REVISITING THE SHAPE CONVENTION OF TRANSFORMER LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

The architectural shape of dense Transformers has remained remarkably stable: narrow-wide-narrow feed-forward networks (FFNs) consume most non-embedding parameters. Motivated by theoretical and empirical evidences that residual wide-narrow-wide (hourglass) MLPs remain expressive despite bottlenecks, we revisit whether this architectural convention is necessary for dense language models. We study Hourglass Transformers, which replace the conventional FFN with residual stacks of hourglass sub-MLPs and use hourglass attention to decouple residual-stream width from attention width. This exposes a practical depth-width trade-off: compressing the FFN intermediate dimension allows wider hidden states and fewer layers at matched parameter budgets. Across model scales from 113M to 8B parameters, Hourglass Transformers achieve language-modeling and downstream performance comparable to conventional Transformers, while improving training compute efficiency by 8.7% at matched average downstream accuracy across the 906M, 3B, and 8B scales. After long-context extension, the 8B Hourglass model also outperforms its matched conventional baseline across 4k–64k context lengths. At 64k context, the reduced attention layer count lowers both computation and KV-cache requirements, yielding up to $1.93\times$ faster token decoding and 50% lower KV-cache memory at the 1B scale. These results identify hourglass structures as a practical architecture-efficiency alternative for compute- and latency-conscious Transformer design.

1 INTRODUCTION

Despite rapid advances in scale and training methodology, the architectural shape of dense Transformer language models has remained remarkably stable since early scaling studies (Kaplan et al., 2020). Modern models consistently adopt a narrow–wide–narrow multilayer perceptron (MLP) in feed-forward network (FFN), expanding the model dimension d_{model} to an intermediate width d_h before projecting back, with expansion ratios typically fixed between 2 and 4 (Vaswani et al., 2017; Kaplan et al., 2020). This design choice has become a de facto standard across contemporary dense LLMs (Touvron et al., 2023; Qwen et al., 2025; Team et al., 2025; OLMo et al., 2025). At the same time, the FFN dominates parameter allocation in Transformers, accounting for a majority of model parameters relative to attention. As a result, the MLP shape implicitly determines how capacity is distributed between depth, width, and attention.

While the shape convention in Transformer is effective, we raise a fundamental question: *is the narrow–wide–narrow MLP shape in FFN essential, or merely one convenient instantiation of a residual-compatible transformation?* Recent results suggest that this convention may be unnecessarily restrictive. Theoretical studies have shown that residual MLPs with wide–narrow–wide (hourglass) structures can act as optimal function approximators, including universality results for networks with extremely narrow intermediate layers (Lin & Jegelka, 2018; Liu et al., 2024a). From an expressivity perspective, such hourglass architectures motivate deeper residual compositions at fixed parameter budgets, a setting known to favor rapid growth in linear regions and representational complexity (Hanin & Rolnick, 2019; Joyce & Verschelde, 2026). Moreover, operating in a wider residual embedding space before and after bottleneck transformations is consistent with the intuition behind Cover's theorem and random feature methods: high-dimensional representations can make useful distinctions easier to express Cover (1965); Rahimi & Recht (2007). These complementary arguments are not a substitute for language-model evidence, but they motivate treating bottleneck residual transformations as viable candidates rather than degenerate approximations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

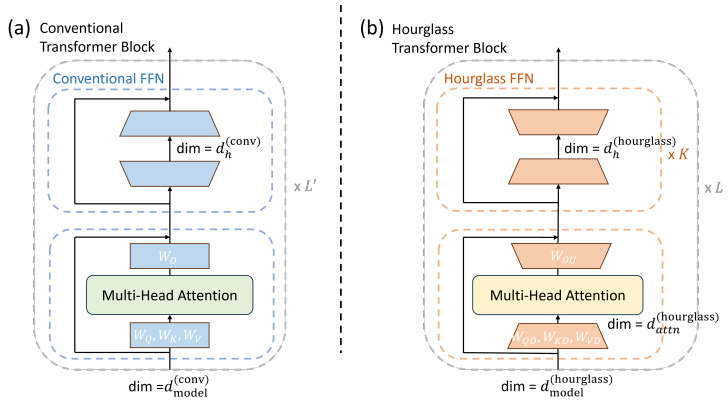


Figure 1: **Overview: revisiting the shape convention of Transformer through studying the relaxation of shape in FFN and attention.** Inspired by (Liu et al., 2024a; Chen et al., 2025), we compare conventional and hourglass variants of Transformer architectures. (a) Conventional Transformer Block with L' layers, consisting of an attention module and a conventional FFN with a narrow-wide-narrow MLP. (b) Hourglass Transformer Block with L layers, consisting of an hourglass attention module and an hourglass FFN with K hourglass-shaped MLP sub-blocks. We explore the design space by tuning parameters such as d_{model}, d_h, K, and L, allowing the Hourglass layer count L to differ from the baseline L' while fixing the Hourglass $d_{attn}^{(\text{hourglass})} = d_{\text{model}}^{(\text{conv})}$.

Empirical evidence has also begun to challenge conventional MLP shape assumptions. Motivated by high-dimensional representation learning, Chen et al. (2025) demonstrates that hourglass-shaped MLPs outperform standard designs in generative vision models. Related work further shows that expert-routed FFNs with hourglass MLPs in Mixture-of-Experts Transformers can surpass dense baselines (Wang et al., 2024) at scale. More broadly, the principle of allocating parameters through wide–narrow–wide pathways has proven effective in convolutional architectures Zagoruyko & Komodakis (2016); Zhou et al. (2020) and parameter-efficient methods such as LoRA Hu et al. (2022). Together, these results indicate that hourglass-shaped transformations constitute a strong and general class of building blocks, motivating a re-examination of shape conventions in Transformer.

In this work, we revisit the shape convention of Transformer by substituting both the conventional FFN and attention module with hourglass variants. The hourglass FFN replaces the single narrow–wide–narrow MLP with a stack of K wide–narrow–wide sub-MLPs connected by residuals; the hourglass attention module projects the embedding to a lower dimension before applying attention, allowing sequence mixing to operate below the widened residual dimension. Stacking K residual sub-blocks deepens the compositional structure of the FFN, a regime where representational complexity can grow rapidly with depth (Joyce & Verschelde, 2026). Practically, the inversion gives an additional architectural knob: FFN bottleneck width, residual-stream width, and the number of sequential Transformer layers can be traded against one another under a fixed parameter or FLOP budget. Furthermore, recent depth–width studies have shown that, at fixed parameter count, deeper Transformers achieve lower perplexity than wider ones, but the benefit of additional layers saturates rapidly while the latency cost grows linearly in depth (Petty et al., 2024). The hourglass FFN shifts this balance by making it possible to use a wider d_{model} with fewer Transformer layers, the regime where latency savings are greatest and depth returns may already be diminished. This leads us to ask: *can inverting the FFN shape exploit the depth–width trade-off more favorably, reducing training compute and latency while preserving quality?*

To answer this question, we construct Transformer variants with hourglass FFN and hourglass attention modules (collectively referred to as Hourglass Transformers) and conduct experiments across model scales from 113M to 8B parameters. [1] Controlled studies at 113M isolate the contribution of FFN parameterization, while design-space sweeps at 403M/906M examine how hourglass FFN/Attn together support fewer-layer, wider-residual configurations. Scaling to 906M, 3B, and 8B parameters, we evaluate the complete Hourglass architecture recipe against standard Transformer baselines: Hourglass Transformers remain competitive across language modeling and question answering benchmarks, while improving training compute efficiency by 8.7% at matched average

[1] Code will be released upon publication.

downstream accuracy. At a 64k-token context, the reduced attention-layer count lowers KV-cache memory by 50.0%, 35.3%, and 31.3% and improves token decoding speed by $1.93\times$, $1.52\times$, and $1.43\times$ for the 906M, 3B, and 8B models, respectively. To test whether fewer attention layers compromise context extension, we additionally apply YaRN to the 8B model. Notably, Hourglass obtains lower long-context language modeling perplexity and retrieval and processing tasks than the conventional baseline at every evaluated length from 4k to 64k tokens.

To summarize, our contributions are as follows:

- We systematically investigate whether the narrow–wide–narrow FFN convention in Transformers can be replaced by an inverted, hourglass shape. We introduce Hourglass Transformers, which pair wide–narrow–wide FFN sub-blocks with hourglass attention modules for efficient sequence mixing in widened residual streams.
- Through controlled experiments at 113M and design-space exploration at 403M/906M, we show that hourglass FFNs are competitive with conventional FFNs while enabling a depth–width trade-off: fewer Transformer layers and wider residual representations at matched parameter budgets.
- We demonstrate that the complete Hourglass recipe persists through scales, yielding a competitive quality frontier with more efficient training FLOPs, faster token decoding at 64k context; an improved long-context performance at 8B.

