

Article

Improving Systematic Generalization of Linear Transformer Using Normalization Layers and Orthogonality Loss Function

Taewon Park and Hyun-Chul Kim * 

Department of Artificial Intelligence, Kyungpook National University, Daegu 41566, Republic of Korea; ptw4570@knu.ac.kr

* Correspondence: hyunchul_kim@knu.ac.kr

Abstract: A Linear Transformer linearizes the attention mechanism of the vanilla Transformer architecture, significantly improving efficiency and achieving linear theoretical complexity with respect to sequence length. However, few studies have explored the capabilities of the Linear Transformer beyond its efficiency. In this work, we investigate the systematic generalization capability of the Linear Transformer, a crucial property for strong generalization to unseen data. Through preliminary experiments, we identify two major issues contributing to its unstable systematic generalization performance: (i) unconstrained norms of *Queries* and *Keys*, and (ii) high correlation among *Values* across the sequence. To address these issues, we propose two simple yet effective methods: normalization layers for *Queries* and *Keys*, and an orthogonality loss function applied to *Values* during training. In experiments, we demonstrate that applying these methods to the Linear Transformer significantly improves its stability and systematic generalization performance across several well-known tasks. Furthermore, our proposed methods outperform the vanilla Transformer on specific systematic generalization tasks, such as the sort-of-CLEVR and SCAN tasks.

Keywords: transformer; linear transformer; systematic generalization; normalization; orthogonality loss



Citation: Park, T.; Kim, H.-C. Improving Systematic Generalization of Linear Transformer Using Normalization Layers and Orthogonality Loss Function. *Mathematics* **2024**, *12*, 3390. <https://doi.org/10.3390/math12213390>

Academic Editors: Evgeny Nikulchev and Ravil Muhamedyev

Received: 8 October 2024

Revised: 28 October 2024

Accepted: 28 October 2024

Published: 30 October 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

MSC: 68T07

1. Introduction

Transformer models [1] have significantly advanced neural networks across various domains, including natural language processing [2,3] and computer vision [4,5]. A key contributor to their success is a scaled dot-product attention mechanism, which enables Transformers to capture contextual information within data, leading to state-of-the-art performance in numerous tasks. However, this attention mechanism has quadratic space and time complexity with respect to the sequence length due to its softmax function, which limits the context length that can be computed [6–10].

To address this limitation, several methods have been developed to improve the efficiency of the attention mechanism, often by linearizing the original attention using kernel methods [6,10] or low-rank approximations [9]. These efficient attention mechanisms achieve linear theoretical complexity with respect to sequence length and exhibit comparable performance to the original attention in long sequential tasks, such as Long-Range Arena tasks [11] and language modeling tasks [12]. Despite these improvements, prior research on linear attention mechanisms has primarily focused on efficiency in handling long sequences while neglecting other critical properties of neural networks, such as systematic generalization.

Systematic generalization refers to the ability to generalize unseen data by combining familiar concepts in novel ways [13–18]. This capability has been extensively studied in Transformers, particularly in enabling them to generalize on out-of-distribution

(OOD) datasets. For instance, Mittal et al. [19] introduced a compositional attention mechanism to enable flexible and dynamic operations among attention heads. Meanwhile, Csordás et al. [20] examined the limitations of Transformers on systematic generalization benchmarks, showing that simple techniques, such as scaling embeddings, can effectively enhance their ability to learn systematic generalization tasks. However, compared to standard Transformers, the systematic generalization capabilities of efficient attention mechanisms remain underexplored.

In this work, we investigate the systematic generalization capabilities of efficient attention mechanisms, with a focus on the Linear Transformer [6], and explore various methods to enhance their performance. In our preliminary experiments on systematic generalization tasks, we identify two major issues in the attention components (*Queries*, *Keys*, and *Values*) that contribute to unstable generalization during training: (i) unconstrained norms of *Queries* and *Keys*, and (ii) high correlation among *Values* across the sequence. The linear attention mechanism operates by computing attention based on *Queries* and accumulated *Keys* over the sequence. As noted in previous research [21,22], this design can lead to instability during training, when the norms of *Queries* and *Keys* increase dramatically. In our preliminary experiments, we observe that these unconstrained values negatively affect the systematic generalization performance of linear attention mechanisms. Additionally, in non-causal settings, the linear attention mechanism often fails to learn distinct features across the sequence, resulting in highly correlated *Values*.

To address the instability in the systematic generalization of the Linear Transformer, we propose two simple yet effective techniques, a normalization term for attention components and an orthogonality loss, both motivated by the issues identified in our preliminary experiments. First, we apply a normalization term to *Queries* and *Keys* (as in [21]) to prevent excessively large norm values. We explore various strategies as the normalization term, including *L1*, *L2*, and *RMS* layer normalization [23]. Additionally, we introduce an orthogonality loss, an auxiliary loss that encourages *Values* within a sequence to be orthogonal during training. This loss reduces the correlation among *Values* generated from distinct input features, thereby improving the generalization performance.

In summary, the main contributions of this work are as follows:

- We investigate the systematic generalization capabilities of linear attention mechanisms and identify key limitations, including unconstrained norms of *Queries* and *Keys* and high correlation among *Values*.
- We propose normalization techniques and auxiliary loss functions to address these limitations and improve the stability and generalization performance of linear attention mechanisms.
- We evaluate our proposed methods on various systematic generalization tasks, including the sort-of-CLEVR [24], SCAN [16], Mathematics dataset [25], and PCFG [18] tasks. Our experimental results demonstrate that the proposed methods enhance the training stability and systematic generalization capabilities of the Linear Transformer.

2. Background and Related Work

2.1. Linear Attention Mechanism

The scaled dot-product attention mechanism is a fundamental component of the vanilla Transformer architecture [1]. This mechanism transforms input sequential data into three key components, *Queries*, *Keys*, and *Values*, through linear projection layers. It then computes a dot product between *Queries* and *Keys*, applying the softmax function to obtain normalized attention scores. The final output is derived by multiplying these attention scores with the *Values*, as expressed below:

$$Q, K, V = XW_q, XW_k, XW_v, \quad (1)$$

$$\text{Causal : } O_n = \frac{\sum_i^n \exp(Q_n K_i^T) V_i}{\sum_i^n \exp(Q_n K_i^T)}, \quad (2)$$

$$\text{Non-Causal : } O_n = \frac{\sum_i^N \exp(Q_n K_i^\top) V_i}{\sum_i^N \exp(Q_n K_i^\top)} \quad (3)$$

where $W_q \in \mathbb{R}^{d \times d_k}$, $W_k \in \mathbb{R}^{d \times d_k}$, and $W_v \in \mathbb{R}^{d \times d_v}$ are trainable parameters, $X \in \mathbb{R}^{N \times d}$ represents the input sequential data, N is the sequence length, and d, d_k , and d_v are the dimensions of the input, *Key*, and *Value* vectors, respectively. $Q \in \mathbb{R}^{N \times d_k}$, $K \in \mathbb{R}^{N \times d_k}$, and $V \in \mathbb{R}^{N \times d_v}$ represent *Queries*, *Keys*, and *Values*, respectively. Additionally, $Q_n \in \mathbb{R}^{d_k}$, $K_n \in \mathbb{R}^{d_k}$, and $V_n \in \mathbb{R}^{d_v}$ are the n -th feature vector within Q , K , and V , respectively.

