

# UnitNorm: Rethinking Normalization for Transformers in Time Series

**Nan Huang**  
Department of Computer Science  
University of North Carolina at Charlotte  
Charlotte, NC 28223  
nhuang1@charlotte.edu

**Christian Kümmerle**  
Department of Computer Science  
University of North Carolina at Charlotte  
Charlotte, NC 28223  
kummerle@charlotte.edu

**Xiang Zhang**  
Department of Computer Science  
University of North Carolina at Charlotte  
Charlotte, NC 28223  
xiang.zhang@charlotte.edu

## Abstract

Normalization techniques are crucial for enhancing Transformer models' performance and stability in time series analysis tasks, yet traditional methods like **batch** and **layer normalization** often lead to **issues** such as **token shift**, **attention shift**, and **sparse attention**. We propose UnitNorm, a novel approach that scales input vectors by their norms and modulates attention patterns, effectively circumventing these challenges. Grounded in existing normalization frameworks, UnitNorm's effectiveness is demonstrated across diverse **time series** analysis **tasks**, including forecasting, classification, and anomaly detection, via a rigorous evaluation on 6 state-of-the-art models and 10 datasets. Notably, UnitNorm shows superior performance, especially in scenarios requiring robust attention mechanisms and contextual comprehension, evidenced by significant improvements by up to a **1.46 decrease in MSE** for forecasting, and a **4.89% increase in accuracy** for classification. This work not only calls for a reevaluation of normalization strategies in time series Transformers but also sets a new direction for enhancing model performance and stability. The source code is available at <https://anonymous.4open.science/r/UnitNorm-5B84>.

## 1 Introduction

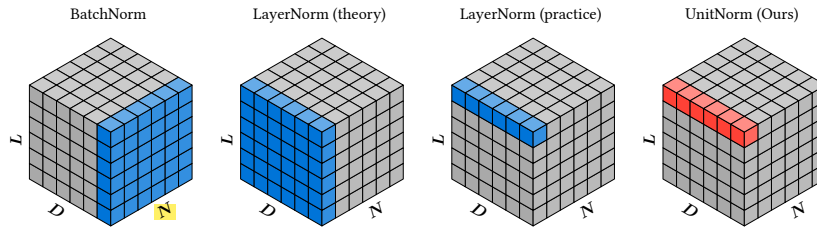


Figure 1: Scheme of different normalization methods. The input to the normalization layers is batched sequences of token vectors  $\mathbf{X} \in \mathbb{R}^{N \times L \times D}$ , where  $N$  is the **batch size**,  $L$  is the **sequence length** and  $D$  is the **dimension** of each **token vector**. The **blue** sections demonstrate a single slice of the input tensor for computing the **mean**  $\mu$  and **variance**  $\sigma^2$ , while the **red** section shows a single slice of data for computing the **vector norm**  $\|\mathbf{x}\|$  (see Appendix C.1).

Transformers have revolutionized sequence modeling, demonstrating unparalleled efficacy across diverse fields such as natural language processing (NLP, Wolf et al. 1), computer vision (CV, Han et al. 2), and recently, time series analysis (TSA, Wen et al. 3). Central to these models is the representation of data as sequences of token vectors, denoted by  $\mathbf{X} \in \mathbb{R}^{N \times L \times D}$ , where  $N$  stands for batch size,  $L$  is the sequence length and  $D$  represents the dimensionality of each token. The core mechanism facilitating the Transformers’ ability to model complex dependencies is the attention mechanism. It computes a weighted sum of value vectors  $\mathbf{V}$ , capturing the sequential relationships between tokens through a scalable dot-product operation of queries  $\mathbf{Q}$  and keys  $\mathbf{K}$  [4]:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D}} \right) \mathbf{V}, \quad (1)$$

To mitigate issues during the training process of Transformers related to **vanishing or exploding gradients** [5, 6], **Layer Normalization** (LayerNorm, LN, Ba et al. 7) plays a significant role and is therefore incorporated at each sub-layer of the architecture (Figure S1)<sup>1</sup>. The **LayerNorm operation follows the center-and-scale standardization paradigm**, by first centering the means to 0 and then rescaling the variances of the input vectors to 1 [7] such that

$$\text{LN}(\mathbf{X}) = \frac{\mathbf{X} - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \varepsilon}}, \quad (2)$$

where  $\boldsymbol{\mu}$  and  $\boldsymbol{\sigma}$  are the mean and standard deviation of the input vector  $\mathbf{X}$ , respectively.

While LayerNorm, compared to other normalization strategies such as batch normalization [8–10], has established itself as the dominant normalization strategy in Transformers, dedicated normalization-specific research has mostly focussed on its impact on model convergence [11], its inner dynamics [10, 9] or its location [12] within the architecture. On the other hand, only few works touch upon the interaction of normalization with the attention mechanism [13] (see also Related Work Section 5.1), which poses specific challenges in TSA (see Section 2) due to the dot product in attention mechanism.

In this work, we provide a new viewpoint on these challenges by first identifying and formalizing Transformer-specific challenges of normalization techniques, highlighting three key issues. Building on these insights, we introduce a novel normalization technique, UnitNorm, designed to address these challenges effectively.

Our contributions lie in: 1) We originally identify two challenges, namely *token shift* and *attention shift*, and reassess the challenge of *sparse attention* in Transformers [14]; 2) We propose a new normalization method, UnitNorm, that can mitigate these issues by design; 3) We empirically validate the effectiveness of UnitNorm on nine datasets spanning three downstream TSA tasks.

## 2 Challenges in Normalization

Transformers rely on attention mechanisms to achieve remarkable performance in time series analysis tasks. However, the interplay between the attention mechanism and the applied normalization methods introduces critical challenges that have yet to be fully addressed. This paper aims to shed light on the complexities of token shift, attention shift, and sparse attention, which arise from the interaction between normalization and the attention mechanism within Transformer models. By presenting a thorough theoretical and empirical analysis, we demonstrate that these challenges are intrinsic to the conventional approaches to normalization, impacting the efficacy of the self-attention mechanism that is central to all Transformer-based architectures.

We explore the **relationship between normalization and the attention mechanism** by examining a simplified equivalent attention process, where the normalization layer precedes the attention computation (Zhang et al. 15, Figure S1). This perspective allows for a detailed exploration of how normalization influences the attention scores derived from the query and key vectors. For simplicity, our discussion will center on a singular instance of self-attention within the encoder layer, assuming identical query and key vectors to streamline our analysis (see Appendix C.2).

Table 1: Effect of input transformations on the softmax function output. **Importance order invariant** refers to whether the relative importance of the tokens is preserved. Of all possible input transformations, only the **reflection** transformation will definitely **change the importance order of the tokens**.

Type	Function	Input	Output	Order invariant?						
None	$f : x \mapsto x$	<table><tr><td>-2</td><td>1</td><td>3</td></tr></table>	-2	1	3	<table><tr><td>0.01</td><td>0.12</td><td>0.88</td></tr></table>	0.01	0.12	0.88	
-2	1	3								
0.01	0.12	0.88								
Stretch	$f : x \mapsto k \cdot x, k \in \mathbb{R}^+$	<table><tr><td>-4</td><td>2</td><td>6</td></tr></table>	-4	2	6	<table><tr><td>0</td><td>0.02</td><td>0.98</td></tr></table>	0	0.02	0.98	✓
-4	2	6								
0	0.02	0.98								
Translate	$f : x \mapsto x + a, a \in \mathbb{R}$	<table><tr><td>-1</td><td>2</td><td>4</td></tr></table>	-1	2	4	<table><tr><td>0.01</td><td>0.12</td><td>0.88</td></tr></table>	0.01	0.12	0.88	✓
-1	2	4								
0.01	0.12	0.88								
Jitter	$f : x \mapsto x + \varepsilon, \varepsilon \sim \mathcal{N}(0, \sigma^2)$	<table><tr><td>-2.1</td><td>1.1</td><td>3</td></tr></table>	-2.1	1.1	3	<table><tr><td>0.01</td><td>0.13</td><td>0.87</td></tr></table>	0.01	0.13	0.87	✓/✗
-2.1	1.1	3								
0.01	0.13	0.87								
Reflection	$f : x \mapsto -x$	<table><tr><td>2</td><td>-1</td><td>-3</td></tr></table>	2	-1	-3	<table><tr><td>0.95</td><td>0.05</td><td>0.01</td></tr></table>	0.95	0.05	0.01	✗
2	-1	-3								
0.95	0.05	0.01								

Table 2: Effect of normalization on the **attention weight distribution** based on empirical results (Figures S6 and S7). UnitNorm shows the most faithful representation of the original attention weights that are cross-validated by various metrics as described in Table S9, while center-and-scale normalization significantly alters the attention weights to an extreme extent as depicted in Figure S5.

Normalization	Chebyshev distance ↓	Cosine similarity ↑	KL divergence ↓	Entropy ↑
None (original)	/	/	/	High
Center-and-scale	High	Low	High	Very Low
UnitNorm	Low	High	Low	High

## 2.1 Token shift

Previous study [16] has attributed LayerNorm’s efficacy to its **center-and-scale** operations: **centering** projects the **input vectors to a hyperplane orthogonal to  $\mathbf{1}$  vector**, and **scaling** normalizes the vectors to **a unit sphere** to prevent any **token vector** being contained in the **convex hull** of the others. However, this can significantly **alter the orientation of input vectors**, especially for those that are near parallel to the hyperplane’s norm vector  $\mathbf{1}$ . This alteration impacts the dot product between vectors, potentially leading to sign flips (Figure 2). Such flips can severely disrupt the softmax function’s output (Table 1), altering the relative importance of tokens **in a catastrophic way that might convert a significant token into an insignificant one, or vice versa** (Table 2). This issue of significant deviations in attention weight distributions caused by token shift will be further explored in Section 2.2.

<sup>1</sup>The LayerNorm used in Transformers, referred to as LayerNorm (practice), computes the statistics within each token rather than over the whole batch as LayerNorm (theory) does (Figure 1). In this paper, we will refer to the LayerNorm (practice) as LayerNorm if no distinction is made.

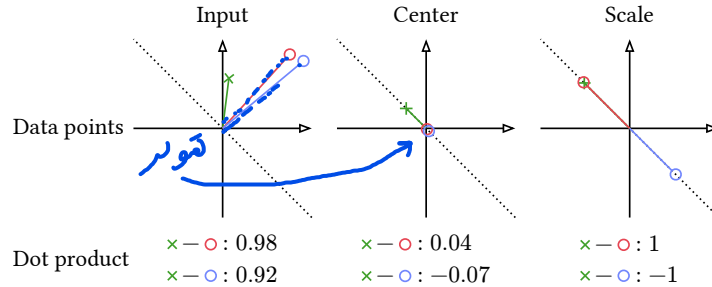


Figure 2: Case of **token shift** in LayerNorm. The green cross denotes a query vector, the red and blue circles denote two key vectors. The **token shift** happens at the **centering step of normalization** and **causes sign flip in dot product**, while the **scale step will not have** such an effect.

Unfortunately, the propensity for "center-and-scale" normalization to induce such undesirable sign flips in the dot product of vectors is not merely theoretical; it occurs with a high probability, as elucidated by the following theorem.

**Theorem 2.1** (High probability of sign flip due to center operation). *Assume that  $\mathbf{x} \sim \mathcal{N}(\mu_x, \text{diag}(\sigma_x^2))$ ,  $\mathbf{y} \sim \mathcal{N}(\mu_y, \text{diag}(\sigma_y^2))$  are two independent token vectors, with  $\mu_x, \mu_y, \sigma_x, \sigma_y \in \mathbb{R}^D$ . Let  $\tilde{\mathbf{x}} = \frac{\mathbf{x} - \mu_x}{\sigma_x}$  and  $\tilde{\mathbf{y}} = \frac{\mathbf{y} - \mu_y}{\sigma_y}$  be the normalized vectors. If*

$$|\mu_x^\top \mu_y| \geq 12 \left( \sqrt{\sigma_x^{2\top} \sigma_y^2} + \|\sigma_x \circ \sigma_y\|_\infty \right) + 5 \left( \sqrt{\sigma_y^{2\top} \mu_x^2} + \sqrt{\sigma_x^{2\top} \mu_y^2} + \|\sigma_y \circ \mu_x\|_\infty + \|\sigma_x \circ \mu_y\|_\infty \right) \quad (3)$$

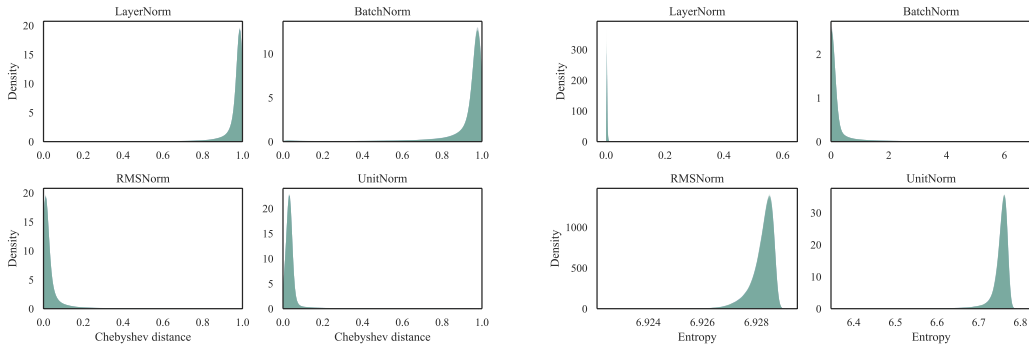
then the probability that the signs of  $\mathbf{x}^\top \mathbf{y}$  and  $\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}$  do not coincide is at least 40%, i.e.,

$$\Pr(\text{sgn}(\mathbf{x}^\top \mathbf{y}) \neq \text{sgn}(\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}})) \geq 0.40. \quad (4)$$

**Remark 2.2.** Derived from the computational methodologies for the statistics of vectors  $\mathbf{x}$  and  $\mathbf{y}$  (Appendix C.1), **BatchNorm** posits that the mean vectors are the same so that  $\mu_x = \mu_y = \mu$ , and similarly  $\sigma_x^2 = \sigma_y^2 = \sigma^2$ , while **LayerNorm** assumes that the mean and standard deviation are shared across feature dimension:  $\mu_x = \mu_x \mathbf{1}$ ,  $\mu_y = \mu_y \mathbf{1}$  and  $\sigma_x^2 = \sigma_x^2 \mathbf{1}$ ,  $\sigma_y^2 = \sigma_y^2 \mathbf{1}$ . Given these assumptions, the condition (3) outlined in Theorem 2.1 is satisfied for many token vector distributions. In fact, we show that in the setup of LayerNorm, the condition (3) allows for the quotients of token means and standard deviations, i.e., for  $\mu_x/\sigma_x$  and  $\mu_y/\sigma_y$ , to decay as  $\Omega(D^{-1/4})$  while still implying a high sign flip probability, cf. Appendix A.

Theorem 2.1 (see Appendix B for proof) underscores the vulnerability of the "center-and-scale" normalization approach to inadvertently altering the attention mechanism's functionality. The potential for such sign flips, demonstrated with significant likelihood, poses a serious risk to the integrity of the attention scores, as it can lead to a complete reordering of the tokens' importance. We shall see that substantial presence of this issue is not only theoretical, but also empirically validated in the following section.

## 2.2 Attention shift



(a) Distribution of Chebyshev distance. UnitNorm and RMSNorm preserves the distribution of attention scores, while others significantly alter the distribution. (b) Distribution of entropy. UnitNorm and RMSNorm preserves the high entropy of attention scores, while others result in a heavily unbalanced distribution.

Figure 3: Empirical statistics for attention scores after each normalization method. Results from 10 independent experiments are overlaid.  $k = 1.5$  is used for UnitNorm.

Attention shift represents a critical challenge in Transformer models, directly stemming from the token shift issue. This shift perturbs the relative significance of tokens, leading to discrepancies in the attention weights generated from normalized inputs compared to those from the original, unnormalized inputs. To validate the prevalence of attention shift across normalization techniques, we conduct a study utilizing pre-trained Word2Vec embeddings [17]. Our analysis includes a comparison of batch normalization (BatchNorm, BN, Ioffe and Szegedy 8), layer normalization (LayerNorm,

LN, Ba et al. 7, Vaswani et al. 4), root mean square layer normalization (RMSNorm, RMSN, Zhang and Sennrich 18), and our proposed unit normalization (UnitNorm, UN; see Section 3).