2 Background and Related Work

2.1 The Narrow-Wide-Narrow FFN Convention

In Transformer architectures, the FFN serves as a position-wise non-linear transformation applied independently to each token representation. Formally, given a hidden state $z \in \mathbb{R}^{d_{\text{model}}}$, the conventional MLP with residual in FFN computes

$$\text{FFN}(z) = z + W_2\sigma(W_1\text{norm}(z)) \tag{1}$$

where $\sigma(\cdot)$ is a non-linear activation function, $\text{norm}(\cdot)$ is a normalization function, $W_1 \in \mathbb{R}^{d_h \times d_{\text{model}}}$, $W_2 \in \mathbb{R}^{d_{\text{model}} \times d_h}$, and $d_h > d_{\text{model}}$. This expansion–projection structure has been retained across most modern dense LLMs Touvron et al. (2023); Qwen et al. (2025); Team et al. (2025).

Beyond functional form, the FFN plays a central role in determining the overall shape of a Transformer. It accounts for a majority of parameters and floating-point operations per layer, often exceeding those of the attention module. As a result, the choice of the shape expansion ratio d_h/d_{model}, together with the model dimension d_{model} and the number of block layers L, implicitly governs how capacity is allocated across width, depth, and attention. While prior work has explored variations in activation functions (Shazeer, 2020), normalization strategies (Xiong et al., 2020; Jiang et al., 2023b), parameter sharing (Lan et al., 2020; Li et al., 2022), and depth-width trade-off (Petty et al., 2024), the overall narrow–wide–narrow shape of the FFN and its interaction with Transformer scaling (Kaplan et al., 2020) have remained largely unchanged.

2.2 Hourglass and Bottleneck Residual Architectures

Theoretical analyses of hourglass[2] MLPs with residual connections have established universal approximation guarantees. With sufficient depth, (Lin & Jegelka, 2018) shows that residual networks with extremely narrow hidden layers retain universal approximation capability, while (Liu et al., 2024a) further shows that residual hourglass MLPs with narrow constant width can achieve optimal approximation of Lebesgue-integrable functions.

Empirically, several efficient architectures have explored hourglass or bottleneck structures with residual connections. In computer vision, U-Net (Ronneberger et al., 2015), MobileNet (Zhou et al., 2020), and Wide ResNets (Zagoruyko & Komodakis, 2016) use bottleneck structures or separable convolutions to improve the parameter-compute ratio. These examples illustrate that carefully structured bottlenecks can reduce computational cost while preserving, or even enhancing, representational power through increased depth or improved gradient flow. In language modeling, low-rank and bottleneck-inspired techniques have primarily appeared in parameter-efficient methods.

[2]Prior work often uses *bottleneck* to describe the wide-narrow-wide shape; we use *hourglass* interchangeably.

LoRA (Hu et al., 2022; Liu et al., 2024b) introduces low-rank projections within linear layers to reduce trainable parameters during fine-tuning, effectively imposing a bottleneck on weight updates. Mixture-of-Experts (MoE) (Shazeer et al., 2017; Mu & Lin, 2025) scales capacity without a proportional increase in inference cost by replacing dense FFNs with sparse expert networks.

Earlier scaling-law experiments reported that hourglass MLP in dense Transformer FFNs underperformed their narrow-wide-narrow counterparts (Kaplan et al., 2020; Petty et al., 2024). More recent work on generative vision models paints a different picture: residual-connected hourglass MLPs that iteratively refine representations in expanded feature spaces can outperform conventional MLPs (Chen et al., 2025). Whereas prior work primarily studies general MLP properties or sparse expert settings, our work studies hourglass MLPs as dense Transformer FFNs, enabling a reconsideration of model shape and parameter and compute allocation between attention and FFN across language-model scales.

3 Hourglass Transformer Architecture

We define Hourglass Transformers by replacing the conventional wide-expansion FFN with a stack of residual bottleneck MLP sub-blocks (Hourglass FFN) and by decoupling residual-stream width from attention width through an hourglass attention module (Hourglass Attention) (See Figure 1).

3.1 Network Architecture

Our Hourglass Transformer follows the conventional LLaMA style Transformer backbone (Grattafiori et al., 2024) consisting of input embeddings, stacked model layers L, and a final output projection.

Hourglass Transformer Layer Each layer consists of an attention module followed by an FFN. The attention module performs global information aggregation across the sequence, while the FFN performs local feature refinement within each token. Both components are wrapped with residual connections and layer normalization.

Specifically, given the input to the l-th layer $\mathbf{z}^{(\ell)}$, the intermediate representation $\mathbf{u}^{(\ell)}$ is computed via the attention mechanism (Attn): $\mathbf{u}^{(\ell)} = \mathbf{z}^{(\ell)} + \text{Attn}(\text{norm}(\mathbf{z}^{(\ell)}))$. Then, we set $\mathbf{h}_0^{(\ell)} = \mathbf{u}^{(\ell)}$ as the input to the FFN.

Hourglass Feed-Forward Network The Hourglass FFN refines the representation through K stacked hourglass-shaped MLP sub-blocks. For $i = 0, \ldots, K-1$:

$$\mathbf{h}_{i+1}^{(\ell)} = \mathbf{h}_i^{(\ell)} + \text{MLP}_i(\mathbf{h}_i^{(\ell)}) \tag{2}$$

where MLP_i denotes the i-th hourglass sub-block. Finally, the output of the layer is $\mathbf{z}^{(\ell+1)} = \mathbf{h}_K^{(\ell)}$.

While the convention expands the hidden dimension d_{model} to a wider d_h, the Hourglass FFN utilizes a compression-expansion structure with a bottleneck dimension $d_h < d_{\text{model}}$. Each sub-block MLP_i consists of a down-projection $W_d^{(i)} \in \mathbb{R}^{d_h \times d_{\text{model}}}$, a non-linear activation σ, and an up-projection $W_u^{(i)} \in \mathbb{R}^{d_{\text{model}} \times d_h}$. Formally:

$$\text{MLP}_i(\mathbf{x}) = W_u^{(i)} \sigma(W_d^{(i)} \text{norm}(\mathbf{x})) \tag{3}$$

This structure allows independent control over the FFN's depth (K) and width (d_h).

In practice, we implemented the hourglass MLP following (Grattafiori et al., 2024) using the SwiGLU activation function. Specifically, each sub-block MLP_i consists of two down-projection matrices $W_{d1}^{(i)}, W_{d2}^{(i)} \in \mathbb{R}^{d_h \times d_{\text{model}}}$ and one up-projection matrix $W_u^{(i)} \in \mathbb{R}^{d_{\text{model}} \times d_h}$. That is

$$\text{MLP}_i(\mathbf{x}) = W_u^{(i)}(\text{SiLU}(W_{d1}^{(i)} \bar{\mathbf{x}}) \odot (W_{d2}^{(i)} \bar{\mathbf{x}})) \tag{4}$$

where $\bar{\mathbf{x}} = \text{norm}(\mathbf{x})$ and $\odot$ denotes element-wise multiplication.

Hourglass Attention Transformer with hourglass FFN enables a rethinking of the attention module width. Narrowing each FFN sub-block makes it possible to explore wider residual representations under a fixed parameter budget. However, increasing d_{model} also increases the computational cost of attention if attention is performed directly in the residual dimension.

We propose Hourglass Attention to project the input to a lower dimensional space, perform attention, and project back to the original dimension, while keeping the computational cost of attention the same as the conventional attention. Specifically, the input $\mathbf{z}$ is first projected to a lower dimensional space by down-projection matrices $W_{QD}, W_{KD}, W_{VD} \in \mathbb{R}^{d_{\text{attn}} \times d_{\text{model}}}$, where $d_{\text{attn}} < d_{\text{model}}$. Then, the attention mechanism is performed on the projected key, query, and value features. Finally, the output of the attention mechanism is projected back to the original dimension by an up-projection output matrix $W_{OU} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}}$. Formally:

$$\text{Attn}(\mathbf{z}) = W_{OU}\text{softmax}\left(\frac{(W_{QD}\mathbf{z})(W_{KD}\mathbf{z})^T}{\sqrt{d_{\text{attn}}}}\right) W_{VD}\mathbf{z}, \tag{5}$$

where QK-norm, RoPE, and multi-head settings are omitted for brevity. In practice, we implement the multi-head attention by splitting the projected key, query, and value features into multiple heads, and perform attention on each head independently. Note that when $d_{\text{attn}} = d_{\text{model}}$, the Hourglass Attention reduces to the conventional multi-head attention. In our main scaling experiments, we set d_{attn} equal to the d_{model} of the corresponding conventional baseline, so attention operates in the same-width attention space while the Hourglass residual stream can be wider. Appendix A.4 provides a sensitivity check where this attention dimension is further reduced.