However, the scaled dot-product attention mechanism suffers from quadratic space and time complexity relative to the sequence length, making it impractical for long sequences. Various efficient attention mechanisms have been proposed to improve the scalability of the vanilla Transformer's attention mechanism [6–10]. Among these, Katharopoulos et al. [6] introduced a linear attention mechanism that replaces the softmax function in the original attention with a kernel-based method. This linear attention mechanism substitutes the softmax function with a kernel function ϕ , as shown below:

$$\text{Causal : } O_n = \frac{\sum_i^n \phi(Q_n) \phi(K_i^\top) V_i}{\sum_i^n \phi(Q_n) \phi(K_i^\top)}, \quad (4)$$

$$= \frac{\phi(Q_n) \sum_i^n \phi(K_i^\top) V_i}{\phi(Q_n) \sum_i^n \phi(K_i^\top)}, \quad (5)$$

$$\text{Non-Causal : } O_n = \frac{\phi(Q_n) \sum_i^N \phi(K_i^\top) V_i}{\phi(Q_n) \sum_i^N \phi(K_i^\top)} \quad (6)$$

In this framework, different kernel functions, such as the $\text{elu} + 1$ [6] and cosine functions [10], have been proposed to enhance the long-term comprehension of the linear attention mechanism. However, most of these studies focused on improving efficiency in handling long sequences, with less attention paid to other important neural network properties. In this work, we go beyond long-term understanding and investigate a critical capability of linear attention mechanisms: systematic generalization.

2.2. Systematic Generalization

The systematic generalization is a crucial property for neural networks, required to achieve strong out-of-distribution (OOD) generalization [13–18]. It has been widely studied to improve model generalization in various fields, including natural language processing [18,20,25], computer vision [26–28], and robotic agents [16,29,30].

Unlike other OOD benchmark datasets that involve adding noise to data [31], evaluating systematic generalization capability requires specifically designed tasks, as it aims to test a model's ability to generalize to unseen combinations of known components. For example, a model should be able to generalize to the concept of a "red box" after learning "red circle" and "black box". Several studies have proposed benchmark datasets designed to test the systematic generalization of models, such as the sort-of-CLEVR [24], SCAN [16], Mathematics dataset [25], and PCFG [18] tasks.

Using these benchmarks, researchers have explored the systematic generalization capabilities of attention mechanisms [19,20]. For instance, Csordás et al. [20] examined the systematic generalization of the vanilla Transformer and found that simple techniques, such as scaling embeddings, can enhance generalization performance without the need for complex modifications.

Similarly, in this work, we examine the underexplored systematic generalization capabilities of linear attention mechanisms and propose methods to enhance their generalization performance.

3. Preliminary Experiments and the Design of Proposed Methods

In this section, we present preliminary experiments aimed at evaluating the systematic generalization capabilities of existing linear attention mechanisms, identifying their potential limitations, and using these findings to motivate the design of our proposed methods.

We employ the sort-of-CLEVR task [24] for the preliminary experiments, which evaluates systematic generalization in models through visual question answering. The task includes different types of questions: *Unary* (questions about the properties of a single visual object), *Binary*, and *Ternary* (questions about relationships among multiple visual objects). We conduct several trials (in each trial, the model is trained on a different training seed) using the Linear Transformer [6] on the sort-of-CLEVR task, following the setup of prior work [19]. Detailed experimental settings can be found in Section 5.

Table 1 and Figure 1 present the results of the vanilla Transformer and the Linear Transformer across five trials. The results indicate that the Linear Transformer exhibits worse systematic generalization performance than the vanilla Transformer and shows instability in generalization depending on the training seed. For example, while the Linear Transformer achieves comparable accuracy to the vanilla Transformer in *Unary* and *Ternary* question types in trials 4 and 5, it fails to achieve a similar performance in trials 1, 2, and 3.

Table 1. The mean accuracy [%] from the preliminary experiments on the sort-of-CLEVR task over five trials.

Model	Trial	Question Type		
		<i>Unary Accuracy</i>	<i>Binary Accuracy</i>	<i>Ternary Accuracy</i>
Transformer		98.5 ± 0.5	85.6 ± 2.9	64.3 ± 2.8
Linear Transformer	1	62.3	76.0	54.6
	2	63.0	76.4	53.9
	3	63.8	74.7	55.0
	4	99.2	78.7	62.9
	5	98.7	79.3	64.7

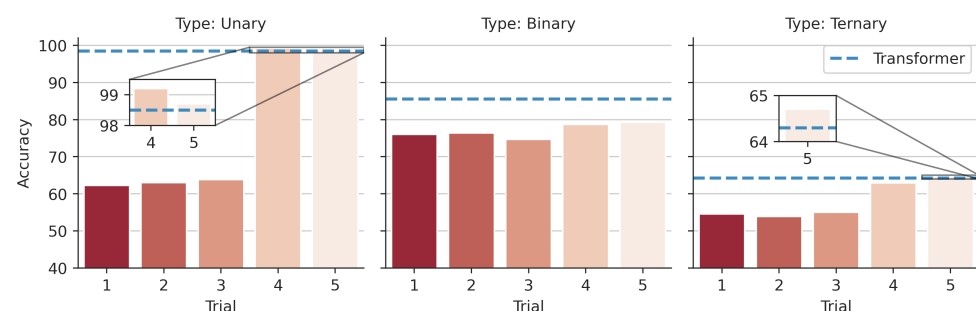


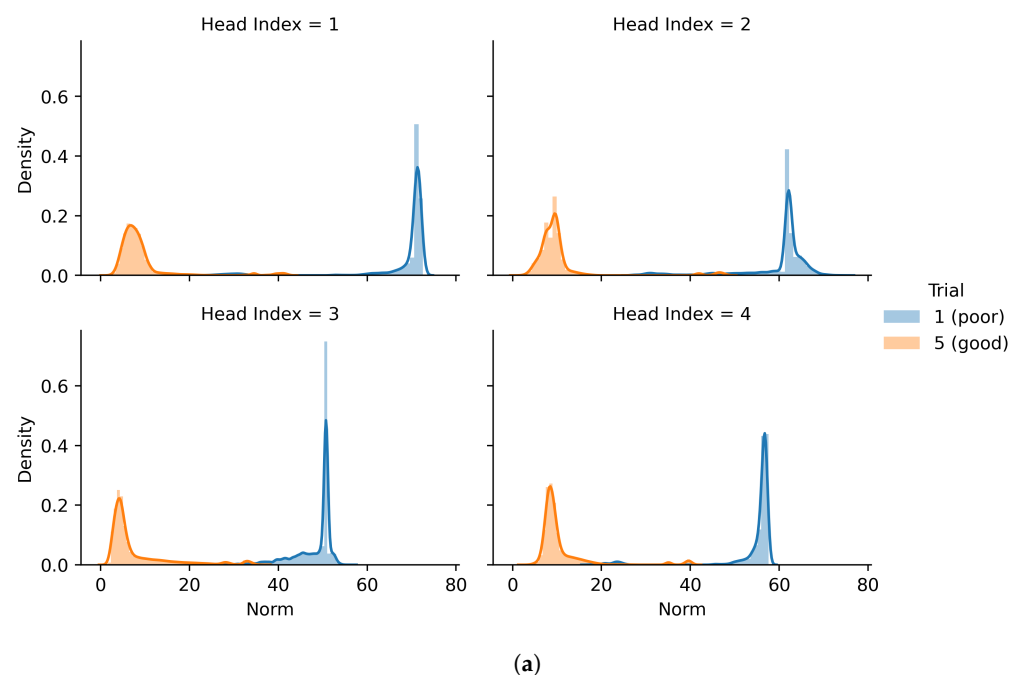
Figure 1. The accuracy [%] of the Linear Transformer from the preliminary experiments on the sort-of-CLEVR task over five trials. The blue dotted line indicates the mean accuracy of the Transformer. These results show that the Linear Transformer exhibits worse systematic generalization performance than the Transformer on the sort-of-CLEVR task.

