FROM TRANSFORMER TO TRANSPONDER: INTRODUCING CONTEXTUAL MODULATION TRAINING FOR RESIDUAL LEARNING IN LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers are the backbone of state-of-the-art systems across language, vision, and multimodal learning tasks, yet the relevance scale of their functional blocks (self-attention and feed-forward networks) is typically constant across inputs and depth. This static design neglects context-sensitive regulation of information flow through residual pathways. We introduce the *contextual modulator*: a lightweight, input-aware mechanism that can scale the outputs of linear sublayers within a block or the entire block output at token- and channel-level granularity. The modulator is implemented via compact parametric functions and adds negligible parameter overhead. Building on this idea, we propose TRANSPONDER, which integrates contextual modulators throughout Transformer blocks to endow functional residual architectures with fine-grained, input-adaptive control. TRANSPONDER provides evident improvement over six other scaling or normalization methods across LLaMA backbones ranging from 60M to 250M parameters, yielding consistent perplexity reductions with $< 1\%$ additional parameters. Analysis reveals depth-, module-, and token-specific scaling patterns, indicating that learned modulators act as input-adaptive regulators of residual information flow. TRANSPONDER provides a simple, general mechanism to augment Transformer-based models with context-sensitive modulators, providing robust and significant performance improvements without substantial architectural changes.

1 INTRODUCTION

Transformer architectures deliver state-of-the-art performance across vision, language, and multimodal tasks (Vaswani et al., 2017; Devlin et al., 2019; Radford et al., 2018; Touvron et al., 2023). In standard designs, information flows through a sequence of linear and nonlinear components—query/key/value projections, attention output projections, and the two feed-forward networks (FFN) linear maps—interleaved with residual connections. After training, the scaling by which these components contribute to the residual stream is effectively static for all the input tokens. Several mechanisms attempt to manipulate residual or path scaling via global reparameterizations (e.g., ReZero (Bachlechner et al., 2021), DeepNorm/DeepNet (Wang et al., 2024)) or depth-aware normalization (Li et al., 2024; Sun et al., 2025). Yet these approaches assign fixed gains in spite of the current token, channel, or depth position.

These designs clash with the core goal of representation learning: the relevance of a component's output is input-dependent. Rare tokens may call for amplifying specific heads; bursty channels may require attenuation; deeper layers may benefit from different mixing strengths than shallow ones. When scaling is static, the model must implicitly encode such selectivity inside the functional maps themselves—mixing representation (what to compute) with control (how much to pass through).

To address this limitation, we introduce TRANSPONDER, whose core principle is simple: **pair each Transformer component with a contextual, input-aware modulator that explicitly controls its contribution at inference**. Specifically, for a component with output $\mathbf{Y} = \mathbf{F}(\mathbf{X})$, we attach a lightweight controller $g(\mathbf{X}) \in [0, 1]$ and compute $\tilde{\mathbf{Y}} = \mathbf{F}(\mathbf{X}) \odot g(\mathbf{X})$. This decouples representation from control, turning residual mixing from a static heuristic into a principled, context-sensitive regulation mechanism.

We instantiate this principle with TRANSPONDER, designed to be minimally invasive and broadly compatible:

1. **Input-aware modulation.** Modulators calculate the scales that multiply functional-path outputs at use time, yielding context-conditioned scaling factors for each component in Transformer.
2. **Granularity and placement.** We systematically study where to modulate (projection- vs. path-level) and at what resolution (scalar vs. channel-wise), providing a comprehensive view of practical design.
3. **Low overhead, easy integration.** TRANSPONDER adds $\leq 1\%$ parameters and integrates into standard Transformer blocks via a fused Triton kernel for each modulator to accelerate inference.

Across large-scale language-modeling benchmarks, TRANSPONDER delivers consistent gains on LLaMA backbones from 60M to 250M parameters, achieving up to 5.8–15.3% relative perplexity reductions over the LLaMA baselines. Extensive ablations spanning placement, granularity, hidden size, and component coverage (attention and FFN) show the robustness of the design. Analysis of learned modulator values uncover depth-, module-, and token-specific patterns that adapt layer-wise contributions to input semantics, providing direct evidence that residual functional transformations benefit from adaptive, context-aware scaling.

2 RELATED WORK

2.1 RESIDUAL CONNECTIONS AND THE TRANSFORMER ARCHITECTURE

Residual connections (He et al., 2016) enable very deep models by adding a learned transformation to an identity shortcut, improving gradient flow, preserving signal, and allowing layers to refine rather than reconstruct representations. The Transformer (Vaswani et al., 2017) instantiates this principle with a persistent residual stream that carries token-wise state across layers while each block applies a functional path—multi-head self-attention (MHA) followed by a position-wise feed-forward network (FFN). Concretely, each block computes linear projections for queries, keys, and values and an attention output, then applies two FFN linear maps with nonlinearities; the results are mixed back into the residual stream through additive shortcuts. Layer normalization (LayerNorm) is interleaved with these sublayers: the original Post-LN normalizes after each sublayer (Vaswani et al., 2017), whereas Pre-LN normalizes before sublayers, improving optimization stability for deeper stacks (Xiong et al., 2020). This decomposition—identity carryover plus functional updates—yields strong gradient propagation and composability, but it also implicitly fixes the contribution of each subcomponent to the residual stream at inference, motivating methods that explicitly control (or scale) block updates relative to the identity path.