3.2 The Hourglass Design Space

The hourglass shape exposes an explicit parameter trade-off between residual width, attention width, FFN bottleneck width, FFN depth, and Transformer depth. Ignoring embeddings, normalization parameters, and biases, the dominant parameter count is $N \approx L\left(4d_{\text{model}}d_{\text{attn}} + 3Kd_h d_{\text{model}}\right)$, where the first term corresponds to the query, key, value, and output attention projections, and the second term corresponds to the three SwiGLU FFN projections in each of the K hourglass sub-blocks. For a token budget T and sequence length S, the corresponding dominant forward-pass FLOP estimate is

$$C_{\text{fwd}} \approx T\left(L\left(8d_{\text{model}}d_{\text{attn}} + 2Sd_{\text{attn}} + 6Kd_h d_{\text{model}}\right) + 2d_{\text{model}}n_{\text{vocab}}\right), \tag{6}$$

where n_{vocab} is the vocabulary size; Appendix A.8 gives the detailed forward–backward FLOP accounting used in our experiments. To achieve strong performance at a given parameter and compute budget, we need to balance the design parameters K, L, d_h, d_{attn}, and d_{model}. This design introduces two fundamental shifts in resource allocation:

Trading FFN Expansion and Layers for Residual Width. The Hourglass FFN ($d_h < d_{\text{model}}$ inside each sub-block) decouples FFN expressivity from a single large expansion ratio. Under a fixed parameter budget, this creates room to widen the residual stream and adjust the number of sequential Transformer layers without requiring attention to operate at the full residual dimension. Section 4 evaluates whether this trade-off preserves quality while reducing training FLOPs and token decode latency.

Trading FFN Width for Depth. Conventional FFNs rely on extreme width within one MLP transformation. In contrast, the Hourglass FFN leverages depth by stacking K residual sub-blocks within a single Transformer layer. This increases the sequence of non-linear transformations without requiring each sub-block to use a wide intermediate dimension. The resulting topology is a practical way to test whether deeper bottleneck transformations can substitute for the conventional per-layer FFN expansion in language models.

Inference Efficiency. Considering the aforementioned parametrized trade-offs, the forward KV memory and compute costs of Hourglass Transformers for inference can be approximated as

$$M_{\text{KV}}(S; g) = \frac{2\beta LSd_{\text{attn}}}{g} \quad \text{and} \quad t_{\text{decode}}(S; g) = t_{\text{fixed}} + \frac{4LSd_{\text{attn}}}{g\,\text{BW}_{\text{eff}}}, \tag{7}$$

where g is the per-layer KV-compression factor given a selected sequence mixing module such as MHA or GQA, t_{fixed} is the fixed decoding time overhead, β is the bytes per cached element, and

BW_{eff} is the effective GPU memory bandwidth. Section B provides detailed inference efficiency analysis, and Section 4.7 validates the theoretical inference efficiency gain with empirical measurements against the convetional baselines.

4 EXPERIMENTS

We evaluate whether the conventional narrow–wide–narrow Transformer FFN can be replaced by an hourglass-shaped alternative without sacrificing language-model quality. We first use controlled settings to isolate the FFN-side effect, then evaluate the complete Hourglass architecture as a depth–width trade-off against standard Transformer baselines.

4.1 EXPERIMENTAL SETUP

Baselines. We compare against published dense Transformer configurations spanning 113M, 403M, 906M, 3B, and 8B parameters (see Table 2). These baselines follow LLaMA/OLMo-style architectures (Touvron et al., 2023; OLMo et al., 2025) and use standard shapes from prior scaling studies and open model reports (Brown et al., 2020; Touvron et al., 2023; Grattafiori et al., 2024). We treat these published shapes as established, reproducible reference points for conventional dense Transformers, not as baselines tuned around our method. As a robustness check, Appendix A.2 reports conventional FFN-ratio variants at 403M and a reduced-depth parameter-matched variant at 906M; these controls shift the quality–compute trade-off but do not uniformly improve over the published-shape baselines.

Hourglass Transformers. For each baseline size, we construct a corresponding variant by replacing the conventional FFN with the Hourglass FFN and using Hourglass Attention. For fair comparison, d_{attn} at each parameter size is chosen to match the d_{model} of the corresponding conventional baseline. Model configurations are chosen with approximately matched total parameter budgets, staying within 0.5% of the baseline parameter count while keeping training FLOPs closely matched or lower.

Training and Evaluation Details. All models are trained on the same dataset and tokenized using identical preprocessing pipelines. Optimization uses AdamW with a cosine learning rate schedule. The details of experiment settings can be found in Appendix A.1, where we follow general settings from (Brown et al., 2020; Hoffmann et al., 2022; OLMo et al., 2025). Following the convention in (Kaplan et al., 2020; Hoffmann et al., 2022), we report only non-embedding parameters in model-size comparisons. For training efficiency, we report training FLOPs per forward-backward pass, following (Hoffmann et al., 2022; McLeish et al., 2025); the calculation script is provided in Appendix A.8. For inference, we report relative token-decode latency compared with the corresponding baseline model. For evaluation, we report average validation loss and validation perplexity (PPL) over validation sets. Section 4.5 additionally reports downstream task performance at larger model scales.

4.2 A CONTROLLED TEST OF FFN SHAPE

To isolate the contribution of the Hourglass FFN from changes in attention capacity and layer count, we evaluate 113M models where $d_{\text{model}} = 768$ and $L = 12$ are kept identical, ensuring that the attention parameter count and train FLOPs are the same across all variants (see Table 1). We vary the Hourglass FFN depth K and adjust the intermediate dimension d_h accordingly, while keeping the total model size fixed at 113M parameters. Under this controlled setting, the Hourglass FFN Transformer consistently achieves lower validation loss and perplexity than the conventional baseline. This provides controlled evidence that the hourglass FFN parameterization can be competitive even when attention capacity is held fixed. As an additional sanity check, Appendix A.5 reports a no-attention Bigram LM comparison, where the Hourglass FFN also remains comparable to the conventional FFN.

Table 1: **Hourglass FFNs vs. Conventional FFNs under fixed $d_{\text{model}} = 768$ and $L = 12$ (113M).** All models use 113M total parameters, with 28M attention parameters and 85M FFN parameters.

Model	d_{h}	K	Val Loss ↓	Val PPL ↓
Conventional	3072	1	3.766	43.19
Hourglass	614	5	3.749	42.46
	512	6	3.752	42.61
	384	8	3.742	42.16
	307	10	3.747	42.38

4.3 Effect of Larger d_{model} in Hourglass Transformers

The Hourglass architecture compresses the FFN intermediate dimension, allowing configurations that widen d_{model} under a matched parameter budget. A wider hidden dimension increases the representational capacity of every residual-stream component, i.e., attention projections, layer norms, and the LM head. To study this effect in isolation, we conduct a frozen LM head experiment on 403M models, which controls for the additional training FLOPs that a larger LM head would otherwise introduce.

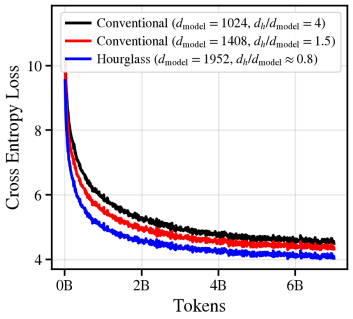


Figure 2: **Frozen LM head Experiment at 403M.** The experiment probes the benefit of a larger d_{model}: the Hourglass model achieves the lowest loss, and the wider conventional variant outperforms the standard one, consistent with the gradient bottleneck hypothesis (Godey & Artzi, 2026).

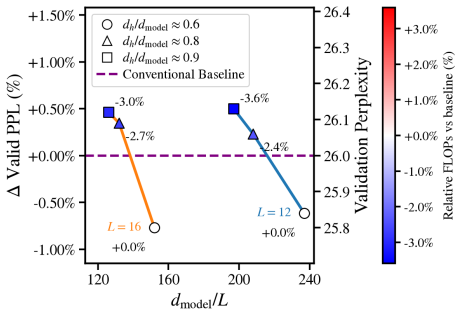


Figure 3: **Width and depth of Hourglass Transformers at 906M.** Validation PPL delta versus d_{model}/L for $L \in \{12, 16\}$ and varying d_h/d_{model} (marker shape). Lower $d_h/d_{\text{model}} \approx 0.6$ gives the best PPL, while higher ratios provide larger FLOPs savings but move PPL above the baseline, revealing a quality–compute trade-off.

