothesis

We begin with a simple observation: in a 512-dimensional layer, the neurons are not independent. They move in *sympathetic clusters*—correlated groups that reduce the intrinsic dimensionality of the representation. Prior work on LoRA [8] demonstrated that weight *updates* during fine-tuning are low-rank (typically $r \leq 64$). Recent work on gradient dynamics [12] shows that optimization naturally collapses to low rank. We extend this observation to argue that the *architecture itself*—specifically the attention mechanism—should be structurally constrained to match this intrinsic rank.

Empirical measurements from our experiments show that the effective rank of W_Q and W_K projections stabilizes around 11-32 dimensions, even when the nominal dimension is 512. This aligns with theoretical analysis by Bhojanapalli et al. [3], who identified a "low-rank bottleneck" in multi-head attention, and recent work by Kobayashi et al. [9] showing that weight decay actively induces rank reduction during training. Wang et al. [16] further demonstrate that attention outputs are approximately 60% low-dimensional, adding to low-dimensional residual subspaces. Crucially, Refael et al. [12] proved that gradient rank *decreases monotonically* during training, asymptotically approaching rank one—providing theoretical justification for why architectural bottlenecks become increasingly appropriate as training progresses. Chiang & Yogatama [4] show that RoPE may cause dimension inefficiency for long-distance retrieval, supporting our use of higher dimensions (64) for the geometric path.

1.2 Comparison with Existing Approaches

Grouped-Query Attention (GQA). While Grouped-Query Attention [2] successfully reduces KV-cache memory by sharing key-value heads across multiple query heads, it maintains the full computational cost of the query projection and attention scoring in the high-dimensional space. Each query still operates in $\mathbb{R}^d$, and every attention score still requires a d-dimensional dot product—GQA merely amortizes the *storage* cost, not the *interaction* cost.

Our Bottleneck approach reduces both memory *and* compute by compressing the interaction manifold. Rather than sharing high-dimensional KV pairs, we project queries and keys into a low-rank semantic subspace ($r \ll d$) *before* computing attention, reducing dot-product complexity from $O(n^2d)$ to $O(n^2r)$.

Multi-Head Latent Attention (MLA). DeepSeek-V2 [5] introduced MLA, which compresses KV storage into a latent vector, achieving 93% cache reduction. However, MLA *up-projects* during the forward pass to perform attention in the original high-dimensional space. Our method remains low-rank throughout, saving both memory and compute.

Disentangled Attention. DeBERTa [7] pioneered the separation of content and position representations in attention scoring. We adopt this disentanglement principle but leverage it for *efficiency*: applying aggressive compression to the semantic (content) path while preserving fidelity in the geometric (position) path.

1.3 Contributions

1. We demonstrate that attention routing can be performed in $\sim$32 dimensions without perplexity degradation, while positional encoding requires $\sim$64 dimensions for RoPE fidelity.

2. We propose **Decoupled Bottleneck Attention**, which separates semantic and geometric scoring paths with asymmetric dimensionality.

3. We introduce a **Null Token** mechanism that stabilizes training by providing an explicit "attend nowhere" option.

4. We show that combined dimension reduction + 4-bit quantization achieves **168**$\times$ KV-cache compression with minimal quality loss (Figure 5).

2 Methodology

2.1 Standard Multi-Head Attention

In standard scaled dot-product attention with H heads:

$$\mathrm{Attn}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{1}$$

where $Q, K, V \in \mathbb{R}^{n\times d}$ are obtained by linear projection from the input $X \in \mathbb{R}^{n\times d_{\text{model}}}$:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \tag{2}$$

with $W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}}\times d}$. For language modeling with context length n and dimension d, the KV-cache requires $O(2\cdot L\cdot n\cdot d)$ memory, where L is the number of layers.

2.2 Bottleneck Attention

We introduce a simple modification: project Q and K to a lower-dimensional space *before* computing attention scores.[1]

$$Q' = XW'_Q, \quad K' = XW'_K \tag{3}$$

where $W'_Q, W'_K \in \mathbb{R}^{d_{\text{model}}\times d_{\text{attn}}}$ with $d_{\text{attn}} \ll d_{\text{model}}$. The attention computation becomes:

$$\mathrm{Attn}_{\text{bottleneck}}(Q',K',V') = \mathrm{softmax}\left(\frac{Q'K'^\top}{\sqrt{d_{\text{attn}}/H}}\right)V' \tag{4}$$

This reduces the dot-product complexity from $O(n^2\cdot d_{\text{model}})$ to $O(n^2\cdot d_{\text{attn}})$ and the KV-cache from $O(n\cdot d_{\text{model}})$ to $O(n\cdot d_{\text{attn}})$.