2.2 SCALING THE BLOCK: STATIC REPARAMETERIZATIONS AND NORMALIZATION PLACEMENT

A substantial line of work manipulates the magnitude of each block's update relative to the residual stream via static, input-agnostic mechanisms. ReZero (Bachlechner et al., 2021) introduces a zero-initialized, learnable scalar per block that gates the residual branch, effectively starting from an identity mapping and allowing depth to emerge during training. DeepNorm/DeepNet (Wang et al., 2024) analytically rescales residual and sublayer outputs to maintain stability in very deep Transformers, enabling substantially deeper stacks without divergence. LAuReL (Menghani et al., 2024) generalizes residual/functional mixing with learned coefficients (e.g., RW/LR/PA variants), providing a tunable trade-off between identity flow and the functional path. While these approaches improve training dynamics and depth, their coefficients are typically block-level and input-agnostic at inference time, leaving the strength of updates fixed for any given input.

2.3 CONTEXT-CONDITIONED GATING METHODS

In convolutional networks, squeeze-and-excitation (SE) (Hu et al., 2019) introduces feature-wise, input-conditioned scaling by pooling global context and learning channel gates, improving repre-

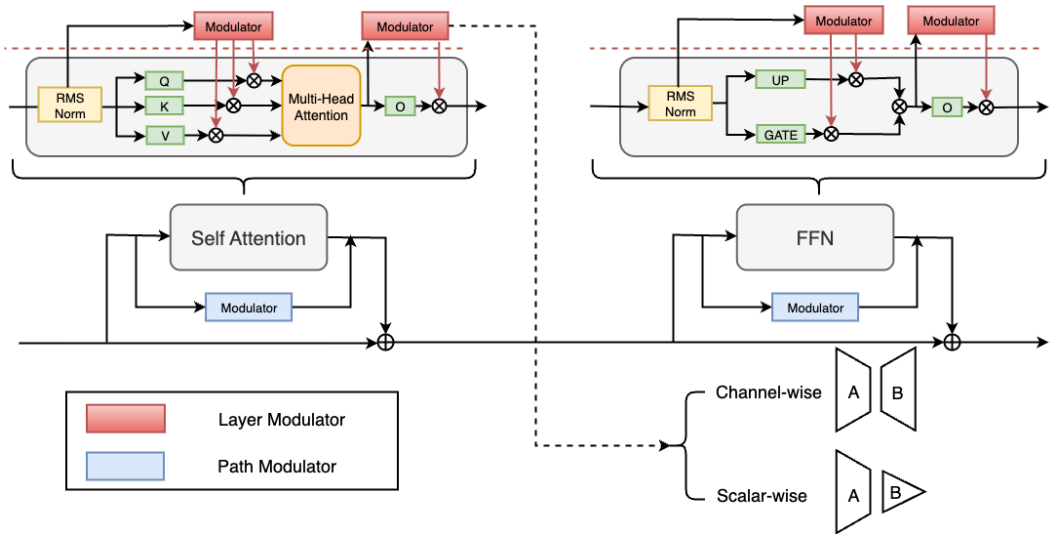

Figure 1: **Illustration of the TRANSPONDER mechanism.** The Figure shows a LLaMA decoder block, consisting of a self-attention module followed by a feed-forward network (FFN). Layer-Modulator (red) dynamically rescales the outputs of key linear transformations (e.g. Q, K, V, O) within both modules, while Path-Modulator (blue) modulates the functional paths directly (e.g. Self-attention, FFN). Each Modulator can operate in two modes: channel-wise control, where the output dimension matches the original feature dimension, and scalar-wise control, where a single scalar is used to regulate the entire function/layer.

sentational efficiency and accuracy. The core idea—context-aware modulation of features—has influenced architectures beyond convolution. Within Transformers, SDPA-gated mechanisms (Qiu et al., 2025) attach the gating mechanism to scaled dot-product attention, commonly parameterized at the head or channels. These gates selectively modulate attention behavior and can stabilize training, but they often incur heavy additional parameters and computation, and primarily target the attention pathway.

3 TRANSPONDER - CONTEXTUAL MODULATION TRAINING

We formalize the TRANSPONDER by introducing the lightweight, input-conditioned modulators that scale the functional information flow inside Transformer blocks. The goal is to convert static functional compositions into context-aware modulation while preserving (i) the inductive bias of the base operator (attention/MLP), (ii) training stability, and (iii) negligible parameter/FLOP overhead. We show the main architecture of the TRANSPONDER in Figure 1.