Figure 2 compares three configurations under the frozen-head setting. Among the two conventional baselines alone, the variant with a wider d_{model} (d_{model}=1408, d_h=2112, d_h/d_{model}=1.5, red) consistently outperforms the standard configuration (d_{model}=1024, d_h=4096, d_h/d_{model}=4, black), suggesting that a larger hidden dimension is beneficial in this setting. The Hourglass model (K=4, L=12, d_{model}=1952, d_h=1564, d_h/d_{model}≈0.8, blue) operates at the widest d_{model} of the three and follows this trend. This illustrates one advantage of the Hourglass architecture: it enables the use of a wider d_{model} within the same parameter budget by compressing the FFN intermediate dimension.

This ordering is consistent with the LM-head gradient bottleneck identified by Godey & Artzi (2026). With a frozen head W, the backbone receives gradients $\nabla_H \mathcal{L} = \text{diag}(\omega)(P - \tilde{N})W$, where H denotes the pre-LM-head hidden states, P the model's next-token probabilities, $\tilde{N}$ the empirical next-token distribution, and ω the per-position weights. The learning signal is therefore compressed through the hidden dimension D=d_{model}; a wider d_{model} alleviates this bottleneck. Because the Hourglass architecture naturally widens d_{model} (while keeping total parameters matched), it may benefit from both a more expressive backbone and a less constrained gradient pathway through the LM head.

4.4 Width and Depth Landscape of Hourglass Transformers

In this section, we study the FFN intermediate ratio d_h/d_{model} and the width-to-depth d_{model}/L ratio of the model. The central finding is that, within the Hourglass design space, models can remain competitive with the conventional baseline while using fewer Transformer layers. By stacking deeper FFN sub-MLPs within each layer, the architecture frees parameters that can be allocated to widening d_{model} rather than increasing the number of layers L. We fix K=4 as a moderate-depth design choice that balances FFN compositional depth against residual-stream width under the matched parameter budget, rather than as the result of an exhaustive search over K at every scale; Appendix A.7.1 reports a small-scale ablation over K. We establish this at the 403M scale (Appendix A.6) and then use a targeted sweep at 906M to identify reference configurations for scaling.

Figure 3 shows the sweep at 906M parameters. Notably, the $(d_h/d_{\text{model}}, L) = (0.6, 12)$ and $(0.6, 16)$ configurations, using only 50% and 67% of the baseline's Transformer layers ($L = 24$), exceed the

Table 2: **Configurations and quality–compute efficiency at 906M, 3B, and 8B.** Hourglass variants use hourglass-shaped FFNs with residual connections (K stacked hourglass blocks), while conventional variants use standard narrow-wide-narrow FFNs ($K = 1$). Training settings are summarized in Table 4. Parenthesized values in the Hourglass rows give the relative change versus the matched conventional baseline. The 64k decode column reports relative token-decode latency, where lower is faster. Training FLOPs are estimated by the script in Appendix A.8; raw validation and downstream metrics are also provided in Table 7.

Size	Model	Attn Params (M)	FFN Params (M)	d_{model}	d_h	L	K	Train FLOPs↓	64k Decode Latency ↓	Val PPL ↓	Avg Downstream Acc. ↑	Avg Downstream PPL ↓
906M	Conventional	227	679	1536	6144	24	1	3.34×10^{16}	1.00×	26.00	0.503	1.524
	Hourglass	184	725	2496	2016	12	4	3.26×10^{16} (−2.40%)	0.52×	26.06 (+0.23%)	0.505 (+0.40%)	1.514 (−0.66%)
3B	Conventional	1057	2114	3072	8192	28	1	1.03×10^{17}	1.00×	16.80	0.626	1.211
	Hourglass	817	2350	3696	2944	18	4	9.92×10^{16} (−3.69%)	0.66×	16.93 (+0.83%)	0.628 (+0.32%)	1.188 (−1.90%)
8B	Conventional	2148	5637	4096	14336	32	1	2.30×10^{17}	1.00×	13.37	0.701	0.955
	Hourglass	1799	5988	4992	4544	22	4	2.25×10^{17} (−2.17%)	0.70×	13.31 (−0.45%)	0.706 (+0.71%)	0.943 (−1.26%)

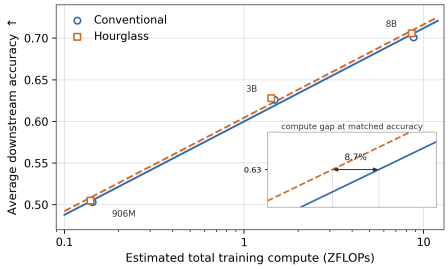


Figure 4: **Average downstream accuracy versus estimated total training compute.** Lines are least-squares fits of $\text{Acc} = a + b \log C$ with $R^2 = 0.996$ and 0.997 for separate fits. Under the test regime, Hourglass reaches a matched average accuracy with 8.7% less training compute.

Table 3: **Long-context evaluation at 8B.** Both models receive an identical 400-step YaRN context extension (Peng et al., 2023); Hourglass uses $L = 22$ attention layers versus 32 for the matched conventional baseline. Perplexity is measured on Proof-pile and RULER accuracy is averaged over its 13 tasks. Hourglass is better at every evaluated length on both metrics.

	8B Model	4k	8k	16k	32k	64k	Avg.
Proof-pile PPL ↓	Conventional	7.39	6.81	5.54	5.05	4.79	5.92
	Hourglass	**7.10**	**6.51**	**5.28**	**4.78**	**4.55**	**5.64**
RULER Acc. (%) ↑	Conventional	9.3	5.8	3.2	2.5	2.2	4.6
	Hourglass	**11.0**	**6.8**	**4.4**	**4.0**	**3.3**	**5.9**

baseline by 0.5%, suggesting that the freed parameters are better spent on wider hidden representations. We also observe a quality–compute trade-off: configurations with lower d_h/d_{model} (circles, $\approx$0.6) reach the lowest validation PPL, while the largest savings (up to $\sim$3% fewer training FLOPs than the baseline) occur at higher ratios (squares, $\approx$0.9), at slightly worse PPL. We therefore use this sweep to select an efficiency-oriented configuration for larger-scale validation.

4.5 Scaling to 8B

Guided by the quality–compute frontier in the previous section, we scale an efficiency-oriented Hourglass architecture and compare it against the standard conventional baseline at 906M, 3B, and 8B parameters. The selected configurations (Table 2) follow the design principles established above: deeper FFN stacking (K=4), fewer Transformer layers (L=12/18/22 at 906M/3B/8B versus L=24/28/32 for the baselines), wider d_{model}, and matched d_{attn} with respective baselines.

Table 2 summarizes the paired quality–compute trade-off as raw values with relative deltas to the corresponding conventional baseline; the raw validation and downstream metrics are also reported in Table 7. Validation PPL remains close to the conventional baselines, with relative changes of +0.23%, +0.83%, and −0.45%. Downstream averages remain competitive: average accuracy changes by +0.40%, +0.32%, and +0.71%, while average downstream PPL improves by 0.66%, 1.90%, and 1.26%. Taking these pairs, we fit quality on total training compute $C = (\text{FLOPs/token}) \times \text{tokens}$ to understand the training efficient improvment between the two architectures at matched performance (Figure 4), revealing that Hourglass reaches a given average accuracy with 8.7% less training compute. Overall, the scaled Hourglass architecture preserves competitive validation and downstream quality while using modestly fewer training FLOPs across 906M, 3B, and 8B. Since FFN shape, residual width, layer count, and attention width are varied jointly, these results support the combined Hourglass design as a whole.

4.6 Long-Context Language Modeling

Hourglass reduces the number of Transformer layers, which gives latency gains but also removes attention applications that might otherwise support long-range mixing. We evaluate the 8B conventional/Hourglass pairs at Table 2 after identical 400-step YaRN finetuning for context extension up to 64k tokens, using Proof-pile following Peng et al. (2023) and RULER (Hsieh et al., 2024), whose 13 tasks probe retrieval, multi-key and multi-query needle search, aggregation, variable tracking, and question answering. Notably, Hourglass obtains lower PPL and higher RULER accuracy at every evaluated context length. One possible explanation is that the wider residual embedding space in Hourglass can preserve token information better across long contexts, and the fewer attention layers reduce accumulated transformation/noise during very long-context decoding.