2.3 Decoupled Bottleneck Attention

The key insight motivating decoupling is that *semantic matching* ("is this token semantically related?") and *geometric positioning* ("how far away is this token?") have different intrinsic dimensionality requirements.

We decompose the attention score into two additive components:

$$\text{Score} = \underbrace{\frac{Q_{\text{sem}}K_{\text{sem}}^\top}{\sqrt{d_{\text{sem}}/H}}}_{\text{Semantic Path}} + \underbrace{\frac{Q_{\text{geo}}K_{\text{geo}}^\top}{\sqrt{d_{\text{geo}}/H}}}_{\text{Geometric Path}} \tag{5}$$

where:

$$Q_{\text{sem}} = XW_{Q,\text{sem}}, \quad K_{\text{sem}} = XW_{K,\text{sem}} \qquad (d_{\text{sem}} = 32) \tag{6}$$

$$Q_{\text{geo}} = XW_{Q,\text{geo}}, \quad K_{\text{geo}} = XW_{K,\text{geo}} \qquad (d_{\text{geo}} = 64) \tag{7}$$

[1] Our use of "bottleneck" refers to dimensionality reduction in the query/key space, distinct from Park et al.'s BAM [11], which applies channel and spatial attention in CNNs for computer vision.

Critically, we apply **Rotary Position Embeddings (RoPE)** [13] *only* to the geometric path:

$$Q_{\text{geo}}, K_{\text{geo}} \leftarrow \text{RoPE}(Q_{\text{geo}}, K_{\text{geo}}, \text{position}) \tag{8}$$

The semantic path operates on pure content similarity, while the geometric path encodes positional relationships. The value projection uses the combined dimension:

$$V = XW_V, \quad W_V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}} \tag{9}$$

where $d_{\text{attn}} = d_{\text{sem}} + d_{\text{geo}} = 96$ in our default configuration.

2.4 The Null Token Mechanism

Low-rank attention can become unstable when queries lack semantically appropriate keys. We introduce a learnable **null token** $k_\emptyset$ providing an explicit "attend nowhere" option:

$$\text{Score}_{\text{null}} = \frac{Q_{\text{sem}} k_{\emptyset,\text{sem}}^\top}{\sqrt{d_{\text{sem}}/H}} + \frac{Q_{\text{geo}} k_{\emptyset,\text{geo}}^\top}{\sqrt{d_{\text{geo}}/H}} \tag{10}$$

This score is concatenated to the attention matrix before softmax, allowing the model to "dump" attention mass when no key is appropriate, which stabilizes training at very low ranks.

2.5 Tied Q-K Projections

For the semantic path, we optionally **tie** the query and key projections: $W_{Q,\text{sem}} = W_{K,\text{sem}}$. This enforces symmetric similarity ("A attends to B iff B attends to A"), which is appropriate for content matching but not for position-dependent relationships.

2.6 Quantized Inference

For inference, we apply aggressive quantization to the KV-cache. Recent work has demonstrated that 4-bit KV cache quantization preserves model quality remarkably well. Turboderp's ExLlamaV2 implementation [14] showed Q4 cache performs comparably to FP16, and this capability has been integrated into production inference engines like llama.cpp [6]. We implement block-wise Q4_0 quantization following this approach:

$$x_{\text{quantized}} = \text{round}\left(\frac{x}{\text{scale}}\right), \quad \text{scale} = \frac{\max(|x_{\text{block}}|)}{7} \tag{11}$$

where each block of 32 elements shares a single FP16 scale factor. Combined with the dimension reduction ($d_{\text{attn}} = 96$ vs $d_{\text{model}} = 512$), this achieves:

$$\text{Compression} = \underbrace{\frac{512}{96}}_{\text{Dimension}} \times \underbrace{\frac{16}{4}}_{\text{Quantization}} = 5.33 \times 4 \approx 21\times \text{ (per-layer)} \tag{12}$$

Calculation of 168×. The KV-cache memory at long context depends on the choice of attention dimension d_{attn} at scale. For a rough Llama-7B-like configuration (32 layers, $d = 4096$, 128k context, batch=1), the FP16 KV cache is:

$$M_{\text{FP16}} \approx 2 \cdot 32 \cdot 4096 \cdot 128\text{k} \cdot 2 \text{ bytes} \approx 64 \text{ GiB}.$$