3.1 PRELIMINARIES: RESIDUAL FUNCTIONAL COMPOSITION IN TRANSFORMERS

Let $\mathbf{Y}_{l-1} \in \mathbb{R}^{T\times d}$ denote the hidden states entering layer l (sequence length T, width d), and let $\mathcal{F}_l$ denote a parameterized sublayer (self–attention or FFN). The standard residual formulation composes identity and functional streams additively:

$$\mathbf{Y}_l \;=\; \mathbf{Y}_{l-1} \;+\; \mathcal{F}_l\big(\mathbf{Y}_{l-1}\big). \tag{1}$$

While Eq. equation 1 ensures stable signal and gradient propagation, the contribution of $\mathcal{F}_l$ is typically static at inference time: once trained, the magnitude with which $\mathcal{F}_l$ perturbs the skip path does not depend on the current token, channel, or context. Prior work improves optimization via block-wise scalars (e.g., ReZero Bachlechner et al. (2021)), depth-aware reparameterizations (e.g., DeepNorm Wang et al. (2024)), or by mixing identity and functional streams with learned but input-agnostic coefficients (e.g., LAuReL Menghani et al. (2024)). Our aim is to endow Eq. equation 1 with contextual modulators applied to, or inside, $\mathcal{F}_l$.

3.2 Definition: Transponder Scheme

A *Modulator* $\mathcal{G}_l$ produces a contextual scaling signal that modulates a sublayer at its point of use. For a generic operator $\mathcal{F}_l$ acting on $\mathbf{X}$,

$$\mathbf{Y} \;=\; \mathcal{F}_l(\mathbf{X}) \;\odot\; \mathcal{G}_l(\mathbf{X}), \tag{2}$$

where $\odot$ denotes broadcasted Hadamard multiplication. In a residual block,

$$\mathbf{Y}_l \;=\; \mathbf{Y}_{l-1} \;+\; \Big[\mathcal{F}_l\big(\mathbf{Y}_{l-1}\big) \odot \mathcal{G}_l\big(\mathbf{Y}_{l-1}\big)\Big]. \tag{3}$$

Modulator parameterization. We parameterize the modulator $\mathcal{G}_l$ with a compact bottleneck applied to each input activation $\mathbf{X}$, followed by a bounded nonlinearity:

$$\mathcal{G}_l(\mathbf{X}) \;=\; \tilde{\sigma}_{\alpha_l}\Big(\mathbf{B}_l\,\phi\big(\mathbf{A}_l\,\mathbf{X}\big)\Big), \qquad \mathbf{A}_l \in \mathbb{R}^{r\times d_{\text{in}}},\;\; \mathbf{B}_l \in \mathbb{R}^{d_{\text{out}}\times r}, \tag{4}$$

where d_{in} is the input width of the summary, d_{out} matches the modulation resolution (one single scalar or channel-wise. ϕ is a pointwise nonlinearity (e.g., Sigmoid), and $\tilde{\sigma}_\alpha$ is a learnable curvature sigmoid,

$$\sigma_\alpha(x) \;=\; \frac{1}{1+\exp(-\alpha x)}, \qquad \alpha > 0, \qquad \tilde{\sigma}_\alpha(x) \;=\; 2\,\sigma_\alpha(x). \tag{5}$$

The base gate σ_α maps to $[0,1]$ and interpolates between soft scaling (small α) and near-binary gating (large α). We use the calibrated gate $\tilde{\sigma}_\alpha \in [0,2]$ so that $\tilde{\sigma}_\alpha(0) = 1$, preserving the expected residual magnitude at initialization.

3.3 Granularity: Where and at What Resolution to Modulate?

We distinguish *where* the Modulator is applied (inter–layer) from *how finely* it is parameterized (intra–layer).

Inter–layer granularity.

- **Projection–level modulation.** Apply a modulator after each linear projection inside a functional path:

$$\underbrace{\mathbf{W}\mathbf{X}}_{\text{projection}} \;\mapsto\; \big(\mathbf{W}\mathbf{X}\big) \odot \mathcal{G}_l(\mathbf{X}). \tag{6}$$

  This affords precise control (e.g., on Q, K, V and MLP up/down projections) with minimal additional computation.
- **Path–level modulation.** Modulate the entire functional path (e.g., self–attention or FFN):

$$\mathbf{Y}_l \;=\; \mathbf{Y}_{l-1} \;+\; \mathcal{F}_l\big(\mathbf{Y}_{l-1}\big) \odot \mathcal{G}_l\big(\mathbf{Y}_{l-1}\big). \tag{7}$$

Intra–layer granularity. Let d_{out} be the dimension of $\mathcal{F}_l$'s output at the modulation site. We instantiate:

- **Layer–wise (scalar) modulation:** $\mathcal{G}_l(\mathbf{X}) \in \mathbb{R}$ via $\mathbf{B}_l \in \mathbb{R}^{1\times r}$. A single scale uniformly modulates the sublayer; parameter and compute overheads are negligible.
- **Channel–wise modulation:** $\mathcal{G}_l(\mathbf{X}) \in \mathbb{R}^{d_{\text{out}}}$ via $\mathbf{B}_l \in \mathbb{R}^{d_{\text{out}}\times r}$, enabling feature–specific modulating with two low–rank matrices.

