ReLU's Revival: On the Entropic Overload in Normalization-Free Large Language Models

Nandan Kumar Jha
New York University
nj2049@nyu.edu

Brandon Reagen
New York University
bjr5@nyu.edu

Abstract

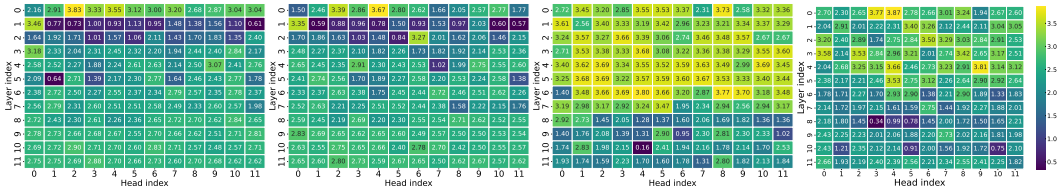
LayerNorm is a critical component in modern large language models (LLMs) for stabilizing training and ensuring smooth optimization. However, it introduces significant challenges in mechanistic interpretability, outlier feature suppression, faithful signal propagation, and computational and communication complexity of private inference. This work explores desirable activation functions in normalization-free decoder-only LLMs. Contrary to the conventional preference for the GELU in transformer-based models, our empirical findings demonstrate an *opposite trend*—ReLU significantly outperforms GELU in LayerNorm-free models, leading to an **8.2%** perplexity improvement. We discover a key issue with GELU, where early layers experience entropic overload, leading to the under-utilization of the representational capacity of attention heads. This highlights that smoother activations like GELU are *ill-suited* for LayerNorm-free architectures, whereas ReLU's geometrical properties—specialization in input space and intra-class selectivity—lead to improved learning dynamics and better information retention in the absence of LayerNorm. This study offers key insights for optimizing transformer architectures where LayerNorm introduces significant challenges. The code and implementation are available at relu-revival-normfree.

1 Introduction

Motivation and challenges. LayerNorm [1] has been a key architectural component contributing to the success of large language models (LLMs) by stabilizing training through normalizing inputs across features within a layer. Additionally, it plays a crucial role in enhancing the models' non-linear representational capabilities [2–5]. Despite its benefits, LayerNorm introduces several practical challenges that become pronounced in specific settings:

1. *Private Inference (PI):* PI protocols enable inference on encrypted data without exposing inputs, ensuring data privacy while protecting model weights [6–14]. Hybrid PI protocols encounter difficulties with the inverse square root computation inherent in LayerNorm, making it the second most costly operation after GELU, contributing to 22% of total latency and communication overhead [11]. Also, Homomorphic Encryption (HE)-only PI requires polynomial approximations of LayerNorm, which are challenging due to the wide variance range [8].
2. *Mechanistic Interpretability:* LayerNorm increases the complexity of the residual stream, making it harder to analyze and understand the internal workings of transformer models [15], limiting the applicability of LLMs for applications requiring transparency and explainability.
3. *Low-Precision Training:* The trainable parameters in LayerNorm are associated with the amplification of outlier features which poses challenges in LLM quantization, as it exacerbates numerical instability and degrades performance in low-precision training regimes [16–19].
4. *Signal Propagation:* LayerNorms shown to negatively impact the faithful signal propagation [20].

2nd Workshop on Attributing Model Behavior at Scale (NeurIPS 2024).



(a) SM + LN + G (b) SM + LN + R (c) SM + G (d) SM + R

Figure 1: Entropy heatmaps of attention for baseline (a, b) and normalization-free (c, d) GPT-2 models with GELU and ReLU in the FFN. In the absence of LayerNorm, GELU in the FFN leads to significantly higher entropic overload (highlighted in yellow, c) compared to ReLU.

These challenges highlight the need for LayerNorm-free architectures that preserve transformer benefits while avoiding its drawbacks. However, this shift introduces new considerations, especially in selecting activation functions for the feed-forward networks (FFNs). Prior work [20–22] has explored various architectural heuristics for designing normalization-free LLMs. However, the impact of removing normalization layers on the choice of FFN activation functions remains underexplored.

Research insights and its implications. In this work, we go beyond prior approaches by conducting an in-depth investigation into the design choices for activation functions in normalization-free LLMs, offering new insights into how these choices impact learning dynamics, internal representations, and overall model performance. Our study reveals several key findings:

- *ReLU Outperforms GELU in LayerNorm-Free Models:* Contrary to conventional practices, we show that models using ReLU in the FFN significantly outperform, with an 8.2% improvement in perplexity, those using GELU in the absence of LayerNorm (see Figure 2 and Table 2).
- *Learning Dynamics with Learnable Negative Slopes:* To explore further, we experimented with a learnable negative slope in the leaky ReLU activation function using two configurations: (1) Layer-wise configuration: Each layer has its independent learnable slope. Initially, early layers learn a positive slope while deeper layers learn a negative slope. However, gradually all layers converge to a near-zero slope (Figure 3a). (2) Global Configuration: A single learnable slope is shared across all layers. The slope initially shifts to positive before converging to near zero (see Figure 3b). These results highlight LayerNorm-free models' inherent preference for ReLU-like activations with zero negative slopes.
- *Entropic Overload with GELU Activation:* To delve deeper, we analyze the head-wise entropy values and find that early layers in normalization-free models with GELU activation experience entropic overload, a significant proportion of attention heads reach near-maximum entropy levels, indicating the under-utilization of the representational capacity of attention heads.

Contributions. Our key contributions are follows:

1. We conduct an in-depth analysis of activation functions in normalization-free, decoder-only models by studying their learning dynamics when trained from scratch.
2. We explore the effect of different activation functions in baseline and normalization-free models on the attention score distribution through the lens of Shannon's entropy, offering valuable insights for advancing the architectural design of LayerNorm-free models.
3. We conducted experiments across various context sizes (128 and 256) on GPT-2 and Pythia [23] model with 2.1B training tokens from the CodeParrot [24].