With 4-bit KV-cache quantization (0.5 bytes/value), the memory becomes:

$$M_{\text{Q4}} \approx 2 \cdot 32 \cdot d_{\text{attn}} \cdot 128\text{k} \cdot 0.5 \text{ bytes}.$$

This yields two useful reference scenarios:

- **Constant-fraction $d_{\mathbf{attn}}$ (e.g., $d_{\mathbf{attn}} = 768$):** $M_{\mathrm{Q4}} \approx 3.0$ GiB, for an overall reduction of $\sim 21\times$.
- **Fixed-rank $d_{\mathbf{attn}}$ (e.g., $d_{\mathbf{attn}} = 96$):** $M_{\mathrm{Q4}} \approx 0.38$ GiB, for an overall reduction of $\sim 168\times$.

The architectural contribution is the *dimension reduction* (the ratio $4096/d_{\mathrm{attn}}$); the additional factor of $4\times$ comes from standard $16 \to 4$-bit quantization.

Heterogeneous KV-cache quantization (decoupled). A practical benefit of decoupling is that it enables *heterogeneous* KV-cache quantization: we can compress the semantic path more aggressively (e.g., Q4) while keeping the geometric (RoPE) path at higher fidelity (e.g., Q8). As a lightweight sanity check, we compared FP16 caches to a decoupled policy with $K_{\mathrm{sem}} = \mathrm{Q4}$, $K_{\mathrm{geo}} = \mathrm{Q8}$, $V = \mathrm{Q4}$ and a 128-token FP16 residual window. On a small held-out calibration slice from FineWeb-Edu, this policy yielded $\Delta\mathrm{NLL} \approx 0.015$ nats/token (perplexity ratio ≈ 1.015) and $\mathrm{KL}(p_{\mathrm{FP16}} \| p_{\mathrm{quant}}) \approx 0.006$ nats/token, while preserving identical greedy decoding for short prompts. These guardrail metrics suggest that aggressive semantic compression can be applied without catastrophic degradation, even though stochastic sampling trajectories may diverge due to small distribution shifts.

While we report training throughput in our experiments, the theoretical FLOPs reduction in the attention mechanism ($O(n^2 d) \to O(n^2 r)$) implies a proportional speedup in the *prefill phase* of inference, where the KV-cache is populated. For autoregressive decoding, the memory bandwidth savings from the smaller cache dominate latency improvements.

3 Experiments

3.1 Experimental Setup

Why two suites? We use a two-stage experimental strategy: the **v21 ablation suite** (WikiText-2, smaller model) enables rapid iteration and mechanistic insight, while the **v29 scaling suite** (FineWeb-Edu, larger model) tests whether the same architectural ideas hold at higher scale. Because these suites differ in model size, data distribution, and training details, we report them separately and avoid direct numerical comparison across suites.

Model Configuration. We report results from two experiment suites:

- **v21 (ablation suite):** a smaller 6-layer model trained on WikiText-2 for rapid architectural iteration.
- **v29 (scaling suite):** a larger 12-layer model trained on FineWeb-Edu (100M tokens) to validate scaling behavior.

Datasets.

- **WikiText-2**: 2M tokens of Wikipedia text used for rapid prototyping in the v21 ablation suite.
- **FineWeb-Edu**: 100M tokens of educational web content used for the v29 scaling suite. In our implementation, we use a 50,257-token vocabulary (GPT-2 compatible) for tokenized data.

Training. Unless otherwise noted, models are trained for 6000 steps with gradient clipping at 1.0. Exact hyperparameters and configurations are recorded in the run logs.

Table 1: WikiText-2 Validation Loss Comparison (v21 suite; 6-layer model)

Model	**Attn Config**	**Params**	**Val Loss**	**Tok/s**
Standard Baseline	$d = 512$	31.8M	5.37	20k
Combined 96	$d_{\text{attn}} = 96$	30.1M	**5.33**	117k
Bottleneck 128	$d_{\text{attn}} = 128$	31.3M	5.48	128k
Decoupled 32/64	d_{sem}=32, d_{geo}=64	30.9M	5.59	106k
GQA (kv=2)	8Q/2KV heads	30.1M	5.63	25k
Small Model	$d_{\text{model}} = 128$	4.2M	5.74	930k

3.2 v21 Ablation Suite: WikiText-2 Results

Key Finding (v21). Table 1 shows that the Combined 96 bottleneck achieves the *lowest* validation loss (5.33), outperforming the full-rank baseline (5.37). This supports the hypothesis that attention routing can be learned in a substantially lower-dimensional interaction space.