4.7 Token Decode Efficiency

A practical benefit of the Hourglass architecture using fewer Transformer layers than the conventional baseline at the same parameter budget is that the reduced L directly translates to lower KV cache size and thus improved decode latency. Figure 5 reports relative token decode latency across sequence lengths from 4k to 64k tokens at 906M, 3B, and 8B. As context length increases, the speedup aymptotically approaches the $L_{\text{conv}}/L_{\text{hg}}$ ratio, which is $2\times$, $1.56\times$, and $1.45\times$ at 906M, 3B, and 8B, respectively. We additionaly evaluate with different sequence mixers (GQA and MLA) in Table 12 for 906M model up to 128k context length and observe the same speedup trend.

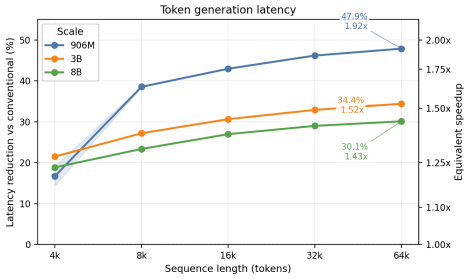


Figure 5: **Token decode latency.** Relative latency is normalized to the conventional baseline; endpoint labels report the 64k-token values.

5 Discussion and Future Work

Our investigation revisits the long-standing shape convention of dense Transformer language models, with particular focus on the narrow-wide-narrow FFN. Our evaluation results suggest that the conventional FFN expansion is not essential for maintaining competitive language-model quality: residual hourglass bottlenecks can provide an alternative token-wise transformation path, while freeing capacity for wider residual representations and fewer Transformer layers. Decoupling attention width from residual-stream width makes this reallocation practical, allowing sequence mixing to remain close to the baseline attention dimension even as the residual stream widens.

Limitations. While these results are promising, specific limitations remain. First, our comprehensive parameter search was constrained to smaller scales, with the selected recipe validated up to 8B parameters trained with 160B tokens. Validation at larger parameter size and training budget to trillion tokens for short and long contexts could further confirm the generality of the hourglass design. Second, while our controlled ablations isolate the Hourglass FFN mechanism at smaller scales, the 906M, 3B, and 8B experiments validate the complete architecture recipe, where FFN shape, residual width, layer count, and attention projection width are selected jointly. Thus, the main large-scale conclusion is an end-to-end architecture-efficiency result. Full factor decomposition across every design axis with detailed analysis remains a future direction. Third, all trained models use multi-head attention as the sequence mixer. Appendix B.1 measures decode cost with Group-Query Attention (Jiang et al., 2023a) and Multi-head Latent Attention (DeepSeek-AI et al., 2025) substituted into both arms, and finds the KV-cache and latency advantages preserved, but those measurements concern inference cost only. How these mixers interact with Hourglass FFNs during training, and whether they shift the attention-FFN ratio landscape, remains unexplored.

Future Work. Continued scaling of model dimensions (d_{model} and K) is a promising avenue, as similar manifold-constrained architectures have shown benefits from expanded width Xie et al. (2025). At the same time, deeper hourglass FFNs may require careful control of signal propagation and residual noise, especially in regimes affected by the "curse of depth" Sun et al. (2025). Broader long-context evaluation is another natural next step, including retrieval-heavy tasks, instruction-following settings, reasoning-oriented benchmarks, and context windows beyond 64k. We view Hourglass Transformers as a practical step toward expanding the design space of efficient dense language models beyond the historical narrow-wide-narrow FFN convention.

AI USE STATEMENT

In this work, we used generative AI tools to help accelerate experiment execution, draft plotting and table-generation code, review the manuscript's wording, and check the accuracy of references. AI tools also assisted with adapting the manuscript's LaTeX formatting to the ICLR 2027 template. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. Proof-pile, 2022. URL https://github.com/zhangir-azerbayev/proof-pile.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Meng-Hsi Chen, Yu-Ang Lee, Feng-Ting Liao, and Da shan Shiu. Rethinking the shape convention of an mlp, 2025. URL https://arxiv.org/abs/2510.01796.

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:1803.05457v1*, 2018.

Thomas M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-14(3):326–334, 1965. doi: 10.1109/PGEC.1965.264137.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003. doi: https://doi.org/10.1002/rsa.10073. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rsa.10073.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin

Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Nathan Godey and Yoav Artzi. Lost in backpropagation: The lm head is a gradient bottleneck, 2026. URL https://arxiv.org/abs/2603.10145.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, An-

nie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Boris Hanin and David Rolnick. Complexity of Linear Regions in Deep Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2596–2604. PMLR, 2019.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZe72R8yS0.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023a. URL https://arxiv.org/abs/2310.06825.

Zixuan Jiang, Jiaqi Gu, Hanqing Zhu, and David Z. Pan. Pre-RMSNorm and pre-CRMSNorm transformers: Equivalent and efficient pre-LN transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=z06npyCwDq.

William B. Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206, 1984. URL https://api.semanticscholar.org/CorpusID:117819162.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. triviaqa: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *arXiv e-prints*, art. arXiv:1705.03551, 2017.

Johnny Joyce and Jan Verschelde. Computing linear regions in neural networks with skip connections. In François Boulier, Chenqi Mou, Timur M. Sadykov, and Evgenii V. Vorozhtsov (eds.), *Computer Algebra in Scientific Computing*, pp. 175–194, Cham, 2026. Springer Nature Switzerland. ISBN 978-3-032-09645-6.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1eA7AEtvS.

Bei Li, Quan Du, Tao Zhou, Yi Jing, Shuhan Zhou, Xin Zeng, Tong Xiao, JingBo Zhu, Xuebo Liu, and Min Zhang. ODE transformer: An ordinary differential equation-inspired model for sequence generation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8335–8351, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.571. URL https://aclanthology.org/2022.acl-long.571/.

Hongzhou Lin and Stefanie Jegelka. Resnet with one-neuron hidden layers is a universal approximator. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/03bfc1d4783966c69cc6aef8247e0103-Paper.pdf.

Chenghao Liu, Enming Liang, and Minghua Chen. Characterizing ResNet's universal approximation capability. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 31477–31515. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/liu24am.html.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024b.

Sean Michael McLeish, John Kirchenbauer, David Yu Miller, Siddharth Singh, Abhinav Bhatele, Micah Goldblum, Ashwinee Panda, and Tom Goldstein. Gemstones: A model suite for multi-faceted scaling laws. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=iZk78dZ1Ap.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Siyuan Mu and Sen Lin. A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications, 2025. URL https://arxiv.org/abs/2503.07137.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, Michal Guerquin, David Heineman, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Jake Poznanski, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models, 2023. URL https://arxiv.org/abs/2309.00071.

Jackson Petty, Sjoerd Steenkiste, Ishita Dasgupta, Fei Sha, Dan Garrette, and Tal Linzen. The impact of depth on compositional generalization in transformer language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7239–7252, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.402. URL https://aclanthology.org/2024.naacl-long.402/.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pp. 1177–1184, Red Hook, NY, USA, 2007. Curran Associates Inc.

Machel Reid, Victor Zhong, Suchin Gururangan, and Luke Zettlemoyer. M2D2: A massively multi-domain language modeling dataset. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.),

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 964–975, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.63. URL https://aclanthology.org/2022.emnlp-main.63/.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://arxiv.org/abs/1701.06538.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Walsh, Luke Zettlemoyer, Noah Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15725–15788, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.840. URL https://aclanthology.org/2024.acl-long.840/.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. *arXiv preprint arXiv:2502.05795*, 2025.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine,

Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. URL https://arxiv.org/abs/2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Siqi Wang, Zhengyu Chen, Bei Li, Keqing He, Min Zhang, and Jingang Wang. Scaling laws across model architectures: A comparative analysis of dense and MoE models in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5583–5595, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.319. URL https://aclanthology.org/2024.emnlp-main.319/.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In Leon Derczynski, Wei Xu, Alan Ritter, and Tim Baldwin (eds.), *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413/.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, et al. mhc: Manifold-constrained hyper-connections. *arXiv preprint arXiv:2512.24880*, 2025.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020. URL https://openreview.net/forum?id=B1x8anVFPr.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.87. URL https://dx.doi.org/10.5244/C.30.87.

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*, 2019.