2 Preliminaries

Notations. We denote the number of layers as L, number of heads as H, model dimensionality as d, head dimension as d_k (where $d_k = \frac{d}{H}$), and context length as T. Table 1 illustrates the abbreviations for architectural configurations with simplified nonlinearities in a transformer-based LLM.

2.1 Overview of Transformer-based Decoder-only Architectures

A transformer-based LLM is constructed by sequentially stacking L transformer blocks, where each block is composed of two sub-blocks: an attention mechanism and a feed-forward network (FFN), both having their own residual connections and normalization layers, positioned in the Pre-LN order

to improves training stability [25]. Formally, transformer blocks take an input sequence $\mathbf{X}_{\text{in}} \in \mathbb{R}^{T\times d}$, consisting of T tokens of dimension d, and transform it into $\mathbf{X}_{\text{out}}$ as follows:

$$\mathbf{X}_{\text{out}} = \hat{\mathbf{X}}_{\text{SA}} + \text{FFN}_{\text{GELU}}(\text{LayerNorm}_2(\hat{\mathbf{X}}_{\text{SA}})), \text{ where } \hat{\mathbf{X}}_{\text{SA}} = \mathbf{X}_{\text{in}} + \text{MHA}(\text{LayerNorm}_1(\mathbf{X}_{\text{in}})). \quad (1)$$

The Multi-Head Attention (MHA) sub-block enables input contextualization by sharing information between individual tokens. MHA employs the self-attention mechanism to compute the similarity score of each token with respect to all other tokens in the sequence. In particular, self-attention mechanism transform the input sequence $\mathbf{X}$ into $\mathbf{Attn}(\mathbf{X})$ as follows:

$$\mathbf{Attn}(\mathbf{X}) = \Big(\text{Softmax}\Big(\frac{1}{\sqrt{d_k}}(\mathbf{X}\mathbf{W}^Q)(\mathbf{X}\mathbf{W}^K)^\top + \mathbf{M}\Big)\Big)\mathbf{X}\mathbf{W}^V. \quad (2)$$

Here, each token generates query(Q), key(K), and value(V) vectors through the linear transformations $\mathbf{W}^Q, \mathbf{W}^K,$ and $\mathbf{W}^V \in \mathbb{R}^{d\times d_h}$, respectively. Then, similarity scores are computed by taking the dot product of the Q and K vectors, scaled by the inverse square root of the K dimension, and passed through a softmax function to obtain the attention weights. These weights are then used to compute a weighted sum of the V vectors, producing the output for each token. For auto-regressive models (e.g., GPT), mask $\mathbf{M} \in \mathbb{R}^{T\times T}$, which has values in $\{0, -\infty\}$ with $\mathbf{M}_{i,j} = 0$ iff $i \geq j$, is deployed to prevent the tokens from obtaining information from future tokens.

The MHA sub-block employs a self-attention mechanism across all the heads, each with its own sets of Q, K, and V. This allows the attention heads to focus on different parts of the input sequence, capturing various aspects of the input data simultaneously. The outputs from all heads are concatenated and linearly transformed ($\mathbf{W}^O \in \mathbb{R}^{d\times d}$) to produce the final MHA output as follows:

$$\text{MHA}(\mathbf{X}) = \text{Concat}\big(\text{Attn}_1(\mathbf{X}),\ \text{Attn}_2(\mathbf{X}),\ \text{Attn}_3(\mathbf{X}), \ldots, \text{Attn}_H(\mathbf{X})\big)\mathbf{W}^O. \quad (3)$$

Following the MHA sub-block, the FFN sub-block transforms each token independently. The FFN sub-blocks have a single hidden layer whose dimension is a multiple of d (e.g., $4d$ in GPT [26] models). Specifically, the FFN sub-block first applies a linear transformation to the input $\mathbf{X}$ using $\mathbf{W}^{\text{ffn}}_{\text{in}} \in \mathbb{R}^{d\times 4d}$, followed by a non-linear transformation using an activation function such as GELU. This is then followed by another linear transformation using $\mathbf{W}^{\text{ffn}}_{\text{out}} \in \mathbb{R}^{4d\times d}$, as follows:

$$\text{FFN}(\mathbf{X}) = (\text{GELU}(\mathbf{X}\mathbf{W}^{\text{ffn}}_{\text{in}}))\mathbf{W}^{\text{ffn}}_{\text{out}} \quad (4)$$

2.2 Entropy as a Metric for Attention Score Distribution

Shannon's entropy quantifies the uncertainty in a probability distribution, measuring the amount of information needed to describe the state of a stochastic system [27, 28]. For a probability distribution $P(x)$, the entropy is defined as $\mathbf{E}(P) = -\sum_i P(x_i)\log P(x_i)$. Refer to [29] for details on entropy.

In a softmax-based attention mechanism, each softmax operation yields an entropy value representing the sharpness or spread of the attention scores for each query position [30, 31]. Higher entropy indicates a more uniform distribution of softmax scores, while lower entropy signifies a more focused distribution on certain features [32].

Let $\mathbf{A}^{(h,l)} \in \mathbb{R}^{T\times T}$ be the attention matrix of h-th head in l-th layer, and each element in the attention matrix, $a^{(l,h)}_{ij}$, are attention weights, which are non-negative and sum to one for a query:

$$\mathbf{A}^{(l,h)} = \Big[a^{(l,h)}_{ij}\Big]_{T\times T}, \quad \text{where} \quad a^{(l,h)}_{ij} \geq 0 \quad \text{and} \quad \sum_{j=1}^{T} a^{(l,h)}_{ij} = 1 \quad (5)$$

This square matrix is generated by applying the softmax operation over the key length for each query position as follows (i.e., $\mathbf{X} \in \mathbb{R}^{T\times T}$ $\mathbf{X}_i \in \mathbb{R}^{1\times T}$):

$$\mathbf{A}^{(h,l)}(\mathbf{X}) = \text{Softmax}\Big(\frac{1}{\sqrt{d_k}}(\mathbf{X}\mathbf{W}^Q)(\mathbf{X}\mathbf{W}^K)^\top\Big), \text{ where } \quad \text{Softmax}(\mathbf{X}_i) = \frac{\exp(x_i)}{\sum_{j=1}^{T}\exp(x_j)} \quad (6)$$