Our investigation utilizes sequences of token vectors,  $\mathbf{X} \in \mathbb{R}^{N \times L \times D}$ , as inputs to the normalization layer, where  $N$  is the batch size,  $L$  is the sequence length, and  $D$  is the dimensionality of each token. The attention scores  $\mathbf{A} \in \mathbb{R}^{N \times L \times L}$ , given as Equation (5), are computed for 10 independent sets of 32 batches, each containing 1,024 randomly sampled embeddings from a total of 2 million. The primary goal is to assess the impact of normalization on the fidelity of attention scores  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ , pre- and post-normalization, using the Chebyshev distance as a metric (Table S9).

$$\mathbf{A}_{n,i} = \text{softmax} \left( \frac{\mathbf{X}_{n,i} \mathbf{X}_n^T}{\sqrt{D}} \right) \quad (5)$$

where  $\mathbf{A}_{n,i} \in \mathbb{R}^L$  is the attention scores for the  $i$ -th anchor token  $\mathbf{X}_{n,i}$  to the context sequence  $\mathbf{X}_n$  from the  $n$ -th batch;  $\tilde{\mathbf{A}}$  is computed similarly from normalization output  $\tilde{\mathbf{X}}$ .

The probability density distributions of Chebyshev distances for each normalization method, depicted in Figure 3(a), reveal significant findings of the inability of maintaining faithful attention distribution of current normalization methods. For BatchNorm and LayerNorm, the Chebyshev distances predominantly span towards the maximum possible value of 1, suggesting a profound alteration in attention weights distribution by normalization. Conversely, UnitNorm and RMSNorm demonstrates a distribution concentrated around zero, indicating minimal disruption to the original attention scores.

The empirical evidence underscores a fundamental issue with current normalization practices in Transformers: they compromise the fidelity of attention scores, leading to distorted relational dynamics between tokens. This distortion challenges not only the model’s interpretability but also its ability to learn and adopt complex dependencies accurately.

### 2.3 Sparse attention

The challenge of sparse attention further complicates the normalization landscape in Transformer models. Traditional "center-and-scale" normalization methods often lead to an undesirable concentration of attention scores, effectively pushing the distribution towards one-hot. This is due to fact that centering removes a degree of freedom from the vectors, and only query that are tightly around the  $\mathbf{1}$  vector can produce uniform attention scores [16]. This can be depicted by the entropy of the attention scores  $\mathbf{A}_i$ :

$$H(\mathbf{A}_i) = - \sum_{j=1}^L \mathbf{A}_{i,j} \log \mathbf{A}_{i,j} \quad (6)$$

A higher entropy value suggests a more uniform attention distribution, enabling models capturing periodicity in time series. Conversely, lower entropy, or a trend towards one-hot distributions, limits its attention to narrow ranges of tokens. While some studies [19, 14] in other fields have shown that Transformer models may benefit from capturing longer-range, denser connections, we will show later that such sparse attention is particularly problematic in TSA tasks and requires finer control over the attention patterns.

Analysis of normalization methods through the lens of attention score entropy (Figure 3(b)) reveals a stark contrast in their effects on model behavior. BatchNorm and LayerNorm significantly skew attention distributions towards minimal entropy. This condition not only narrows the model’s focus but may also precipitate training instability [14]. In contrast, UnitNorm and RMSNorm maintain higher entropy levels, suggesting a more balanced and contextually aware attention mechanism. Notably, the key deviation in attention entropy between UnitNorm and RMSNorm is the former’s ability to modulate the entropy pattern by adjusting the  $k$  parameter, as discussed in Section 3, while RMSNorm maintains a consistent high entropy level close to the theoretical upper bound  $\log L$ .

### 3 Methodology

To mitigate the challenges identified with traditional normalization methods, we introduce a novel approach called **unit normalization (UnitNorm, UN)**, formulated such that

$$\text{UN}(\mathbf{X}) = D^{\frac{k}{2}} \frac{\mathbf{X}}{\|\mathbf{X}\|_2}, \quad (7)$$

Diverging from the conventional center-and-scale paradigm, UnitNorm omits the center operation entirely. Similar to RMSNorm, UnitNorm focuses solely on scaling the input vectors, first normalizing the input vectors by their  $\ell^2$  norm. However, UnitNorm is different to RMSNorm through subsequently scaling them by a factor of  $D^{\frac{k}{2}}$ , where  $k$  is a hyperparameter dictating the sparsity of the resulting attention scores.

#### 3.1 Theoretical foundation

UnitNorm is theoretically grounded as a variant of LayerNorm and RMSNorm. Specifically, when taking  $k = 1$ , UnitNorm is effectively acting as LayerNorm with asserted zero mean, and the RMSNorm can be seen as a special case of UnitNorm with  $k = 1$ .

This equivalence suggests that UnitNorm inherits the beneficial properties of LayerNorm and RMSNorm, such as mitigating gradient vanishing or exploding and stabilizing training. It maintains consistent forward pass and gradient propagation regardless of scaling in learnable parameters, while scaling down the gradient to these parameters when they are large (proved in Appendix B), thus ensuring stable training conditions:

**Theorem 3.1** (UnitNorm preserves the gradient to the input and stabilize the gradient to the learnable parameters). *Given the output of an affine transformation  $\mathbf{x} = \mathbf{W}\mathbf{v} + \mathbf{b}$ , where  $\mathbf{W}$  and  $\mathbf{b}$  are learnable parameters. If  $\mathbf{x}' = (\alpha\mathbf{W})\mathbf{v} + (\alpha\mathbf{b})$ , then the output of UnitNorm is unchanged, i.e.,  $\tilde{\mathbf{x}}' = \tilde{\mathbf{x}}$ , while the gradients to loss  $\mathcal{L}$  are given as follows:*

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial (\alpha\mathbf{W})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{W}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \mathbf{J}\mathbf{v}^\top \\ \frac{\partial \mathcal{L}}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial (\alpha\mathbf{b})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{b}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \mathbf{J} \\ \frac{\partial \mathcal{L}}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial \mathbf{v}} &= \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{v}} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \cdot \mathbf{J}\mathbf{W}^\top \end{aligned} \quad (8)$$

where  $\mathbf{J}$  is the Jacobian matrix of  $\mathbf{x}$  w.r.t.  $\mathbf{x}$ .

#### 3.2 Overcoming defects

While UnitNorm shares similar learning dynamics with RMSNorm, by omitting the center operation, it preserves the directions of original input vectors, directly addressing the token and attention shift problems by maintaining the dot product's sign (Figure S3). This allows UnitNorm to serve as a drop-in replacement for LayerNorm and RMSNorm in time series Transformer architectures, requiring no structural modifications.

Additionally, UnitNorm confronts the sparse attention issue by introducing an entropy lower bound (ELB) for attention scores, modulated by the hyperparameter  $k$  (proved in Appendix B). This feature enables the control of attention patterns, from dense as uniform to sparse as one-hot, offering versatility in modeling attention dynamics:

**Theorem 3.2** (UnitNorm guarantees an entropy lower bound independent of the input). *For a given set of  $L, D$  and a given  $k$ , there exists an entropy lower bound (ELB) of the attention scores, i.e.*

$$\text{ELB}(k; L, D) = \log(L - 1 + e^d) - \frac{de^d}{L - 1 + e^d}, \quad (9)$$

where  $d = 2D^{k-\frac{1}{2}}$ .

**Corollary 3.3** (The ELB of UnitNorm can be any possible value by modulating  $k$ ). *The ELB is a monotonically decreasing function of  $k$  for a given  $L, D$ . Furthermore, it is bounded that  $\forall k$ :*

$$0 < \text{ELB}(k; L, D) < \log L \quad (10)$$



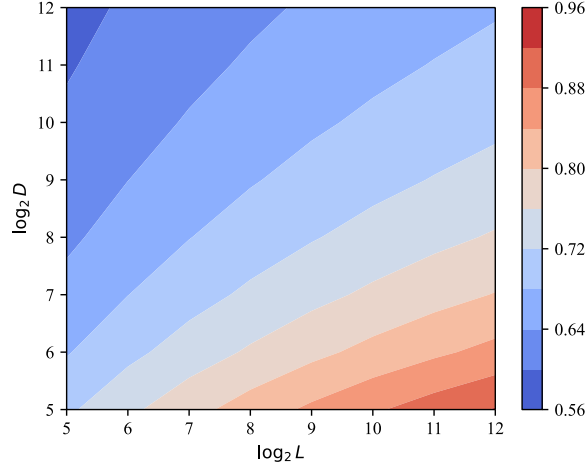


Figure 4: Landscape of  $k_{50}$  for different  $L, D$ . The  $k_{50}$  is the value of  $k$  that achieves an ELB of half of the theoretical maximum  $\log L$  for a given  $L, D$  pair. The landscape of  $k_{50}$  is rather smooth and insensitive to the sequence length  $L$ , indicating UnitNorm with fixed  $k$  can be applied to sequences with variable length without significant change in the attention pattern.

The adaptability of UnitNorm is further exemplified by its applicability across variable sequence lengths, with the entropy lower bound’s sensitivity to  $k$  remaining relatively consistent irrespective of sequence length (Figure S4), along with the smooth landscape of  $k_{50}$ , the value of  $k$  that achieves an ELB of  $\frac{1}{2} \log L$  for a given  $L, D$  pair (Figure 4), particularly with larger  $D$ . This property, combined with the option of setting  $k$  as a learnable parameter, empowers the model to dynamically adjust its attention pattern, optimizing performance across different tasks and data sets.

## 4 Experiments

In our experimental evaluation, UnitNorm is rigorously tested across a spectrum of TSA tasks to illustrate its theoretical advantages in practical applications, including long term forecasting (ETTh1, ETTh2, ECL, Exchange), classification (FaceDetection, Heartbeat, PEMS-SF, UWaveGestureLibrary) and anomaly detection (MSL). We integrate UnitNorm into various Transformer models, namely Crossformer [20], FEDformer [21], Informer [22], PatchTST [23] and the vanilla Transformer [4], all with same set of hyperparameter as described in [24]. For comparison, we also include BatchNorm, LayerNorm, RMSNorm and various settings of UnitNorm (see figure legends). By doing so, we aim to demonstrate its superior ability to address normalization-related challenges, enhancing model performance in these tasks. Detailed experimental settings and full results are provided in Tables S2 to S4 and S6 to S8. Below, we outline the significance of these tasks and the specific benefits UnitNorm brings.

**Long-term forecasting:** Long-term forecasting represents a significant challenge for Transformer models, primarily due to the difficulty in maintaining periodic pattern recognition over extended sequences [25]. The conventional normalization methods often exacerbate the sparse attention problem, hindering the model’s capability to capture periodicity. In contrast, UnitNorm demonstrates exceptional performance in mitigating this issue, as indicated by its superior rank over longer prediction horizon and slower increase in prediction error across various datasets (Figure 5). With a maximum increase of 1.46/0.45 in MSE/MAE on ETTh2, and 1.27/0.36 in MSE/MAE on Exchange at the longest prediction horizon, it substantiates UnitNorm’s ability to preserve the attention mechanism’s effectiveness, even with increasing prediction horizons, due to its ability to maintain a balanced attention distribution and omission of token shift and attention shift disturbances.

**Classification:** In classification tasks, the key challenge lies in effectively capturing long-range dependencies within sequences [26], a task at which Transformers excel. However, the efficacy of this capability can be significantly impacted by the choice of normalization method. UnitNorm, with its unique approach to normalization, has been shown to enhance model performance across multiple datasets, outperforming traditional methods in 3 out of 4 datasets on average (Figure S8), with a significant increase in accuracy of up to 4.90% on UWaveGestureLibrary, 1.95% on Heartbeat and

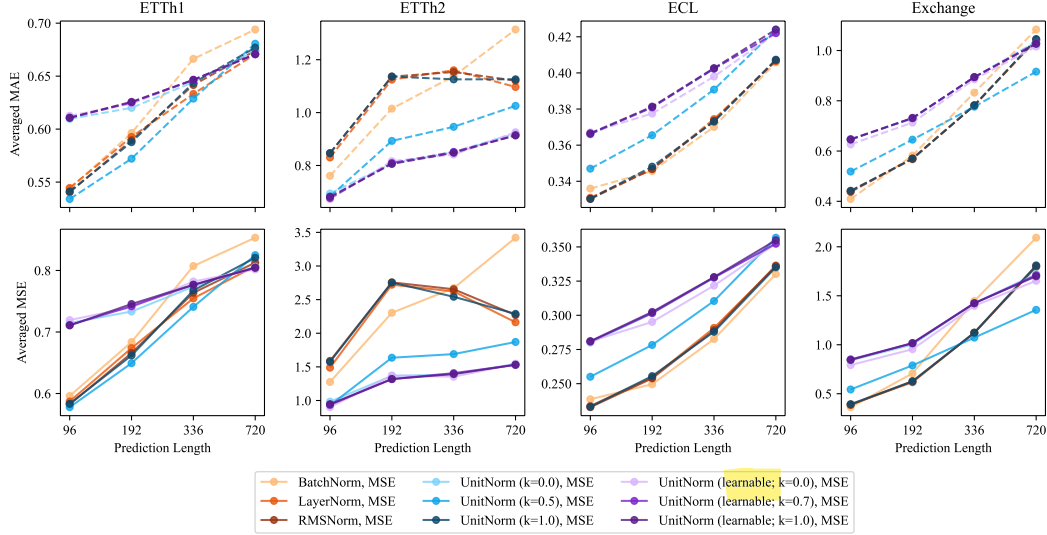


Figure 5: Average rank of normalization methods on the long-term forecasting tasks. X-axis: number of tokens to forecast, Y-axis: average rank over models. Ranks are computed based on the MAE or MSE of each model on each task with different normalization methods (lower is better). UnitNorm and UnitNorm (learnable) achieve better results with the increase of prediction horizon, and have a slower increase in prediction error compared to other normalization methods.

0.48% on FaceDetection. This underscores the versatility of UnitNorm in adapting to varied datasets, offering improved accuracy by enabling a more robust, contextually aware attention mechanism.

**Anomaly detection:** Anomaly detection in time series data demands robust model sensitivity to subtle deviations [27, 28], a requirement often compromised by normalization-induced shifts in attention. The token and attention shift problems, in particular, pose significant challenges in learning stable representations. UnitNorm addresses these challenges head-on, providing a more stable foundation for anomaly detection models to operate on, therefore gaining a maximum of 7.32% in recall, 5.58% in F-score, and 2.81% in precision. Its effectiveness is dominant in all accuracy, recall, precision, and F-score metrics (Figure S9), showcasing its capacity to facilitate more accurate and reliable time series modeling for anomaly detection.

## 5 Discussion

This study introduces UnitNorm, a normalization method tailored to Transformers that addresses the inherent limitations of currently prevalent normalization techniques. Here, we reflect on the broader implications of our findings and chart potential avenues for future research.

### 5.1 Related work

The development of effective normalization techniques is crucial in the optimization of neural network training, particularly for Transformer models [11]. This section reviews notable contributions in this field, providing a context for our proposed UnitNorm method.

The quest for effective normalization in neural networks, particularly Transformers, is ongoing, with significant strides made in understanding and optimizing these models’ training dynamics [11]. However, we can see that UnitNorm is fundamentally different from existing research directions, and provides a novel perspective on the role of normalization in Transformer models.

**Normalization layer placement:** The discourse around normalization in Transformer models has predominantly revolved around its placement: Post-Layer Normalization (Post-LN) versus Pre-Layer Normalization (Pre-LN), highlighting its impact on training stability and gradient flow [12]. Our approach with UnitNorm shifts focus from placement to the essence of normalization itself,



emphasizing the importance of preserving token vector direction being crucial for the attention mechanism, a perspective that can be applied to both Post-LN and Pre-LN Transformers.

**Normalization layer design for Transformers:** Following RMSNorm [18], UnitNorm eliminates the center operation in normalization and alleviate token shift and attention shift problem, while owning a fundamental departure by introducing a hyperparameter  $k$  to modulate the sparsity of attention scores. This design choice is unique to aid capturing periodicity in time series data.

**Normalization on model weight:** Another parallel can be drawn with Weight Normalization [29], which, despite its computational similarity to UnitNorm, applies to model parameters rather than inputs. Weight Normalization also focused on re-parameterizing for training acceleration, and thus still requires a modified BatchNorm for normalization on layer data. This distinction underscores UnitNorm’s unique approach to addressing the input-specific challenges in Transformers, diverging from methods that primarily focused on parameter optimization.

## 5.2 Adopting UnitNorm in Transformer models

UnitNorm invites reconsideration of standard normalization practices in Transformers, suggesting alternatives that might enhance model performance and stability. Its simplicity and versatility suggest it could be readily adopted across various Transformer applications. The broader impact of UnitNorm lies in its potential to improve the applicability and efficiency of Transformers in fields where precision and model stability are paramount. By addressing specific normalization-related challenges, UnitNorm can make Transformers more suitable for tasks with complex sequential relationships.

## 6 Limitations

While UnitNorm represents a significant advancement in normalization techniques for Transformers, several areas still warrant further investigation:

- **Dynamic and Adaptive Normalization:** Investigating UnitNorm’s adaptability, particularly the dynamic adjustment of the hyperparameter  $k$ , could lead to performance optimizations tailored to specific tasks.
- **Broader Application Scope:** Extending the application of UnitNorm beyond Transformers to other neural network architectures could provide valuable insights into the fundamental principles of normalization across deep learning models.
- **Cross Domain Validation:** Applying UnitNorm across diverse domains and challenging datasets beyond TSA, e.g., NLP [30, 31] and CV [32], will further elucidate its effectiveness and generalizability, providing insights into its broad utility in deep learning.
- **Problem characterization:** Understanding how and what certain dataset characteristics influence the efficacy of normalization methods, including quantitatively assess the presence of token shift, attention shift, and sparse attention in the dynamic interplay of attention mechanisms and normalization during training, can guide the community in selecting appropriate techniques for varied deep learning challenges.

