Bottleneck Attention: Hard-Wiring Low-Dimensional Routing in GPT-Style Transformers

Daniel Owen van Dommelen
(contact omitted)

December 2025

Abstract

Adaptive Low-Rank Training (ALRT) suggests that Transformer attention contains extreme redundancy: in prior experiments, query/key (Q/K) projections compress to ranks ≈ 9–14 with mean ≈ 11.2 out of 512, implying that attention pattern computation may live in a small comparison subspace. Motivated by this signal and a first-principles "question everything" approach to architecture design, we test a minimal hypothesis: decouple residual stream width d_{model} from attention subspace dimension d_{attn}, using $Q, K, V : \mathbb{R}^{d_{\text{model}}} \to \mathbb{R}^{d_{\text{attn}}}$ with $d_{\text{attn}} \ll d_{\text{model}}$, and map attention output back via $\mathbb{R}^{d_{\text{attn}}} \to \mathbb{R}^{d_{\text{model}}}$. On a controlled WikiText-2 word-level setup (6-layer GPT-style LM; $d_{\text{model}} = 512$, 8 heads, context 256), reducing d_{attn} from 512 to 128 ($4\times$) incurs only a modest degradation in best validation loss (+1.8% with a learned "null" key/value sink), while compressing to $d_{\text{attn}} = 32$ ($16\times$, with null) remains viable (+4.4%). Notably, bottleneck configurations *learn faster early*: within the first few hundred steps, validation perplexity is consistently lower than the baseline, suggesting that the conventional full-width attention subspace may introduce optimization noise. Because attention-module compute and KV-cache memory scale linearly with d_{attn}, these results support the interpretation that (in this regime) attention is primarily a low-dimensional routing primitive distinct from high-rank feature processing in MLP blocks.

1 Introduction

Transformers remain architecturally conservative despite dramatic changes in scale, hardware, and use cases since their introduction for machine translation [8]. ALRT was motivated by a simple suspicion: much of this architecture may be redundant, and the redundancy may not be uniform across components. Empirically, ALRT finds strong projection-type asymmetry: attention Q/K projections compress far more aggressively than attention output and MLP projections, suggesting that *routing* may be intrinsically low-dimensional while *writing back* is not [6, 7].

This work tests the most literal architectural consequence of that observation: if attention pattern computation truly lives in a small subspace, we can *hard-wire* that by setting an explicit attention subspace dimension d_{attn} that is decoupled from the residual stream width d_{model}. Unlike dynamic resizing (which complicates optimizer state), this is a static architectural change: train from scratch with the bottleneck baked in.

2 Background: ALRT and the low-rank attention signal

ALRT parameterizes linear maps as low-rank factors $W = UV$ and adapts rank during training using stable rank estimation:

$$r_{\text{stable}}(W) = \frac{\|W\|_F^2}{\sigma_{\max}(W)^2}.$$

Across experiments on GPT-style language models, ALRT reports that Q/K projections are the most compressible, with mean ranks ≈ 11.2 out of 512, while attention output and MLP projections retain higher ranks [6]. A natural architectural interpretation is that attention's *comparison space* (used to compute patterns) is low-dimensional, even if the residual stream is not.

3 Method: Bottleneck Attention

3.1 Decoupled attention subspace

We replace full-width Q/K/V with projections into a smaller subspace:

$$W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}}, \qquad W_O \in \mathbb{R}^{d_{\text{attn}} \times d_{\text{model}}}.$$

Attention computation (scores and application) now scales in d_{attn}, while the residual stream stays at d_{model}. This directly targets compute, parameters, and KV-cache memory in attention.

3.2 "Attend nowhere" via a null key/value (optional)

Softmax attention forces each query to allocate probability mass across keys. We optionally append learnable per-head vectors $k_\emptyset, v_\emptyset$ to the Key/Value sequence:

$$K' = [K;\, k_\emptyset], \qquad V' = [V;\, v_\emptyset],$$

so that the attention distribution can allocate probability mass $p_\emptyset$ to a "no-op" sink without simulating this behavior implicitly. In practice, $v_\emptyset$ is initialized near zero so that attending to the null slot initially contributes little and becomes useful only if learned.

Table 1: WikiText-2 results (word-level). Best validation metrics over training.

Model	d_{attn}	Params (M)	Best val
Baseline	512	36.06	5.
Bottleneck	128	31.34	5.
Bottleneck + Null	128	31.34	5.
Bottleneck + Null + Tied QK	128	30.95	5.
Extreme + Null	32	30.16	5.

3.3 Learned per-head temperature (optional)

When d_{attn} is very small, dot products can become high-variance and softmax can saturate. We include a learned per-head logit scale (equivalently, a learned temperature) to stabilize training in tiny head dimensions.