3.4 Fused Projection for Projection–Level Transponder

For each linear layer, the dominant cost is the matrix product $\mathbf{X}\mathbf{W}_\star$. When the summary $u_\star(\mathbf{X})$ is token-wise, we also need $\mathbf{X}\mathbf{A}_\star^\top$. Instead of launching two kernels, we concatenate along the output dimension and perform a single tiled GEMM:

$$\big[\,\mathbf{Z}_\star \;\; \mathbf{U}_\star\,\big] \;=\; \mathbf{X}\;\underbrace{\big[\,\mathbf{W}_\star \,\big|\, \mathbf{A}_\star^\top\,\big]}_{\mathbf{C}_\star \in \mathbb{R}^{d_{\text{in}}\times(d_{\text{out}}+r)}}, \qquad \mathbf{Z}_\star \in \mathbb{R}^{B\times T\times d_{\text{out}}},\;\; \mathbf{U}_\star \in \mathbb{R}^{B\times T\times r}. \tag{8}$$

Here $\mathbf{A}_\star \in \mathbb{R}^{r \times d_{\text{in}}}$ is the rank-r bottleneck, $\mathbf{W}_\star \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ is the projection, B is batch size, and T is sequence length. After the fused GEMM, we compute the modulator logits and modulate in-register per tile:

$$\text{logits}_\star \;=\; \mathbf{U}_\star\, \mathbf{B}_\star^\top, \qquad \mathbf{B}_\star \in \mathbb{R}^{d_{\text{out}} \times r}, \qquad \boldsymbol{\gamma}_\star \;=\; \tilde{\sigma}_{\alpha_\star}\big(\text{logits}_\star\big) \in \begin{cases} \mathbb{R}^{B \times T \times 1}, & \text{scalar gate,} \\ \mathbb{R}^{B \times T \times d_{\text{out}}}, & \text{channel-wise gate.} \end{cases} \tag{9}$$

Finally, we apply broadcasted modulation to the projected activations:

$$\widehat{\mathbf{Z}}_\star \;=\; \boldsymbol{\gamma}_\star \odot \mathbf{Z}_\star. \tag{10}$$

This fusion amortizes the summary computation into the main GEMM and keeps the modulator arithmetic on-chip, yielding precise projection-level control at minimal overhead.

4 Experiments

4.1 Setup

We evaluate Transponder on standard language modeling with LLaMA backbones Touvron et al. (2023) ranging from $\sim$ 60M to $\sim$ 250M parameters, trained on the C4 Raffel et al. (2020) and OpenWebText Gokaslan & Cohen (2019) corpus. All models are optimized with Adam and a base learning rate of 3×10^{-3} for Openwebtext dataset and 1×10^{-3} for C4 dataset. All the experiments are trained with bfloat16 for all the activations, weights, and optimization states. The hidden dimension of the contextual modulator is set to $r = 8$ in the main comparisons to emphasize gains from contextual modulation rather than capacity. For fairness, we keep training data, tokenization, optimizer settings, and learning-rate schedules identical across baselines and Transponder. We report the perplexity (the lower the better) as the standard language-modeling metrics and measure parameter overhead relative to the corresponding backbone. We initialize A, B of the Transponder with Kaiming uniform He et al. (2015) and zero biases; set α=1 initially. Detailed hyperparameter settings can be found in Table 7.

4.2 Baselines

We compare against widely used residual/normalization designs and recent residual-scaling or layer-wise contextual control methods:

- **DeepNorm/DeepNet** Wang et al. (2024): rescales the residual branch with depth-dependent constants and tailored initialization to stabilize training of very deep Transformers.
- **LayerNorm Scaling** Sun et al. (2025): multiplies each layer's normalized output by a depth-aware factor (e.g., $\propto 1/\sqrt{L}$) to curb variance growth across residual connections.
- **LAuREL** Menghani et al. (2024): learns a lightweight mixer that blends the skip and transformed paths (LR/PA variants), improving layer-wise signal routing with minimal extra parameters.
- **SDPA-Gate** Qiu et al. (2025): applies a per-head sigmoid gate to the scaled dot-product attention output to modulate contribution and mitigate attention-sink effects.

To match SDPA-Gate's mechanism more broadly, we also implement an **ALL-Gate** variant that applies the same gating scheme to all linear sublayers, including the MLP projections, providing a stronger control baseline.

4.3 Modulator Variants Validation

We instantiate five different variants of the contextual modulator in the framework of the Transponder to probe where and at what resolution modulation helps most:

- **Modulator-path-scalar**: a single scalar Modulator modulates the entire functional path (attention output or MLP output) per block.

Table 1: Validation perplexity (PPL, lower is better) of different variants of contextual modulators on Openwebtext. "Param ↑" denotes the relative parameter increase vs. LLaMA baseline. The best performance is highlighted in bold.

	60M		130M		250M	
Model	PPL	Param ↑	PPL	Param ↑	PPL	Param ↑
Modulator-path-scalar	62.91	$<1\%$	18.29	$<1\%$	1088	$<1\%$
Modulator-path-channel	23.25	$<1\%$	18.91	$<1\%$	16.14	$<1\%$
Modulator-layer-scalar	23.15	$<1\%$	17.76	$<1\%$	15.21	$<1\%$
Modulator-layer-channel	23.08	1%	17.56	1%	15.17	1%
Modulator-layer-channel-scalar	**22.50**	1%	**17.45**	1%	**14.93**	1%

- **Modulator-path-channel**: a channel-wise Modulator modulates the entire functional path (attention output or MLP output) output channels.
- **Modulator-layer-scalar**: a single scalar Modulator per linear projection.
- **Modulator-layer-channel**: a channel-wise Modulator per linear projection.
- **Modulator-layer-channel-scalar (TRANSPONDER)**: a channel-wise Modulator per linear projection and a single scalar output Modulator per linear projection.