Daquan Zhou, Qibin Hou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Rethinking bottleneck structure for efficient mobile network design. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III*, pp. 680–697, Berlin, Heidelberg, 2020. Springer-Verlag. ISBN 978-3-030-58579-2. doi: 10.1007/978-3-030-58580-8_40. URL https://doi.org/10.1007/978-3-030-58580-8_40.

A Appendix

A.1 Experimental Details

A.1.1 Experimental Environment and Training Corpus

Training is conducted with NVIDIA NeMo Megatron Bridge[3], while evaluation uses the OLMo codebase[4]. Experiments are executed on NVIDIA RTX 6000 Ada and B200 GPUs. The 3B and 8B runs use 8 B200 GPUs and take approximately 3 and 8 days, respectively. The fixed random seed 6198 is used in all experiments.

For training data, we adopt the Stage 1 pre-training corpus used to train the original OLMo-2 1B checkpoint. To control for the effects of stochasticity arising from data shuffling, we replicate the exact data ordering employed during the Stage 1 pre-training of OLMo-2 1B. From this fixed-order dataset, we select the first 2.5B and 7B tokens for the 113M and 403M studies, respectively. The token budgets for the 906M, 3B, and 8B scaling experiments are reported in Table 4. The choice of the number of training tokens for each model size follows (Hoffmann et al., 2022) reported ratio, giving approximately 20 tokens per parameter at every scale.

The finetuning and evaluation codebase[5] of the long-context experiments follows (Peng et al., 2023) with necessary adaptation to use the NeMo checkpoints and model. We convert the NeMo checkpoints to HuggingFace format for finetuning and evaluation. For finetuning and evaluation, we employ the exact same hyperparameters as (Peng et al., 2023). The finetuning corpus follows (Peng et al., 2023), which employs PG19 from (Rae et al., 2020), and the evaluation is performed on Proof-pile from (Azerbayev et al., 2022). Both finetuning and evaluation dataset are processed with the same tokenizer as the training data, which is the tokenizer used in OLMo-2 1B model.

A.1.2 Validation Datasets

We conduct validation using the following datasets: Dolma Common Crawl Soldaini et al. (2024), Dolma The Stack Soldaini et al. (2024), M2D2 Reid et al. (2022), and WikiText Merity et al. (2016). The validation splits follow the official OLMo-2 1B configuration. For both cross-entropy loss and perplexity, we report the average scores computed across these four datasets. Additional validation results are provided in Section A.3.

A.1.3 Downstream Evaluation Datasets

We conduct downstream task evaluation on the following benchmark datasets: Arc Easy Clark et al. (2018), HellaSwag Zellers et al. (2019), PIQA Bisk et al. (2020), SciQ Welbl et al. (2017), CommonsenseQA Talmor et al. (2019), TriviaQA Joshi et al. (2017), and NaturalQS Kwiatkowski et al. (2019). Among these, TriviaQA and NaturalQS are evaluated using perplexity, while all remaining tasks are assessed using accuracy as the evaluation metric.

A.1.4 Training Hyperparameters

Across all experiments, we use AdamW as the optimizer and adopt the cross-entropy loss augmented with an auxiliary softmax loss as the training objective. For AdamW, the hyperparameters are set to $\beta_1 = 0.9$, $\beta_2 = 0.95$, and $\epsilon = 1 \times 10^{-8}$, with a weight decay coefficient of $\lambda = 0.1$. SwiGLU is employed as the activation function for all models. Rotary Position Embedding (RoPE) is utilized for positional encoding in both the Transformer architecture and the proposed hourglass blocks.

[3] https://github.com/NVIDIA-NeMo/Megatron-Bridge

[4] https://github.com/allenai/OLMo

[5] https://github.com/jquesnelle/yarn

A cosine learning rate scheduler with warmup is applied in all experiments, where the number of warmup tokens varies according to model size. Additional hyperparameters are reported in Table 4, and model-size-dependent architectural configurations are reported in Table 2.

Table 4: **Additional experimental hyperparameters for different model sizes**, including learning rates, batch tokens, trained tokens, warmup tokens, attention heads, and sequence lengths.

Model Size	Learning Rate	Batch Tokens (K)	Trained Tokens (B)	WarmUp Tokens	Attention Heads	Max Sequence Size
113M	6×10^{-4}	512	2.5	50M	12	2048
403M	3×10^{-4}	1024	7	50M	16	4096
906M	2.5×10^{-4}	1024	18	100M	16	4096
3B	1.6×10^{-4}	1024	60	400M	24	4096
8B	1.2×10^{-4}	1024	160	400M	32	4096

A.2 Conventional Baseline Controls

We use standard published Transformer configurations as the primary baselines because they provide fixed, reproducible reference points for the established conventional design. As targeted controls, Table 5 evaluates two classes of conventional alternatives. At 403M, varying the conventional FFN ratio yields similar quality, while the standard $d_h/d_{\text{model}} = 4$ ratio has the lowest training FLOPs among these comparable variants. At 906M, a reduced-depth conventional variant produces a mixed trade-off: the parameter-matched $L = 12$ variant reduces training FLOPs by about 10%, but it has worse validation PPL and average downstream PPL. These checks support the use of standard configurations as fair primary baselines, while a full conventional depth–width frontier remains outside the scope of this study.

A.3 Detailed Validation Results

Table 6 presents the complete validation loss and perplexity results for the 906M and 3B scaling experiments evaluated across four distinct validation datasets.

A.4 Sensitivity to Reduced Attention Dimension

Random projection analyses provide useful context: geometric structure can be preserved in lower-dimensional projections when the target dimension is sufficiently large (Johnson & Lindenstrauss, 1984; Dasgupta & Gupta, 2003). The Hourglass architecture operates at a wider d_{model} than the conventional baseline, so attention queries and keys are projected from a higher-dimensional residual stream into the same attention width as the baseline. As a small sensitivity check, we reduce the attention dimension d_{attn} to $0.9\times$ the default value at 906M parameters and compare the resulting degradation in Table 8. Reducing d_{attn} increases validation PPL by $+0.43$ for the conventional baseline and by $+0.21$ for the Hourglass model, with a similar pattern in validation loss. This shows lower degradation for the Hourglass variant under this moderate attention-width compression check, although this experiment is not intended as a complete attention-width sweep.

A.5 No-Attention FFN Control

As a minimal FFN-only control, we train 113M-parameter models with all self-attention layers removed, reducing the architecture to a token-level feed-forward network. This experiment is intentionally limited as language modeling evidence because it removes sequence mixing, but it directly tests whether the hourglass FFN parameterization remains stable in the absence of attention. As shown in Table 9, the Hourglass FFN remains comparable to the conventional FFN under the same FFN parameter budget.

A.6 FFN intermediate ratio at 403M.

We fix the total model size to 403M parameters and sweep d_h/d_{model} for Hourglass Transformers at two layer counts, as shown in Figure 6. Both $(K, L) = (4, 12)$ and $(K, L) = (4, 24)$ are competitive with or better than the conventional baseline ($L = 24$, $d_h/d_{\text{model}} = 4$) across most d_h/d_{model}

Table 5: **Conventional baseline controls.** Results for conventional baseline models with different internal FFN ratios and an exploratory reduced-depth 906M variant. Arrows indicate the preferred direction for each metric. Parenthesized values report the relative change versus the standard setting at the same parameter scale. The reduced-depth variant illustrates that reallocating the same parameter budget into fewer conventional layers can lower training FLOPs while worsening validation PPL and average downstream PPL.

Params	Setting	L	d_h/d_{model}	FLOPs ↓	Val Loss ↓	Val PPL ↓	Avg Downstream Acc ↑	Avg Downstream PPL ↓
403M	Standard	24	4	1.87e+15	3.081	25.21	0.525	1.472
403M	FFN-ratio	24	3	1.93e+15 (+3.21%)	3.081 (+0.00%)	25.30 (+0.36%)	0.525 (+0.00%)	1.503 (+2.11%)
403M	FFN-ratio	24	2	2.04e+15 (+9.09%)	3.067 (−0.45%)	24.97 (−0.95%)	0.523 (−0.38%)	1.486 (+0.95%)
403M	FFN-ratio	24	1.5	2.10e+15 (+12.30%)	3.060 (−0.68%)	24.80 (−1.63%)	0.528 (+0.57%)	1.518 (+3.12%)
906M	Standard	24	4	3.34e+16	3.092	26.00	0.503	1.524
906M	FFN-ratio	12	9.33	3.01e+16 (−9.88%)	3.127 (+1.13%)	26.85 (+3.27%)	0.503 (+0.00%)	1.578 (+3.54%)

Table 6: **Detailed validation loss and perplexity on our 4 validation datasets**, including Dolma Common Crawl, Dolma The Stack, M2D2 and WikiText.