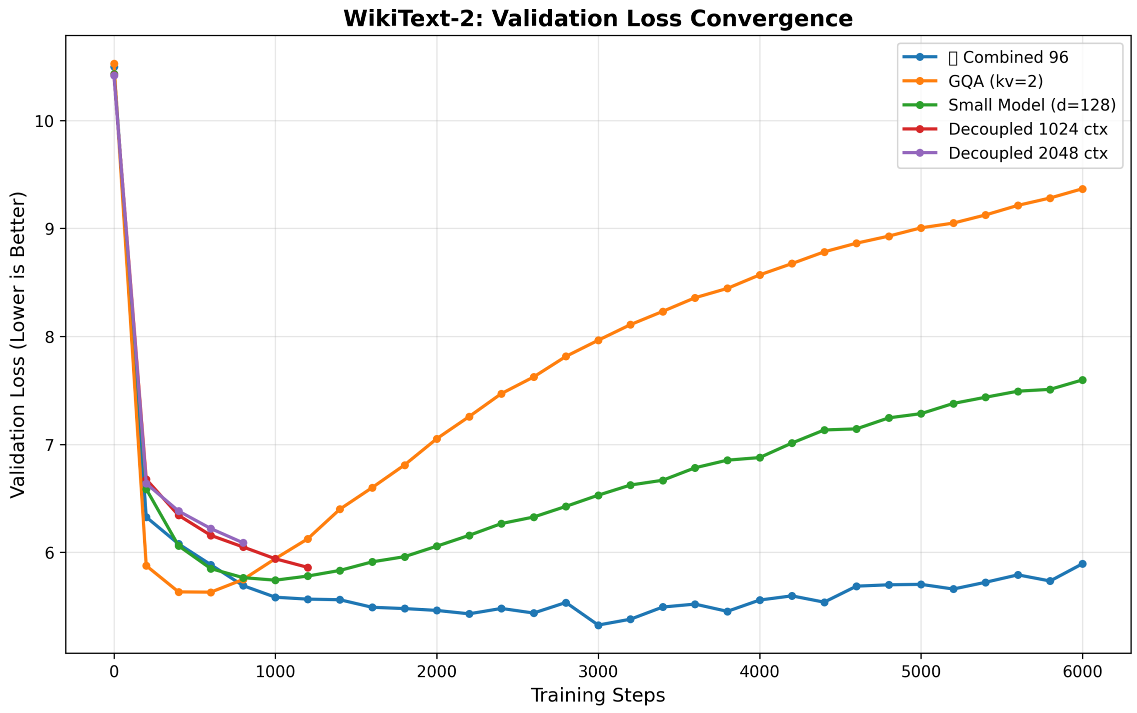

Figure 1: Validation loss curves on WikiText-2 (v21 suite). Bottlenecked models converge rapidly; Combined 96 achieves the best final loss.

3.3 v29 Scaling Suite: FineWeb-Edu Results

To validate that our findings generalize beyond small datasets, we train on 100M tokens from FineWeb-Edu (Figure 2).

Scaling Observation. Surprisingly, on FineWeb-Edu (100M tokens), the Decoupled Bottleneck model *outperformed* the Standard Baseline, a parameter-matched GQA model, and a matched-rank Bottleneck baseline, achieving a **37% reduction in perplexity**. This contradicts the common assumption that compression must hurt performance. We hypothesize that standard high-dimensional attention can be sufficiently over-parameterized to hinder optimization at this scale, whereas explicitly reducing the interaction rank (and disentangling content from positional geometry) acts as a regularizer that forces the model to learn more robust features.

Table 2: FineWeb-Edu Validation Loss (100M tokens, 1024 context, 6000 steps; v29 suite)

Model	**Attn Config**	**Params**	**Val Loss**	**Val PPL**	**Δ**
Standard Baseline	$d = 512$	139.8M	6.326	559.0	—
GQA (kv=2)	12Q/2KV	128.0M	6.302	545.9	-2.4%
Bottleneck (rank 144)	$d_{\text{attn}} = 144$	116.8M	6.127	458.1	-18.0%
Decoupled 48/96	d_{sem}=48, d_{geo}=96	**116.8M**	**5.862**	**351.5**	**-37.1%**