Table 1 shows two consistent trends across model scales. (i) The functional-path modulator does not aid training and can even destabilize optimization, whereas the layer-wise modulator consistently improves both training dynamics and final performance. (ii) With a layer-wise modulator in place, both channel-wise and scalar modulators further benefit TRANSPONDER: the scalar provides coarse, global control, while the channel-wise variant offers finer-grained adjustments. Using both together yields an additional, robust gain with only a modest $\sim 1\%$ parameter overhead.

Given its consistently best validation perplexity under comparable budgets and its favorable integration cost, we adopt **Modulator–layer–channel–scalar** as the default TRANSPONDER configuration for all subsequent comparisons against baseline methods.

4.4 MAIN RESULTS FOR LANGUAGE MODELING.

Table 2 compares validation perplexity (PPL; lower is better) across three LLaMA model scales. From the results, TRANSPONDER attains the best PPL wherever reported with only $\sim$!1% parameter overhead. Compared to the LLaMA baseline, TRANSPONDER reduces PPL on OpenWebText by **15.3%** at 60M (26.56→22.50), **9.4%** at 130M (19.27→17.45), and **13.6%** at 250M (17.28→14.93); on **C4** the reductions are **5.8%** at 60M (30.31→28.55), **17.4%** at 130M (26.73→22.09), and **14.6%** at 250M (21.92→18.72). These gains persist through 250M under larger token budgets, indicating strong large-scale behavior.

Capacity-heavy gates (SDPA-Gate, ALL-Gate) improve PPL but require 15–80% extra parameters. In contrast, TRANSPONDER matches or exceeds these improvements with only 1% overhead. Among minimal/zero-overhead baselines, LayerNorm Scaling is competitive yet still trails.

Several baselines that attempt to learn the mixing between the residual path and functional blocks exhibit instability at some scales (e.g., DeepNet and LAuReL-LR explode on C4-130M, 143.65 and 106.94 PPL; LAuReL-PA diverges on C4-130M with 1355 PPL and collapses on OpenWebText-250M to 257 PPL), whereas TRANSPONDER delivers consistent improvements across all reported settings.

Under comparable budgets, TRANSPONDER achieves state-of-the-art PPL with minimal overhead, strong large-scale behavior, and improved robustness, establishing it as our default configuration for subsequent comparisons.

4.5 STABILITY WITH POST-LAYERNORM

Beyond the Pre-LN setting, we evaluate TRANSPONDER under the more delicate Post-LN regime. As shown in Table 3, vanilla LLaMA with Post-LN is unstable at scale: PPL explodes at 250M

Table 2: Validation perplexity (PPL, lower is better) on C4 and Openwebtext. "Param ↑" denotes the relative parameter increase vs. LLaMA.

Training tokens		60M 1.2B		130M 2.2B		250M 3.9B	
Dataset	Model	PPL	Param ↑	PPL	Param ↑	PPL	Param ↑
Openwebtext	LLaMA (baseline)	26.56	-	19.27	-	17.28	-
	DeepNet	23.78	$<1\%$	18.74	$<1\%$	16.53	$<1\%$
	LAuReL-LR	23.81	$<1\%$	18.19	$<1\%$	16.75	$<1\%$
	LAuReL-PA	23.37	$<1\%$	18.22	$<1\%$	257	$<1\%$
	LayerNorm Scaling	23.31	0%	18.28	0%	16.16	0%
	SDPA-Gate	23.19	15%	18.03	22%	15.44	23%
	ALL-Gate	22.96	44%	17.91	65%	14.97	80%
	TRANSPONDER	**22.50**	1%	**17.45**	1%	**14.93**	1%
C4	LLaMA (baseline)	30.31	-	26.73	-	21.92	-
	DeepNet	30.18	$<1\%$	143.65	$<1\%$	21.72	$<1\%$
	LAuReL-LR	30.05	$<1\%$	106.94	$<1\%$	39.14	$<1\%$
	LAuReL-PA	29.50	$<1\%$	1355	$<1\%$	20.88	$<1\%$
	LayerNorm Scaling	29.77	0%	25.76	0%	20.35	0%
	SDPA-Gate	29.53	15%	23.18	22%	18.95	23%
	ALL-Gate	28.98	44%	22.92	65%	18.82	80%
	TRANSPONDER	**28.55**	1%	**22.09**	1%	**18.72**	1%

Table 4: Validation perplexity (PPL, lower is better) on Openwebtext comparing non-contextual and contextual variants.