Thus, each element $a^{(l,h)}_{ij}$ of the attention matrix can be represented as follows:

$$a^{(l,h)}_{ij} = \frac{\exp\Big(\frac{1}{\sqrt{d_k}}(\mathbf{X}_i\mathbf{W}^Q)(\mathbf{X}_j\mathbf{W}^K)^\top\Big)}{\sum_{k=1}^{T}\exp\Big(\frac{1}{\sqrt{d_k}}(\mathbf{X}_i\mathbf{W}^Q)(\mathbf{X}_k\mathbf{W}^K)^\top\Big)}. \quad (7)$$

Table 1: Architectural configurations of nonlinearities in LLMs, illustrating the combinations of Softmax (SM), LayerNorm (LN), GELU (G), and ReLU (R) functions (see Eq. 1, 2, 3 and 4).

Abbreviation	Architectural configuration
SM + LN + G	$\mathbf{X}_{\text{out}} = \text{FFN}_{\text{GELU}}(\text{LayerNorm}_2(\text{MHA}(\text{Attn}_{\text{Softmax}}(\text{LayerNorm}_1(\mathbf{X}_{\text{in}})))))$
SM + LN + R	$\mathbf{X}_{\text{out}} = \text{FFN}_{\text{ReLU}}(\text{LayerNorm}_2(\text{MHA}(\text{Attn}_{\text{Softmax}}(\text{LayerNorm}_1(\mathbf{X}_{\text{in}})))))$
SM + G	$\mathbf{X}_{\text{out}} = \text{FFN}_{\text{GELU}}(\text{MHA}(\text{Attn}_{\text{Softmax}}(\mathbf{X}_{\text{in}})))$
SM + R	$\mathbf{X}_{\text{out}} = \text{FFN}_{\text{ReLU}}(\text{MHA}(\text{Attn}_{\text{Softmax}}(\mathbf{X}_{\text{in}})))$

Following [33], we compute the mean of entropy values across all query positions to obtain a single entropy value for each head. The entropy $\mathbf{E}^{(l,h)}$ for the h-th head in the l-th layer of an attention matrix is given by:

$$\mathbf{E}^{(l,h)} = -\frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{T} a_{ij}^{(l,h)} \log\left(a_{ij}^{(l,h)} + \epsilon\right) \tag{8}$$

where ϵ is a small constant added for numerical stability to prevent taking the log of zero.

The combination of MHA and FFN sub-blocks, along with residual connections and normalization layers, allows transformer models to learn the contextual relationships between tokens effectively.

2.3 Dataset and Training Methodology

We train our models from scratch using the CodeParrot dataset [24], a standard benchmark for LLMs [22, 16]. The dataset, derived from 20 million Python files on GitHub, consists of 8 GB of data with 16.7 million examples, each containing 128 tokens, amounting to a total of 2.1 billion training tokens. For tokenization, we utilize a tokenizer with a vocabulary size of 50K.

For training on the CodeParrot dataset, we adopt the settings from [22], ensuring consistency across all architectural variations to isolate the effects of the changes. In line with prior works [22, 34, 35], all models are trained using a single RTX 3090 GPU.

3 Activation Functions and Their Impact Through Shannon's Entropy

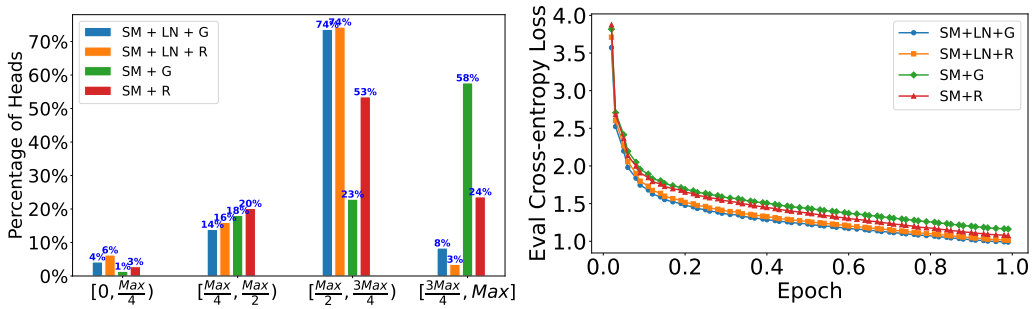
In this section, we investigate the role of activation functions in baseline and normalization-free decoder-only LLMs. Specifically, we examine the learning dynamics and internal representations of activation functions, using entropy as a metric to highlight key observations and insights.

Well-behaved entropy distribution We begin by analyzing the headwise entropy distribution of baseline architecture with GELU and ReLU in the FFN, i.e., configurations $\texttt{SM} + \texttt{LN} + \texttt{G}$ and $\texttt{SM} + \texttt{LN} + \texttt{R}$ respectively. We find that the majority of heads ($\approx$90%) possess entropy values between $\frac{\texttt{max}}{4}$ and $\frac{\texttt{3max}}{4}$, where $\texttt{max}$ is maximum observed entropy value among all heads (Figure 2a). This concentration in the mid-entropy range, avoiding extremes, demonstrates a well-behaved distribution, providing as a benchmark for assessing the impact of architectural modifications, such as activation function simplification, on model behavior.

Entropic overload We observed that in certain nonlinearity configurations, a disproportionately large fraction of the attention heads exhibit higher entropy values (between $\frac{\texttt{3max}}{4}$ and $\texttt{max}$). We term this phenomenon as entropic overload and hypothesize that this imbalance results in *under-utilization* of the network's representational capacity, as too many heads engaged in exploration, hindering the model from effectively leveraging the diversity of attention heads.

To investigate further, we examined how entropy values evolve during training. Typically, all heads start with higher entropy values, indicating an initial exploration phase, and gradually adapt to balance exploration and exploitation in baseline networks (see Figure 4). However, in the absence of certain nonlinearities, this balance is disrupted, preventing attention heads from specializing and refining their focus on critical aspects of the input, thereby diminishing overall performance.