To further investigate the source of this instability, we perform additional analyses comparing the attention components from a successful trial (trial 5) and a poor-performing trial (trial 1). These analyses are informed by prior research [21,22] suggesting that unconstrained attention components can lead to unstable performance in linear attention. Through this investigation, we identify two key flaws in the attention components—*Queries*, *Keys*, and *Values*—that may contribute to the instability of the linear attention mechanism.

First, we examine the norm distributions of *Queries* and *Keys*. The results, as shown in Figure 2, clearly indicate that the *Keys*' norms in the poorly performing trial (trial 1) are significantly higher than those in the successful trial (trial 5) across all attention heads. Similarly, the *Queries*' norms in the poor trial are higher than in the successful trial for all heads, except for head index 4. These findings suggest that the lack of constraints on *Queries* and *Keys* may lead to unstable generalization performance. Based on this observation, we hypothesize that the relatively high norm distribution contributes to the instability of the linear attention mechanism and investigate normalization methods applied to *Queries* and *Keys* as a potential solution.

Next, we analyze the representational quality of *Values*, which is closely linked to the overall performance of the attention mechanism, through the lens of similarity. Figure 3 shows cosine similarity heatmaps for *Values* across the sequence in the non-causal setting. In the poorly performing trial (Figure 3a), the *Values* exhibit a high correlation, whereas in the successful trial (Figure 3b), the *Values* show a significantly lower correlation. This high correlation may suggest that the attention mechanism fails to transfer distinct information about individual tokens to subsequent layers, ultimately leading to poor generalization performance. To address this issue, we explore the use of auxiliary objective functions to encourage the model to learn more distinct representations across the sequence during training.

In the following section, we build upon these preliminary findings and propose representation learning methods to enhance the systematic generalization performance of the linear attention mechanism.



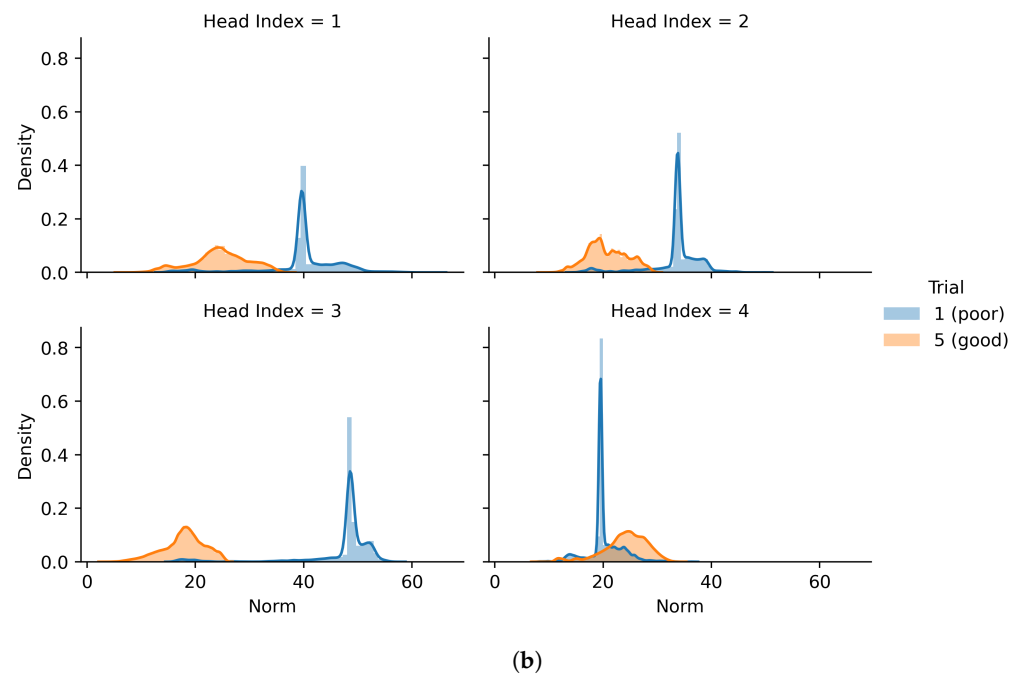


Figure 2. Norm distributions of (a) *Keys* and (b) *Queries* for a poor-performing trial (trial 1) and a successful trial (trial 5). The figure shows that the norms of *Keys* and *Queries* are significantly higher in the poorly performing trial compared to the successful trial. These results suggest that the lack of constraints on *Queries* and *Keys* may contribute to poor systematic generalization performance.

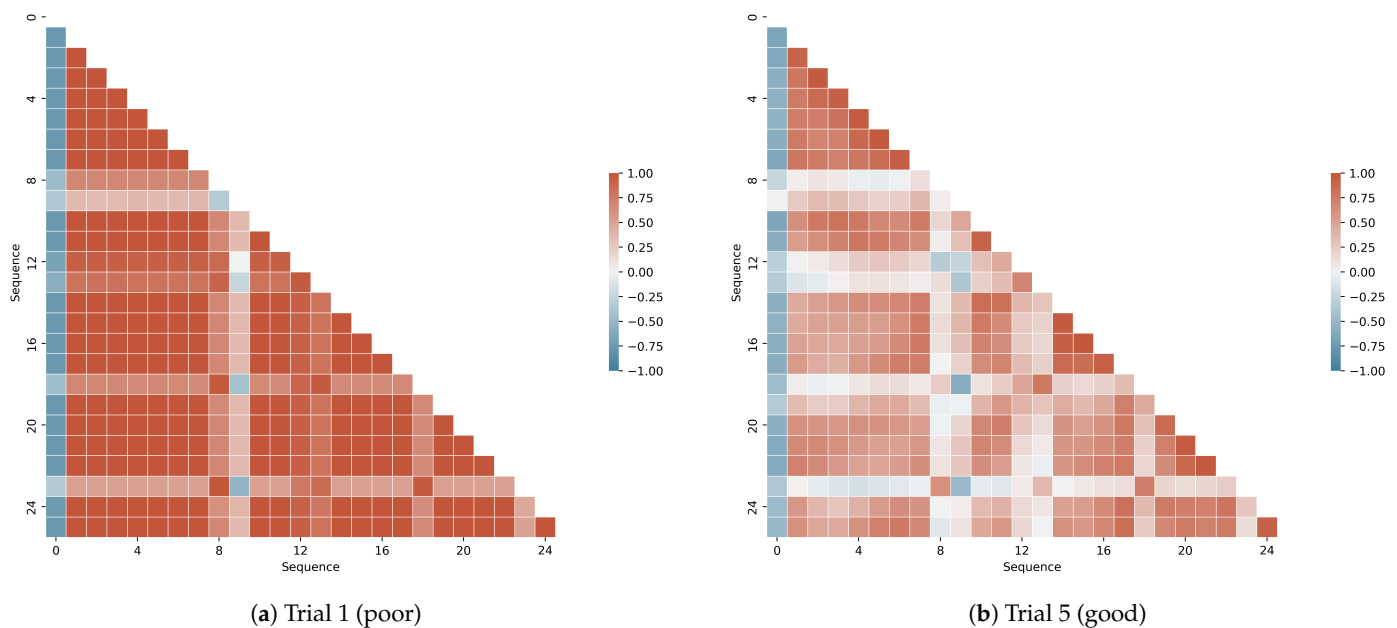


Figure 3. The heatmap visualizes the cosine similarity of *Values* across the sequence for (a) a poor-performing trial (trial 1) and (b) a successful trial (trial 5). In the poorly performing trial, the *Values* exhibit a high correlation, whereas in the successful trial, the *Values* show a relatively lower correlation. These results suggest that a high correlation among *Values* may contribute to poor systematic generalization performance.