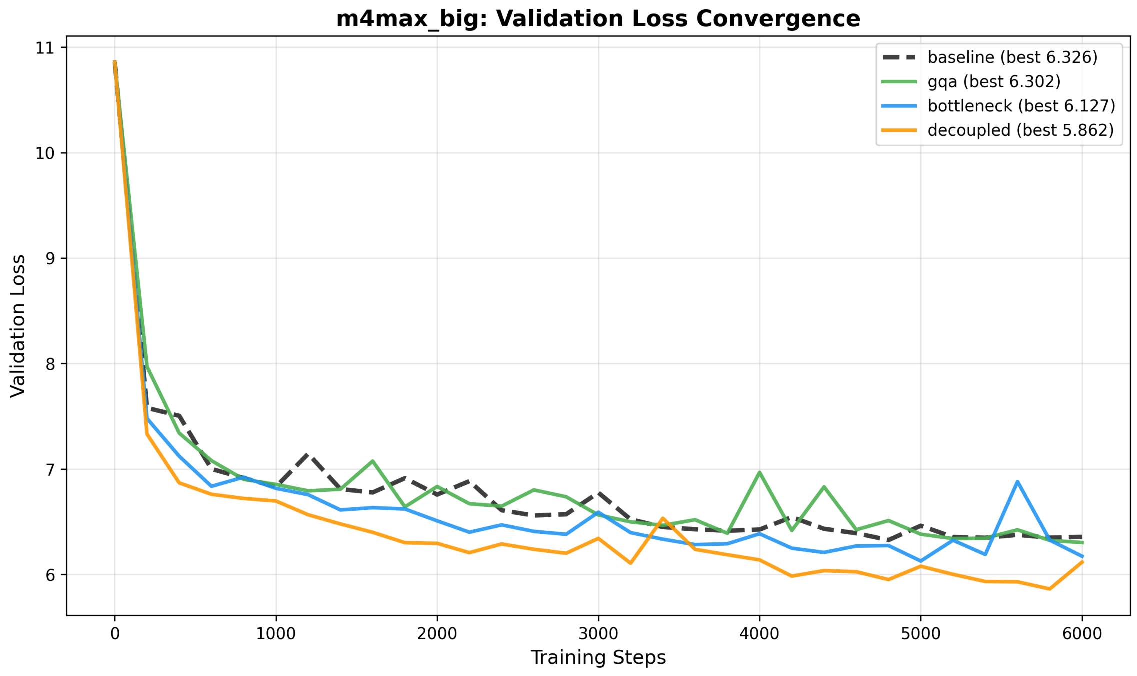

Figure 2: Validation loss curves on FineWeb-Edu (100M tokens). Under the same training setup, the Decoupled Bottleneck model converges to the lowest validation loss, outperforming both the standard baseline and GQA.

3.4 Final Loss Comparison

Figure 3 provides a direct comparison of final validation losses across our FineWeb-Edu (v29) architectures. The Decoupled Bottleneck (48/96) achieves the best validation loss among baseline, GQA, and bottleneck variants.

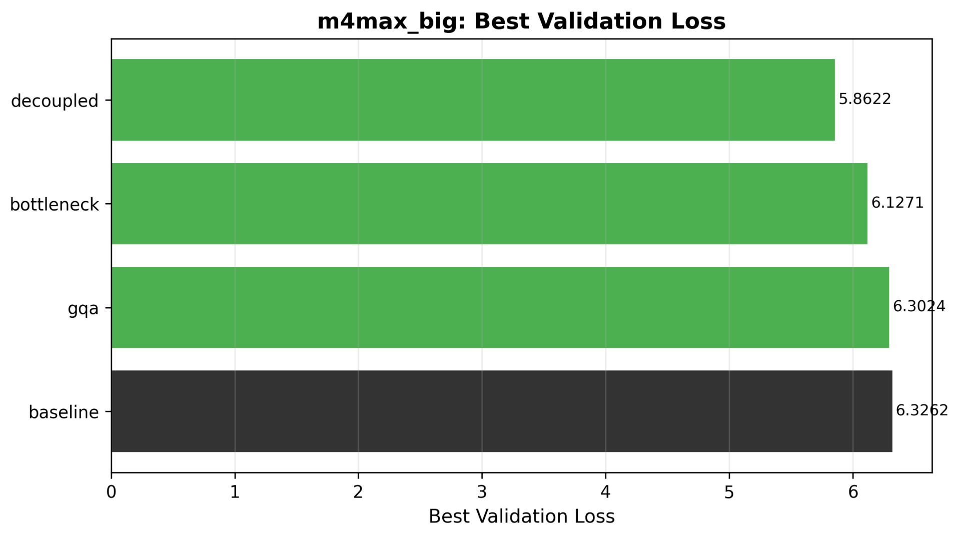

Figure 3: Final validation loss comparison (FineWeb-Edu, v29). Lower is better. Decoupled 48/96 achieves the best loss, outperforming both the standard baseline and a parameter-matched GQA baseline (kv=2).