Observation 1: ReLU significantly outperforms GELU in LayerNorm-Free LLMs. While GELU is typically preferred over ReLU in conventional transformer-based models due to its smooth and differentiable properties that improve performance and optimization, our empirical findings indicate



(a) Headwise entropy distribution

(b) Evaluation loss curve

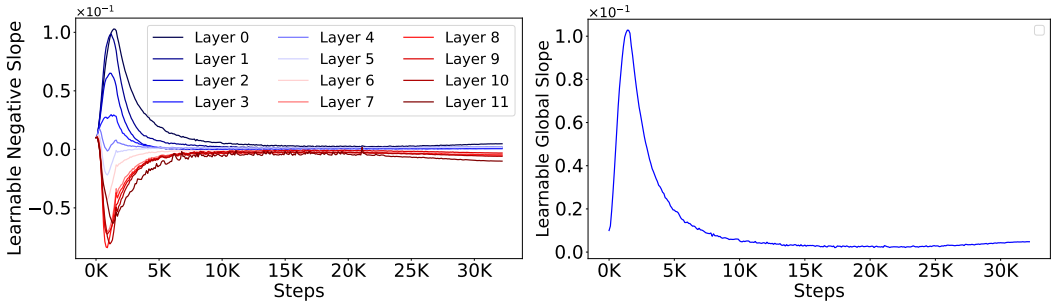
Figure 2: Headwise entropy distribution and evaluation loss for baseline and normalization-free GPT-2 models, using GELU and ReLU activations, trained from scratch on CodeParrot dataset.

the *opposite trend* for LayerNorm-free models— using ReLU in the FFN exhibit better learning dynamics than their GELU counterpart. This leads to an **8.2%** improvement in perplexity for GPT-2 (see Figure 2 and Table 2). A similar trend has been observed on the normalization-free Pythia-70M model across various context lengths.

Table 2: Perplexity comparison between baseline and normalization-free GPT-2 (L=12, H=12, d=768) and Pythia-70M (L=6, H=8, d=512) models, using GELU and ReLU activations in the FFN, trained from scratch on CodeParrot dataset. While GELU outperforms ReLU in baseline models, the normalization-free models exhibit the *opposite trend*.

	GPT-2 (T=128)		Pythia-70M (T=128)		Pythia-70M (T=256)	
	Eval PPL	+Δ(%)	Eval PPL	+Δ(%)	Eval PPL	+Δ(%)
SM+LN+G	2.688	0.00	3.512	0.00	3.054	0.00
SM+LN+R	2.757	2.53	3.590	2.22	3.107	1.73
SM+G	3.197	18.92	4.086	16.35	3.570	16.87
SM+R	2.936	9.20	3.736	6.36	3.273	7.17

To further strengthen our findings, we conducted experiments with a learnable negative slope in the leaky ReLU activation function with two configurations: 1) layer-wise, where each layer has its independent learnable slope, and 2) global, where a single learnable slope is shared across all layers. Interestingly, in the layerwise setting, the early layers initially learn a positive slope while the deeper layers learn a negative slope. However, as training progresses, all layers converge to a near-zero slope. In the global setting, the slope first shifts to positive before converging to near zero (see Figure 3).