4. Proposed Method

In this section, we introduce simple yet effective techniques to enhance the systematic generalization capabilities of the linear attention mechanism, as shown in Figure 4.

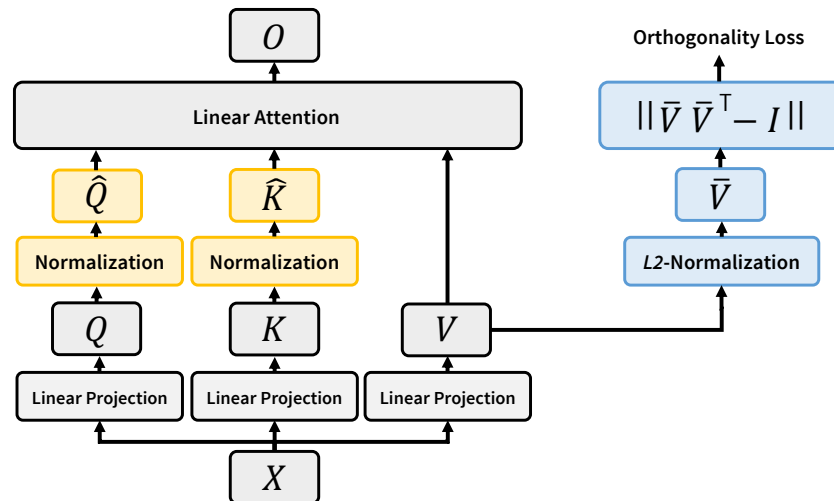


Figure 4. Overview of the proposed normalization layers and orthogonality loss function. The yellow box illustrates the application of normalization layers to *Queries* and *Keys*. The blue box depicts the orthogonality loss function applied to *Values*, encouraging them to be orthogonal during training. X , Q , K , V , O , and I denote input data, *Queries*, *Keys*, *Values*, an output, and an identity matrix, respectively.

4.1. Normalization Layers for Queries and Keys

As discussed in Section 3, we hypothesize that the relatively high norms of *Queries* and *Keys* are closely related to the instability in the systematic generalization of the Linear Transformer. To address this issue, we introduce normalization layers to prevent the norms of *Queries* and *Keys* from becoming excessively large during training, thereby mitigating the instability in the Linear Transformer.

Normalization can be applied at different levels, including the input level [23], group level [32], or batch level [33]. This study focuses on input-level normalization since the Linear Transformer processes sequential data with variable input lengths. We employ several well-known input-level normalization strategies, including $L1$, $L2$, and RMS layer normalization [23].

In the linear attention mechanism, normalization layers can be applied to the feature vectors of *Queries* and *Keys* either before or after the kernel function. In this work, we apply normalization after the kernel function, consistent with previous research [21].

In the linear attention mechanism with normalization applied, once the kernel function is applied to *Queries* and *Keys*, the normalization layer normalizes each feature vector individually, as expressed by the following equations:

$$\hat{Q}_n = \text{Normalization}(\phi(Q_n)), \quad (7)$$

$$\hat{K}_n = \text{Normalization}(\phi(K_n)), \quad (8)$$

$$\text{Causal : } O_n = \frac{\hat{Q}_n \sum_i^n \hat{K}_i^\top V_i}{\hat{Q}_n \sum_i^n \hat{K}_i^\top}, \quad (9)$$

$$\text{Non-Causal : } O_n = \frac{\hat{Q}_n \sum_i^N \hat{K}_i^\top V_i}{\hat{Q}_n \sum_i^N \hat{K}_i^\top} \quad (10)$$

Normalization(x) refers to one of the following methods:

$$L1\text{-Normalization}(x) = \frac{x}{(\|x\|_1 + \epsilon)} \quad (11)$$

$$L2\text{-Normalization}(x) = \frac{x}{(\|x\|_2 + \epsilon)} \quad (12)$$

$$RMS\text{-Normalization}(x) = \frac{x}{(\text{RMS}[x] + \epsilon)} * \gamma \quad (13)$$

where $x \in \mathbb{R}^d$ represents a feature vector, ϵ is a small positive constant to avoid division by zero, $\|\cdot\|_1$ is an L1 norm, $\|\cdot\|_2$ is an L2 norm, $\text{RMS}[\cdot]$ is a root mean square function, and $\gamma \in \mathbb{R}^d$ is a trainable parameter.

Unlike L1 and L2 normalization, RMS normalization includes the trainable parameter γ . In our approach, we apply separate RMS normalization layers to *Queries* and *Keys*, but the normalization is shared across all attention heads.

4.2. Orthogonality Loss for Values

In our preliminary experiments (Section 3), we observe a high correlation among distinct *Values* across the sequence, which may negatively impact the Linear Transformer's systematic generalization capability. To address this issue, we introduce an orthogonality loss function that encourages the *Values* to be orthogonal to each other. Unlike traditional orthogonality regularization methods that are applied to network weights [34], our proposed orthogonality loss function is applied directly to the *Values*. This approach directly regularizes the high correlation among *Values* during training.

To calculate the orthogonality loss function, first, we normalize the *Value* feature vectors using L2 normalization:

$$\tilde{V}_n = \frac{V_n}{(\|V_n\|_2 + \epsilon)}, \quad (14)$$

where V_n is the n -th *Value* vector, and ϵ is a small positive constant to prevent division by zero.

Using the normalized *Values*, the orthogonality loss function, $\mathcal{L}_{\text{orthogonality}}$, is defined as follows:

$$\mathcal{L}_{\text{orthogonality}} = \|\tilde{V}\tilde{V}^\top - I\|_F^2, \quad (15)$$

where $I \in \mathbb{R}^{N \times N}$ is the identity matrix, $\tilde{V} \in \mathbb{R}^{N \times d}$ is the matrix of normalized *Value* vectors, and $\|\cdot\|_F$ denotes the Frobenius norm.

We then combine the orthogonality loss with the original task loss, $\mathcal{L}_{\text{task}}$, to train the model:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{orthogonality}} \quad (16)$$

where $\mathcal{L}_{\text{task}}$ is the original task-specific loss function, and λ is a regularization coefficient that controls the strength of the orthogonality penalty.

While the orthogonality loss helps prevent the model from learning highly correlated *Values*, applying too strong a penalty may interfere with learning the correct attention mechanisms. Therefore, we adjust the value of the regularization coefficient λ to balance the trade-off between enforcing orthogonality and maintaining the performance of the linear attention mechanism.

5. Experiment Setup

In this section, we describe the systematic generalization tasks used in this work, along with details of the model implementation and training procedures.

5.1. Sort-of-CLEVR Task

The sort-of-CLEVR task [24] evaluates the systematic generalization of models in visual relational reasoning using images, questions, and corresponding answers. Each image contains six distinct objects, each varying in color (red, blue, green, orange, yellow, or gray) and shape (square or circle). As described by Mittal et al. [19], the questions are categorized into three types: *Unary* (focusing on the properties of a single object), *Binary*, and *Ternary* (focusing on relationships between multiple objects). Following Mittal et al. [19], we evaluate the generalization performance of models for each question type individually.