3.5 Ablation Studies

Wide Residual Stream Hypothesis. Comparing "Small Model" ($d_{\text{model}} = 128$) to "Bottleneck 128" ($d_{\text{model}} = 512$, $d_{\text{attn}} = 128$), we observe a 0.26 loss gap (5.74 vs 5.48) and severe overfitting in the small model. This confirms that the *residual stream* must remain wide; only the *attention interaction* can be compressed.

GQA vs. Bottleneck (Head Sharing vs. Interaction Rank). We compared our method against Grouped-Query Attention (GQA) with matched KV memory footprints on FineWeb-Edu. While GQA reduces KV cache storage by sharing key-value heads, it retains full-rank query projections and attention scoring in high-dimensional space. In contrast, the Bottleneck architecture reduces the interaction dimension directly. In our runs, GQA (kv=2) achieved val loss 6.302 (ppl 545.9) with a measured KV cache of $\sim$ 0.75 GB at 128k context, while the Bottleneck (rank 144) achieved val loss 6.127 (ppl 458.1) with a comparable KV cache of $\sim$ 0.84 GB. This suggests that reducing the *interaction rank* is more beneficial than merely reducing the *head count*.

Decoupled vs. Bottleneck (Separating Content and Geometry). Holding the same total attention dimension fixed ($d_{\text{attn}} = 144$), the Decoupled Bottleneck model (48/96) further improves validation loss to 5.862 (ppl 351.5) at essentially the same KV cache footprint ($\sim$ 0.84 GB at 128k context). This suggests that the semantic/geometric separation is not only a memory strategy but can also improve optimization and generalization at this scale.

Long Context Stability. We verify that the geometric path handles extended context correctly:

- 1024 context: Val Loss 5.86 (converged smoothly)

- 2048 context: Val Loss 6.09 (converged smoothly)

The higher loss is expected due to reduced batch size; the key observation is stable training with RoPE on 64 dimensions.

3.6 Memory-Quality Trade-off

Figure 4 shows the quality–efficiency trade-off for our FineWeb-Edu (v29) runs. For these experiments, KV-cache memory at 128k context is measured directly from the implementation, and we compare it against the best validation loss achieved by each architecture. Notably, Decoupled 48/96 matches the Bottleneck's KV footprint (same total d_{attn}) while improving quality, and it outperforms GQA despite GQA's smaller KV cache.

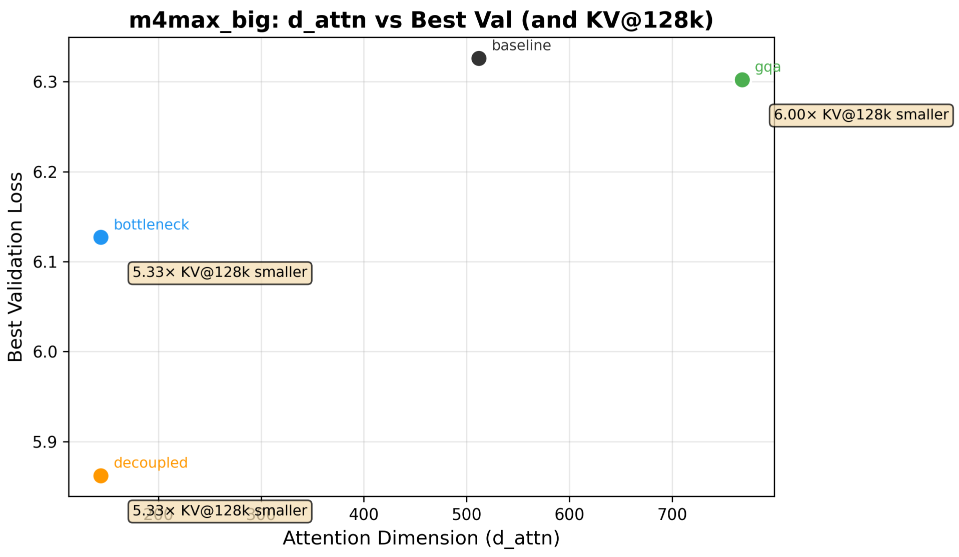

Figure 4: Quality–efficiency comparison on FineWeb-Edu (v29). Points show best validation loss versus attention dimension (d_{attn}), with annotations derived from measured KV-cache memory at 128k context.