(a) Layerwise learnable slope

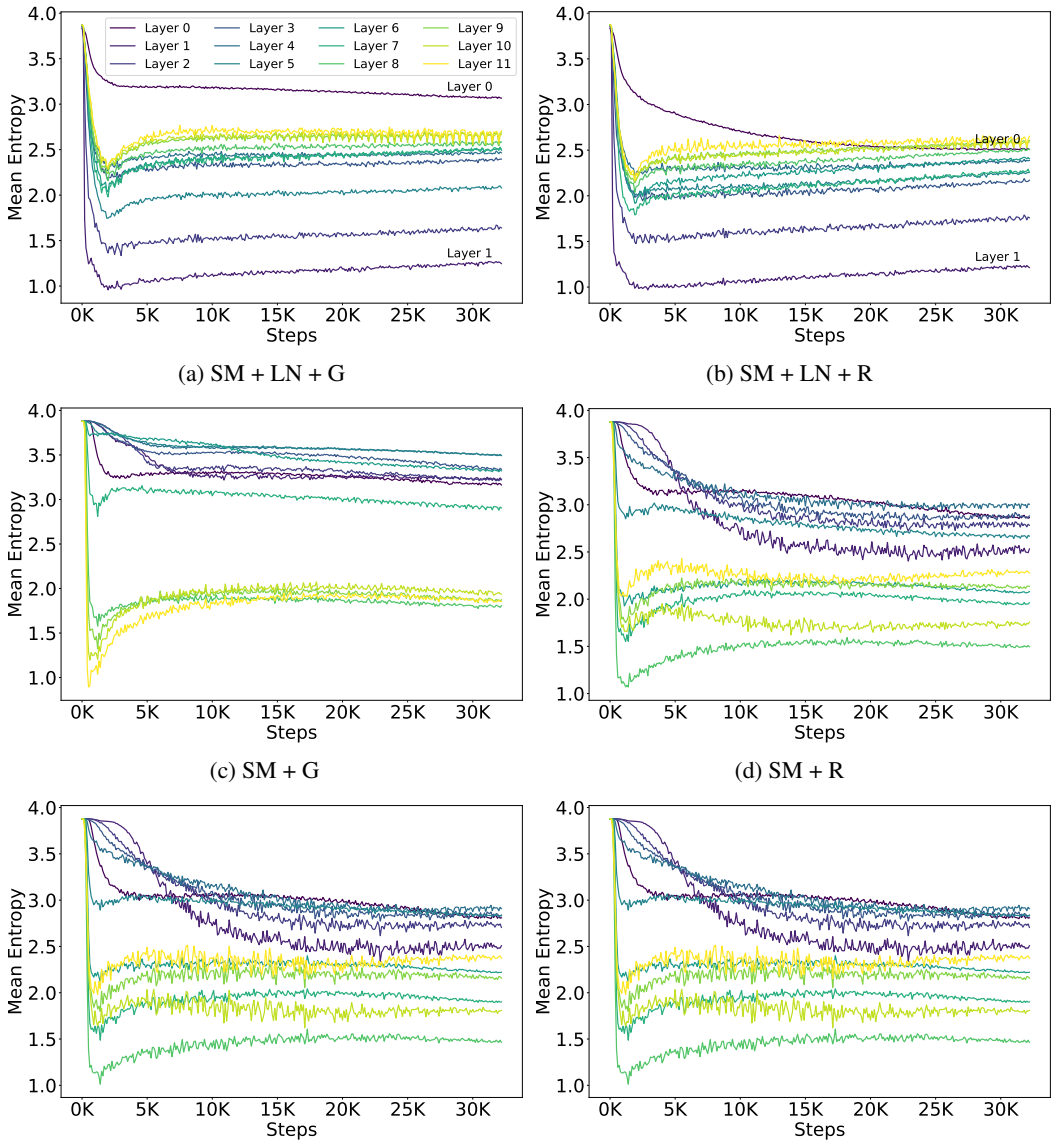
(b) Global learnable slope

Figure 3: Learnable negative slope for leaky ReLU in FFN of LN-free GPT-2 model. (a) Layerwise slopes showing initial variability and convergence towards zero. (b) Global slope trend towards zero over training steps, indicating a preference for zero negative slope in LN-free architectures.

When comparing the layerwise entropy dynamics in both cases (Figure 4e and Figure 4f) with the normalization-free model using ReLU activations (Figure 4d), we observed near-identical patterns.

This highlights the a *natural preference* for zero negative slope, similar to ReLU, in the FFN activation function of the normalization-free model.

Observation 2: Early layers in the LayerNorm-Free model with GELU in FFN experience entropic overload. To understand the zero negative slope preference for the FFN activation function in LN-free architecture, we analyzed the headwise entropy values of LN-free models with GELU and ReLU, when trained from scratch, and compared them to their baseline counterparts. Our analysis revealed a significant divergence in the headwise entropy distributions of the LN-free GELU model (see Figure 1). While baseline models with GELU and ReLU exhibit a balanced entropy distribution, by avoiding the extreme values, the LN-free GELU model shows entropic overload in early layers.



(a) SM + LN + G

(b) SM + LN + R

(c) SM + G

(d) SM + R

(e) SM + LearnableNegativeSlope (Layerwise)

(f) SM + LearnableNegativeSlope (Global)

Figure 4: Evolution of Layerwise entropy when GPT-2 (L=12, H=12, d=768) models with various nonlinearity configurations are trained from scratch on CodeParrot dataset. Evolution of layer-wise entropy during training of GPT-2 models (L=12, H=12, d=768) with different nonlinearity configurations on the CodeParrot dataset. The near-identical entropy dynamics in Figures d, e, and f underscore a *natural preference* for a zero negative slope, similar to ReLU, in the FFN activation function of the normalization-free model.

Quantitatively, 58% of heads in the LN-free GELU model have entropy values between $\frac{3\texttt{max}}{4}$ and $\texttt{max}$, compared to only 23% in the LN-free ReLU model (Figure 2a). More importantly, very few heads in the latter approach maximum entropy compared to the former (see yellow regions in Figure 1c), indicating more severe entropic overload in the LN-free model with GELU.

These observations align with geometrical properties of ReLUs: it preserve more information about the structure of the raw input, encouraging neurons to specialize in different regions of the input space, leading to a higher intra-class *selectivity* and *specialization* [36]. Thus, the lack of LayerNorm makes the geometry and specialization effects of ReLU more beneficial, while GELU's smoother nonlinearity causes issues in maintaining distinct attention head behaviors.

Training instability and entropy dynamics with a fixed negative slope In normalization-free LLMs, ReLU-like activation functions, with a near-zero negative slope, naturally stand out as a preferred choice compared to the conventional GELU, offering both improved predictive performance and stable training dynamics. This makes exploring fixed negative slopes in leaky ReLU activations particularly intriguing.

To systematically investigate this, we conducted a series of experiments on normalization-free GPT-2 models, adjusting the negative slopes to fixed values of $1e$-2, $5e$-2, $1e$-1, and $2e$-1. We assessed training instability by monitoring the frequency and distribution of NaN values across model layers and evaluated entropy dynamics across the model's depth. The results are shown in Figure 5.

For a negative slope of $1e$-2, we observed sporadic occurrences of NaNs primarily in the last layer (Layer 11), as shown in Figure 5a, with no entropy collapse (Figure 5b). However, as the negative slope increased to $5e$-2, $1e$-1, and $2e$-1, NaNs began to appear consistently in deeper layers, as evidenced by the NaN counts in Figures 5c, 5e, and 5g, respectively. These consistent NaNs correlated with entropy collapses in the deeper layers, indicating a strong relationship between increased negative slope and training instability (Figures 5d, 5f, and 5h).

An interesting trend emerged: the larger the negative slope, the earlier the training instability and entropy collapse occurred. For example, with a slope of $2e$-1, instability occurred almost immediately (Figure 5g), and entropy collapses in deeper layers (Figure 5h) occurred much sooner compared to lower negative slopes. This suggests that as the slope increases, the window of stable training narrows, making it crucial to choose the appropriate negative slope to avoid early instability and entropy collapse in normalization-free models.

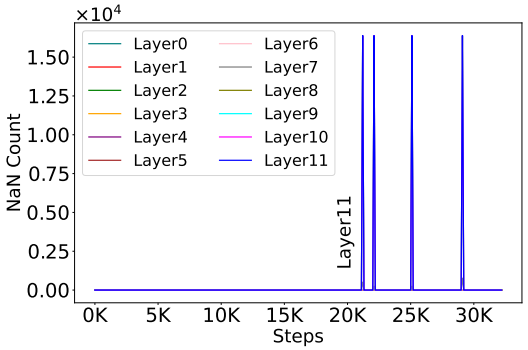
Broader implications of activation function characteristics in normalization-free models The emergence of training instability and entropy collapse at larger negative slopes underscores the sensitivity of normalization-free LLMs to the choice of activation function parameters. The strong correlation between larger negative slopes and earlier training instability suggests that even seemingly minor changes to the negative slope of the leaky ReLU function can significantly influence the model's ability to maintain stability during training. Specifically, larger negative slopes in leaky ReLU activations *aggravate the proliferation of NaNs and exacerbate entropy collapse in deeper layers*, leading to earlier and more pronounced training instability.