For this task, we follow the experimental settings of Mittal et al. [19], using a four-layer shared Transformer and Linear Transformer (encoder only) with four attention heads and a model dimension of 256, and an $\mathcal{L}_{\text{task}}$ of a cross-entropy loss. We use a CNN-based visual encoder with a kernel size of 15 and a stride of 15. The models are trained using the Adam optimizer with a batch size of 64 and a learning rate of 1×10^{-4} for 100 epochs. We also experiment with several regularization coefficients: $\{1 \times 10^{-2}, 1 \times 10^{-3}, \text{ and } 1 \times 10^{-4}\}$.

5.2. SCAN Task

The SCAN task [16] is a natural language task that evaluates the systematic generalization of models. In this task, models receive natural language input and are required to translate it into an action sequence for a robotic agent. For example, the input “look around left and jump opposite left” should be translated into the action sequence “LTURN LOOK LTURN LOOK LTURN LOOK LTURN LOOK LTURN LTURN JUMP”. The SCAN task provides a length-based data split where the action sequence lengths in testing are longer than those in training, based on a specific cutoff length. In this work, we adopt a cutoff length of 26 and evaluate the models in two settings: (1) the independent and identically distributed (IID) setting, where the testing action sequence length matches the training length, and (2) the OOD setting, where the testing sequence length exceeds the training sequence length.

For this task, we use the experimental settings from Csordás et al. [20], which include a three-layer shared Linear Transformer with eight attention heads and a model dimension of 128, and an $\mathcal{L}_{\text{task}}$ of a token-wise cross-entropy loss. The models are trained using the Adam optimizer with a batch size of 256 and a learning rate of 1×10^{-3} for 50,000 steps. We apply the orthogonality loss function only to the encoder of the Linear Transformer (in a non-causal configuration) with a regularization coefficient of 1×10^{-4} .

5.3. Mathematics Dataset

The Mathematics dataset [25] is used to evaluate mathematical reasoning skills across various problem categories. The task involves translating mathematical instructions into correct answers, such as “Calculate $-502 + 1523 \rightarrow 1021$ ”. Following Csordás et al. [20], we focus on two categories: “add_or_sub” (addition and subtraction problems for pairs of integers or decimals) and “place_value” (problems asking for the value of a specific digit). We evaluate the models in both IID and OOD settings, where the testing values are larger than those encountered during training.

For this task, we follow the experimental settings from Csordás et al. [20], using a six-layer shared Linear Transformer with eight attention heads and a model dimension of 512, and an $\mathcal{L}_{\text{task}}$ of a token-wise cross-entropy loss. The models are trained with the Adam optimizer, using a batch size of 256 and a learning rate of 1×10^{-4} for 50,000 steps. We apply the orthogonality loss function to the encoder of the Linear Transformer (non-causal configuration) with a regularization coefficient of 1×10^{-4} .

5.4. PCFG Task

The PCFG task [18] evaluates models in sequence-to-sequence scenarios, testing properties such as systematicity and productivity. In this task, models receive natural language instructions involving functions (e.g., append, copy, repeat) and strings (e.g., A, B, C), and are required to translate them into the correct sequence. For example, the input “re-

peat A B C” should be translated to “A B C A B C”. The PCFG task provides several data splits targeting specific aspects of generalization, and, in this work, we focus on the “systematicity” split.

For this task, we follow the experimental settings from Csordás et al. [20], using a six-layer shared Linear Transformer with eight attention heads and a model dimension of 512, and an $\mathcal{L}_{\text{task}}$ of a token-wise cross-entropy loss. The models are trained using the Adam optimizer with a batch size of 64 and a learning rate of 1×10^{-4} for 300,000 steps. The orthogonality loss function is applied only to the encoder of the Linear Transformer (in a non-causal configuration) with a regularization coefficient of 1×10^{-4} .

6. Experiment Result

In this section, we demonstrate the effectiveness of our proposed methods in enhancing the systematic generalization performance of the Linear Transformer across the several tasks introduced in Section 5. We organize the experiments into two categories based on model architectures: (i) encoder-only architecture and (ii) encoder–decoder architecture.

6.1. Experiments with Encoder-Only Architecture

We first evaluate the proposed methods on the sort-of-CLEVR task, as conducted in the preliminary experiments. Table 2 presents the mean accuracy of our proposed methods over five trials for each question type. The results indicate that the proposed normalization layers and orthogonality loss function significantly improve both the generalization performance and stability of the Linear Transformer across all question types.

Table 2. The mean accuracy [%] on the sort-of-CLEVR task over five trials. ✓ indicates that the method is used, while ✗ indicates it is not. The bold text highlights the best accuracy across models.

Model	Normalization	Orthogonality Loss	Question Type		
			Unary Accuracy	Binary Accuracy	Ternary Accuracy
Transformer	✗	✗	98.5 ± 0.5	85.6 ± 2.9	64.3 ± 2.8
Linear Transformer	✗	✗	77.4 ± 17.6	77.0 ± 1.7	58.2 ± 4.6
		✓	98.7 ± 0.4	81.7 ± 1.9	66.5 ± 2.4
	RMS	✗	98.7 ± 0.3	83.5 ± 2.1	66.0 ± 2.3
		✓	98.8 ± 0.1	86.6 ± 2.0	68.5 ± 0.5
	L1	✗	98.8 ± 0.2	81.9 ± 2.8	66.8 ± 1.9
		✓	99.0 ± 0.2	84.2 ± 3.9	66.8 ± 1.7
	L2	✗	99.1 ± 0.2	83.7 ± 3.3	67.8 ± 1.4
		✓	99.0 ± 0.1	87.6 ± 1.5	69.0 ± 0.6
Compositional Transformer [19]	✗	✗	98.9 ± 0.2	89.9 ± 1.4	66.3 ± 0.9
Performer [35]	✗	✗	98.9 ± 0.1	82.6 ± 1.8	58.3 ± 3.1
Cosformer [10]	✗	✗	98.5 ± 0.1	78.2 ± 0.8	66.0 ± 0.6

First, applying the proposed normalization layers to *Queries* and *Keys* proves effective in enhancing the generalization performance of the Linear Transformer, regardless of the specific normalization method used. While the vanilla Linear Transformer achieves an accuracy of 77.4% on *Unary* questions, 77.0% on *Binary* questions, and 58.2% on *Ternary* questions, the *L2* normalization layer improves these results to 99.1%, 83.7%, and 67.8%, respectively. Other normalization methods also show performance gains, with the *RMS* layer normalization method achieving accuracies of 98.7%, 83.5%, and 66.0%, and the *L1* normalization layer reaching 98.8%, 81.9%, and 66.8%.

Second, the proposed orthogonality loss function with a regularization coefficient of 1×10^{-3} also enhances the vanilla Linear Transformer’s performance, achieving accuracies of 98.7%, 81.7%, and 66.5%. Furthermore, combining the orthogonality loss function with

the normalization layers results in a significant performance boost with reduced variance. Notably, the combined method with the $L2$ normalization layer achieves accuracies of 99.0%, 87.6%, and 69.0%, surpassing the systematic generalization performance of the vanilla Transformer.

Next, we extend our comparison to include the Compositional Transformer [19], which is specifically designed to enhance systematic generalization in Transformers. As shown in Table 2, our proposed methods—particularly the combination of $L2$ normalization and the orthogonality loss—achieve comparable performance with the Compositional Transformer. Additionally, we compare our methods to other efficient attention mechanisms, including the Performer [35] and Cosformer [10]. As shown in Table 2, while both the Performer and Cosformer outperform the Linear Transformer, their generalization performance falls short of the vanilla Transformer. Applying our proposed methods to the Linear Transformer yields a better performance than the Performer and Cosformer, demonstrating the effectiveness of our approach.