	60M	130M	250M
Original LLaMA	26.56	19.27	17.28
TRANSPONDER w/o contextual control	23.40	19.29	16.29
TRANSPONDER w/o learnable sigmoid	23.56	17.70	**14.93**
TRANSPONDER	**22.50**	**17.45**	**14.93**

(1409.79), and even at 130M performance degrades to 26.95 in comparison to the Pre-LN LLaMA. In contrast, TRANSPONDER stabilizes training and improves accuracy in both cases, yielding a **4.6%** PPL reduction at 130M (26.95→25.71) and preventing divergence at 250M, where it achieves **20.28** PPL instead of catastrophic failure. While Post-LN remains slightly behind strong Pre-LN baselines at a similar scale, these results indicate that TRANSPONDER substantially enlarges the viable training regime for Post-LN models with only $\sim 1\%$ overhead, mitigating the well-known optimization fragility of Post-LN Transformers.

4.6 ABLATIONS AND SENSITIVITY TESTS

Contextual vs. Non-Contextual. To assess the contribution of contextual modulation, we compare TRANSPONDER with and without contextual signals. In the non-contextual variant, for each layer l, we replace the contextual pathway with learnable parameters—a layer-wise scalar $\beta_1^{(l)}$ and a channel-wise vector $\boldsymbol{\beta}_2^{(l)}$—and set

$$\mathcal{G}_l \;=\; \tilde{\sigma}_{\alpha_l}\Big(\beta_1^{(l)} \cdot \boldsymbol{\beta}_2^{(l)}\Big),$$

where $\beta_1^{(l)}$ applies uniform scaling and $\boldsymbol{\beta}_2^{(l)}$ provides per-channel modulation.

Table 3: Validation perplexity of LLaMA 130M and 250M on C4 with Post-Layer Norm.

	130M	250M
LLaMA w. Post-LN	26.95	1409.79
TRANSPONDER w. Post-LN	**25.71**	**20.28**

Table 5: Ablation study on modulators on different functional modules for the TRANSPONDER on C4 dataset. The reported metric is Perplexity. The best performance is highlighted in bold.

Modules	LLaMA60M	LLaMA130M
Baseline	30.31	26.07
self-attention	29.04	23.21
mlp	29.43	23.36
only_last	29.88	23.97
only_first	29.91	24.11
only_qk	30.01	24.03
w/o up and gate	29.12	22.93
all	**28.55**	**22.09**

Table 6: Ablation study on the hidden dimension r of both the channel-wise and scalar-wise modulators for the TRANSPONDER on C4 dataset. The reported metric is Perplexity. The best performance is highlighted in bold.

Hidden Size	LLaMA60M	LLaMA130M
Baseline	30.31	26.07
2	28.91	23.01
4	28.65	22.64
8	28.55	22.09
16	**28.41**	22.12
32	28.73	**22.07**

As shown in Table 4, the non-contextual variant already improves over the LLaMA baseline at certain scales, confirming the utility of the proposed modulator design. However, adding contextual control consistently delivers much larger gains across all model sizes, establishing it as the key driver of performance. This result directly supports our initial motivation: each component benefits from input-aware, contextual control rather than relying solely on static modulation.

Sigmoid and Learnable Sigmoid. A key innovation of TRANSPONDER is the introduction of a learnable sigmoid. This learnable non-linear modulation mechanism allows each modulator to dynamically adjust its sensitivity to the input, either amplifying or attenuating modulation strength as needed. As shown in Table 4, incorporating the learnable sigmoid consistently improves (except for 250M that is comparable)perplexity across model sizes, validating our design intuition that adaptive nonlinearity provides a crucial layer of flexibility for effective modulation.

Hidden dimensions. We further examine the effect of the hidden dimension r in the contextual modulator. Table 6 shows that even with an extremely compact setting ($r = 2$), TRANSPONDER already delivers large gains over the LLaMA baseline. Scaling up the hidden size from 8 to 32 provides marginal yet consistent improvements. These findings suggest that the modulator does not require a large intermediate capacity to capture input-dependent scaling, underscoring the efficiency of our design: lightweight modulators suffice to model contextual dependencies while adding negligible parameter overhead.

Contribution of functional components. We next investigate which functional modules benefit most from modulation by selectively applying the contextual modulators to individual subcomponents (Table 5). Applying modulators to either the self-attention or MLP block alone yields clear improvements, while restricting modulation to partial components (e.g., only_first, only_last, only_qk) provides only limited gains.

We further evaluate the effect of removing the up- and gate-projections. In the original LLaMA design, the gate projection serves as a channel-wise modulator to the up-projection, though implemented as a full-rank matrix. Interestingly, excluding these projections leads to strong performance, second only to the full "all" configuration. This suggests that the gate-projections already play a key role in channel-wise contextual modulation.

Overall, the best results are consistently achieved when all components are modulated jointly ("all"), yielding the lowest PPL on both LLaMA-60M and LLaMA-130M. This confirms that full-path functional modulation is necessary to provide comprehensive control over residual transformations, enabling robust and consistent improvements across scales.