This suggests that a near-zero negative slope strikes a crucial balance in normalization-free LLMs, offering sufficient nonlinearity while maintaining training stability.

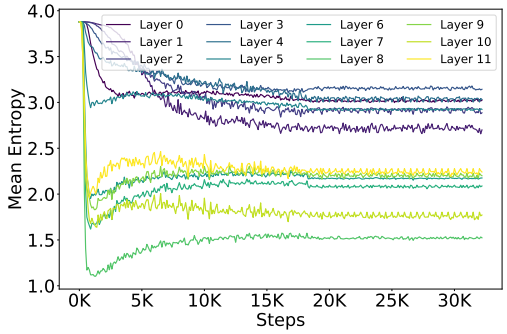
4 Conclusion

In this paper, we investigated the design of normalization-free decoder-only language models and highlighted the critical role of activation functions in such architectures. Our empirical studies revealed that, contrary to conventional practices, the ReLU activation significantly outperforms, an 8.2% improvement in perplexity, the GELU in normalization-free models. We found that models with learnable negative slopes in leaky ReLU activations naturally converge toward zero negative slopes, effectively resembling ReLU.

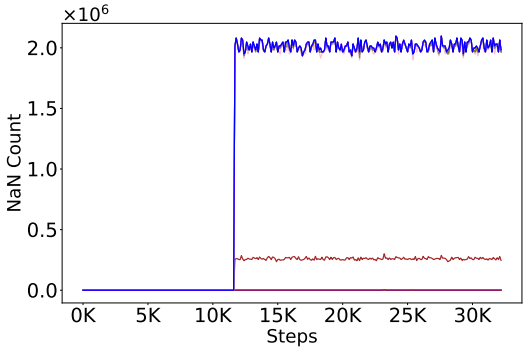
Additionally, we discovered that LayerNorm-free models with GELU activation suffer from entropic overload in early layers, leading to under-utilization of their representational capacity. These findings underscore the necessity of rethinking activation function choices when LayerNorm is absent and suggest that selecting appropriate activations like ReLU enables the development of transformer



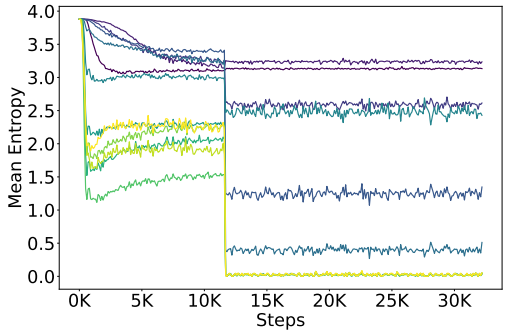
(a) NaNs observed with a negative slope of $1e$-2



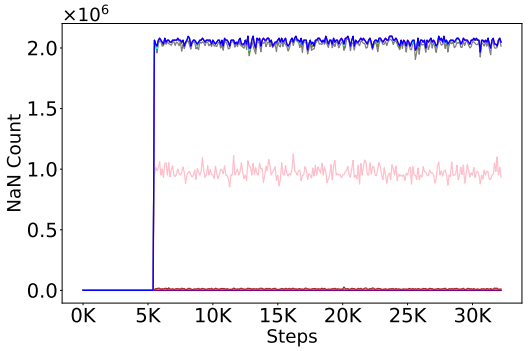
(b) Entropy dynamics with a negative slope of $1e$-2



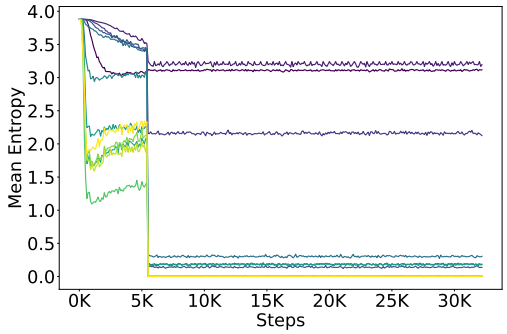
(c) NaNs observed with a negative slope of $5e$-2



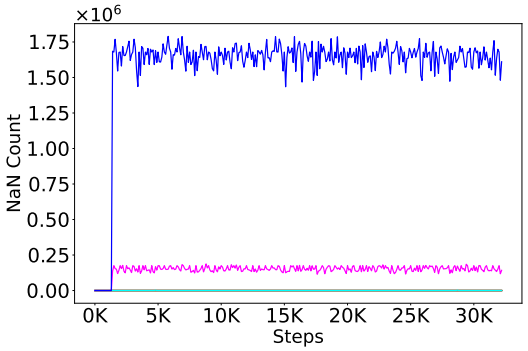
(d) Entropy dynamics with a negative slope of $5e$-2



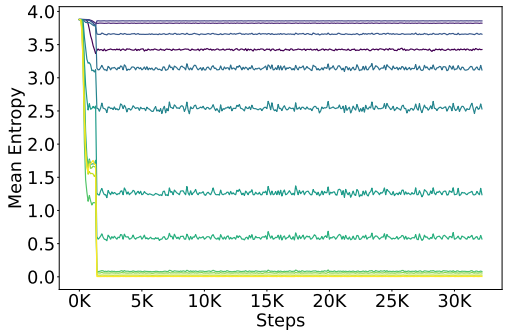
(e) NaNs observed with a negative slope of $1e$-1



(f) Entropy dynamics with a negative slope of $1e$-1



(g) NaNs observed with a negative slope of $2e$-1



(h) Entropy dynamics with a negative slope of $2e$-1

Figure 5: Training instability, indicated by NaNs, and corresponding entropy dynamics in Normalization-Free GPT-2 (L=12, H=12, d=768) models with fixed negative slopes in the leaky ReLU. The larger the negative slope, the earlier the training instability and entropy collapse occurred.

models that are more efficient, interpretable, and better suited for applications such as private inference and quantization.