Much as UnitNorm marks a promising advancement in normalization for Transformers, its exploration is far from complete. The limitations identified herein not only highlight the need for further empirical validation across domains but also the potential for refining and extending the methodology to accommodate a wider array of neural network architectures and applications.

## 7 Conclusion

Through the introduction of UnitNorm, this study challenges prevailing norms around normalization in Transformer models for TSA tasks, underscoring the importance of a tailored approach to normalization. UnitNorm’s innovative strategy, eschewing the conventional center operation, directly addresses the critical issues of token shift, attention shift that we have identified, along with sparse attention, which have been overlooked in traditional normalization practices.

Our contribution extends beyond the theoretical introduction of UnitNorm; it includes empirical evidence showcasing its efficacy across various tasks, setting a new precedent for normalization techniques within the Transformer architecture. By facilitating a more stable and faithful represen-

tation learning, UnitNorm paves the way for enhanced performance and broader applicability of Transformer models in complex sequential data analysis.

While there are also many potential ethical consequences of our work, given the theoretical nature of this work, a detailed discussion on ethical impacts falls beyond its scope. Future endeavors that leverage UnitNorm in application-specific contexts should carefully assess these considerations.

## References

- [1] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-Art Natural Language Processing. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.
- [2] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A Survey on Vision Transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45 (1):87–110, January 2023. ISSN 1939-3539.
- [3] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in Time Series: A Survey. volume 6, pages 6778–6786, August 2023.
- [4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [5] Ekdeep S Lubana, Robert Dick, and Hidenori Tanaka. Beyond batchnorm: towards a unified understanding of normalization in deep learning. *Advances in Neural Information Processing Systems*, 34:4778–4791, 2021.
- [6] Greg Yang and Samuel S. Schoenholz. Mean Field Residual Networks: On the Edge of Chaos, December 2017.
- [7] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization, July 2016.
- [8] Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML’15*, pages 448–456, Lille, France, July 2015. JMLR.org.
- [9] Sheng Shen, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. PowerNorm: rethinking batch normalization in transformers. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML’20*, pages 8741–8751. JMLR.org, July 2020.
- [10] Jiayi Wang, Ji Wu, and Lei Huang. Understanding the Failure of Batch Normalization for Transformers in NLP, October 2022.
- [11] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. Learning Deep Transformer Models for Machine Translation. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy, July 2019. Association for Computational Linguistics.
- [12] 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. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML’20*, pages 10524–10533. JMLR.org, July 2020.
- [13] Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Incorporating Residual and Normalization Layers into Analysis of Masked Language Models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4547–4568, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

- [14] 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*, pages 40770–40803. PMLR, 2023.
- [15] Lily Zhang, Veronica Tozzo, John Higgins, and Rajesh Ranganath. Set Norm and Equivariant Skip Connections: Putting the Deep in Deep Sets. In *Proceedings of the 39th International Conference on Machine Learning*, pages 26559–26574. PMLR, June 2022.
- [16] Shaked Brody, Uri Alon, and Eran Yahav. On the Expressivity Role of LayerNorm in Transformers’ Attention. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14211–14221, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [17] Murhaf Fares, Andrey Kutuzov, Stephan Oepen, and Erik Velldal. Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In Jörg Tiedemann and Nina Tahmasebi, editors, *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 271–276, Gothenburg, Sweden, May 2017. Association for Computational Linguistics.
- [18] Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [19] Nam Hyeon-Woo, Kim Yu-Ji, Byeongho Heo, Dongyoon Han, Seong Joon Oh, and Tae-Hyun Oh. Scratching Visual Transformer’s Back with Uniform Attention, October 2022.
- [20] Yunhao Zhang and Junchi Yan. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. September 2022.
- [21] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR, July 2022.
- [22] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021. ISSN 2374-3468.
- [23] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. September 2022.
- [24] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations*, 2023.
- [25] Yan Li, Xinjiang Lu, Haoyi Xiong, Jian Tang, Jiantao Su, Bo Jin, and Dejing Dou. Towards Long-Term Time-Series Forecasting: Feature, Pattern, and Distribution. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 1611–1624, April 2023.
- [26] Jayant Vyas, Nishit Bhardwaj, Bhumika, and Debasis Das. TransDBC: Transformer for Multivariate Time-Series based Driver Behavior Classification. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2022.
- [27] Ijaz Ul Haq and Byung Suk Lee. TransNAS-TSAD: Harnessing Transformers for Multi-Objective Neural Architecture Search in Time Series Anomaly Detection, December 2023.
- [28] Chaocheng Yang, Tingyin Wang, and Xuanhui Yan. DDMT: Denoising Diffusion Mask Transformer Models for Multivariate Time Series Anomaly Detection, October 2023.
- [29] Tim Salimans and Diederik P. Kingma. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks, June 2016.

- [30] 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 *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [32] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. October 2020.
- [33] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2018. ISBN 978-1-108-41519-4.
- [34] Sjoerd Dirksen. Tail bounds via generic chaining. *Electronic Journal of Probability*, 20(none): 1–29, January 2015.
- [35] Artur Trindade. ElectricityLoadDiagrams20112014, 2015.
- [36] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '18*, pages 95–104, New York, NY, USA, June 2018. Association for Computing Machinery. ISBN 978-1-4503-5657-2.
- [37] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018, October 2018.
- [38] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18*, pages 387–395, New York, NY, USA, July 2018. Association for Computing Machinery. ISBN 978-1-4503-5552-0.

## A Dimension Dependence of Sign-Flip Probability

We recall that Theorem 2.1 provided a condition for token vector means and variances, condition (3), to imply that the sign of the token dot product  $\mathbf{x}^\top \mathbf{y}$  is flipped by center-and-scale standardization as in LayerNorm [7].

In this section, we elucidate the dimension dependence of the required relationship between token means and standard deviations implied by this condition in the case of shared means and standard deviations across feature dimensions, such as implicitly assumed by LayerNorm.

**Corollary A.1.** *Assume that the mean and variance vectors of independent token vectors  $\mathbf{x}$  and  $\mathbf{y}$  satisfy  $\boldsymbol{\mu}_x = \mu_x \mathbf{1}$ ,  $\boldsymbol{\mu}_y = \mu_y \mathbf{1}$  and  $\boldsymbol{\sigma}_x^2 = \sigma_x^2 \mathbf{1}$ ,  $\boldsymbol{\sigma}_y^2 = \sigma_y^2 \mathbf{1}$ . Then the mean-variance condition (3) of Theorem 2.1 is satisfied for all  $L \geq 77$  if*

$$\frac{\mu_x}{\sigma_x} \geq \frac{6}{D^{1/4}} \quad \text{and} \quad \frac{\mu_y}{\sigma_y} \geq \frac{6}{D^{1/4}}, \quad (11)$$

Furthermore, if additionally the independent token vectors are distributed as  $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \text{diag}(\boldsymbol{\sigma}_x^2))$ ,  $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, \text{diag}(\boldsymbol{\sigma}_y^2))$ , then the dot product  $\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}$  of normalized vectors  $\tilde{\mathbf{x}} = \frac{\mathbf{x} - \boldsymbol{\mu}_x}{\boldsymbol{\sigma}_x}$  and  $\tilde{\mathbf{y}} = \frac{\mathbf{y} - \boldsymbol{\mu}_y}{\boldsymbol{\sigma}_y}$  attains a sign flip with respect to the original inner products  $\mathbf{x}^\top \mathbf{y}$  with probability of at least 40%.

Corollary A.1 implies that for high-dimensional token vectors with  $D \gg 1$ , it might become easier to satisfy (11) given an empirical token distribution, which means that sign flips of dot products after LayerNorm-style normalization might become even more prevalent in that case.

*Proof of Corollary A.1.* For the case of  $\boldsymbol{\mu}_x = \mu_x \mathbf{1}$ ,  $\boldsymbol{\mu}_y = \mu_y \mathbf{1}$  and  $\boldsymbol{\sigma}_x^2 = \sigma_x^2 \mathbf{1}$ ,  $\boldsymbol{\sigma}_y^2 = \sigma_y^2 \mathbf{1}$ , it follows that

$$\begin{aligned} & 12 \left( \sqrt{\boldsymbol{\sigma}_x^{2\top} \boldsymbol{\sigma}_y^2} + \|\boldsymbol{\sigma}_x \circ \boldsymbol{\sigma}_y\|_\infty \right) + 5 \left( \sqrt{\boldsymbol{\sigma}_y^{2\top} \boldsymbol{\mu}_x^2} + \sqrt{\boldsymbol{\sigma}_x^{2\top} \boldsymbol{\mu}_y^2} + \|\boldsymbol{\sigma}_y \circ \boldsymbol{\mu}_x\|_\infty + \|\boldsymbol{\sigma}_x \circ \boldsymbol{\mu}_y\|_\infty \right) \\ &= 12 \left( \sqrt{D \sigma_x^2 \sigma_y^2} + \sigma_x \sigma_y \right) + 5 \left( \sqrt{D \sigma_y^2 \mu_x^2} + \sqrt{D \sigma_x^2 \mu_y^2} + \sigma_y |\mu_x| + \sigma_x |\mu_y| \right) \\ &\leq 12 \left( \sqrt{D \frac{D \mu_x^2 \mu_y^2}{36^2}} + \frac{D^{1/4} \mu_x}{6} \frac{D^{1/4} \mu_y}{6} \right) \\ &\quad + 5 \left( \sqrt{D \frac{D^{1/2} \mu_y^2}{36} \mu_x^2} + \sqrt{D \frac{D^{1/2} \mu_x^2}{36} \mu_y^2} + \frac{D^{1/4} \mu_y}{6} |\mu_x| + \frac{D^{1/4} \mu_x}{6} |\mu_y| \right) \\ &= 12 \left( D \frac{\mu_x \mu_y}{36} + D^{1/2} \frac{\mu_x \mu_y}{36} \right) + \frac{5}{6} \left( \sqrt{D D^{1/2} \mu_y^2 \mu_x^2} + \sqrt{D D^{1/2} \mu_x^2 \mu_y^2} + D^{1/4} \mu_y |\mu_x| + D^{1/4} \mu_x |\mu_y| \right) \\ &= \mu_x \mu_y \left( \frac{1}{3} D + \frac{1}{3} D^{1/2} + \frac{5}{3} (D^{3/4} + D^{1/4}) \right) \leq D \mu_x \mu_y = |\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y|. \end{aligned}$$

Here, we used in the first inequality the assumption Equation (11) and the fact that  $\frac{1}{3} D^{1/2} + \frac{5}{3} (D^{3/4} + D^{1/4}) \leq \frac{2}{3} D$  for  $D \geq 77$  in the last inequality. The last assertion of the theorem then follows by application of Theorem 2.1.  $\square$

## B Proofs

In this section, we detail the proofs of the theoretical results of this paper. In particular, we present the proofs of Theorem 2.1, Theorem 3.1, Theorem 3.2, Corollary 3.3, as well as of auxiliary lemmas.

### B.1 Proof of Theorem 2.1

*Proof of Theorem 2.1.* Let  $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \text{diag}(\boldsymbol{\sigma}_x^2))$  and  $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, \text{diag}(\boldsymbol{\sigma}_y^2))$  be independent, and write  $\mathbf{x} = (X_1, \dots, X_D)$  and  $\mathbf{y} = (Y_1, \dots, Y_D)$ , respectively.



then we can compute the expectation  $\mathbb{E} [\mathbf{x}^\top \mathbf{y}]$  of the dot product of  $\mathbf{x}$  and  $\mathbf{y}$  as

$$\begin{aligned}\mathbb{E} [\mathbf{x}^\top \mathbf{y}] &= \mathbb{E} \left[ \sum_{i=1}^D X_i Y_i \right] \\ &= \sum_{i=1}^D \mathbb{E} [X_i Y_i] \\ &= \sum_{i=1}^D \mathbb{E} [X_i] \mathbb{E} [Y_i] \\ &= \sum_{i=1}^D (\boldsymbol{\mu}_x)_i (\boldsymbol{\mu}_y)_i \\ &= \boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y.\end{aligned}$$

$$\begin{aligned}\text{Var} (\mathbf{x}^\top \mathbf{y}) &= \mathbb{E} \left[ (\mathbf{x}^\top \mathbf{y})^2 \right] - (\mathbb{E} [\mathbf{x}^\top \mathbf{y}])^2 \\ &= \mathbb{E} \left[ \left( \sum_{i=1}^D X_i Y_i \right)^2 \right] - (\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y)^2 \\ &= \sum_{i,j=1}^D \mathbb{E} [X_i Y_i X_j Y_j] - (\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y)^2 \\ &= \sum_{i,j=1}^D \mathbb{E} [X_i X_j] \mathbb{E} [Y_i Y_j] - (\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y)^2\end{aligned}\tag{12}$$

By definition of covariance, we have  $\boldsymbol{\sigma}_x = \mathbb{E} [\mathbf{x} \mathbf{x}^\top] - \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top$ , and here  $\boldsymbol{\sigma}_x = \text{diag}(\boldsymbol{\sigma}_x^2)$ , then Equation (12) can be simplified as follows:

$$\begin{aligned}\text{Var} (\mathbf{x}^\top \mathbf{y}) &= \sum_{i,j=1}^D (\boldsymbol{\sigma}_x + \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top)_{ij} (\boldsymbol{\sigma}_y + \boldsymbol{\mu}_y \boldsymbol{\mu}_y^\top)_{ij} - (\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y)^2 \\ &= \langle \boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y \rangle_F + \langle \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top, \boldsymbol{\sigma}_y \rangle_F + \langle \boldsymbol{\sigma}_x, \boldsymbol{\mu}_y \boldsymbol{\mu}_y^\top \rangle_F + \langle \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top, \boldsymbol{\mu}_y \boldsymbol{\mu}_y^\top \rangle_F - (\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y)^2 \\ &= (\boldsymbol{\sigma}_x^2)^\top (\boldsymbol{\sigma}_y^2) + (\boldsymbol{\sigma}_y^2)^\top (\boldsymbol{\mu}_x^2) + (\boldsymbol{\sigma}_x^2)^\top (\boldsymbol{\mu}_y^2)\end{aligned}\tag{13}$$

where  $\langle \cdot, \cdot \rangle_F$  is the Frobenius inner product.

Consider now the normalized random vectors  $\tilde{\mathbf{x}} = \frac{\mathbf{x} - \boldsymbol{\mu}_x}{\boldsymbol{\sigma}_x}$  and  $\tilde{\mathbf{y}} = \frac{\mathbf{y} - \boldsymbol{\mu}_y}{\boldsymbol{\sigma}_y}$ . Due to the Gaussianity assumption on  $\mathbf{x}$  and  $\mathbf{y}$ , it follows that the normalized vectors are also Gaussian, and in particular, are distributed as  $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \sim \mathcal{N}(\mathbf{0}, \mathbb{I})$ . Plugging the respective mean and variance values into the formulas for the expectation and variance for dot products above, we obtain that

$$\mathbb{E} [\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}] = 0 \quad \text{and} \quad \text{Var} (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}) = 1\tag{14}$$

As  $\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}$  is a symmetric random variable, it follows that

$$\Pr (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}) = 0.5.\tag{15}$$

Next, due to the definition of the random vectors  $\mathbf{x}$  and  $\mathbf{y}$ , it holds that  $\mathbf{x}^\top \mathbf{y} = \sum_{i=1}^D X_i Y_i$ , where  $X_i \sim \mathcal{N}((\boldsymbol{\mu}_x)_i, (\boldsymbol{\sigma}_x)_i^2)$  and  $Y_i \sim \mathcal{N}((\boldsymbol{\mu}_y)_i, (\boldsymbol{\sigma}_y)_i^2)$  are independent normal random variables. Going forward, we will use the  $\psi_1$ -Orlicz norm

$$\|X\|_{\psi_1} := \inf \{t > 0 : \mathbb{E}[\exp(|X|/t)] \leq 2\},\tag{16}$$

cf. Definition 2.7.5 of [33]. We call a random variable for which  $\|\cdot\|_{\psi_1}$  is finite sub-exponential, following, e.g., [33].

Define now  $Z_i := X_i Y_i - (\boldsymbol{\mu}_x)_i (\boldsymbol{\mu}_y)_i$ . We observe that

$$Z_i = X_i Y_i - (\boldsymbol{\mu}_x)_i (\boldsymbol{\mu}_y)_i = X_i (Y_i - (\boldsymbol{\mu}_y)_i) + (X_i - (\boldsymbol{\mu}_x)_i) (\boldsymbol{\mu}_y)_i = Z_i^{(1)} + Z_i^{(2)}$$

with  $Z_i^{(1)} := X_i (Y_i - (\boldsymbol{\mu}_y)_i)$  and  $Z_i^{(2)} := (X_i - (\boldsymbol{\mu}_x)_i) (\boldsymbol{\mu}_y)_i$ . To bound the  $\psi_1$ -norm of  $Z_i$ , we bound this norm for  $Z_i^{(1)}$  and  $Z_i^{(2)}$  separately.