Size	Model	Validation Loss				Validation Perplexity			
		Dolma Common Crawl	Dolma The Stack	M2D2	WikiText	Dolma Common Crawl	Dolma The Stack	M2D2	WikiText
906M	Conventional	3.554	1.981	3.581	3.253	34.95	7.25	35.93	25.86
	Hourglass	3.555	1.984	3.582	3.260	34.98	7.27	35.96	26.04
3B	Conventional	3.139	1.545	3.201	2.696	23.07	4.69	24.60	14.82
	Hourglass	3.140	1.546	3.225	2.694	23.10	4.69	25.15	14.80

Table 7: **Full per-task performance for the 906M, 3B, and 8B scaling experiments.** The main text reports the relative deltas; this table provides the raw validation and downstream metrics.

Size	Model	Validation		Accuracy Tasks					PPL Tasks		Avg	
		Val Loss	Val PPL	Arc Easy	HellaSwag	PIQA	SciQ	CommonsenseQA	TriviaQA	NaturalQS	Avg Acc	Avg PPL
906M	Conventional	**3.092**	**26.00**	**0.504**	0.331	0.643	**0.735**	0.301	**1.645**	1.404	0.503	1.524
	Hourglass	3.095	26.06	0.498	**0.337**	**0.654**	0.727	**0.309**	1.657	**1.372**	**0.505**	**1.514**
3B	Conventional	**2.646**	**16.80**	0.628	**0.519**	**0.726**	**0.859**	0.399	1.233	1.190	0.626	1.211
	Hourglass	2.651	16.93	**0.654**	0.517	0.716	0.852	**0.402**	**1.215**	**1.161**	**0.628**	**1.188**
8B	Conventional	2.414	13.37	**0.723**	**0.667**	0.754	0.911	0.453	0.909	1.000	0.701	0.955
	Hourglass	**2.409**	**13.31**	0.721	0.661	**0.764**	**0.921**	**0.462**	**0.900**	**0.986**	**0.706**	**0.943**

Table 8: **Sensitivity to reduced d_{attn} ($0.9\times$) at 906M parameters.** The Δ columns show the degradation from reducing d_{attn}. In this setting, the Hourglass model shows roughly half the validation-loss and perplexity degradation of the conventional baseline.

Model	d_{model}	d_{attn}	Params	FLOPs	Val Loss	Δ	Val PPL	Δ
Conventional	1536	1536	906.1M	3.34e+16	3.092		26.00	
Conventional (0.9)	1536	1376	882.5M	3.21e+16	3.110	+0.018	26.43	+0.43
Hourglass	2496	1536	908.8M	3.26e+16	3.095		26.06	
Hourglass (0.9)	2496	1376	889.6M	3.17e+16	3.103	+0.008	26.27	+0.21

Table 9: **Toy Bigram Language Model Results.** With self-attention removed, the Hourglass FFN remains comparable to the conventional FFN under a matched FFN parameter budget.

Model	FFN Size	d_{model}	d_{h}	K	L	Val Loss	Val PPL
Conventional FFN	85M	768	3072	1	12	6.47	717.12
Hourglass FFN ($K = 4$)	85M	854	690	4	12	6.467	715.09

ratios. Notably, the $(K, L) = (4, 12)$ configuration—which uses only half the Transformer layers of the baseline—performs comparably or better than the $(K, L) = (4, 24)$ variant across the sweep, suggesting that the freed parameters can be more effectively spent on wider hidden representations.

A.7 Additional Hourglass FFN Ablations

We conduct a series of ablation studies on the 113M hourglass FFN Transformer to investigate the impact of key architectural hyperparameters. All experiments use the same dataset, tokenization, and training setup as described in Section 4.1.

A.7.1 Varying Hourglass FFN Depth K

With $d_{\text{model}} = 1032$, $d_h = 418$, and $L = 12$ fixed, we vary $K \in \{1, 2, 4, 6, 8\}$ to study the effect of Hourglass FFN depth. As shown in Table 10, increasing K consistently improves performance: validation perplexity drops from 40.153 at $K = 1$ to 32.832 at $K = 8$, with corresponding reductions

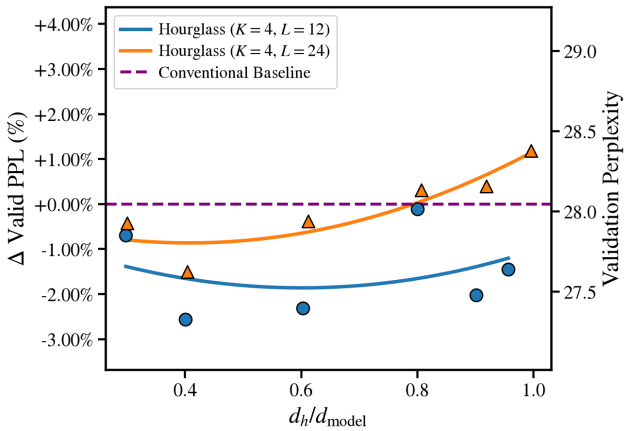


Figure 6: d_h/d_{model} **search at 403M.** Validation perplexity across different d_h/d_{model} ratios for Hourglass Transformers at 403M parameters. The $(K, L) = (4, 12)$ curve (blue) remains below the conventional baseline across the sweep and is consistently lower than the $(K, L) = (4, 24)$ curve (orange).

Table 10: **Impact of increasing hourglass depth** K**.** Performance (validation loss and perplexity) improves as depth K increases from 1 to 8. We fix $d_{\text{model}} = 1032$, $d_h = 418$, and $L = 12$.

K	Model Size	Attention Size	FFN Size	Val Loss	Val PPL
1	67M	51M	16M	3.551	40.153
2	82M	51M	31M	3.489	37.632
4	113M	51M	62M	3.426	35.335
6	144M	51M	93M	3.391	34.051
8	175M	51M	124M	3.357	32.832

in both training and validation loss. The gains are most pronounced when moving from shallow configurations ($K = 1$ or $K = 2$) to moderate depth ($K = 4$ or $K = 6$), after which improvements become more incremental.

Notice that under this setting, larger K also increases model size from 67M at $K = 1$ to 175M at $K = 8$, suggesting a trade-off between depth and efficiency. This indicates that deeper Hourglass FFNs yield better performance in this sweep, while moderate K values (e.g., $K = 4$ or $K = 6$) provide a strong accuracy–parameter trade-off for the main architecture design.

Table 11: **Impact of varying** d_h**.** The validation perplexity increases gradually as the ratio decreases from 0.8 to 0.1. We fixed $d_{\text{model}} = 1032$, $K = 4$, and $L = 12$.

d_{h}	d_h/d_{model}	Model Size	Attention Size	FFN Size	Val Loss	Val PPL
836	0.8	175M	51M	124M	3.355	32.788
627	0.6	144M	51M	93M	3.384	33.747
418	0.4	113M	51M	62M	3.426	35.335
209	0.2	82M	51M	31M	3.500	38.075
103	0.1	66M	51M	15M	3.541	39.741

A.7.2 Varying FFN Bottleneck Width d_h

With $d_{\text{model}} = 1032$, $L = 12$, and $K = 4$ fixed, we vary $d_h/d_{\text{model}} \in \{0.8, 0.6, 0.4, 0.2, 0.1\}$ to study the effect of reducing the intermediate FFN dimension relative to the hidden dimension. As shown in Table 11, decreasing the ratio leads to smaller model sizes from 175M at ratio $= 0.8$ down to 66M at ratio $= 0.1$ while performance degrades gradually. Validation perplexity increases from 32.788 at ratio $= 0.8$ to 39.741 at ratio $= 0.1$, with the largest jumps occurring when the ratio drops below 0.4.

These results characterize the trade-off between FFN width and model efficiency under a fixed architectural configuration. Reducing d_h leads to substantial parameter savings in the FFN, while

validation performance degrades gradually over a broad range of ratios. In particular, configurations with $d_h/d_{\text{model}} \geq 0.4$ maintain competitive performance despite significant reductions in model size. Below this range, performance degradation becomes more pronounced, indicating a transition where further FFN compression is less effective. Overall, the results identify a practical operating range for FFN width reduction in this small-scale hourglass setting.

A.8 Training FLOP Accounting

We adapted the flop estimation reported by (Hoffmann et al., 2022; McLeish et al., 2025). We report the train FLOPs for the forward and backward passes.