Limitations. This study mainly focuses on pre-training performance, with perplexity as the primary metric, and does not include experiments to evaluate other capabilities such as transfer learning or few-shot learning. Additionally, our findings are been validated on models with fewer than 1B parameters. Future work will explore broader experimental evaluations, including the large-scale models (see Appendix D).

Notes. This workshop submission delves into one of the key findings—the LayerNorm-free design– from our comprehensive paper AERO: Softmax-Only LLMs for Efficient Private Inference. The code and implementation are available at relu-revival-normfree.

References

[1] Jimmy Lei Ba. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Xinyi Wu, Amir Ajorlou, Yifei Wang, Stefanie Jegelka, and Ali Jadbabaie. On the role of attention masks and layernorm in transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[3] Yunhao Ni, Yuxin Guo, Junlong Jia, and Lei Huang. On the nonlinearity of layer normalization. In *Forty-first International Conference on Machine Learning (ICML)*, 2024.

[4] Bingchen Zhao, Haoqin Tu, Chen Wei, Jieru Mei, and Cihang Xie. Tuning LayerNorm in attention: Towards efficient multi-modal llm finetuning. In *International Conference on Learning Representations (ICLR)*, 2024.

[5] Amir Joudaki, Hadi Daneshmand, and Francis Bach. On the impact of activation and normalization in obtaining isometric embeddings at initialization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[6] Jiawen Zhang, Jian Liu, Xinpeng Yang, Yinghao Wang, Kejia Chen, Xiaoyang Hou, Kui Ren, and Xiaohu Yang. Secure transformer inference made non-interactive. In *Annual Network and Distributed System Security Symposium (NDSS)*, 2025.

[7] Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Kui Ren, Cheng Hong, Tao Wei, and WenGuang Chen. Bumblebee: Secure two-party inference framework for large transformers. In *Annual Network and Distributed System Security Symposium (NDSS)*, 2025.

[8] Itamar Zimerman, Moran Baruch, Nir Drucker, Gilad Ezov, Omri Soceanu, and Lior Wolf. Converting transformers to polynomial form for secure inference over homomorphic encryption. In *International Conference on Machine Learning (ICML)*, 2024.

[9] Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. Bolt: Privacy-preserving, accurate and efficient inference for transformers. In *IEEE Symposium on Security and Privacy (SP)*, 2024.

[10] Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. SIGMA: secure GPT inference with function secret sharing. In *Proceedings on Privacy Enhancing Technologies (PETs)*, 2024.

[11] Xiaoyang Hou, Jian Liu, Jingyu Li, Yuhan Li, Wen-jie Lu, Cheng Hong, and Kui Ren. Ciphergpt: Secure two-party gpt inference. *Cryptology ePrint Archive*, 2023.

[12] Nandan Kumar Jha and Brandon Reagen. DeepReShape: Redesigning neural networks for efficient private inference. In *Transactions on Machine Learning Research (TMLR)*, 2024.

[13] Zahra Ghodsi, Nandan Kumar Jha, Brandon Reagen, and Siddharth Garg. Circa: Stochastic relus for private deep learning. In *Advances in Neural Information Processing Systems*, 2021.

[14] Nandan Kumar Jha, Zahra Ghodsi, Siddharth Garg, and Brandon Reagen. DeepReDuce: Relu reduction for fast private inference. In *International Conference on Machine Learning (ICML)*, 2021.

[15] Neel Nanda. Attribution patching: Activation patching at industrial scale. *URL: https://www. neelnanda. io/mechanistic-interpretability/attribution-patching*, 2023.

[16] Bobby He, Lorenzo Noci, Daniele Paliotta, Imanol Schlag, and Thomas Hofmann. Understanding and minimising outlier features in neural network training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[17] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Quantizable transformers: Removing outliers by helping attention heads do nothing. In *Advances in Neural Information Processing Systems*, 2023.

[18] Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. In *Advances in Neural Information Processing Systems*, 2022.

[19] Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. Bert busters: Outlier dimensions that disrupt transformers. In *Findings of the Association for Computational Linguistics (ACL-IJCNLP)*, 2021.

[20] Bobby He, James Martens, Guodong Zhang, Aleksandar Botev, Andrew Brock, Samuel L Smith, and Yee Whye Teh. Deep transformers without shortcuts: Modifying self-attention for faithful signal propagation. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[21] Lorenzo Noci, Chuning Li, Mufan Li, Bobby He, Thomas Hofmann, Chris J Maddison, and Dan Roy. The shaped transformer: Attention models in the infinite depth-and-width limit. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[22] Bobby He and Thomas Hofmann. Simplifying transformer blocks. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[23] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning (ICML)*, 2023.

[24] Hugging Face. Codeparrot. `https://huggingface.co/learn/nlp-course/chapter7/6`.

[25] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning (ICML)*, 2020.

[26] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[27] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 1948.

[28] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 1957.

[29] John C. Baez. What is entropy? *arXiv preprint arXiv:2409.09232*, 2024. `https://arxiv.org/abs/2409.09232`.

[30] Hamidreza Ghader and Christof Monz. What does attention in neural machine translation pay attention to? In *Proceedings of the The 8th International Joint Conference on Natural Language Processing*, 2017.

[31] Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019.

[32] Yury Nahshan, Joseph Kampeas, and Emir Haleva. Linear log-normal attention with unbiased concentration. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[33] Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning (ICML)*, 2023.

[34] Aleksandar Stanić, Dylan Ashley, Oleg Serikov, Louis Kirsch, Francesco Faccio, Jürgen Schmidhuber, Thomas Hofmann, and Imanol Schlag. The languini kitchen: Enabling language modelling research at different scales of compute. *arXiv preprint arXiv:2309.11197*, 2023.