Indeed, due to Lemma 2.7.7 of [33], it holds that

$$\|Z_i^{(1)}\|_{\psi_1} \leq \|X_i\|_{\psi_2} \|Y_i - (\boldsymbol{\mu}_y)_i\|_{\psi_2},$$

where

$$\|X\|_{\psi_2} := \inf\{t > 0 : \mathbb{E}[\exp(X^2/t^2)] \leq 2\}, \quad (17)$$

is the  $\psi_2$ -Orlicz norm [33] characterizing sub-Gaussian random variables  $X$ . From Lemma B.2, it follows therefore that

$$\|Z_i^{(1)}\|_{\psi_1} \leq \max \left( 2(\boldsymbol{\sigma}_x)_i, \sqrt{\frac{(\boldsymbol{\mu}_x)_i^2}{\log 2} + (\boldsymbol{\sigma}_x)_i^2} \right) \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_y)_i.$$

For the second part, since  $\|\cdot\|_{\psi_2}$  is a norm, we estimate that

$$\|Z_i^{(2)}\|_{\psi_1} \leq \|X_i - (\boldsymbol{\mu}_x)_i\|_{\psi_2} \|(\boldsymbol{\mu}_y)_i\|_{\psi_2} \leq \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_x)_i \|(\boldsymbol{\mu}_y)_i\|_{\psi_2} \leq \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_x)_i \frac{|(\boldsymbol{\mu}_y)_i|}{\sqrt{\log 2}},$$

where we used again Lemma 2.7.7 and (2.17) of [33] in the first and last inequality, respectively, and Lemma B.2 in the second inequality.

From this, it follows that

$$\begin{aligned} \|Z_i\|_{\psi_1} &\leq \|Z_i^{(1)}\|_{\psi_1} + \|Z_i^{(2)}\|_{\psi_1} \leq \max \left( 2(\boldsymbol{\sigma}_x)_i, \sqrt{\frac{(\boldsymbol{\mu}_x)_i^2}{\log 2} + (\boldsymbol{\sigma}_x)_i^2} \right) \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_y)_i + \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_x)_i \frac{|(\boldsymbol{\mu}_y)_i|}{\sqrt{\log 2}} \\ &\leq \left( 2(\boldsymbol{\sigma}_x)_i + \sqrt{\frac{(\boldsymbol{\mu}_x)_i^2}{\log 2} + (\boldsymbol{\sigma}_x)_i^2} \right) \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_y)_i + \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_x)_i \frac{|(\boldsymbol{\mu}_y)_i|}{\sqrt{\log 2}} \\ &\leq 2\sqrt{6} (\boldsymbol{\sigma}_x)_i (\boldsymbol{\sigma}_y)_i + \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_y)_i \frac{|(\boldsymbol{\mu}_x)_i|}{\sqrt{\log 2}} + \sqrt{\frac{8}{3}} (\boldsymbol{\sigma}_x)_i \frac{|(\boldsymbol{\mu}_y)_i|}{\sqrt{\log 2}}, \end{aligned} \quad (18)$$

using that  $\sqrt{a^2 + b^2} \leq a + b$  for any non-negative  $a, b \geq 0$  in the last inequality. We next establish a lower bound on the probability of a sign flip through normalization, i.e., for  $\Pr(\text{sgn}(\mathbf{x}^\top \mathbf{y}) \neq \text{sgn}(\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}))$ . Assuming without loss of generality that  $|\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y| = \boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y$ , we observe that

$$\begin{aligned} \Pr(\text{sgn}(\mathbf{x}^\top \mathbf{y}) \neq \text{sgn}(\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}})) &= \Pr((\mathbf{x}^\top \mathbf{y} > 0) \wedge (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} < 0)) + \Pr((\mathbf{x}^\top \mathbf{y} < 0) \wedge (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} > 0)) \\ &\geq \Pr((\mathbf{x}^\top \mathbf{y} > 0) \wedge (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} < 0)). \end{aligned}$$

Furthermore, since the distribution of the normalized vectors  $\tilde{\mathbf{x}}$  and  $\tilde{\mathbf{y}}$  is symmetric, the same holds true for the dot product  $\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}}$ , which implies that

$$\begin{aligned} \Pr((\mathbf{x}^\top \mathbf{y} > 0) \wedge (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} < 0)) &= 1 - \Pr((\mathbf{x}^\top \mathbf{y} \leq 0) \vee (\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} \geq 0)) \geq 1 - \Pr(\mathbf{x}^\top \mathbf{y} \leq 0) - \Pr(\tilde{\mathbf{x}}^\top \tilde{\mathbf{y}} \geq 0) \\ &\geq 1 - 0.5 - \Pr(\mathbf{x}^\top \mathbf{y} \leq 0) = 0.5 - \Pr(\mathbf{x}^\top \mathbf{y} \leq 0). \end{aligned}$$

It remains to show that

$$\Pr(\mathbf{x}^\top \mathbf{y} \leq 0) \leq 0.1. \quad (19)$$

To establish this, we see that

$$\begin{aligned} \Pr(\mathbf{x}^\top \mathbf{y} \leq 0) &= \Pr(\mathbf{x}^\top \mathbf{y} - \mathbb{E}[\mathbf{x}^\top \mathbf{y}] \leq -\mathbb{E}[\mathbf{x}^\top \mathbf{y}]) = \Pr(\mathbf{x}^\top \mathbf{y} - \boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y \leq -\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y) \\ &= \Pr\left(\sum_{i=1}^D Z_i \leq -\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y\right) \end{aligned}$$

with the random variables  $Z_i$  defined above. Using the triangle inequality of the  $\ell_2$ -norm, it follows from (18) that

$$\sqrt{\sum_{i=1}^D \|Z_i\|_{\psi_1}^2} \leq 2\sqrt{6}\sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\sigma}_y^2} + \sqrt{\frac{8}{3\log 2}} \left( \sqrt{(\boldsymbol{\sigma}_y^2)^\top \boldsymbol{\mu}_x^2} + \sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\mu}_y^2} \right)$$

and that

$$\max_{i=1}^D \|Z_i\|_{\psi_1} \leq 2\sqrt{6}\|\boldsymbol{\sigma}_x \circ \boldsymbol{\sigma}_y\|_\infty + \sqrt{\frac{8}{3\log 2}} (\|\boldsymbol{\sigma}_y \circ |(\boldsymbol{\mu}_x)|\|_\infty + \|\boldsymbol{\sigma}_x \circ |(\boldsymbol{\mu}_y)|\|_\infty),$$

which implies that

$$\begin{aligned} & \sqrt{2 \sum_{i=1}^D \|Z_i\|_{\psi_1}^2} \sqrt{\log(10)} + \max_{i=1}^D \|Z_i\|_{\psi_1} \log(10) \\ & \leq 4\sqrt{3\log(10)} \sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\sigma}_y^2} + \frac{4\sqrt{\log(10)}}{\sqrt{3\log(2)}} \left( \sqrt{(\boldsymbol{\sigma}_y^2)^\top \boldsymbol{\mu}_x^2} + \sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\mu}_y^2} \right) \\ & \quad + 2\sqrt{6}\log(10)\|\boldsymbol{\sigma}_x \circ \boldsymbol{\sigma}_y\|_\infty + \sqrt{\frac{8}{3\log 2}} \log(10) (\|\boldsymbol{\sigma}_y \circ |(\boldsymbol{\mu}_x)|\|_\infty + \|\boldsymbol{\sigma}_x \circ |(\boldsymbol{\mu}_y)|\|_\infty) \\ & \leq 12 \left( \sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\sigma}_y^2} + \|\boldsymbol{\sigma}_x \circ \boldsymbol{\sigma}_y\|_\infty \right) + 5 \left( \sqrt{(\boldsymbol{\sigma}_y^2)^\top \boldsymbol{\mu}_x^2} + \sqrt{(\boldsymbol{\sigma}_x^2)^\top \boldsymbol{\mu}_y^2} + \|\boldsymbol{\sigma}_y \circ |(\boldsymbol{\mu}_x)|\|_\infty + \|\boldsymbol{\sigma}_x \circ |(\boldsymbol{\mu}_y)|\|_\infty \right) \\ & \leq |\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y|, \end{aligned}$$

using the assumption (3) in the last inequality. With this inequality, we can use the fact that the  $Z_i$  are independent mean-zero sub-exponential random variables and Bernstein's inequality as stated in Lemma B.1 to conclude that

$$\begin{aligned} \Pr \left( \sum_{i=1}^D Z_i \leq -\boldsymbol{\mu}_x^\top \boldsymbol{\mu}_y \right) & \leq \Pr \left( \sum_{i=1}^D Z_i \leq - \left( \sqrt{2 \sum_{i=1}^D \|Z_i\|_{\psi_1}^2} \sqrt{\log(10)} + \max_{i=1}^D \|Z_i\|_{\psi_1} \log(10) \right) \right) \\ & \leq \exp(-\log(10)) = 0.1. \end{aligned}$$

This establishes (19), which concludes the proof.  $\square$

**Lemma B.1** (Bernstein's Inequality, cf. Lemma 5.1 of [34]). *Let  $Z_1, \dots, Z_D$  be independent mean-zero sub-exponential random variables. Then for every  $t \geq 0$ ,*

$$\Pr \left( \sum_{i=1}^D Z_i \leq - \left( \sqrt{2 \sum_{i=1}^D \|Z_i\|_{\psi_1}^2} \sqrt{t} + \max_{i=1}^D \|Z_i\|_{\psi_1} t \right) \right) \leq \exp(-t).$$

**Lemma B.2** (Bounds on  $\psi_2$ -norm of Gaussians [33]). *1. If  $X \sim \mathcal{N}(0, \sigma^2)$  is a centered Gaussian random variable with variance  $\sigma^2$ , then its  $\psi_2$ -norm (17) satisfies*

$$\|X\|_{\psi_2} \leq \sqrt{\frac{8}{3}}\sigma.$$

*2. If  $X \sim \mathcal{N}(\mu, \sigma^2)$  is a Gaussian random variable with mean  $\mu$  and variance  $\sigma^2$ , then its  $\psi_2$ -norm (17) satisfies*

$$\|X\|_{\psi_2} \leq \max \left( 2\sigma, \sqrt{\frac{\mu^2}{\log 2} + \sigma^2} \right).$$

## B.2 Proof of Theorem 3.1

*Proof of Theorem 3.1.* Given the output of an affine transformation  $\mathbf{x} = \mathbf{W}\mathbf{v} + \mathbf{b}$ , where  $\mathbf{W}$  and  $\mathbf{b}$  are learnable parameters. If  $\mathbf{x}' = (\alpha\mathbf{W})\mathbf{v} + (\alpha\mathbf{b})$ , then the output of UnitNorm is unchanged, i.e.,

$\tilde{\mathbf{x}}' = \tilde{\mathbf{x}}$ , while the gradients to loss  $\mathcal{L}$  are given as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha \mathbf{W})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{W}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{v}^\top \\
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha \mathbf{b})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{b}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \\
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{v}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{v}} = \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{W}^\top
\end{aligned} \tag{20}$$

Proof: First we will show  $\tilde{\mathbf{x}}' = \tilde{\mathbf{x}}$ , for which we have:

$$\begin{aligned}
\tilde{\mathbf{x}}' &= D^{\frac{k}{2}} \frac{\mathbf{x}'}{\|\mathbf{x}'\|} \\
&= D^{\frac{k}{2}} \frac{\alpha \mathbf{x}}{\alpha \|\mathbf{x}\|} \\
&= D^{\frac{k}{2}} \frac{\mathbf{x}}{\|\mathbf{x}\|} \\
&= \tilde{\mathbf{x}}
\end{aligned} \tag{21}$$

And thus for the gradients to loss  $\mathcal{L}$ , we have  $\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} = \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}}$ . Also, for the Jacobian matrix  $\mathbf{J}$  of  $\tilde{\mathbf{x}}$  w.r.t.  $\mathbf{x}$ , we have

$$\begin{aligned}
\mathbf{J} &= \frac{\partial D^{\frac{k}{2}} \frac{\mathbf{x}}{\|\mathbf{x}\|}}{\partial \mathbf{x}} \\
&= D^{\frac{k}{2}} \left( \frac{\mathbf{I}}{\|\mathbf{x}\|} - \frac{\mathbf{x} \mathbf{x}^\top}{\|\mathbf{x}\|^3} \right)
\end{aligned} \tag{22}$$

And the Jacobian matrix of  $\tilde{\mathbf{x}}'$  w.r.t.  $\mathbf{x}'$  is given as:

$$\begin{aligned}
\frac{\partial D^{\frac{k}{2}} \frac{\mathbf{x}'}{\|\mathbf{x}'\|}}{\partial \mathbf{x}'} &= D^{\frac{k}{2}} \left( \frac{\mathbf{I}}{\|\mathbf{x}'\|} - \frac{\mathbf{x}' \mathbf{x}'^\top}{\|\mathbf{x}'\|^3} \right) \\
&= D^{\frac{k}{2}} \left( \frac{\mathbf{I}}{\alpha \|\mathbf{x}\|} - \frac{\alpha^2 \mathbf{x} \mathbf{x}^\top}{\alpha^3 \|\mathbf{x}\|^3} \right) \\
&= \frac{1}{\alpha} D^{\frac{k}{2}} \left( \frac{\mathbf{I}}{\|\mathbf{x}\|} - \frac{\mathbf{x} \mathbf{x}^\top}{\|\mathbf{x}\|^3} \right) \\
&= \frac{1}{\alpha} \mathbf{J}
\end{aligned} \tag{23}$$

Then we have the gradient of loss w.r.t.  $\mathbf{W}$  and  $\alpha \mathbf{W}$ :

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{W}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{W}} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{v}^\top \\
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha \mathbf{W})} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial (\alpha \mathbf{W})} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{1}{\alpha} \mathbf{J} \mathbf{v}^\top \\
\Rightarrow \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha \mathbf{W})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{W}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{v}^\top
\end{aligned} \tag{24}$$

Similarly, for  $\mathbf{b}$  and  $\alpha\mathbf{b}$  we have:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{b}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{b}} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \\
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha\mathbf{b})} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial (\alpha\mathbf{b})} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{1}{\alpha} \mathbf{J} \\
\Rightarrow \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial (\alpha\mathbf{b})} &= \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{b}} = \frac{1}{\alpha} \cdot \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J}
\end{aligned} \tag{25}$$

And for  $\mathbf{v}$ , we have:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{v}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{v}} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{W}^\top \\
\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{v}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{x}'} \cdot \frac{\partial \mathbf{x}'}{\partial \mathbf{v}} \\
&= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{1}{\alpha} \mathbf{J} (\alpha \mathbf{W})^\top \\
\Rightarrow \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}'} \cdot \frac{\partial \tilde{\mathbf{x}}'}{\partial \mathbf{v}} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{v}} = \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}} \cdot \mathbf{J} \mathbf{W}^\top
\end{aligned} \tag{26}$$

□

### B.3 Proofs of Theorem 3.2 and Corollary 3.3

*Proof of Theorem 3.2.* Let  $\mathbf{X} \in \mathbb{R}^{L \times D}$  be a single sequence of token vectors, and let  $\tilde{\mathbf{X}}$  be the unit normalized output with modulus  $k$ , the entropy lower bound (ELB) of the attention scores is given by the following expression:

$$\begin{aligned}
\text{ELB}(k; L, D) &= \min_{i=1}^L H(\mathbf{A}_i) \\
&= \min_{i=1}^L \left( - \sum_{j=1}^L \mathbf{A}_{i,j} \log \mathbf{A}_{i,j} \right) \\
&= \log \left( L - 1 + \exp \left( 2D^{k-\frac{1}{2}} \right) \right) - \frac{2D^{k-\frac{1}{2}} \exp \left( 2D^{k-\frac{1}{2}} \right)}{L - 1 + \exp \left( 2D^{k-\frac{1}{2}} \right)}
\end{aligned} \tag{27}$$

Proof: Let  $\tilde{\mathbf{X}} = D^{\frac{k}{2}} \mathbf{e}$  where  $\mathbf{e}$  are the vectors of unit norm. Without loss of generality, we can assume the ELB is achieved at anchor index  $i$ , where we can compute the attention scores as follows:

$$\begin{aligned}
\mathbf{A}_i &= \text{softmax} \left( \frac{\tilde{\mathbf{X}}_i \tilde{\mathbf{X}}^\top}{\sqrt{D}} \right) \\
&= \text{softmax} \left( \frac{D^k \mathbf{e}_i \mathbf{e}^\top}{\sqrt{D}} \right) \\
&= \text{softmax} \left( D^{k-\frac{1}{2}} \mathbf{e}_i \mathbf{e}^\top \right)
\end{aligned} \tag{28}$$