Ablation Study and Analysis

We perform an ablation study on the orthogonality loss function combined with the $L2$ normalization layer, varying the regularization coefficient from 1×10^{-4} to 1×10^{-2} . As shown in Figure 5, the results demonstrate that the proposed orthogonality loss function consistently improves the performance of the Linear Transformer across all coefficient values in the sort-of-CLEVR task. However, the results also suggest that an optimal regularization coefficient is required to achieve the best performance, with 1×10^{-3} yielding the highest accuracy on the sort-of-CLEVR task.

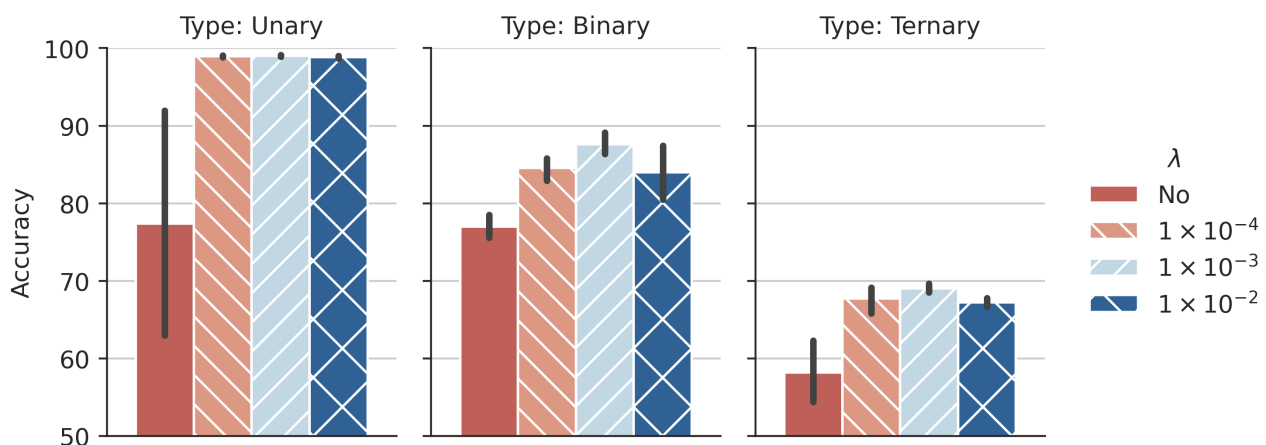


Figure 5. The mean accuracy [%] of the normalized Linear Transformer with $L2$ normalization and varying λ on the sort-of-CLEVR task over five trials.

Additionally, we investigate the effect of the proposed methods on different hyperparameters of the Linear Transformer, specifically the model dimension and the number of attention heads. As shown in Table 3, the proposed methods improve the generalization performance of the Linear Transformer across all hyperparameter settings. These results demonstrate the efficacy of the proposed methods across a wide range of conditions.

Next, we analyze the effect of the orthogonality loss function on the *Values*. Similar to our approach in Section 3, we examine the correlation among *Values*. As shown in Figure 6, the proposed methods effectively reduce the correlation among *Values*. These results confirm that the orthogonality loss function operates as intended by lowering correlation, thereby enhancing generalization performance.

Table 3. The mean accuracy [%] on the sort-of-CLEVR task over five trials in ablation study varying the model dimension and the number of attention heads. The bold text highlights the best accuracy across models.

Model Dimension	Attention Head	Model	Question Type		
			Unary Accuracy	Binary Accuracy	Ternary Accuracy
256	4	Linear Transformer	77.4 ± 17.6	77.0 ± 1.7	58.2 ± 4.6
		Linear Transformer + L_2 & $\mathcal{L}_{\text{orthogonality}}$	99.0 ± 0.1	87.6 ± 1.5	69.0 ± 0.6
	8	Linear Transformer	84.7 ± 17.1	78.4 ± 2.1	61.0 ± 5.7
		Linear Transformer + L_2 & $\mathcal{L}_{\text{orthogonality}}$	98.6 ± 0.2	86.1 ± 2.4	66.4 ± 1.2
512	4	Linear Transformer	58.6 ± 1.3	66.0 ± 6.6	53.0 ± 0.9
		Linear Transformer + L_2 & $\mathcal{L}_{\text{orthogonality}}$	98.3 ± 0.3	76.9 ± 0.8	60.7 ± 2.7
	8	Linear Transformer	58.6 ± 2.4	68.3 ± 6.2	53.4 ± 0.8
		Linear Transformer + L_2 & $\mathcal{L}_{\text{orthogonality}}$	97.7 ± 0.7	77.4 ± 0.9	59.9 ± 2.9

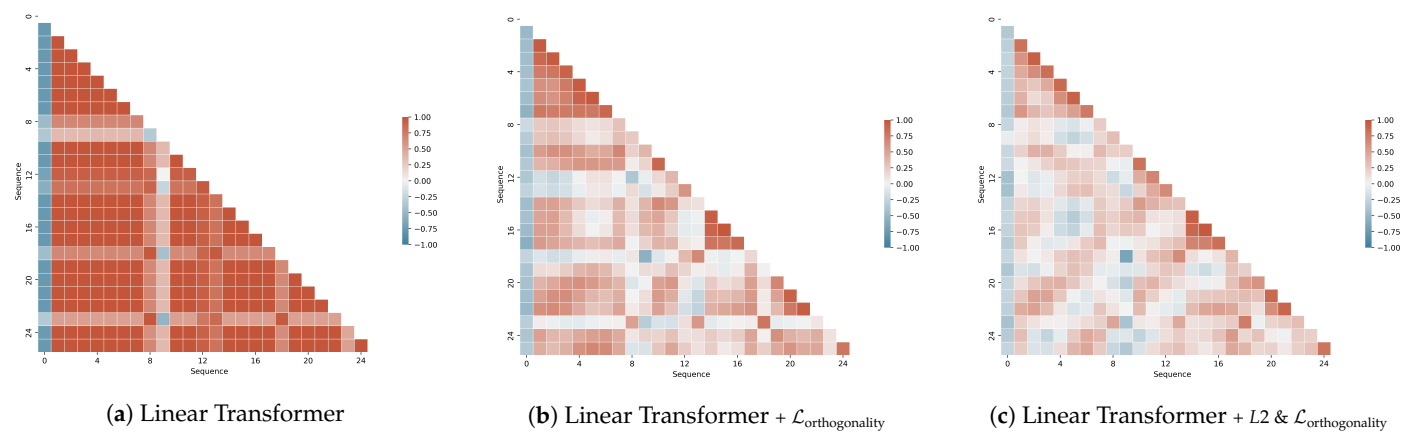


Figure 6. The heatmap visualizes the cosine similarity of *Values* across the sequence for (a) the vanilla Linear Transformer, (b) applying orthogonality loss, and (b) applying L_2 normalization and orthogonality loss. These results indicate that the orthogonality loss function effectively reduces correlation among *Values* across the sequence.

6.2. Experiments with Encoder–Decoder Architecture

Next, we evaluate the proposed methods in an encoder–decoder architecture on the SCAN, Mathematics dataset, and PCFG tasks.

6.2.1. SCAN Task

Table 4 presents the mean accuracy of the proposed methods on the SCAN task over five trials for both the IID and OOD settings. Interestingly, unlike for the sort-of-CLEVR task, the vanilla Linear Transformer outperforms the vanilla Transformer in the OOD setting, achieving an accuracy of 46.9%. This suggests that the Linear Transformer may have potential in some systematic generalization tasks. Our proposed methods further improve the performance of the Linear Transformer.