A python excerpt is as follows:

```python
def calculate_train_flops(
    d_model, d_model_attn, d_h, k_mlp_layers,
    n_layers, batch_size, seq_len, num_heads,
    use_qk_norm=True, vocab_size=100278,
):
    num_qheads = num_heads
    num_kvheads = 2 * num_qheads
    head_size = d_model_attn // num_heads
    d_h = d_h / 2  # SwiGLU splits d_h in half

    # === Forward Pass ===
    # Attention
    qkv_proj = batch_size * 3 * (2.0 * seq_len * d_model * d_model_attn)
    rope = batch_size * 3.5 * seq_len * (num_qheads + num_kvheads / 2) * head_size
    attention = qkv_proj + rope
    kq_logits = batch_size * seq_len * seq_len * d_model_attn
    softmax = batch_size * 3.0 * seq_len * seq_len * num_qheads
    attn_v = batch_size * 2.0 * seq_len * seq_len * d_model_attn
    attn_out = batch_size * 2.0 * seq_len * d_model * d_model_attn
    qk_norm = batch_size * 7.0 * (2 * d_model_attn) * seq_len if use_qk_norm else 0

    # MLP (K hourglass sub-layers)
    mlp_proj = k_mlp_layers * (batch_size * 6.0 * seq_len * d_model * d_h)
    mlp_act = k_mlp_layers * (batch_size * 10.0 * seq_len * d_h)

    # RMSNorm and residuals
    attn_norm = batch_size * 7.0 * d_model * seq_len
    attn_res = batch_size * d_model * seq_len
    mlp_norm = k_mlp_layers * (batch_size * 7.0 * d_model * seq_len)
    mlp_res = k_mlp_layers * (batch_size * d_model * seq_len)
    total_norm = attn_norm + mlp_norm
    total_res = attn_res + mlp_res

    # LM head
    lm_head = batch_size * (2 * seq_len * d_model * vocab_size)

    layer_fwd = (attention + kq_logits + softmax + attn_v + attn_out
                 + qk_norm + mlp_proj + mlp_act + total_norm + total_res)
    final_norm = batch_size * 7.0 * d_model * seq_len
    total_fwd = n_layers * layer_fwd + final_norm

    # === Backward Pass ===
    layer_attn_bwd = (2.0 * attention
        + 2.5 * (kq_logits + softmax + attn_v)
        + 2.0 * attn_out + 2.0 * qk_norm)
```

```
    layer_nonattn_bwd = 2.0 * (mlp_proj + mlp_act + total_res +
        total_norm)
    total_bwd = (n_layers * (layer_attn_bwd + layer_nonattn_bwd)
                 + 2.0 * final_norm)

    # === Total training FLOPs ===
    total = total_fwd + total_bwd + lm_head + 2.0 * lm_head
    return total
```

B Inference Cost Analysis

Formally speaking, let S be the cached context length, β the bytes per cached element ($\beta = 2$ for BF16), and g the per-layer KV-compression factor for different sequence mixers ($g = 1$ for MHA, $g = h/n_{\text{kv}}$ for GQA, and $g = 2d_{\text{attn}}/(r_{\text{KV}} + d_{\text{rope}})$ for the MLA cache layout considered here). The total KV-cache memory and the KV bytes read for one decoded token are

$$M_{\text{KV}}(S; g) = \frac{2\beta LSd_{\text{attn}}}{g} = \frac{4LSd_{\text{attn}}}{g} \text{ bytes in BF16.}$$

For MHA and GQA, the sequence-length-dependent QK and AV operations require approximately

$$F_{\text{attn}}(S) \simeq 4LSd_{\text{attn}} \text{ FLOPs}$$

per decoded token. Equivalently, the leading total decode costs in terms of compute and memeory can be written as

$$F_{\text{decode}}(S) \simeq F_{\text{fixed}} + 4LSd_{\text{attn}}, \qquad M_{\text{decode}}(S; g) \simeq M_{\text{weights}} + \frac{4LSd_{\text{attn}}}{g},$$

where the fixed terms include the FFN and projection operations and the model-weight traffic and BW_{eff} is the effective GPU memory bandwidth. MLA has additional implementation-dependent projection terms, but its context-dependent cache traffic remains linear in L, S, and its compressed KV width.

In the memory-bound regime, the corresponding first-order latency model is

$$t_{\text{decode}}(S; g) = t_{\text{fixed}} + \frac{4LSd_{\text{attn}}}{g\,\text{BW}_{\text{eff}}},$$

where t_{fixed} collects context-independent work such as weight access and kernel launches. Because the compared models match d_{attn} and use the same mixer, their relative KV-cache size is

$$\frac{M_{\text{KV,hg}}}{M_{\text{KV,conv}}} = \frac{L_{\text{hg}}}{L_{\text{conv}}}, \tag{8}$$

independent of g. Thus Hourglass reduces both the attention FLOPs and KV traffic associated with the removed layers: by 50.0% at 906M, 35.7% at 3B, and 31.3% at 8B.

As to the latency speed up, it can be written as

$$\text{speedup}(S; g) = \frac{c_{\text{fixed,conv}} + L_{\text{conv}}S}{c_{\text{fixed,hg}} + L_{\text{hg}}S}, \tag{9}$$

where we simplify the expression with $c_{\text{fixed}} = t_{\text{fixed}}(g\text{BW}_{\text{eff}})/(4d_{\text{attn}})$. The t_{fixed} is hard to quantify analytically due to consideration of the kernel operations and the sequential operations across layers. Nevertheless, as the sequence length increases, LS term should dominate, therefore approximating $\text{speedup}(S; g) \approx \frac{L_{\text{conv}}}{L_{\text{hg}}}$. This matches what is observed empirically for MHA at 64k context length, where speedups are achieved by $1.93\times$ at 906M, $1.52\times$ at 3B, and $1.43\times$ at 8B, respectively.

B.1 Empirical Comparison Across Sequence Mixers

The model above predicts that changing the sequence mixer moves g but leaves both the KV-cache ratio and the asymptotic speedup at $L_{\text{conv}}/L_{\text{hg}}$, shifting only the context length at which the asymptote is approached. We test this at 906M by swapping the mixer in both arms of the matched pair while holding $d_{\text{attn}} = 1536$, $L_{\text{conv}} = 24$, and $L_{\text{hg}} = 12$ fixed, and measuring single-token decode latency at 32k, 64k, and 128k. The GQA configuration uses $h = 16$ query heads over $n_{\text{kv}} = 4$ key–value heads ($g = 4$), and the MLA configuration uses $r_{\text{KV}} = 256$ and $d_{\text{rope}} = 32$ ($g \approx 10.7$).

Inference setting. Measurements use an NVIDIA RTX A6000 with batch size 1 and BF16, timing a single decoded token against a fixed cache of the stated length, with 5 warm-up and 20 timed iterations pinned to an otherwise idle device. Both architectures run through one identical code path per mixer, with the same attention backend selection and causal-mask handling and no architecture-specific kernel tuning. To avoid an $O(S^2)$ prefill at every length, we build the cache directly: a short prefill records each layer's exact key/value structure, which is then rebuilt at the target sequence length and decoded against once. The setup has two limitations: all measurements use batch size 1, and the decode kernel is not optimized, both of which reduce the observed speedup at a fixed context relative to the asymptote.

Table 12: **Decode speedup and KV-cache reduction across sequence mixers at 906M.** Speedup is conventional latency divided by Hourglass latency for a single decoded token, so higher is faster; the two arms differ only in layer count ($L = 24$ vs. 12). For every mixer the speedup rises toward the layer-count ratio $L_{\text{conv}}/L_{\text{hg}} = 2$ as context grows, and does so later for the mixers with larger g. The KV-cache reduction is exactly $1 - L_{\text{hg}}/L_{\text{conv}}$ for every mixer and every context length, as the memory model predicts.

	Decode speedup ↑			KV-cache
Sequence mixer	32k	64k	128k	reduction ↑
MHA ($g = 1$)	$1.93\times$	$1.96\times$	$1.98\times$	50.0%
GQA ($g = 4$)	$1.87\times$	$1.93\times$	$1.97\times$	50.0%
MLA ($g \approx 10.7$)	$1.87\times$	$1.94\times$	$1.97\times$	50.0%

Table 12 matches the inference memory model at Equation 8. The KV-cache reduction is 50.0% for every mixer at every length, confirming that the relative memory saving is independent of g. Latency behaves as predicted as well. At a fixed context, mixers with larger g sit further from the asymptote, because a smaller cache lets the context-independent term t_{fixed} occupy a larger share of the step; by 128k every mixer is within $0.03\times$ of the layer-count ratio and the layer ratio in Equation 9 dominates.