Since  $\mathbf{e}_i \mathbf{e}_j^\top \in (-1, 1)$ ,  $\forall i, j = 1, 2, \dots, L$ , the entropy of the attentions scores is lower bounded by the following expression when it satisfies that  $\mathbf{e}_i \mathbf{e}_j = \begin{cases} 1, & j = i \\ -1, & j \neq i \end{cases}$ :

$$\begin{aligned}
& H(\mathbf{A}_i) \\
&= - \sum_{j=1}^L \mathbf{A}_{i,j} \log \mathbf{A}_{i,j} \\
&= -(L-1) \cdot \frac{\exp\left(-D^{k-\frac{1}{2}}\right)}{(L-1) \exp\left(-D^{k-\frac{1}{2}}\right) + \exp\left(D^{k-\frac{1}{2}}\right)} \log \frac{\exp\left(-D^{k-\frac{1}{2}}\right)}{(L-1) \exp\left(-D^{k-\frac{1}{2}}\right) + \exp\left(D^{k-\frac{1}{2}}\right)} \\
&\quad - \frac{\exp\left(D^{k-\frac{1}{2}}\right)}{(L-1) \exp\left(-D^{k-\frac{1}{2}}\right) + \exp\left(D^{k-\frac{1}{2}}\right)} \log \frac{\exp\left(D^{k-\frac{1}{2}}\right)}{(L-1) \exp\left(-D^{k-\frac{1}{2}}\right) + \exp\left(D^{k-\frac{1}{2}}\right)} \\
&= \frac{L-1}{L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)} \log\left(L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)\right) \\
&\quad + \frac{\exp\left(2D^{k-\frac{1}{2}}\right)}{L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)} \log \frac{L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)}{\exp\left(2D^{k-\frac{1}{2}}\right)} \\
&= \log\left(L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)\right) - \frac{2D^{k-\frac{1}{2}} \exp\left(2D^{k-\frac{1}{2}}\right)}{L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)}
\end{aligned} \tag{29}$$

Therefore, the entropy lower bound (ELB) for any  $L, D$  and  $k$  is:

$$\text{ELB}(k; L, D) = \log\left(L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)\right) - \frac{2D^{k-\frac{1}{2}} \exp\left(2D^{k-\frac{1}{2}}\right)}{L-1 + \exp\left(2D^{k-\frac{1}{2}}\right)} \tag{30}$$

□

*Proof of Corollary 3.3.* The ELB is a monotonically decreasing function of  $k$  bounded between 0 and  $\log L$ .

Proof: Let  $d = 2D^{k-\frac{1}{2}}$ , then it is obvious that  $d$  is monotonically increasing with  $k$ , therefore we only need to prove that  $\text{ELB}(k; L, D)$  is monotonically decreasing with  $d$ . The derivative of  $\text{ELB}(k; L, D)$  with respect to  $d$  is given as follows:

$$\begin{aligned}
\frac{\partial \text{ELB}(k; L, D)}{\partial d} &= \frac{e^d}{L-1 + e^d} - \frac{(L-1 + e^d)(d+1)e^d - (de^d)e^d}{(L-1 + e^d)^2} \\
&= \frac{e^d}{(L-1 + e^d)^2} ((L-1 + e^d) - (L-1 + e^d)(d+1) + de^d) \\
&= \frac{de^d}{(L-1 + e^d)^2} (1-L) \\
&(\forall L > 1) < 0
\end{aligned} \tag{31}$$

Therefore,  $\text{ELB}(k; L, D)$  is monotonically decreasing with  $d$  and with  $k$ . If the limits of  $\text{ELB}(k; L, D)$  as  $k \rightarrow -\infty$  and  $k \rightarrow +\infty$  exist, then  $\text{ELB}(k; L, D)$  is bounded between these two limits. The limits are given as follows:

$$\begin{aligned}
\lim_{k \rightarrow -\infty} \text{ELB}(k; L, D) &= \lim_{d \rightarrow 0^+} \left( \log(L-1 + e^d) - \frac{de^d}{L-1 + e^d} \right) \\
&= \log(L-1 + 1) - \frac{0}{L-1 + 1} \\
&= \log L
\end{aligned} \tag{32}$$



$$\begin{aligned}
\lim_{k \rightarrow +\infty} \text{ELB}(k; L, D) &= \lim_{d \rightarrow +\infty} \left( \log(L - 1 + e^d) - \frac{de^d}{L - 1 + e^d} \right) \\
&= \lim_{d \rightarrow +\infty} \log e^d - \lim_{d \rightarrow +\infty} \frac{d}{(L - 1)e^{-d} + 1} \\
&= d - d \\
&= 0
\end{aligned} \tag{33}$$

Therefore,  $\text{ELB}(k; L, D)$  is bounded between 0 and  $\log L$ .  $\square$

## C Discussion

### C.1 Difference between the proposed normalization and the other normalization

BatchNorm and LayerNorm are all normalization methods that are widely used in deep learning. They share the same center-and-scale normalization paradigm by first subtracting the mean and then divide by standard deviation. The only difference between them in terms of computation is the dimensions of data used to compute these statistics, as shown in Table S1.

In terms of application, BatchNorm is often used in fully connected layers and convolution layers, while LayerNorm is often used in recurrent neural networks and Transformers. The subtle difference between LayerNorm (theory) and LayerNorm (practice) might be attributed to the fact that the sequence length  $L$  is often variable in Transformers, thus normalization within each token might be more stable. But this will require further investigation to come to a conclusion.

The proposed UnitNorm is a normalization method that is used to normalize the input data to have unit norm, which takes the same dimension for computation as LayerNorm, yet it distinguishes itself from LayerNorm by the fact that it does not subtract the mean and divide by standard deviation. Also, UnitNorm discard the center operation on the normalized output, as it will also cause the problem of token shift (Section 2.1).

### C.2 Feasibility of switching the order of normalization and projection in theoretical analysis

Let  $\mathbf{X} \in \mathbb{R}^{L \times D}$  be a single sequence of token vectors, and the normalization operation is given in the following form:

$$f : \mathbf{X} \mapsto \frac{\mathbf{X} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \equiv \mathbf{X}\mathbf{W} + \mathbf{b} \tag{34}$$

where  $\boldsymbol{\mu}$  and  $\boldsymbol{\sigma}$  are the mean and standard deviation of the input vector  $\mathbf{X}$ , respectively, and  $\mathbf{W} = \boldsymbol{\sigma}^{-1}$  and  $\mathbf{b} = \boldsymbol{\mu}\boldsymbol{\sigma}^{-1}$ . Depending on the normalization method, the mean and standard deviation can be computed over different dimensions.

The projection in the attention mechanism maps the input vectors to query, key and value vectors, and here we only consider the query and key vectors for this discussion, which are computed as follows:

$$\begin{aligned}
\mathbf{Q} &= \mathbf{X}\mathbf{W}_Q + \mathbf{b}_Q \\
\mathbf{K} &= \mathbf{X}\mathbf{W}_K + \mathbf{b}_K
\end{aligned} \tag{35}$$

where  $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{D \times D}$  are the projection matrices and  $\mathbf{b}_Q, \mathbf{b}_K \in \mathbb{R}^D$  are the bias vectors for query and key, respectively. As the normalization and projection are both linear operations, we can combine them into a single linear operation as follows:

$$\begin{aligned}
\mathbf{Y} &= \tilde{\mathbf{X}}\mathbf{W}_Y + \mathbf{b}_Y \\
&= (\mathbf{X}\mathbf{W} + \mathbf{b})\mathbf{W}_Y + \mathbf{b}_Y \\
&= \mathbf{X}(\mathbf{W}\mathbf{W}_Y) + (\mathbf{b}\mathbf{W}_Y + \mathbf{b}_Y)
\end{aligned} \tag{36}$$

for  $Y \in \{Q, K\}$ . Therefore, there must exist some  $\mathbf{W}'$ ,  $\mathbf{b}'$ ,  $\mathbf{W}'_Y$  and  $\mathbf{b}'_Y$  such that:

$$\begin{aligned}
\mathbf{W}'_Y\mathbf{W}' &= \mathbf{W}\mathbf{W}_Y \\
\mathbf{b}_Y\mathbf{W}' + \mathbf{b}' &= \mathbf{b}\mathbf{W}_Y + \mathbf{b}_Y
\end{aligned} \tag{37}$$

Table S1: Computation of the statistics for different normalization methods. Input data  $\mathbf{X} \in \mathbb{R}^{N \times L \times D}$ , where  $N$  is the batch size,  $L$  is the sequence length and  $D$  is the feature dimension.  $\mathbf{X}_{n,l,d}$  denotes the  $d$ -th feature of the  $l$ -th token in the  $n$ -th sequence. Normalization is broadcasted over the same dimension as the statistics and mathematical operations are done element-wise. For BatchNorm, LayerNorm (theory) and LayerNorm (practice),  $\gamma$  and  $\beta$  are optional learnable parameters that will re-scale and re-center the normalized output element-wise, which is enabled by default in the PyTorch’s implementation.

Method	Statistics	Normalization
BatchNorm	$\mu_d = \frac{1}{NL} \sum_{n=1}^N \sum_{l=1}^L \mathbf{X}_{n,l,d}$ $\sigma_d^2 = \frac{1}{NL} \sum_{n=1}^N \sum_{l=1}^L (\mathbf{X}_{n,l,d} - \mu_d)^2$ $\boldsymbol{\mu} = [\mu_1 \quad \mu_2 \quad \cdots \quad \mu_D]^\top \in \mathbb{R}^{1 \times 1 \times D}$ $\boldsymbol{\sigma}^2 = [\sigma_1^2 \quad \sigma_2^2 \quad \cdots \quad \sigma_D^2]^\top \in \mathbb{R}^{1 \times 1 \times D}$	
LayerNorm (theory)	$\mu_n = \frac{1}{LD} \sum_{l=1}^L \sum_{d=1}^D \mathbf{X}_{n,l,d}$ $\sigma_n^2 = \frac{1}{LD} \sum_{l=1}^L \sum_{d=1}^D (\mathbf{X}_{n,l,d} - \mu_n)^2$ $\boldsymbol{\mu} = [\mu_1 \quad \mu_2 \quad \cdots \quad \mu_N]^\top \in \mathbb{R}^{N \times 1 \times 1}$ $\boldsymbol{\sigma}^2 = [\sigma_1^2 \quad \sigma_2^2 \quad \cdots \quad \sigma_N^2]^\top \in \mathbb{R}^{N \times 1 \times 1}$	$\tilde{\mathbf{X}} = \frac{\mathbf{X} - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \varepsilon}}$ $\mathbf{Y} = \tilde{\mathbf{X}} \odot \boldsymbol{\gamma} + \boldsymbol{\beta}$
LayerNorm (practice)	$\mu_{n,l} = \frac{1}{D} \sum_{d=1}^D \mathbf{X}_{n,l,d}$ $\sigma_{n,l}^2 = \frac{1}{D} \sum_{d=1}^D (\mathbf{X}_{n,l,d} - \mu_{n,l})^2$ $\boldsymbol{\mu} = \begin{bmatrix} \mu_{1,1} & \mu_{1,2} & \cdots & \mu_{1,L} \\ \mu_{2,1} & \mu_{2,2} & \cdots & \mu_{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{N,1} & \mu_{N,2} & \cdots & \mu_{N,L} \end{bmatrix}^\top \in \mathbb{R}^{N \times L \times 1}$ $\boldsymbol{\sigma}^2 = \begin{bmatrix} \sigma_{1,1}^2 & \sigma_{1,2}^2 & \cdots & \sigma_{1,L}^2 \\ \sigma_{2,1}^2 & \sigma_{2,2}^2 & \cdots & \sigma_{2,L}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{N,1}^2 & \sigma_{N,2}^2 & \cdots & \sigma_{N,L}^2 \end{bmatrix}^\top \in \mathbb{R}^{N \times L \times 1}$	
RMSNorm	$\ \mathbf{X}\ _{n,l} = \sqrt{\sum_{d=1}^D \mathbf{X}_{n,l,d}^2}$	$\tilde{\mathbf{X}} = \sqrt{D} \frac{\mathbf{X}}{\ \mathbf{X}\ }$ $\mathbf{Y} = \tilde{\mathbf{X}} \odot \boldsymbol{\gamma} + \boldsymbol{\beta}$
UnitNorm	$\ \mathbf{X}\  = \begin{bmatrix} \ \mathbf{X}\ _{1,1} & \ \mathbf{X}\ _{1,2} & \cdots & \ \mathbf{X}\ _{1,L} \\ \ \mathbf{X}\ _{2,1} & \ \mathbf{X}\ _{2,2} & \cdots & \ \mathbf{X}\ _{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ \ \mathbf{X}\ _{N,1} & \ \mathbf{X}\ _{N,2} & \cdots & \ \mathbf{X}\ _{N,L} \end{bmatrix}^\top \in \mathbb{R}^{N \times L \times 1}$	$\tilde{\mathbf{X}} = D^{\frac{1}{2}} \frac{\mathbf{X}}{\ \mathbf{X}\ }$ $\mathbf{Y} = \tilde{\mathbf{X}}$

Therefore, the order of normalization and projection does not affect the theoretical analysis. And in favor of simplicity, we can assume the normalization is performed after the projection.

## D Supplementary Figures

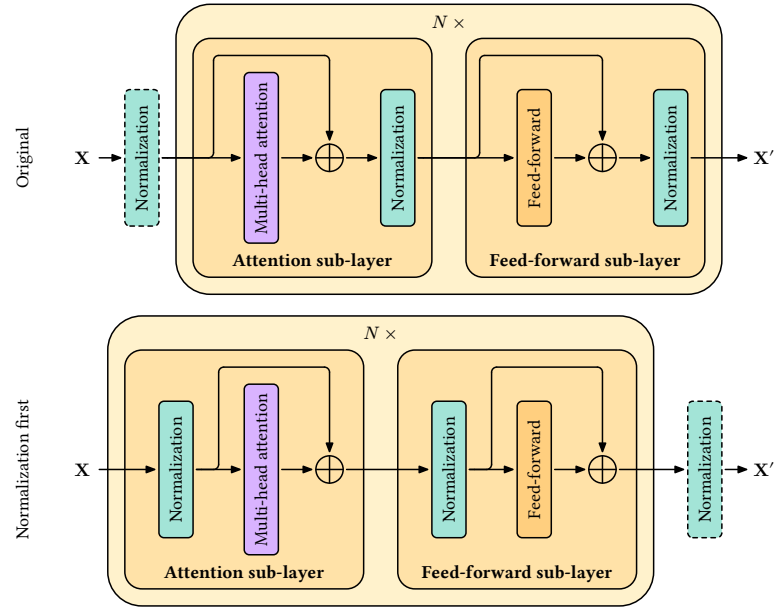


Figure S1: Transformer layer architecture. The original architecture is equivalent to a normalization-first sub-layer design for simpler analysis.

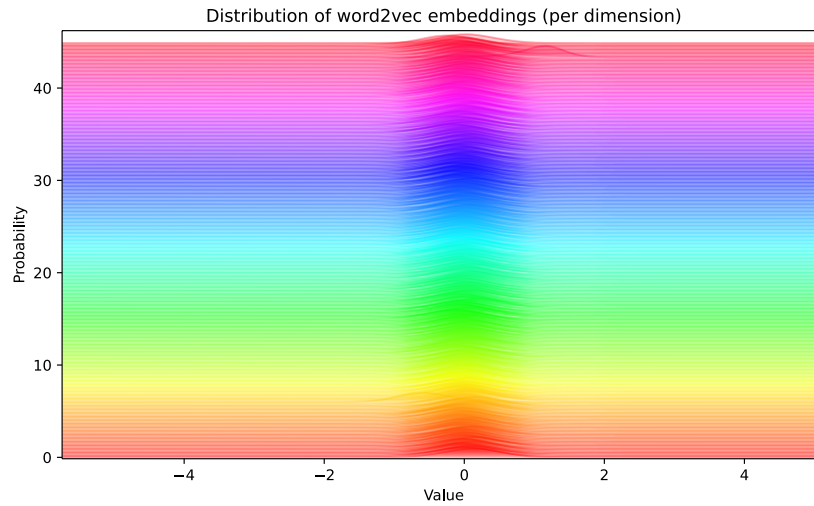


Figure S2: Distribution of values in each dimension of the word2vec embedding. The word2vec embedding is a 300-dimensional vector, and the distribution follows a normal distribution with means mostly around 0.