Table 4. The mean accuracy [%] on the SCAN task with a cutoff length of 26 over five trials. [†] indicates that results are derived from Csordás et al. [20]. ✓ indicates that the method is used, while ✗ indicates it is not. The bold text highlights the best accuracy across models.

Model	Normalization	Orthogonality Loss	Accuracy (IID)	Accuracy (OOD)
Transformer [†]	✗	✗	100.0 ± 0.0	21.0 ± 1.0
Linear Transformer	✗	✗	99.8 ± 0.1	46.9 ± 8.9
		✓	99.8 ± 0.1	48.4 ± 11.5
	RMS	✗	99.8 ± 0.0	48.4 ± 8.2
		✓	99.8 ± 0.1	50.4 ± 7.5
	L1	✗	99.7 ± 0.2	49.5 ± 4.1
		✓	99.7 ± 0.2	44.9 ± 4.1
	L2	✗	99.7 ± 0.1	50.6 ± 8.6
		✓	99.7 ± 0.1	56.3 ± 5.4

First, applying normalization layers improves OOD generalization performance across all normalization strategies, with accuracies of 48.4% for RMS, 49.5% for L1, and 50.6% for L2. Second, the orthogonality loss function also enhances the baseline model, resulting in an OOD accuracy of 48.4%. Finally, combining both methods further improves generalization performance (except for the L1 case), with OOD accuracies of 50.4% for RMS, 44.9% for L1, and 56.3% for L2. Notably, as with the sort-of-CLEVR task, the best performance on the SCAN task is achieved by combining the orthogonality loss function with the L2 normalization layer.

6.2.2. “Add_or_Sub” Problem of the Mathematics Dataset

We also evaluate the proposed methods on the “add_or_sub” problem of the Mathematics dataset. As shown in Table 5, the performance improvement depends on the combination of the proposed methods.

Table 5. The mean accuracy [%] on the “add_or_sub” problem of the Mathematics dataset over five trials. [†] indicates that results are derived from Csordás et al. [20]. ✓ indicates that the method is used, while ✗ indicates it is not. The bold text highlights the best accuracy across models.

Model	Normalization	Orthogonality Loss	Accuracy (IID)	Accuracy (OOD)
Transformer [†]	✗	✗	100.0 ± 0.0	94.0 ± 1.0
Linear Transformer	✗	✗	87.9 ± 0.5	66.7 ± 0.7
		✓	87.4 ± 1.2	64.3 ± 0.3
	RMS	✗	87.5 ± 0.6	65.2 ± 0.7
		✓	87.8 ± 0.7	67.1 ± 1.4
	L1	✗	87.5 ± 0.6	65.6 ± 1.5
		✓	88.1 ± 0.4	65.5 ± 1.4
	L2	✗	88.0 ± 0.9	66.0 ± 3.2
		✓	88.4 ± 0.3	67.2 ± 1.3

First, applying normalization layers alone degrades the Linear Transformer’s generalization performance, with OOD accuracies of 65.2% for RMS, 65.6% for L1, and 66.0% for L2. Similarly, the orthogonality loss function also reduces the baseline model’s performance, yielding an OOD accuracy of 64.3%. However, when both methods are applied together, the generalization performance improves (except for in the L1 case), with OOD accuracies of 67.1% for RMS, 65.5% for L1, and 67.2% for L2. These results suggest that the proposed methods, while designed to address issues observed in the preliminary experiments, may

not be effective for all systematic generalization tasks. Furthermore, they highlight the importance of identifying the optimal combination of proposed methods to enhance the Linear Transformer’s performance properly.

6.2.3. “Place_Value” Problem of the Mathematics Dataset

Next, we evaluate the proposed methods on the “place_value” problem of the Mathematics dataset. As shown in Table 6, the vanilla Linear Transformer struggles with training instability and fails to learn the “place_value” problem where the target output length is extremely short (1 in this case). While *RMS* normalization does not resolve this instability, the *L1* and *L2* normalization layers effectively address the issue by preventing the norms of attention components from becoming excessively large, achieving OOD accuracies of 21.6% for *L1* and 18.0% for *L2*.

Table 6. The mean accuracy [%] on the “place_value” problem of the Mathematics dataset over five trials. [†] indicates that results are derived from Csordás et al. [20]. ✓ indicates that the method is used, while ✗ indicates it is not. The bold text highlights the best accuracy across models. Fail indicates that the model diverged during training.

Model	Normalization	Orthogonality Loss	Accuracy (IID)	Accuracy (OOD)
Transformer [†]	✗	✗	100.0 ± 0.0	20. ± 2.0
Linear Transformer	✗	✗		Fail
		✓		Fail
	<i>RMS</i>	✗		Fail
		✓		Fail
	<i>L1</i>	✗	97.1 ± 0.2	21.6 ± 3.5
		✓	97.1 ± 0.2	17.9 ± 1.4
	<i>L2</i>	✗	97.0 ± 0.1	18.0 ± 4.7
		✓	97.0 ± 0.1	20.5 ± 6.9

In contrast, applying the orthogonality loss function alone does not resolve the instability of the “place_value” problem. Furthermore, as with the “add_or_sub” problem, performance varies depending on the combination of methods. When both normalization layers and the orthogonality loss are applied together, OOD accuracies of 17.9% for *L1* and 20.5% for *L2* are achieved. These results indicate that normalization strategies without additional trainable parameters, such as *L1* and *L2*, are effective for addressing training instability.

6.2.4. PCFG Task

Finally, we evaluate the proposed methods for the PCFG task. As shown in Table 7, the vanilla Linear Transformer suffers from training instability, and neither *RMS* normalization nor the orthogonality loss resolves this issue, similar to for the “place_value” problem. On the other hand, applying *L1* and *L2* normalization layers effectively mitigates the instability, achieving accuracies of 56.9% and 45.4%, respectively. Furthermore, combining the orthogonality loss function with the normalization layers leads to further improvement, achieving 58.4% for *L1* and 47.4% for *L2*.

Table 7. The mean accuracy [%] on the PCFG task over five trials. [†] indicates that results are derived from Csordás et al. [20]. ✓ indicates that the method is used, while ✗ indicates it is not. The bold text highlights the best accuracy across models. Fail indicates that the model diverged during training.

Model	Normalization	Orthogonality Loss	Accuracy
Transformer [†]	✗	✗	93.0 ± 1.
Linear Transformer	✗	✗	Fail
		✓	Fail
	RMS	✗	Fail
		✓	Fail
	L1	✗	56.9 ± 1.9
		✓	58.4 ± 2.1
	L2	✗	45.4 ± 1.8
		✓	47.4 ± 1.4

7. Discussion

In this section, we discuss the limitations of the proposed methods.

First, although the proposed methods effectively improve the systematic generalization of the Linear Transformer, the degree of improvement varies depending on the combination of normalization strategies and orthogonality loss used for each task. While the combination of L2 normalization and orthogonality loss generally performs better than others, the underlying reasons for this are still unclear. Moreover, other combinations sometimes outperform L2 normalization with orthogonality loss on specific tasks, such as the “place_value” problem of the Mathematics dataset and the PCFG task. This need for heuristic tuning to find the optimal configuration can impose an additional burden on researchers wishing to utilize these methods. In future work, we will investigate the relationship between normalization strategies and orthogonality loss.

Second, applying the proposed methods to the Linear Transformer introduces additional computational overhead. In our study, adding normalization layers slightly increased training time, whereas applying the orthogonality loss resulted in, on average, a 1.5 times longer training time across all tasks. Furthermore, although the orthogonality loss function can improve the Linear Transformer’s generalization performance, its computational overhead may become more severe as the sequence length increases.