5 CONTEXTUAL MODULATOR VALUES ANALYSIS

The results in Section 4 suggest that, beyond static architectural scaling (e.g., fixed layer-normalization schemes), context-/input-sensitive modulation is a powerful way to improve optimization and expressivity in deep language models. TRANSPONDER learns, for each token and

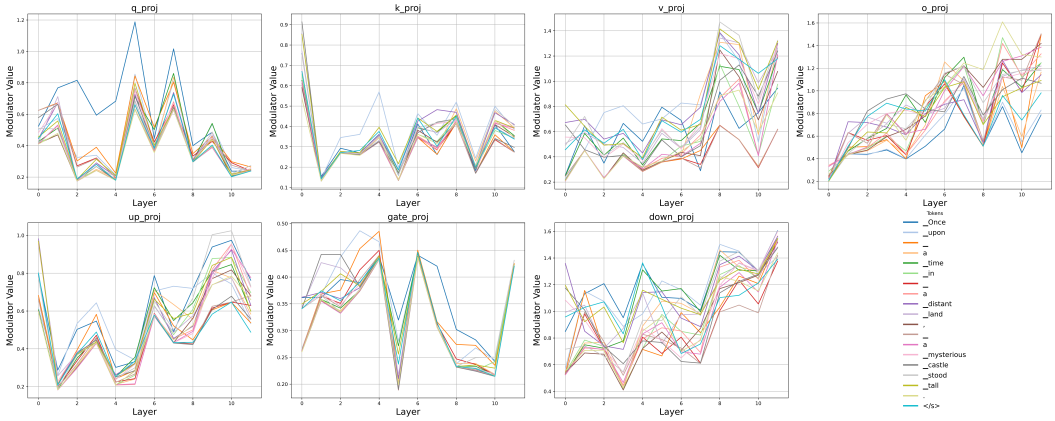

Figure 2: Token-wise modulator values across layers and modules in a LLaMA-130M model trained with TRANSPONDER. We report the values of the scalar modulator as the representative. Each subplot corresponds to a specific linear projection of the decoder layer. The plotted curves represent different tokens from the input sequence.

sub-layer (self-attention and FFN), both a layer scalar and channel-wise modulation. The question is whether these modulators truly encode context rather than acting as static rescalers.

Figure 2 (layer-scalar, per module) shows clear depth- and module-specific structure with token-dependent variation. The traces for different tokens diverge at many layers, indicating context sensitivity. We observe systematic trends across modules: `o_proj` and `down_proj` steadily strengthen with depth, consistent with greater late-layer amplification; `v_proj` exhibits a pronounced surge in upper layers; `up_proj` follows a U-shaped pattern (early attenuation, late amplification); `gate_proj` remains in a tighter band with a mid-depth peak; and `q_proj`/`k_proj` show early suppression followed by recovery.

Figure 3 further confirms the effectiveness of the channel-wise modulation. Early layers are relatively flat, but mid–late layers develop structured ridges over both tokens and channels, with increasing contrast across depth. In particular, we observe a similar trend at some specific layers. For instance, at the 5th layer, one token exhibits very large modulator values across all channels, whereas in the 11th layer, one channel displays very large modulation values across the tokens.

Taken together, the two views indicate that TRANSPONDER learns structured, context-aware gains that evolve with depth and specialize by module and channel—precisely the behavior we would expect if the modulators were encoding useful contextual signals rather than acting as fixed, global scalars.

6 CONCLUSION

In this article, we proposed TRANSPONDER, a simple but effective way to modulate the sub-layers in Transformers *context-aware*. By pairing key projections and paths with compact modulators that produce scalar- and channel-wise gates, TRANSPONDER explicitly controls each component's contribution at inference, separating representation from control. The design is minimally invasive—$\leq 1\%$ parameter overhead—and broadly compatible with standard decoder blocks. Empirically, TRANSPONDER consistently improves language modeling quality across LLaMA backbones (60M–250M) on OpenWebText and C4. Under comparable budgets, it surpasses stronger, capacity-heavy gating schemes while remaining far lighter, and it outperforms static depth-aware scaling methods. Ablations map a practical design space. We find (i) layer-wise placement is preferable to coarse functional-path-only control; (ii) combining channel-wise modulator with a single scalar modulator per layer yields the best accuracy-cost trade-off; (iii) modulating all the layers is crucial. Modulator value analyses further show structured, token- and depth-dependent behaviors, providing direct evidence that adaptive scaling learns semantically meaningful control. The limitation of this work can be found in Appendix A.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, pp. 1352–1361. PMLR, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015. URL https://arxiv.org/abs/1502.01852.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019. URL https://arxiv.org/abs/1709.01507.

Pengxiang Li, Lu Yin, and Shiwei Liu. Mix-ln: Unleashing the power of deeper layers by combining pre-ln and post-ln. *arXiv preprint arXiv:2412.13795*, 2024.

Gaurav Menghani, Ravi Kumar, and Sanjiv Kumar. Laurel: Learned augmented residual layer. *arXiv preprint arXiv:2411.07501*, 2024.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free, 2025. URL https://arxiv.org/abs/2505.06708.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

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.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

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

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020. URL https://arxiv.org/abs/2002.04745.