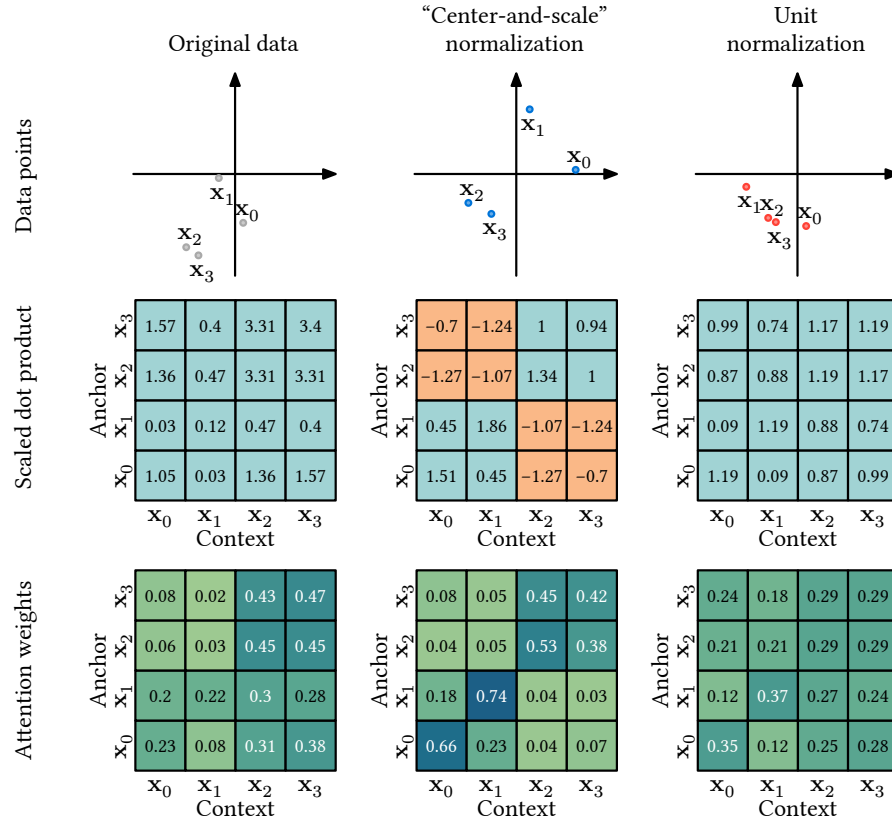


Figure S3: Demonstration of the token shift and attention shift problems using artificial data. The  $x_0$  and  $x_1$  exhibit typical token shift as shifting away from their original quadrants, resulting in sign flip in scaled dot product (marked in orange), and leading to less attention weights distributed to  $x_2$  and  $x_3$  than original. Attention shift and sparse attention problem can also be observed as the maximum attention weight is altered from  $x_2$  and  $x_3$  to nearly solely onto themselves.

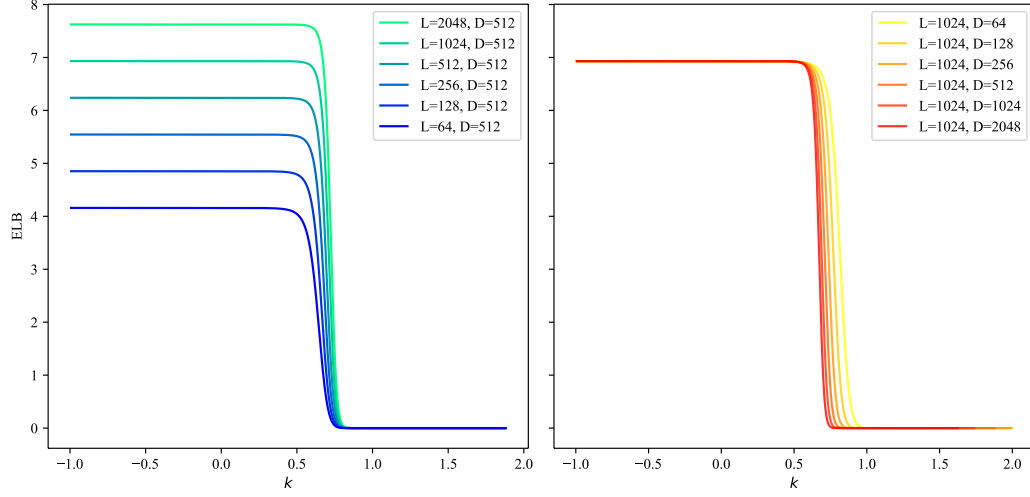


Figure S4: Entropy lower bound (ELB) against  $k$  for different  $L, D$ . The left figure shows the curve for fixed  $D = 512$  and varying  $L$ , and the right figure shows the curve for fixed  $L = 1024$  and varying  $D$ .

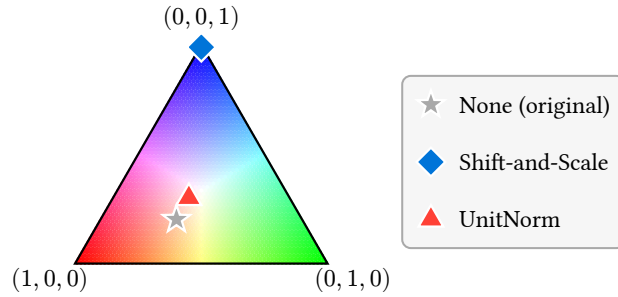


Figure S5: Graphical representation of attention weights showing a simple scenario of 3 tokens. Each corner represents a one-hot distribution (red, blue and green) and the center representing a uniform distribution (white). Gray star, blue diamond, and red triangle mark the attention weights with no normalization, center-and-scale normalization, and UnitNorm, respectively.

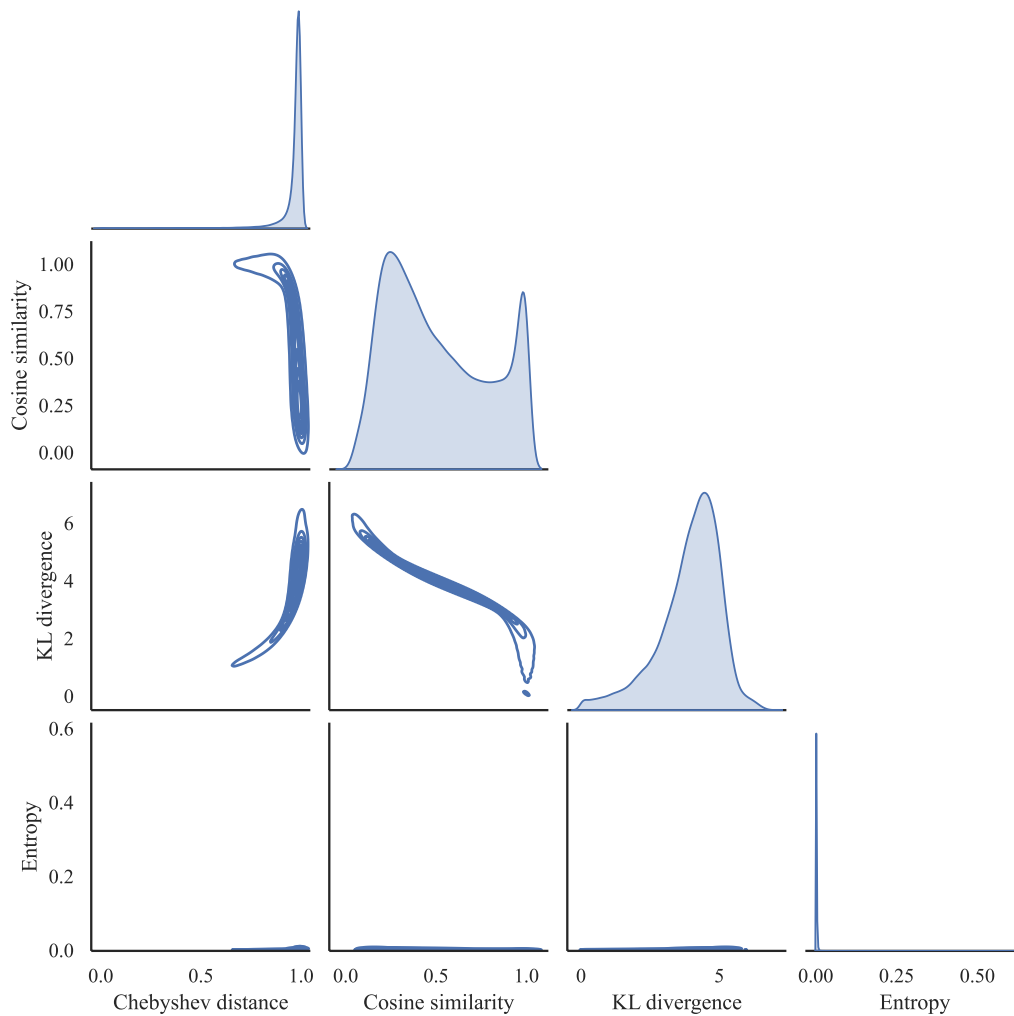


Figure S6: Joint distribution of metrics for LayerNorm (practice). Metrics used are defined as in Table S9.



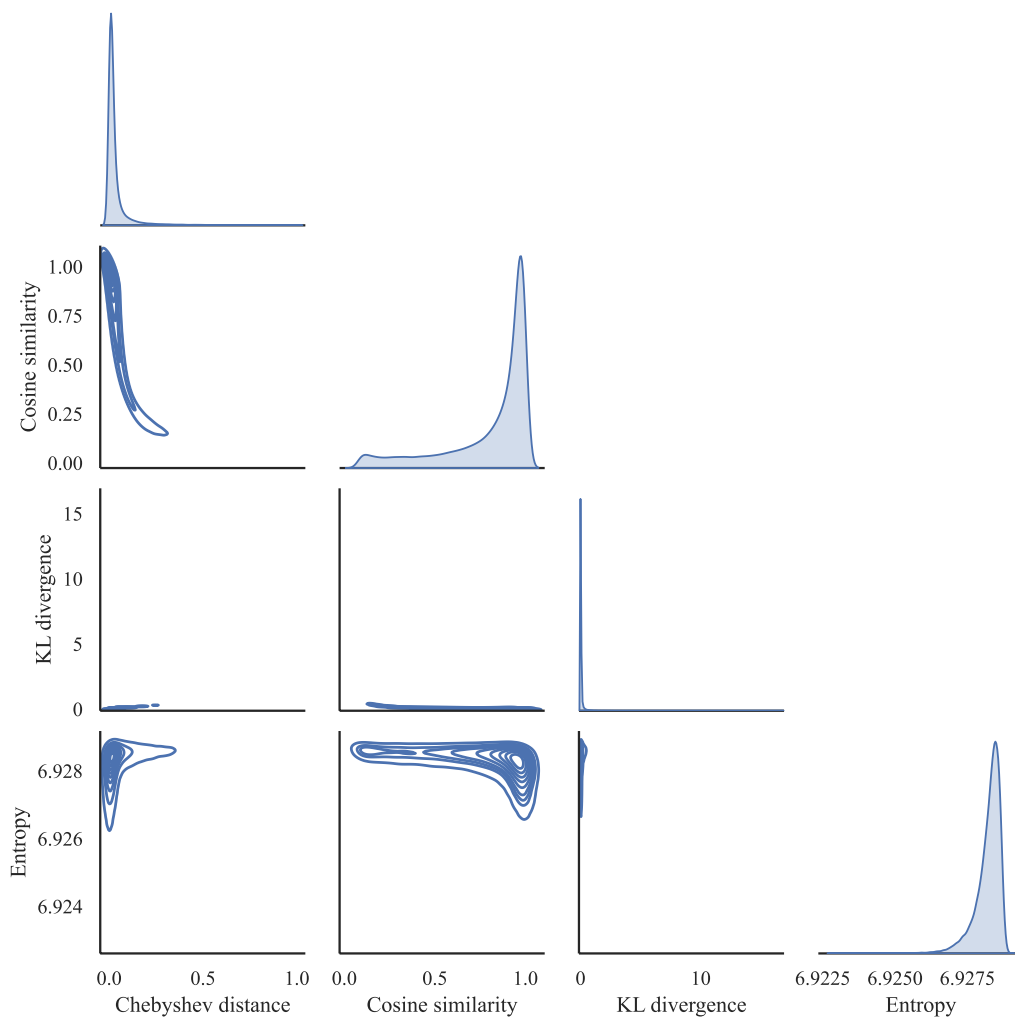


Figure S7: Joint distribution of metrics for UnitNorm. Metrics used are defined as in Table S9.

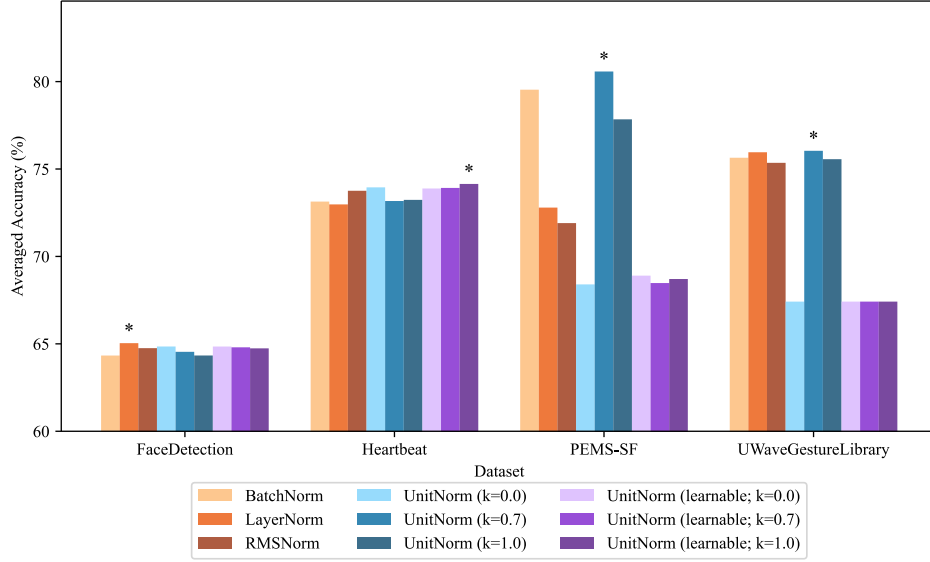


Figure S8: Average rank of normalization methods on the classification tasks. X-axis: Dataset with different normalization, Y-axis: average rank over models. Ranks are computed based on the accuracy of each model on each task with different normalization methods (lower is better). \* indicates the best performing normalization method(s) on each task. UnitNorm and UnitNorm (learnable) outperform other normalization methods on 3 out of 5 datasets, showing its potential in classification tasks.

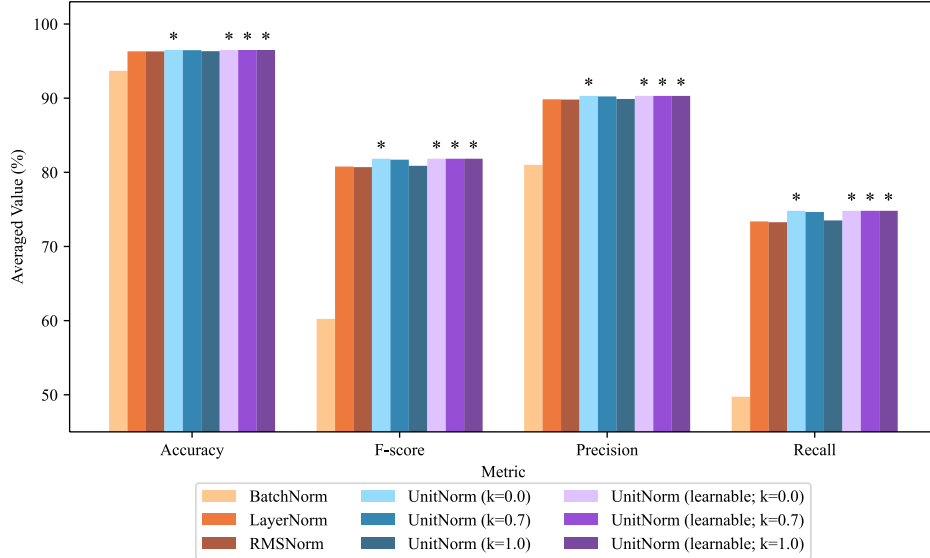


Figure S9: Average rank of normalization methods on the anomaly detection tasks. X-axis: Metrics under different normalization, Y-axis: average rank over models. Ranks are computed based on every metric of each model with different normalization methods (lower is better). \* indicates the best performing normalization method(s) on each metric. UnitNorm and UnitNorm (learnable) show a dominating performance gain over the other normalization methods in all metrics.

## E Supplementary Tables

Table S2: Summary of long term forecasting benchmark settings. The sequence length is the number of historical time steps fed into the encoder, and the label length is the number of time steps fed into the decoder as the ground truth output of the decoder. The prediction length is the number of time steps to be predicted by the decoder.

Datasets	Feature number	Sequence length	Label length	Prediction length	Metrics	License
ETTh1, ETTh2 [22]	7	384	96	96, 192, 384, 720	MSE, MAE	CC BY-ND 4.0
ECL [35]	321					CC BY 4.0
Exchange [36]	8					N/A

Table S3: Summary of classification benchmark settings. All datasets are from UEA Archive [37]. The sequence length is the number of time steps in each sequence fed into the encoder, and the prediction is made on the flattened output of the encoder by a fully connected layer.