4 Experimental setup

Data and tokenization. WikiText-2 with a minimal word-level whitespace tokenizer with `<eos>` and `<unk>`. A single `wiki.train.tokens` file is split 90/10 into train/validation.

Model. Decoder-only GPT-style Transformer with pre-norm blocks: 6 layers, $d_{\text{model}} = 512$, 8 heads, $d_{\text{ff}} = 2048$, context length 256, dropout 0.1, tied input/output embeddings.

Optimization. AdamW with learning rate 3×10^{-4}, weight decay 0.01, batch size 32, 200 steps/epoch, 30 epochs (6000 steps), gradient clipping at 1.0. Evaluation occurs every 200 steps.

5 Results

5.1 Accuracy–efficiency trade-off

Table 1 reports the main comparisons and ablations at $d_{\text{attn}} = 128$.

5.2 Early convergence anomaly

A striking and strategically important observation is that bottleneck models often *outperform* the baseline during early training. For example, at step 200 the baseline achieves validation perplexity 582.1, while $d_{\text{attn}} = 128$ achieves 518.6 and $d_{\text{attn}} = 32$ with a null slot achieves 519.2. This contradicts a common "wide hallway" intuition (that larger attention subspaces are needed to find good solutions even if the endpoint is low-rank). Instead, it suggests that over-provisioned attention subspaces may introduce gradient noise or redundant degrees of freedom that slow early feature acquisition.

Figure 1 plots validation perplexity over training, with an inset focusing on the first 500 steps.

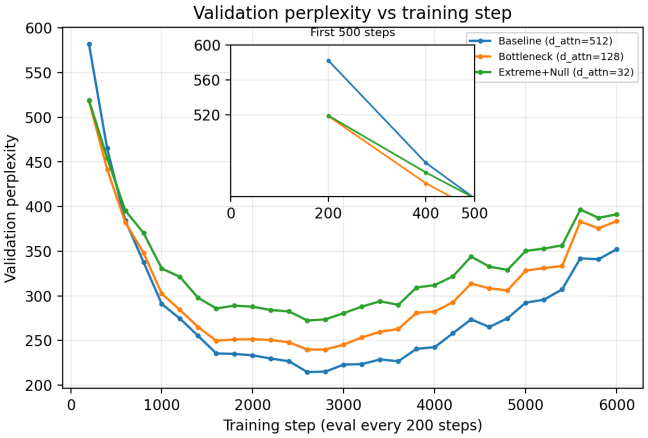

Figure 1: Validation perplexity over training. Inset highlights the first 500 steps, where bottleneck variants are consistently below baseline (e.g., step 200: 582.1 vs 518.6/519.2).

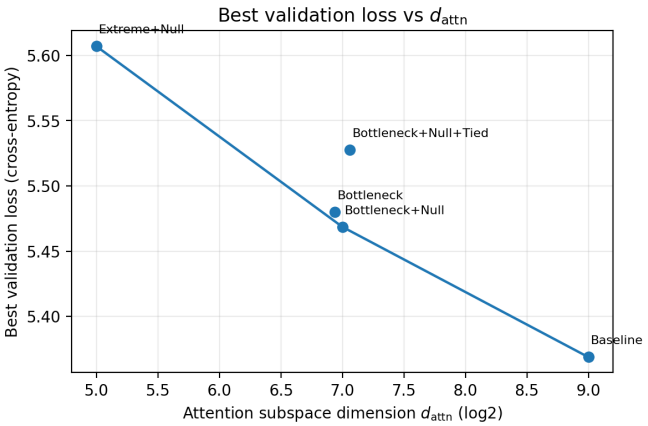

Figure 2: Best validation loss vs d_{attn} ($\log_2$ x-axis).

5.3 Pareto curve

Figure 2 shows best validation loss vs d_{attn} on a log scale. Figure 3 shows a simple parameter–loss view.

6 Discussion and limitations

Why early convergence improves. The early-training advantage of bottleneck variants suggests that, in this regime, the conventional choice $d_{\text{attn}} = d_{\text{model}}$ may over-parameterize the routing mechanism, making optimization less well-conditioned. A plausible hypothesis is that constraining the interaction subspace reduces redundant degrees of freedom in Q/K (which ALRT already indicates are effectively low-rank), thereby reducing gradient noise and accelerating the acquisition of simple routing behaviors (e.g., local n-gram and syntactic dependencies).

Concrete KV-cache math. In decoder-only inference with KV caching, each layer stores Key and Value vectors per token.