3.7 Memory Footprint Analysis

Table 3 shows the KV-cache memory requirements for a 128k context at Llama-7B scale (32 layers, $d = 4096$):

Table 3: KV-Cache Memory for 128k Context (Llama-7B Scale)

Architecture	**VRAM**	**Compression**
Standard (FP16)	64.0 GB	1×
GQA 8× (FP16)	16.0 GB	4×
MLA (FP16)	4.3 GB	15×
Bottleneck (FP16)	1.5 GB	43×
Decoupled (Q4, fixed-rank d_{attn}=96)	**0.38 GB**	**168×**
Decoupled (Q4, constant-fraction d_{attn}=768)	**3.0 GB**	**21×**

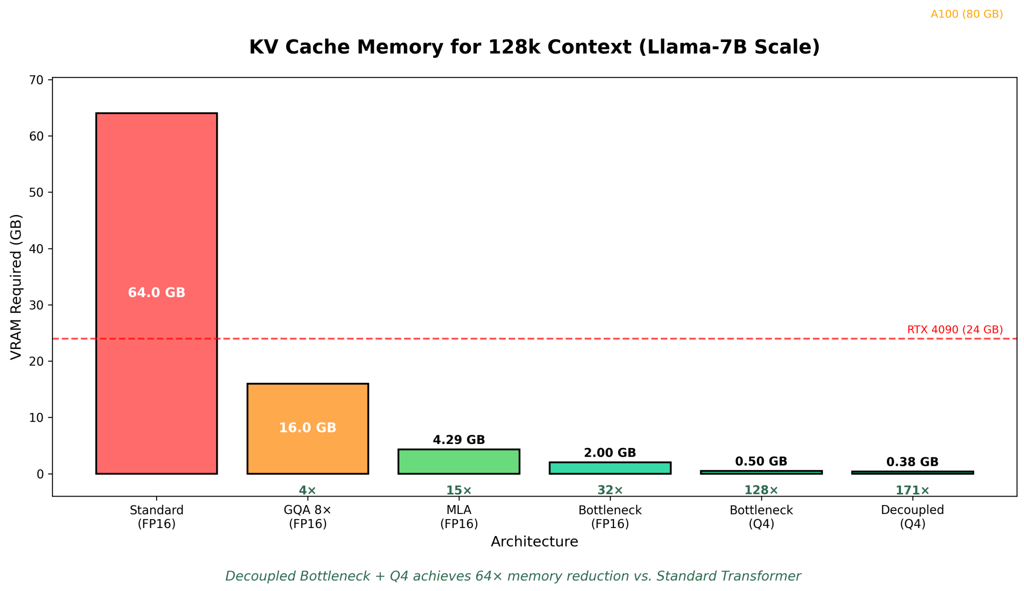

Figure 5: KV-cache memory comparison at 128k context (illustrative). Depending on whether d_{attn} scales with d_{model} or is held fixed, the end-to-end FP16→Q4 reduction ranges from $\sim 21\times$ (constant fraction) up to $\sim 168\times$ (fixed-rank).

4 Discussion

4.1 Why Does Low-Rank Attention Work?

We hypothesize two complementary explanations:

Intrinsic Dimensionality. Following Aghajanyan et al. [1], natural language representations lie on low-dimensional manifolds. The attention mechanism's role is *routing*—selecting which tokens to aggregate—not computing complex transformations. Routing decisions are inherently low-entropy and thus low-rank.

Regularization Effect. The bottleneck acts as an implicit regularizer, preventing the model from memorizing spurious token-pair correlations. In our v29 FineWeb runs, reducing the interaction rank improves generalization relative to the full-rank baseline.

Gradient Rank Dynamics. AdaRankGrad [12] proves that gradient rank decreases monotonically during training, eventually approaching rank one. This suggests that *architectural* bottlenecks become increasingly appropriate as training progresses—the model naturally "wants" to operate in a low-rank subspace. By hard-wiring this constraint from the start, we may accelerate convergence by matching the architecture to the optimization landscape.

4.2 When to Use Each Architecture

Our experiments reveal a two-stage story: v21 ablations on WikiText-2 are useful for rapid iteration and mechanistic insight, while v29 FineWeb-Edu results validate the key claims at larger scale.

- **Decoupled Bottleneck:** On FineWeb-Edu (v29), Decoupled 48/96 achieves the best validation loss among the tested variants (baseline, GQA, and bottleneck) while preserving the KV memory benefits of low-rank attention. It is also the *only* architecture enabling

heterogeneous quantization—e.g., Q4 for semantic and Q8 for geometric paths—which is critical for long-context inference deployments.