Datasets	Feature number	Class number	Sequence length	Metrics	License
FaceDetection	144	5890	96	Accuracy	N/A
Heartbeat	61	204			
PEMS-SF	963	173			
UWaveGestureLibrary	3	320			

Table S4: Summary of anomaly detection benchmark settings. The sequence length is the number of time steps in each sequence fed into the model for reconstruction using MSE as loss. The threshold is determined by the distribution of reconstruction error on the training set, and the metrics are computed on the test set based on this threshold.

Datasets	Feature number	Sequence length	Reconstruction error	Metrics	License
MSL [38]	55	100	MSE	Accuracy, F1-score, Precision, Recall	N/A

Table S5: Summary of compute resources used for the experiments. Depending on the dataset and model, the GPU memory usage varies from 4G to 64G.

CPU	Memory	GPU	GPU Memory
AMD Threadripper 3995WX	512G	4 × NVIDIA RTX A5000	4 × 24G

Table S6: Long term forecasting test losses on different datasets using different models and normalization methods. For each dataset, prediction length, metric and for each model, the best performing normalization method(s) are bolded, and the second best are underlined.

dataset	ECL				ETTh1				ETTh2				Exchange			
	prediction length				prediction length				prediction length				prediction length			
	metric				metric				metric				metric			
BatchNorm	Crossformer	0.408	0.278	0.435	0.311	<b>0.246</b>	0.310	0.184	0.357	0.198	<b>0.144</b>	0.409	0.285	0.448	<b>0.322</b>	0.265
	FEDformer	0.409	0.285	0.448	<b>0.322</b>	0.265	<b>0.321</b>	0.191	<b>0.364</b>	<b>0.211</b>	0.162	<b>0.463</b>	0.300	0.455	<b>0.342</b>	<b>0.290</b>
	Informer	<b>0.407</b>	0.207	0.376	<b>0.234</b>	<b>0.189</b>	0.510	0.332	0.478	0.372	<b>0.338</b>	0.461	0.248	0.417	0.270	0.255
	PatchTST	0.814	<b>0.395</b>	<b>0.661</b>	0.422	<b>0.426</b>	1.008	<b>0.374</b>	<b>0.790</b>	0.389	0.417	0.805	0.429	0.792	0.449	0.507
	Transformer	0.805	0.429	0.792	0.449	0.507	1.014	0.426	1.015	0.433	0.531	0.886	0.462	0.886	0.477	0.620
	Transformer	1.161	0.478	1.230	0.483	0.684	0.819	0.497	0.940	0.507	0.707	1.054	0.506	1.357	0.516	0.834
LayerNorm	Crossformer	1.093	0.346	1.414	0.395	<b>0.557</b>	1.852	0.292	3.319	0.352	<b>0.563</b>	1.258	0.399	2.105	<b>0.434</b>	0.878
	FEDformer	2.488	0.376	6.878	<b>0.423</b>	1.355	1.603	0.435	2.215	<b>0.473</b>	0.961	3.874	0.421	6.966	<b>0.469</b>	1.626
	Informer	1.833	0.453	2.307	0.486	1.491	4.782	0.435	8.194	<b>0.479</b>	3.213	0.608	0.204	<b>0.672</b>	<b>0.297</b>	<b>0.265</b>
	PatchTST	0.665	0.088	<b>0.770</b>	<b>0.169</b>	<b>0.122</b>	0.921	0.300	0.861	<b>0.383</b>	<b>0.445</b>	1.458	0.180	1.308	<b>0.276</b>	<b>0.315</b>
	Transformer	1.242	0.419	1.367	<b>0.487</b>	<b>0.646</b>	2.578	0.336	3.221	0.438	<b>0.658</b>	1.444	<b>0.703</b>	1.602	0.823	<b>0.846</b>
	Transformer	3.069	0.874	4.284	1.151	<b>1.086</b>	0.416	<b>0.270</b>	<b>0.410</b>	<b>0.309</b>	<b>0.249</b>	0.320	<b>0.181</b>	<b>0.325</b>	<b>0.195</b>	<b>0.148</b>
RMSNorm	Crossformer	0.436	<b>0.276</b>	<b>0.436</b>	<b>0.324</b>	<b>0.262</b>	0.436	<b>0.276</b>	<b>0.436</b>	<b>0.324</b>	<b>0.262</b>	0.358	<b>0.187</b>	<b>0.355</b>	<b>0.211</b>	<b>0.162</b>
	FEDformer	0.499	<b>0.291</b>	<b>0.443</b>	<b>0.347</b>	0.291	0.458	<b>0.204</b>	<b>0.364</b>	0.236	0.192	0.542	<b>0.325</b>	<b>0.455</b>	0.366	0.344
	Informer	0.514	<b>0.246</b>	<b>0.391</b>	0.262	0.266	0.514	<b>0.246</b>	<b>0.391</b>	0.262	0.266	0.748	<b>0.396</b>	0.720	<b>0.420</b>	0.439
	PatchTST	0.884	0.377	0.877	<b>0.379</b>	0.419	1.003	0.426	1.013	<b>0.420</b>	0.511	0.802	0.426	0.792	<b>0.444</b>	0.500
	Transformer	0.834	<b>0.447</b>	0.842	<b>0.471</b>	<b>0.572</b>	1.081	<b>0.460</b>	1.148	0.464	<b>0.621</b>	0.833	0.492	<b>0.855</b>	0.507	0.667
	Transformer	1.078	0.511	1.174	0.507	0.765	1.188	0.345	1.502	0.395	0.721	2.205	0.297	3.574	0.351	1.014
UnitNorm (k=0.0)	Crossformer	1.820	0.399	2.056	0.438	0.909	5.161	0.380	6.156	0.425	1.487	1.699	0.431	1.864	0.475	1.331
	FEDformer	4.549	<b>0.417</b>	4.990	<b>0.469</b>	2.686	1.390	<b>0.447</b>	1.652	0.488	1.506	2.908	0.429	3.943	0.481	3.057
	Informer	0.589	0.209	0.733	0.297	0.378	0.581	0.091	<b>0.830</b>	0.169	0.274	0.786	0.301	<b>0.836</b>	<b>0.383</b>	0.531
	PatchTST	1.047	0.180	<b>1.081</b>	<b>0.276</b>	0.502	1.051	0.411	<b>1.073</b>	0.488	0.890	1.745	0.324	<b>1.817</b>	0.440	1.288
	Transformer	1.212	0.713	1.409	0.824	1.047	2.297	0.901	2.967	1.153	1.701	2.354	0.904	2.875	1.153	<b>1.695</b>
	Transformer	2.297	0.901	2.967	1.153	1.701	0.403	0.271	<b>0.417</b>	0.309	0.249	0.303	0.182	<b>0.334</b>	0.195	0.150
UnitNorm (k=0.0)	Crossformer	0.427	0.277	<b>0.444</b>	0.324	<b>0.262</b>	0.427	0.277	<b>0.444</b>	0.324	<b>0.262</b>	0.343	<b>0.188</b>	0.366	0.211	<b>0.162</b>
	FEDformer	0.490	0.293	0.448	0.346	0.292	0.441	0.204	0.370	0.235	0.196	0.539	<b>0.327</b>	<b>0.460</b>	<b>0.365</b>	0.344
	Informer	0.511	<b>0.247</b>	<b>0.399</b>	<b>0.261</b>	0.264	0.511	<b>0.247</b>	<b>0.399</b>	<b>0.261</b>	0.264	0.745	0.396	<b>0.709</b>	0.421	<b>0.431</b>
	PatchTST	0.882	0.377	<b>0.864</b>	0.380	<b>0.409</b>	0.802	<b>0.425</b>	0.781	<b>0.445</b>	<b>0.495</b>	0.984	<b>0.421</b>	0.995	<b>0.420</b>	0.508
	Transformer	0.870	0.448	<b>0.822</b>	<b>0.471</b>	<b>0.597</b>	1.123	0.463	<b>1.112</b>	<b>0.462</b>	0.658	1.030	0.520	<b>1.172</b>	0.505	0.836
	Transformer	1.211	0.345	1.573	<b>0.393</b>	0.709	2.276	0.297	4.058	<b>0.350</b>	0.963	1.815	0.399	2.126	0.437	0.904
UnitNorm (k=0.0)	Crossformer	5.030	0.379	6.495	0.425	1.451	1.806	0.431	1.941	0.475	1.116	5.043	0.418	5.358	0.469	1.985
	FEDformer	1.480	0.448	1.663	0.488	1.525	3.239	0.429	3.887	0.481	3.315	1.480	0.448	1.663	0.488	1.525
	Informer	0.580	0.209	<b>0.730</b>	0.297	<b>0.371</b>	0.580	0.209	<b>0.730</b>	0.297	<b>0.371</b>	0.580	0.209	<b>0.730</b>	0.297	<b>0.371</b>
	PatchTST	0.569	0.091	0.841	0.169	<b>0.261</b>	0.569	0.091	0.841	0.169	<b>0.261</b>	0.569	0.091	0.841	0.169	<b>0.261</b>
	Transformer	0.783	0.300	0.866	0.383	<b>0.522</b>	0.783	0.300	0.866	0.383	<b>0.522</b>	0.783	0.300	0.866	0.383	<b>0.522</b>
	Transformer	1.046	0.179	1.158	0.277	<b>0.485</b>	1.046	0.179	1.158	0.277	<b>0.485</b>	1.046	0.179	1.158	0.277	<b>0.485</b>
UnitNorm (k=0.0)	Crossformer	1.045	0.411	1.077	0.488	<b>0.882</b>	1.045	0.411	1.077	0.488	<b>0.882</b>	1.045	0.411	1.077	0.488	<b>0.882</b>
	FEDformer	1.742	0.322	1.840	0.440	<b>1.262</b>	1.742	0.322	1.840	0.440	<b>1.262</b>	1.742	0.322	1.840	0.440	<b>1.262</b>
	Informer	1.246	0.715	1.386	0.824	<b>1.045</b>	1.246	0.715	1.386	0.824	<b>1.045</b>	1.246	0.715	1.386	0.824	<b>1.045</b>
	PatchTST	2.354	0.904	2.875	1.153	<b>1.695</b>	2.354	0.904	2.875	1.153	<b>1.695</b>	2.354	0.904	2.875	1.153	<b>1.695</b>
	Transformer	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711
	Transformer	1.023	0.206	1.016	0.298	0.688	1.535	0.088	1.497	0.170	0.913	1.121	<b>0.299</b>	1.066	0.385	0.772
UnitNorm (k=0.0)	Crossformer	1.848	<b>0.178</b>	1.646	0.279	1.057	1.203	0.418	1.186	0.487	1.177	2.122	0.331	2.042	0.438	2.187
	FEDformer	1.111	0.707	1.200	<b>0.820</b>	1.338	1.111	0.707	1.200	<b>0.820</b>	1.338	1.111	0.707	1.200	<b>0.820</b>	1.338
	Informer	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711
	PatchTST	2.354	0.904	2.875	1.153	<b>1.695</b>	2.354	0.904	2.875	1.153	<b>1.695</b>	2.354	0.904	2.875	1.153	<b>1.695</b>
	Transformer	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711	1.717	0.876	2.155	<b>1.145</b>	2.711
	Transformer	1.023	0.206	1.016	0.298	0.688	1.535	0.088	1.497	0.170	0.913	1.121	<b>0.299</b>	1.066	0.385	0.772