[35] Jonas Geiping and Tom Goldstein. Cramming: Training a language model on a single gpu in one day. In *International Conference on Machine Learning (ICML)*, 2023.

[36] Matteo Alleman, Jack Lindsey, and Stefano Fusi. Task structure and nonlinearity jointly determine learned representational geometry. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[37] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, 2013.

[38] David Peer, Bart Keulen, Sebastian Stabinger, Justus Piater, and Antonio Rodriguez-sanchez. Improving the trainability of deep neural networks through layerwise batch-entropy regularization. In *Transactions on Machine Learning Research (TMLR)*, 2022.

[39] Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Re. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[40] Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 1977.

[41] DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-recurrent transformers. In *Advances in neural information processing systems (NeurIPS)*, 2022.

Appendix

Table of Contents

A Why Training from Scratch to Study Nonlinearities?

Understanding the intricate roles of architectural components and nonlinearities—such as activation functions (e.g., GELU, ReLU) in FFN, normalization layers (e.g., LayerNorm), etc.—in transformer-based language models necessitates a methodical and detailed investigative approach. Training models from scratch is essential for this purpose, as it allows us to delve into the internal mechanisms of the model using quantitative measures like entropy. Below, we present a justification for our methodology:

- *Nonlinearities' impact on the fundamental learning dynamics:* Nonlinearities significantly influence the optimization landscape by affecting gradient flow and the model's ability to navigate non-convex loss surfaces. Training models from scratch allow us to observe the fundamental learning dynamics that emerge during the initial stages of training. Thus, constructing models with controlled variations, such as substituting or excluding specific nonlinearities, enables us to isolate their direct effects impact on convergence behavior and training stability.
- *Understanding internal mechanisms through entropy analysis:* Training from scratch enables us to navigate the evolution of entropy values across the layers and assess how architectural components influence information flow within the model. This analysis provides deep insights into the internal workings of models that may not be accessible when starting from pre-trained checkpoints.
- *Limitations of fine-tuning approaches:* The aforementioned granular level of analysis is unattainable when starting from pre-trained models, where the optimization trajectory has already been largely determined. In contrast, training models from scratch eliminates confounding variables that could arise from pre-existing weights and learned representations, ensuring that any observed effects are solely due to the architectural modifications introduced.

B Why Use Entropy to Evaluate the Impact of Nonlinearities?

We use entropy as a metric to study the impact of nonlinearities on the transformer-based LLMs for the following reasons:

- *Quantifying attention distribution:* As the attention mechanism is fundamental to all transformer-based architecture, computing the entropy of attention score distributions reveals how nonlinearities affect attention concentration. High entropy quantifies exploration and low entropy indicates exploitation.
- *Feature selection:* Nonlinearities like ReLU enable feature selectivity by amplifying important features and suppressing less relevant ones [37]. Entropy can measure this selectivity across layers and heads, providing insights into the model's prioritization of information. Previously, entropy has been used to quantify the layerwise information flow in neural networks [38].
- *Exploration vs. exploitation*: Nonlinear operators like the self-attention mechanism, LayerNorm, and GELU balance exploration and exploitation by selecting relevant features while considering a broader context. For instance, heads in the first layer focus on exploration, while those in the second layer focus on exploitation. (see Figures 1a, 1b, 4a and 4b).
- *Systematic assessment:* Prior work [39, 32, 33, 31, 30] also used entropy to analyze the behavior of transformer-based models; thus, enhancing validity and comparability of our findings.

C Perplexity as a Reliable Metric to Evaluate the LLMs' Performance

Perplexity [40] is a widely adopted metric to evaluate the predictive performance of auto-regressive language models, reflecting the model's ability to predict the next token in a sequence. However, for perplexity to serve as a meaningful comparative metric across different architectures, it is critical to ensure consistency in the tokenizer, and vocabulary size and quality [41]. Any variation in these components can potentially skew the results by inflating or deflating perplexity scores; thus, obfuscating the true effects of architectural changes.

In our work, we maintain tokenization schemes and vocabulary attributes as invariant factors across all experiments within a dataset. This isolation of architectural modifications ensures that any observed variations in perplexity are directly attributable to changes in the model design. Thus, by enforcing a consistent tokenization scheme and vocabulary within a dataset, we ensure that perplexity remains a

reliable metric for comparing model architectures. Consequently, lower perplexity in our evaluations reliably reflects improved token-level predictions.

D Future Work

Scaling up and generalizing to larger models This research opens several avenues for optimizing LayerNorm-free transformer architectures. A primary direction is scaling up experiments to larger models. Evaluating whether the benefits of ReLU activation persist in models with significantly more parameters will determine the applicability of our findings to state-of-the-art language models. Additionally, extending the analysis to other architectures, such as encoder-only or encoder-decoder transformers, could help generalize our insights across different model types.

Downstream task performance While our study focused on perplexity and entropy metrics, future work should analyze how the choice of activation function affects performance on various downstream tasks. Investigating the implications for fine-tuning processes could also provide valuable insights for practical applications.

Hybrid activation strategies Exploring hybrid activation strategies presents another promising research direction. By using different activation functions in different parts of the model—such as combining GELU in the earlier layers with ReLU in the later layers—we could strike a balance between the benefits observed in our study and the traditional advantages of GELU. This approach may enhance model performance while maintaining computational efficiency.

Interpretability and practical applications Given the observed differences in entropy distribution between ReLU and GELU models, future research could explore how different activation functions impact the interpretability of LayerNorm-free models. This could potentially lead to more explainable model design, addressing a critical need in the field. Integrating these findings into practical applications like private inference and quantization is also promising, as it could improve both model efficiency and security.

Knowledge distillation for performance enhancement By distilling knowledge from a larger, LayerNorm-equipped teacher model to a smaller, LayerNorm-free student model with appropriate activation functions, the performance and generalization capabilities of the student model can be improved. This approach could mitigate any performance gaps arising from the absence of LayerNorm while maintaining the benefits of simplified computation and improved interpretability.