- **Standard Attention:** A strong baseline and simplest implementation, but can be memory-inefficient for long contexts.

Recommendation. For *training*, use the v21 ablation suite to quickly explore attention-rank and decoupling choices, then validate at scale with the v29 FineWeb suite. For *inference* under memory constraints, convert to Decoupled with heterogeneous quantization (aggressively compress semantic, preserve geometric fidelity).

4.3 Limitations

- **Sensitivity to setup:** Results can depend on the training recipe and implementation details. Earlier (v21) FineWeb runs showed a degradation for decoupled attention, whereas the v29 decoupled model improves substantially. This motivates replication across seeds, longer training, and additional datasets.
- Experiments are limited to 512-dim models. Verification at 7B+ scale is needed.
- The optimal $(d_{\text{sem}}, d_{\text{geo}})$ split may vary with model scale.
- We have not evaluated on downstream tasks (e.g., MMLU, HellaSwag).
- Throughput measurements are from training; inference latency benchmarks are future work.

5 Conclusion

We have demonstrated that attention in Transformers contains significant redundancy. On FineWeb-Edu (100M tokens), the Decoupled Bottleneck architecture **surpassed** the standard baseline, a parameter-matched GQA baseline, and a matched-rank Bottleneck baseline, suggesting that for many practical scales, standard attention is not just inefficient but suboptimal.

The core insight is architectural: **Attention is a router, not a processor.** The heavy computation should happen in the feedforward layers (which we leave at full rank), while attention merely selects which tokens to aggregate. By matching the architecture to this functional role, we unlock dramatic efficiency gains.

Our Decoupled Bottleneck Attention separates semantic matching from positional geometry, allowing aggressive compression on the former while preserving RoPE fidelity on the latter. Combined with 4-bit quantization, this enables 128k-context inference on consumer hardware (Figure 5)—transforming a datacenter problem into a laptop-solvable one.

Future Work. We plan to: (1) validate at 7B+ scale where the efficiency gains compound; (2) explore learned mixing weights between semantic and geometric paths; (3) test robustness across seeds, longer training, and additional datasets to understand when decoupling improves optimization; and (4) benchmark inference latency on production hardware.

Statements and Declarations

Conflict of Interest. The author declares no competing interests. This research was conducted independently without corporate affiliation or funding from entities with financial interests in the outcomes.

Data Availability. All datasets used in this study are publicly available: WikiText-2 [10] is available from Salesforce Research, and FineWeb-Edu is available from Hugging Face. The code, trained model checkpoints, and all experimental logs are available at `https://github.com/theapemachine/experiments`.

Funding. This research was conducted without external funding. All computational resources were provided by the author.

References

[1] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *ACL*, 2021.

[2] 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. *EMNLP*, 2023.

[3] Srinadh Bhojanapalli et al. Low-rank bottleneck in multi-head attention models. *ICML*, 2020.

[4] David Chiang and Dani Yogatama. The rotary position embedding may cause dimension inefficiency in attention heads for long-distance retrieval. *arXiv preprint*, 2025.

[5] DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

[6] Georgi Gerganov et al. 4-bit kv cache implementation. `https://github.com/ggml-org/llama.cpp/pull/7412`, 2024. llama.cpp PR#7412: Production Q4_0/Q8_0 KV cache support.

[7] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *ICLR*, 2021.

[8] Edward J Hu et al. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[9] Seijin Kobayashi, Johannes von Oswald, and João Sacramento. Weight decay induces low-rank attention layers. *NeurIPS*, 2024.

[10] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016. Introduces WikiText-2 and WikiText-103 benchmarks.

[11] Jongchan Park et al. Bam: Bottleneck attention module. *BMVC*, 2018.

[12] Yehonathan Refael, Jonathan Svirsky, Boris Shustin, Wasim Huleihel, and Ofir Lindenbaum. Adarankgrad: Adaptive gradient-rank and moments for memory-efficient llms training and fine-tuning. *arXiv preprint arXiv:2410.17881*, 2024.

[13] Jianlin Su et al. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[14] Turboderp. Quantized kv cache evaluation. `https://github.com/turboderp/exllamav2/blob/master/doc/qcache_eval.md`, 2024. ExLlamaV2 implementation showing Q4 cache matches FP16 quality.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[16] Yizhou Wang et al. Attention layers add into low-dimensional residual subspaces. *arXiv preprint*, 2025. Shows attention outputs are approximately 60% low-dimensional.