Continued on next page

	dataset	ECL				ETTh1				ETTh2				Exchange			
		prediction length				prediction length				prediction length				prediction length			
		metric				metric				metric				metric			
UnitNorm (k=0.5)	Crossformer	0.420	0.274	0.463	0.320	0.257	0.420	0.274	0.463	0.320	0.257	0.420	0.274	0.463	0.320	0.257	0.420
	FEDformer	0.331	0.183	0.401	0.204	0.156	0.331	0.183	0.401	0.204	0.156	0.331	0.183	0.401	0.204	0.156	0.331
	Informer	0.446	0.280	0.492	0.335	0.274	0.446	0.280	0.492	0.335	0.274	0.446	0.280	0.492	0.335	0.274	0.446
	PatchTST	0.371	0.189	0.434	0.222	0.176	0.371	0.189	0.434	0.222	0.176	0.371	0.189	0.434	0.222	0.176	0.371
	Transformer	0.504	0.296	0.501	0.354	0.299	0.504	0.296	0.501	0.354	0.299	0.504	0.296	0.501	0.354	0.299	0.504
	Transformer	0.460	0.205	0.441	0.242	0.204	0.460	0.205	0.441	0.242	0.204	0.460	0.205	0.441	0.242	0.204	0.460
UnitNorm (k=1.0)	Crossformer	0.508	0.328	0.558	0.385	0.339	0.508	0.328	0.558	0.385	0.339	0.508	0.328	0.558	0.385	0.339	0.508
	FEDformer	0.459	0.247	0.543	0.283	0.251	0.459	0.247	0.543	0.283	0.251	0.459	0.247	0.543	0.283	0.251	0.459
	Informer	0.662	0.397	0.748	0.428	0.436	0.662	0.397	0.748	0.428	0.436	0.662	0.397	0.748	0.428	0.436	0.662
	PatchTST	0.751	0.381	0.963	0.386	0.408	0.751	0.381	0.963	0.386	0.408	0.751	0.381	0.963	0.386	0.408	0.751
	Transformer	0.736	0.428	0.751	0.448	0.498	0.736	0.428	0.751	0.448	0.498	0.736	0.428	0.751	0.448	0.498	0.736
	Transformer	0.917	0.427	0.981	0.423	0.499	0.917	0.427	0.981	0.423	0.499	0.917	0.427	0.981	0.423	0.499	0.917
UnitNorm (learnable; k=0.0)	Crossformer	0.821	0.451	0.800	0.471	0.600	0.821	0.451	0.800	0.471	0.600	0.821	0.451	0.800	0.471	0.600	0.821
	FEDformer	1.073	0.466	1.061	0.462	0.643	1.073	0.466	1.061	0.462	0.643	1.073	0.466	1.061	0.462	0.643	1.073
	Informer	0.874	0.484	0.871	0.502	0.673	0.874	0.484	0.871	0.502	0.673	0.874	0.484	0.871	0.502	0.673	0.874
	PatchTST	1.186	0.493	1.190	0.497	0.759	1.186	0.493	1.190	0.497	0.759	1.186	0.493	1.190	0.497	0.759	1.186
	Transformer	0.914	0.343	1.099	0.401	0.657	0.914	0.343	1.099	0.401	0.657	0.914	0.343	1.099	0.401	0.657	0.914
	Transformer	1.243	0.291	1.884	0.356	0.863	1.243	0.291	1.884	0.356	0.863	1.243	0.291	1.884	0.356	0.863	1.243
UnitNorm (learnable; k=0.7)	Crossformer	1.391	0.393	1.410	0.438	0.832	1.391	0.393	1.410	0.438	0.832	1.391	0.393	1.410	0.438	0.832	1.391
	FEDformer	3.050	0.365	3.014	0.427	1.326	3.050	0.365	3.014	0.427	1.326	3.050	0.365	3.014	0.427	1.326	3.050
	Informer	1.356	0.433	1.343	0.480	1.120	1.356	0.433	1.343	0.480	1.120	1.356	0.433	1.343	0.480	1.120	1.356
	PatchTST	2.854	0.420	2.696	0.475	2.004	2.854	0.420	2.696	0.475	2.004	2.854	0.420	2.696	0.475	2.004	2.854
	Transformer	1.379	0.449	1.448	0.483	1.369	1.379	0.449	1.448	0.483	1.369	1.379	0.449	1.448	0.483	1.369	1.379
	Transformer	2.669	0.431	2.927	0.474	2.851	2.669	0.431	2.927	0.474	2.851	2.669	0.431	2.927	0.474	2.851	2.669
UnitNorm (learnable; k=1.0)	Crossformer	0.782	0.202	0.838	0.298	0.472	0.782	0.202	0.838	0.298	0.472	0.782	0.202	0.838	0.298	0.472	0.782
	FEDformer	0.965	0.085	1.068	0.170	0.430	0.965	0.085	1.068	0.170	0.430	0.965	0.085	1.068	0.170	0.430	0.965
	Informer	0.915	0.300	0.983	0.385	0.644	0.915	0.300	0.983	0.385	0.644	0.915	0.300	0.983	0.385	0.644	0.915
	PatchTST	1.328	0.178	1.457	0.279	0.702	1.328	0.178	1.457	0.279	0.702	1.328	0.178	1.457	0.279	0.702	1.328
	Transformer	0.993	0.411	1.028	0.487	0.964	0.993	0.411	1.028	0.487	0.964	0.993	0.411	1.028	0.487	0.964	0.993
	Transformer	1.484	0.321	1.595	0.438	1.526	1.484	0.321	1.595	0.438	1.526	1.484	0.321	1.595	0.438	1.526	1.484
UnitNorm (learnable; k=0.5)	Crossformer	0.911	0.698	1.023	0.820	1.128	0.911	0.698	1.023	0.820	1.128	0.911	0.698	1.023	0.820	1.128	0.911
	FEDformer	1.196	0.861	1.608	1.145	1.971	1.196	0.861	1.608	1.145	1.971	1.196	0.861	1.608	1.145	1.971	1.196
	Informer	0.782	0.202	0.838	0.298	0.472	0.782	0.202	0.838	0.298	0.472	0.782	0.202	0.838	0.298	0.472	0.782
	PatchTST	0.965	0.085	1.068	0.170	0.430	0.965	0.085	1.068	0.170	0.430	0.965	0.085	1.068	0.170	0.430	0.965
	Transformer	0.915	0.300	0.983	0.385	0.644	0.915	0.300	0.983	0.385	0.644	0.915	0.300	0.983	0.385	0.644	0.915
	Transformer	1.328	0.178	1.457	0.279	0.702	1.328	0.178	1.457	0.279	0.702	1.328	0.178	1.457	0.279	0.702	1.328
UnitNorm (learnable; k=0.0)	Crossformer	0.580	0.209	0.750	0.297	0.373	0.580	0.209	0.750	0.297	0.373	0.580	0.209	0.750	0.297	0.373	0.580
	FEDformer	0.569	0.091	0.877	0.169	0.263	0.569	0.091	0.877	0.169	0.263	0.569	0.091	0.877	0.169	0.263	0.569
	Informer	0.783	0.300	0.860	0.383	0.522	0.783	0.300	0.860	0.383	0.522	0.783	0.300	0.860	0.383	0.522	0.783
	PatchTST	1.046	0.179	1.137	0.277	0.486	1.046	0.179	1.137	0.277	0.486	1.046	0.179	1.137	0.277	0.486	1.046
	Transformer	1.046	0.411	1.077	0.488	0.886	1.046	0.411	1.077	0.488	0.886	1.046	0.411	1.077	0.488	0.886	1.046
	Transformer	1.742	0.322	1.841	0.440	1.270	1.742	0.322	1.841	0.440	1.270	1.742	0.322	1.841	0.440	1.270	1.742
UnitNorm (learnable; k=0.7)	Crossformer	1.246	0.715	1.396	0.824	1.048	1.246	0.715	1.396	0.824	1.048	1.246	0.715	1.396	0.824	1.048	1.246
	FEDformer	2.354	0.904	2.936	1.153	1.702	2.354	0.904	2.936	1.153	1.702	2.354	0.904	2.936	1.153	1.702	2.354
	Informer	0.580	0.209	0.750	0.297	0.373	0.580	0.209	0.750	0.297	0.373	0.580	0.209	0.750	0.297	0.373	0.580
	PatchTST	0.569	0.091	0.877	0.169	0.263	0.569	0.091	0.877	0.169	0.263	0.569	0.091	0.877	0.169	0.263	0.569
	Transformer	0.783	0.300	0.860	0.383	0.522	0.783	0.300	0.860	0.383	0.522	0.783	0.300	0.860	0.383	0.522	0.783
	Transformer	1.046	0.179	1.137	0.277	0.486	1.046	0.179	1.137	0.277	0.486	1.046	0.179	1.137	0.277	0.486	1.046
UnitNorm (learnable; k=1.0)	Crossformer	0.985	0.341	1.093	0.396	0.568	0.985	0.341	1.093	0.396	0.568	0.985	0.341	1.093	0.396	0.568	0.985
	FEDformer	1.495	0.290	1.861	0.355	0.673	1.495	0.290	1.861	0.355	0.673	1.495	0.290	1.861	0.355	0.673	1.495
	Informer	1.160	0.394	1.251	0.445	0.797	1.160	0.394	1.251	0.445	0.797	1.160	0.394	1.251	0.445	0.797	1.160
	PatchTST	2.112	0.369	2.407	0.434	1.268	2.112	0.369	2.407	0.434	1.268	2.112	0.369	2.407	0.434	1.268	2.112
	Transformer	1.229	0.429	1.268	0.480	0.834	1.229	0.429	1.268	0.480	0.834	1.229	0.429	1.268	0.480	0.834	1.229
	Transformer	2.378	0.417	2.314	0.475	1.379	2.378	0.417	2.314	0.475	1.379	2.378	0.417	2.314	0.475	1.379	2.378
UnitNorm (learnable; k=0.5)	Crossformer	1.290	0.449	1.305	0.485	1.041	1.290	0.449	1.305	0.485	1.041	1.290	0.449	1.305	0.485	1.041	1.290
	FEDformer	2.399	0.429	2.436	0.480	1.914	2.399	0.429	2.436	0.480	1.914	2.399	0.429	2.436	0.480	1.914	2.399
	Informer	1.023	0.202	1.031	0.293	0.688	1.023	0.202	1.031	0.293	0.688	1.023	0.202	1.031	0.293	0.688	1.023
	PatchTST	1.535	0.084	1.527	0.164	0.913	1.535	0.084	1.527	0.164	0.913	1.535	0.084	1.527	0.164	0.913	1.535
	Transformer	1.121	0.300	1.075	0.385	0.772	1.121	0.300	1.075	0.385	0.772	1.121	0.300	1.075	0.385	0.772	1.121
	Transformer	1.848	0.180	1.725	0.280	1.057	1.848	0.180	1.725	0.280	1.057	1.848	0.180	1.725	0.280	1.057	1.848
UnitNorm (learnable; k=0.0)	Crossformer	1.203	0.418	1.186	0.487	1.177	1.203	0.418	1.186	0.487	1.177	1.203	0.418	1.186	0.487	1.177	1.203
	FEDformer	2.122	0.331	2.042	0.438	2.187	2.122	0.331	2.042	0.438	2.187	2.122	0.331	2.042	0.438	2.187	2.122
	Informer	1.111	0.703	1.163	0.830	1.338	1.111	0.703	1.163	0.830	1.338	1.111	0.703	1.163	0.830	1.338	1.111
	PatchTST	1.717	0.869	2.031	1.167	2.711	1.717	0.869	2.031	1.167	2.711	1.717	0.869	2.031	1.167	2.711	

Table S7: Classification accuracies of different datasets using different models and normalization methods. For each dataset and for each model, the best performing normalization method(s) are bolded, and the second best are underlined.

	dataset	Face Detection	Heartbeat	PEMS-SF	UWave Gesture Library
BatchNorm	Crossformer	50.435	<b>75.122</b>	<b>68.401</b>	<b>83.438</b>
	FEDformer	68.275	<u>73.984</u>	78.035	47.812
	Informer	68.606	73.984	<b>87.476</b>	82.083
	PatchTST	65.683	66.992	79.576	81.354
	Transformer	68.663	<u>75.610</u>	<u>84.200</u>	83.542
LayerNorm	Crossformer	<b>52.176</b>	73.008	26.397	<u>82.708</u>
	FEDformer	<u>68.861</u>	73.659	84.393	<u>48.438</u>
	Informer	68.076	73.821	<u>85.742</u>	81.979
	PatchTST	67.329	69.919	<b>84.971</b>	81.562
	Transformer	<u>68.757</u>	74.472	82.466	<u>85.104</u>
RMSNorm	Crossformer	<u>51.693</u>	<u>73.659</u>	23.699	81.042
	FEDformer	68.000	<u>72.846</u>	<u>85.164</u>	47.708
	Informer	68.275	<u>75.447</u>	84.586	83.125
	PatchTST	66.648	70.894	82.852	80.729
	Transformer	<b>69.154</b>	<b>75.935</b>	83.237	84.167
UnitNorm (k=0.0)	Crossformer	50.000	72.195	16.763	29.167
	FEDformer	<b>69.041</b>	<b>74.146</b>	83.430	<b>53.333</b>
	Informer	<b>69.088</b>	75.285	78.035	<b>85.000</b>
	PatchTST	<u>67.546</u>	72.195	81.118	<u>82.396</u>
	Transformer	<u>68.568</u>	<b>75.935</b>	82.659	<b>87.188</b>
UnitNorm (k=0.7)	Crossformer	50.236	73.008	<u>65.896</u>	81.667
	FEDformer	68.067	72.846	84.586	47.083
	Informer	68.142	72.846	83.237	<u>83.958</u>
	PatchTST	<b>67.641</b>	<u>72.358</u>	<b>84.971</b>	<b>82.604</b>
	Transformer	68.634	74.797	<b>84.200</b>	84.896
UnitNorm (k=1.0)	Crossformer	50.019	72.195	56.455	80.521
	FEDformer	67.357	<u>73.984</u>	<b>85.356</b>	48.125
	Informer	68.492	73.984	84.008	83.646
	PatchTST	67.452	72.033	<u>83.044</u>	81.146
	Transformer	68.350	73.984	80.347	84.375
UnitNorm (learnable; k=0.0)	Crossformer	50.000	72.195	16.763	29.167
	FEDformer	<b>69.041</b>	<b>74.146</b>	83.430	<b>53.333</b>
	Informer	<u>69.079</u>	74.959	80.539	<b>85.000</b>
	PatchTST	<u>67.546</u>	72.195	81.118	<u>82.396</u>
	Transformer	68.568	<b>75.935</b>	82.659	<b>87.188</b>
UnitNorm (learnable; k=0.7)	Crossformer	50.000	72.195	16.763	29.167
	FEDformer	<b>69.041</b>	<b>74.146</b>	83.430	<b>53.333</b>
	Informer	68.852	75.122	78.420	<b>85.000</b>
	PatchTST	<u>67.546</u>	72.195	81.118	<u>82.396</u>
	Transformer	68.568	<b>75.935</b>	82.659	<b>87.188</b>
UnitNorm (learnable; k=1.0)	Crossformer	50.000	72.195	16.763	29.167
	FEDformer	<b>69.041</b>	<b>74.146</b>	83.430	<b>53.333</b>
	Informer	68.558	<b>75.610</b>	79.576	<b>85.000</b>
	PatchTST	<u>67.546</u>	<b>72.846</b>	81.118	<u>82.396</u>
	Transformer	68.568	<b>75.935</b>	82.659	<b>87.188</b>



Table S8: Anomaly detection accuracies of MSL dataset using different models and normalization methods. For each metric and for each model, the best performing normalization method(s) are bolded, and the second best are underlined.

	metric	Accuracy	F-score	Precision	Recall
BatchNorm	Crossformer	93.507	59.110	81.410	47.903
	FEDformer	95.543	75.820	88.630	66.240
	Informer	93.040	56.927	81.760	43.733
	PatchTST	95.947	78.613	88.603	70.650
	Transformer	90.417	30.680	64.623	20.123
LayerNorm	Crossformer	96.313	80.640	90.330	72.823
	FEDformer	<b>96.603</b>	<b>82.427</b>	<b>90.697</b>	<b>75.537</b>
	Informer	96.390	81.193	90.120	73.877
	PatchTST	95.950	78.727	88.347	70.993
	Transformer	96.333	80.910	89.740	73.660
RMSNorm	Crossformer	96.307	80.613	90.323	72.790
	FEDformer	96.573	82.263	90.647	75.300
	Informer	96.373	81.067	90.097	73.680
	PatchTST	95.940	78.670	88.307	70.927
	Transformer	96.330	80.883	89.677	73.657
UnitNorm (k=0.0)	Crossformer	<b>96.533</b>	<b>82.000</b>	<u>90.610</u>	<b>74.880</b>
	FEDformer	96.547	82.097	90.653	75.013
	Informer	<u>96.543</u>	<u>82.067</u>	<u>90.640</u>	<u>74.973</u>
	PatchTST	<b>96.317</b>	<b>80.943</b>	<b>88.990</b>	<b>74.227</b>
	Transformer	<u>96.540</u>	<b>82.060</b>	<b>90.637</b>	<b>74.960</b>
UnitNorm (k=0.7)	Crossformer	<u>96.523</u>	<u>81.943</u>	<b>90.617</b>	<u>74.783</u>
	FEDformer	96.537	82.040	90.620	74.943
	Informer	<b>96.557</b>	<b>82.147</b>	<b>90.643</b>	<b>75.103</b>
	PatchTST	<u>96.213</u>	<u>80.360</u>	<u>88.713</u>	<u>73.440</u>
	Transformer	<b>96.540</b>	<b>82.043</b>	<u>90.580</u>	<u>74.977</u>
UnitNorm (k=1.0)	Crossformer	96.347	80.857	90.380	73.143
	FEDformer	<u>96.587</u>	<u>82.350</u>	<u>90.670</u>	<u>75.433</u>
	Informer	96.470	81.653	90.400	74.450
	PatchTST	95.933	78.607	88.273	70.847
	Transformer	96.337	<u>80.933</u>	89.753	73.687
UnitNorm (learnable; k=0.0)	Crossformer	<b>96.533</b>	<b>82.000</b>	<u>90.610</u>	<b>74.880</b>
	FEDformer	96.547	82.097	90.653	75.013
	Informer	<u>96.543</u>	<u>82.067</u>	<u>90.640</u>	<u>74.973</u>
	PatchTST	<b>96.317</b>	<b>80.943</b>	<b>88.990</b>	<b>74.227</b>
	Transformer	<u>96.540</u>	<b>82.060</b>	<b>90.637</b>	<b>74.960</b>
UnitNorm (learnable; k=0.7)	Crossformer	<b>96.533</b>	<b>82.000</b>	<u>90.610</u>	<b>74.880</b>
	FEDformer	96.547	82.097	90.653	75.013
	Informer	<u>96.543</u>	<u>82.067</u>	<u>90.640</u>	<u>74.973</u>
	PatchTST	<b>96.317</b>	<b>80.943</b>	<b>88.990</b>	<b>74.227</b>
	Transformer	<u>96.540</u>	<b>82.060</b>	<b>90.637</b>	<b>74.960</b>
UnitNorm (learnable; k=1.0)	Crossformer	<b>96.533</b>	<b>82.000</b>	<u>90.610</u>	<b>74.880</b>
	FEDformer	96.547	82.097	90.653	75.013
	Informer	<u>96.543</u>	<u>82.067</u>	<u>90.640</u>	<u>74.973</u>
	PatchTST	<b>96.317</b>	<b>80.943</b>	<b>88.990</b>	<b>74.227</b>
	Transformer	<u>96.540</u>	<b>82.060</b>	<b>90.637</b>	<b>74.960</b>

Table S9: Metrics used for characterizing the distribution of attention scores from the original and normalized data, denoted as  $\mathbf{A}_{n,i}$  and  $\tilde{\mathbf{A}}_{n,i}$ , respectively, for the  $i$ -th sample in the  $n$ -th batch.

Metric	Definition	Evaluation
Chebyshev distance	$D_{\text{Chebyshev}}(\mathbf{A}_{n,i}, \tilde{\mathbf{A}}_{n,i}) = \max_{j=1}^L \left  \mathbf{A}_{n,i,j} - \tilde{\mathbf{A}}_{n,i,j} \right $	Lower is Better
Cosine similarity	$D_{\text{Cosine}}(\mathbf{A}_{n,i}, \tilde{\mathbf{A}}_{n,i}) = \frac{\mathbf{A}_{n,i}^\top \tilde{\mathbf{A}}_{n,i}}{\ \mathbf{A}_{n,i}\  \ \tilde{\mathbf{A}}_{n,i}\ }$	Higher is Better
KL divergence	$D_{\text{KL}}(\mathbf{A}_{n,i} \parallel \tilde{\mathbf{A}}_{n,i}) = \sum_{j=1}^L \mathbf{A}_{n,i,j} \left( \log \mathbf{A}_{n,i,j} - \log \tilde{\mathbf{A}}_{n,i,j} \right)$	Lower is Better
Entropy	$E(\tilde{\mathbf{A}}_{n,i}) = - \sum_{j=1}^L \tilde{\mathbf{A}}_{n,i,j} \log \tilde{\mathbf{A}}_{n,i,j}$	Higher is Better

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: We identify the critical issues of token shift, attention shift and re-evaluate the attention pattern problem in Section 2, and provide experimental results in Section 4 to demonstrate the effectiveness of UnitNorm in addressing these challenges.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We provide the limitations of this study in Section 6.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: Assumptions and proofs are provided in Section 3, Appendix C.2 and Appendix B

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The design of UnitNorm is fully disclosed in Section 3, and related code and data are provided in <https://anonymous.4open.science/r/UnitNorm-5B84>.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: See code in <https://anonymous.4open.science/r/UnitNorm-5B84>.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Important training details are provided in Appendix E. Others remain the same as in [24].

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Error bars are not reported, as the mean value are calculated over different model architectures given the same hyperparameter and the same normalization method. Therefore, calculating the standard deviation is doable but not meaningful as it is not following a clear distribution.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The computation resources information is provided in Table S5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The authors follow the NeurIPS Code of Ethics in conducting the research.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This paper is a theoretical study and does not have direct societal impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper focus on the theoretical side of the normalization method in time series Transformers and does not have high risks for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The authors properly credit the original owners of the assets and respect the license and terms of use, despite some assets used in this paper are missing the license information.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

## 13. New Assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not introduce new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

#### 14. Crowdsourcing and Research with Human Subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

#### 15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve studies on human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.