Third, as shown in Figure 6, while the orthogonality loss function effectively reduces correlation among *Values* across the sequence, some features still exhibit relatively high correlations. An over-penalizing configuration (i.e., a high value of γ) in the orthogonality loss function can negatively impact features that should remain similar, thereby degrading the model’s generalization performance, as shown in Figure 5. One possible solution is incorporating the similarity between input features to implement context-aware regularization methods. We leave this for future work.

8. Conclusions

In this work, we propose normalization layers and an orthogonality loss function to enhance the systematic generalization capability of the Linear Transformer. These methods are applied to the attention components of the Linear Transformer and aid in learning systematic generalization tasks. In our experiments, we demonstrate that the normalization layers mitigate instability problems in the Linear Transformer on systematic generalization tasks. Additionally, we show that the orthogonality loss function reduces the high correlation among *Values* across the sequence. When both proposed methods are applied together, they effectively improve the Linear Transformer’s generalization performance across a wide range of systematic generalization tasks.

Author Contributions: Conceptualization, T.P.; methodology, T.P.; software, T.P.; validation, T.P. and H.-C.K.; formal analysis, T.P.; investigation, T.P.; resources, H.-C.K.; data curation, T.P.; writing—original draft preparation, T.P.; writing—review and editing, T.P. and H.-C.K.; visualization, T.P.; supervision, H.-C.K.; project administration, H.-C.K.; funding acquisition, H.-C.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) and grant funded by the Korea government (MSIT) (no. RS-2022-00166735 and no. RS-2023-00218987).

Data Availability Statement: The sort-of-CLEVR dataset is available at [24], the SCAN dataset is available at [16], the Mathematics dataset is available at [25], and the PCFG dataset is available at [18].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; Volume 30.
2. Kenton, J.D.M.W.C.; Toutanova, L.K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019*; pp. 4171–4186.
3. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
4. Dosovitskiy, A. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
5. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021*; pp. 10012–10022.
6. Katharopoulos, A.; Vyas, A.; Pappas, N.; Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020*; pp. 5156–5165.
7. Zaheer, M.; Guruganesh, G.; Dubey, K.A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; et al. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2020; Volume 33, pp. 17283–17297.
8. Kitaev, N.; Kaiser, Ł.; Levskaya, A. Reformer: The efficient transformer. *arXiv* **2020**, arXiv:2001.04451.
9. Wang, S.; Li, B.Z.; Khabsa, M.; Fang, H.; Ma, H. Linformer: Self-attention with linear complexity. *arXiv* **2020**, arXiv:2006.04768.
10. Qin, Z.; Sun, W.; Deng, H.; Li, D.; Wei, Y.; Lv, B.; Yan, J.; Kong, L.; Zhong, Y. Cosformer: Rethinking softmax in attention. *arXiv* **2022**, arXiv:2202.08791.
11. Tay, Y.; Dehghani, M.; Abnar, S.; Shen, Y.; Bahri, D.; Pham, P.; Rao, J.; Yang, L.; Ruder, S.; Metzler, D. Long range arena: A benchmark for efficient transformers. *arXiv* **2020**, arXiv:2011.04006.
12. Merity, S.; Xiong, C.; Bradbury, J.; Socher, R. Pointer sentinel mixture models. *arXiv* **2016**, arXiv:1609.07843.
13. Fodor, J.A.; Pylyshyn, Z.W. Connectionism and cognitive architecture: A critical analysis. *Cognition* **1988**, *28*, 3–71. [[CrossRef](#)]
14. Lake, B.M.; Ullman, T.D.; Tenenbaum, J.B.; Gershman, S.J. Building machines that learn and think like people. *Behav. Brain Sci.* **2017**, *40*, e253. [[CrossRef](#)]
15. Liška, A.; Kruszewski, G.; Baroni, M. Memorize or generalize? Searching for a compositional rnn in a haystack. *arXiv* **2018**, arXiv:1802.06467.
16. Lake, B.; Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018*; pp. 2873–2882.
17. Greff, K.; Van Steenkiste, S.; Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv* **2020**, arXiv:2012.05208.
18. Hupkes, D.; Dankers, V.; Mul, M.; Bruni, E. Compositionality decomposed: How do neural networks generalise? *J. Artif. Intell. Res.* **2020**, *67*, 757–795. [[CrossRef](#)]
19. Mittal, S.; Raparthy, S.C.; Rish, I.; Bengio, Y.; Lajoie, G. Compositional attention: Disentangling search and retrieval. *arXiv* **2021**, arXiv:2110.09419.
20. Csordás, R.; Irie, K.; Schmidhuber, J. The devil is in the detail: Simple tricks improve systematic generalization of transformers. *arXiv* **2021**, arXiv:2108.12284.
21. Schlag, I.; Irie, K.; Schmidhuber, J. Linear transformers are secretly fast weight programmers. In *Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021*; pp. 9355–9366.
22. Qin, Z.; Han, X.; Sun, W.; Li, D.; Kong, L.; Barnes, N.; Zhong, Y. The devil in linear transformer. *arXiv* **2022**, arXiv:2210.10340.
23. Zhang, B.; Sennrich, R. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2019; Volume 32.
24. Santoro, A.; Raposo, D.; Barrett, D.G.; Malinowski, M.; Pascanu, R.; Battaglia, P.; Lillicrap, T. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; Volume 30.

25. Saxton, D.; Grefenstette, E.; Hill, F.; Kohli, P. Analysing mathematical reasoning abilities of neural models. *arXiv* **2019**, arXiv:1904.01557.
26. Locatello, F.; Weissenborn, D.; Unterthiner, T.; Mahendran, A.; Heigold, G.; Uszkoreit, J.; Dosovitskiy, A.; Kipf, T. Object-centric learning with slot attention. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2020; Volume 33, pp. 11525–11538.
27. Webb, T.W.; Sinha, I.; Cohen, J.D. Emergent symbols through binding in external memory. *arXiv* **2020**, arXiv:2012.14601.
28. Webb, T.W.; Frankland, S.M.; Altabaa, A.; Segert, S.; Krishnamurthy, K.; Campbell, D.; Russin, J.; Giallanza, T.; O'Reilly, R.; Lafferty, J.; et al. The relational bottleneck as an inductive bias for efficient abstraction. *Trends Cogn. Sci.* **2024**, *28*, 829–843. [[CrossRef](#)]
29. Hill, F.; Tieleman, O.; Von Glehn, T.; Wong, N.; Merzic, H.; Clark, S. Grounded language learning fast and slow. *arXiv* **2020**, arXiv:2009.01719.
30. Wang, R.; Mao, J.; Hsu, J.; Zhao, H.; Wu, J.; Gao, Y. Programmatically grounded, compositionally generalizable robotic manipulation. *arXiv* **2023**, arXiv:2304.13826.
31. Song, H.; Kim, M.; Park, D.; Shin, Y.; Lee, J.G. Learning from noisy labels with deep neural networks: A survey. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 8135–8153. [[CrossRef](#)]
32. Wu, Y.; He, K. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, Munich, Germany, 8–14 September 2018; pp. 3–19.
33. Ioffe, S. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
34. Bansal, N.; Chen, X.; Wang, Z. Can we gain more from orthogonality regularizations in training deep networks? In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2018; Volume 31.
35. Choromanski, K.; Likhoshesterov, V.; Dohan, D.; Song, X.; Kane, A.; Sarlos, T.; Hawkins, P.; Davis, J.; Mohiuddin, A.; Kaiser, L.; et al. Rethinking attention with performers. *arXiv* **2020**, arXiv:2009.14794.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.