DAPE: Data-Adaptive Positional Encoding for Length Extrapolation

Chuanyang Zheng[1][*][†] Yihang Gao[2][*] Han Shi[3] Minbin Huang[1] Jingyao Li[1]
Jing Xiong[4] Xiaozhe Ren[3] Michael Ng[5] Xin Jiang[3] Zhenguo Li[3] Yu Li[1]
[1]CUHK   [2]NUS   [3]Noah's Ark Lab   [4]HKU   [5]HKBU

Abstract

Positional encoding plays a crucial role in transformers, significantly impacting model performance and length generalization. Prior research has introduced absolute positional encoding (APE) and relative positional encoding (RPE) to distinguish token positions in given sequences. However, both APE and RPE remain fixed after model training regardless of input data, limiting their adaptability and flexibility. Hence, we expect that the desired positional encoding should be data-adaptive and can be dynamically adjusted with the given attention. In this paper, we propose a **D**ata-**A**daptive **P**ositional **E**ncoding (DAPE) method, which dynamically and semantically adjusts based on input context and learned fixed priors. Experimental validation on real-world datasets (Arxiv, Books3, and CHE) demonstrates that DAPE enhances model performances in terms of trained length and length generalization, where the improvements are statistically significant. The model visualization suggests that our model can keep both local and anti-local information. Finally, we successfully train the model on sequence length 128 and achieve better performance at evaluation sequence length 8192, compared with other static positional encoding methods, revealing the benefit of the adaptive positional encoding method.

1   Introduction

Transformer-based models have shown state-of-the-art performances in many language processing tasks, including translation [6], question-and-answer [82, 29, 3], and commonsense reasoning [65]. The transformer mainly consists of attention block, feed-forward block, and positional encoding. Recent works [8] have proved that quadratic-cost attention from the softmax is necessary for better performance, especially in long-context processing. The attention block was originally designed by applying softmax to the key-query multiplication, which requires quadratic computational cost. To address such challenges, some efficient transformers were proposed, including sliding window transformers (e.g., Streaming LLMs [77]), linear transformers (e.g., Performer [17]), and sparse transformers (e.g., Reformer and sparse Sinkhorn transformer [66, 25]), etc. However, some negative results exist regarding efficient transformers' performances [80].

It has been noticed recently that well-designed positional encoding significantly improves the model performances, especially in the long-context tasks [33]. While transformer-based models exhibit satisfying performances in tasks of consistent length and distribution, their effectiveness tends to diminish sharply when the input length exceeds the training length, e.g., long document summarization, "needle in a haystack" search, and long text generation. To avoid the expensive computation in training, the training length is usually preferred to be relatively small due to the quadratic cost of

---

[*]Equal Contribution

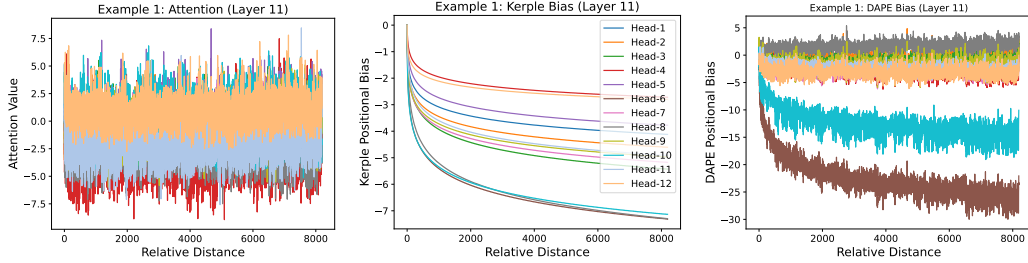[†]Contact Email: cyzheng21@link.cuhk.edu.hk