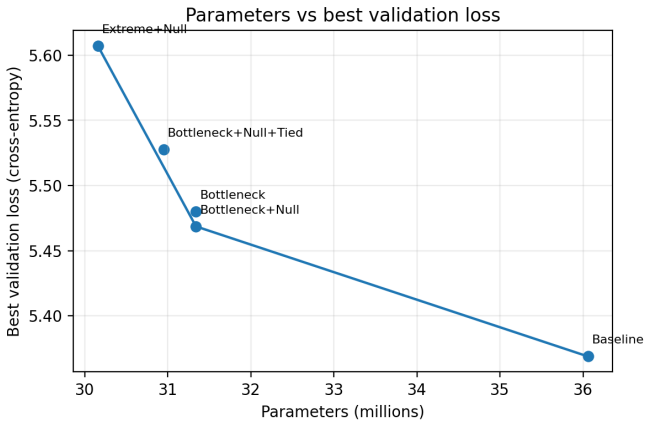

Figure 3: Parameters (M) vs best validation loss.

For standard multi-head attention, per-token KV state is

$$\text{KV per token per layer} = 2 \cdot n_{\text{head}} \cdot d_{\text{head}} = 2\, d_{\text{attn}}.$$

Therefore total KV-cache memory for batch size B, context length T, L layers, and element size b bytes is

$$M_{\text{KV}} = 2\, B\, T\, L\, d_{\text{attn}}\, b,$$

which scales *linearly* in d_{attn}. Reducing d_{attn} from 512 to 128 cuts KV-cache by 75% (4× smaller); at a hypothetical 128k-context regime, this allows $\approx 4\times$ larger batch size (or $\approx 4\times$ longer context) for the same memory budget, directly targeting a dominant modern inference bottleneck.

Long-context scaling risk (Johnson–Lindenstrauss). These experiments use context length 256 and learned absolute positional embeddings. If attention needs to reliably discriminate among N candidate keys, random-projection theory suggests the dimensionality required to preserve pairwise distances scales as $O(\log N)$ (e.g., Johnson–Lindenstrauss-type bounds [4]). Thus $d_{\text{attn}} = 32$ may saturate at much longer sequence lengths, and the optimal d_{attn} may increase (slowly) with context length. Interactions with alternative positional encodings (RoPE/ALiBi) and long-context regimes remain future work. We report single runs; multi-seed sweeps are needed for confidence intervals.

7 Related work

Low-rank adaptation methods (e.g., LoRA [3]) show that many learned transformations can be modified via low-dimensional updates. Separately, efficient-attention methods address quadratic attention cost via sparsity or kernel/low-rank approximations (e.g., Reformer [5], Linformer [9], Performer [1]). Our goal differs: we test whether *standard attention* is over-provisioned in its internal dimensionality even at modest context length.

Multi-Head Latent Attention (MLA). DeepSeek-V2 introduces Multi-Head Latent Attention (MLA), which compresses the Key–Value cache into a latent vector for more efficient inference [2]. Bottleneck Attention is complementary: rather than only compressing the *stored* KV state, we compress the *interaction space* itself by decoupling d_{attn} from d_{model}. Together, these lines of work suggest that both the memory representation and the routing computation in attention may be substantially lower-dimensional than standard practice assumes.

8 Conclusion

ALRT indicates that attention pattern computation is intrinsically low-rank, with Q/K compressing to mean ranks $\approx 11.2/512$. Bottleneck Attention operationalizes this as a static architectural change: compute attention in a reduced subspace $d_{\text{attn}} \ll d_{\text{model}}$ while preserving a wide residual stream. On WikiText-2 (word-level), $d_{\text{attn}} = 128$ and even $d_{\text{attn}} = 32$ (with a null slot) remain viable with modest loss penalties, implying that the standard convention $d_{\text{attn}} = d_{\text{model}}$ is significant structural bloat in this regime. The early-training speedup further suggests that full-width attention subspaces may not just be redundant, but actively suboptimal for optimization.

Reproducibility

```
python3.12 v19_transformer_attn_bottleneck.py \
  --data ./wiki.train.tokens --out-dir runs/v19_bas

python3.12 v19_transformer_attn_bottleneck.py \
  --data ./wiki.train.tokens --out-dir runs/v19_att

python3.12 v19_transformer_attn_bottleneck.py \
  --data ./wiki.train.tokens --out-dir runs/v19_att
  --attn-dim 32 --null-attn
```

References

[1] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Łukasz Kaiser, et al. Rethinking attention with performer's random features. In *International Conference on Learning Representations (ICLR)*, 2021.

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

[3] Edward J Hu, Yelong Shen, Phil Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[4] William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. In *Conference in*

Modern Analysis and Probability, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society, 1984. doi: 10.1090/conm/026/737400.

[5] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations (ICLR)*, 2020.

[6] Daniel Owen van Dommelen. Adaptive low-rank training via spectral rank estimation: Dynamic parameter efficiency in transformer language models (technical report), 2025. Internal technical report, December 2025.

[7] Daniel Owen van Dommelen. Adaptive low-rank training — technical report v2, 2025. Internal technical report, December 2025.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[9] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.