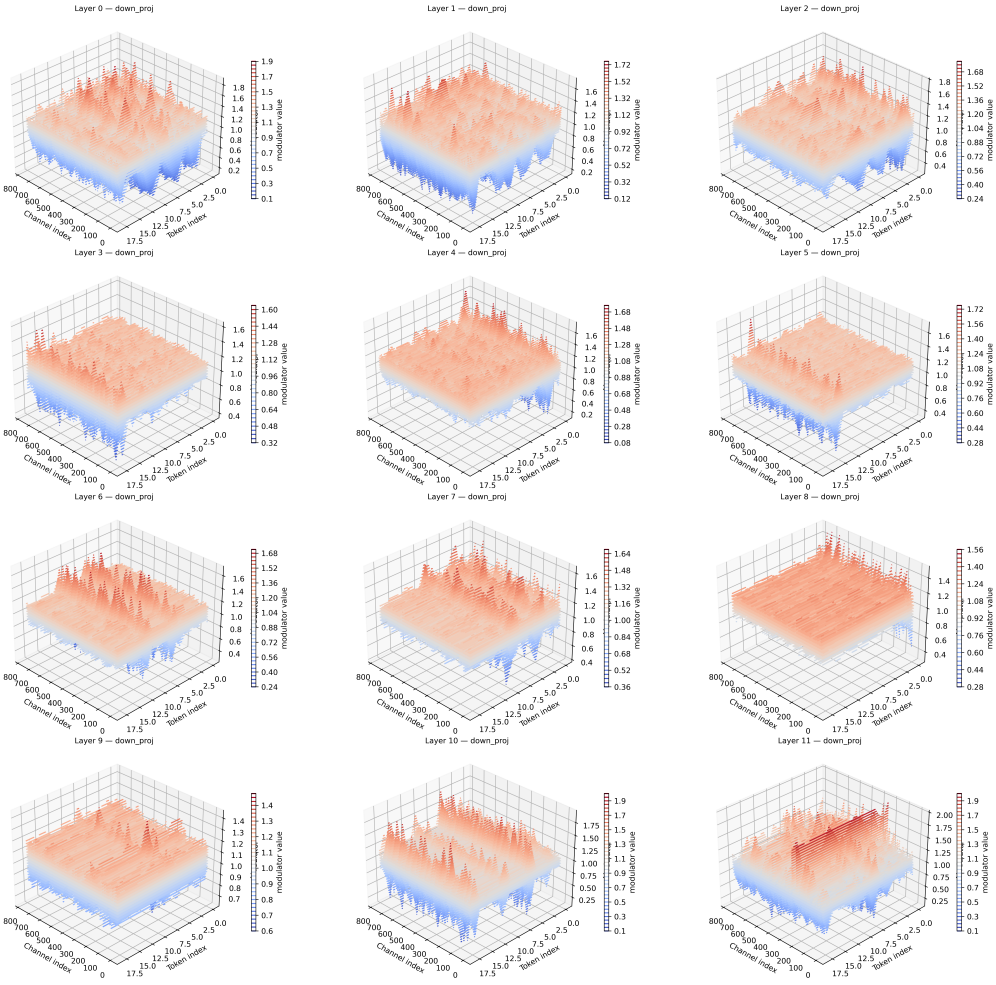

Figure 3: Channel-wise modulator values on down_proj across layers and tokens in a LLaMA-130M model trained with TRANSPONDER. The X and Y are the channel index and the token index, while the Z-axis are the channel-wise modulator values.

Table 7: **Hyperparameters of LLaMA-60M, LLaMA-130M, and LLaMA-250M on OpenWebText and C4.**

Hyper-parameter	LLaMA-60M	LLaMA-130M	LLaMA-250M
Embedding Dimension	512	768	1024
Feed-forward Dimension	1376	2048	2560
Global Batch Size	512	512	512
Sequence Length	256	256	256
Training Steps	10000	20000	40000
Learning Rate	3e-3	3e-3 (1e-3 for C4)	3e-3 (1e-3 for C4)
Warmup Steps	1000	2000	4000
Learning rate Decay Method	noam	noam	noam
Optimizer	Adam	Adam	Adam
Layer Number	8	12	24
Head Number	8	12	16

A LIMITATION

Our study is compute-limited: we did not scale beyond $\sim$1B parameters or train for insufficient tokens because of the limits of the computational resources. In this regime, several zero-shot classification benchmarks yield Matthews correlation coefficient (MCC) scores near 0, which we attribute to undertraining rather than an inherent limitation of the current investigation of TRANSPONDER. Future work should evaluate TRANSPONDER at larger scales and with substantially longer training runs (more steps and tokens), and reassess zero-shot (and few-shot) generalization under those settings.

B CLAIM OF THE LLM USAGE

We used LLM-based tools to improve the language and flow; the principles, core logic, and innovations are entirely the authors'.

C REPORDUCTION STATEMENT

All experiments were conducted using the Hugging Face pretraining framework with data parallelism over $8\times$ NVIDIA A100 (40 GB) GPUs. For DeepNorm and LayerNorm scaling, we report the results from Sun et al. (2025), as we adopt identical hyperparameter settings. To ensure reproducibility, we include our code and step-by-step instructions in the supplementary materials. Because LAuReL has not released its source code, we reimplemented the method based on our understanding of the paper. Detailed training hyperparameters are provided in Table 7.