Figure 1: **Visualization of DAPE learned positional biases for the 8192th query position with key positions between 1 and 8192, while the training length is 512**. We notice that DAPE learns both local and anti-local position patterns. The model is trained with Equation 2: (1) The Attention is $\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top$; (2) The Kerple bias is $\boldsymbol{B}$; (3) The DAPE (with Kerple) bias is $f(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top, \boldsymbol{B})$. More examples are shown in Appendix I

softmax-based transformers. However, real-world applications often require processing longer input sequences, posing a significant challenge. Therefore, there is a growing interest in evaluating model performance by training on shorter sequences while testing on longer inputs. Standard transformers may not distinguish the ordering of tokens without external assistance. In practice, they depend on positional encoding to incorporate positional information, enabling the model to make meaningful token predictions. Without these encodings, token generation would lack the necessary contextual order, rendering the outputs nonsensical. The RoPE [62] positional encoding method demonstrated a notable performance degradation, failing entirely when the input length is double that of the training length [51, 10, 24]. A common characteristic among these positional encodings is their pre-defined and static nature. Specifically, they are fixed across various tasks and models, which may lead to their inability to adapt to varying input lengths and contexts effectively. To address this issue, recent works have introduced Functional Interpolation for Relative Positional Encoding (FIRE) [41], which utilizes a neural network to learn an implicit mapping from input positions to positional bias. A functional approach to positional encoding that dynamically adjusts positional biases based on semantic information (input context) allows the model to empower adaptability beyond the fixed inductive bias as adopted in previous studies (such as RoPE [62] and Alibi [52]). Although FIRE utilizes MLPs to learn positional embeddings, these embeddings remain fixed across different tasks once the training is completed. Intuitively, the learned static positional encoding (such as Kerple and FIRE) is an average optimal solution across all training samples. Consequently, while they might be generally effective, they are inherently suboptimal for any specific instance. This static nature limits their flexibility and applicability in various real-world scenarios that deviate from the training context. In this paper, we introduce a data-adaptive positional encoding (DAPE) method, inspired by the limitations of static PEs. DAPE dynamically adjusts the PE based on the semantic information (e.g., the current attention value) a and the positional indicator b. The proposed PE is represented by MLPs due to their universal approximatability, i.e., MLPs(a, b). We note that DAPE is compatible with all additive relative PEs and offers advantages in terms of interpretability and ease of implementation. The proposed DAPE incorporates both the semantic and the positional information, making the PE adaptive with the input data. The adaptivity allows DAPE to overcome the inflexibility and achieve relatively optimal performance for each individual instance by dynamically adjusting on each specific input data. To the best of our knowledge, this is the first semantically dependent and adaptive positional encoding method introduced in transformer architectures.

The paper is organized as follows. In Section 2, we review some related works on positional encoding methods, including absolute and relative positional encodings as well as the potentially no positional encoding in some transformer models. In Section 3, we introduce the proposed DAPE method with implementation on multi-head attention and analysis on computational costs. We conduct comprehensive experiments on DAPE, validating its effectiveness and performances on various language tasks and datasets, as reported in Section 4. In Section 6, some concluding remarks and potential future works are presented.

2 Related Works

No positional encoding. Haviv et al. [30] show that decoder-only Transformers with causal attention masks can learn positional information even without any explicit positional encoding. Recently, Kazemnejad et al. [33] proved the effectiveness of no positional encoding (NoPE) [71]. Although

the NoPE can implicitly catch the positional information, it performs poorly compared with some explicit positional encoding methods [41].

Absolute positional encoding. Vaswani et al. [69] proposed Absolute positional encoding (APE) to endow transformers with positional information. In particular, in the first layer, a (learnable or fixed sinusoidal) real-valued encoding [69, 35, 42, 70, 47] $e_i \in \mathbb{R}^d$ is assigned to each position i, leading to an APE matrix $E = [e_1, \cdots, e_n]^\top$, which will be added to the input sequence. Though simple and straightforward, APE-based Transformers usually generalize poorly to longer sequences [52].

Relative positional encoding. Relative Positional Encoding (RPE) is another popular way to encode positional information [58, 56, 52], One popular RPE method in large language models is rotary positional encoding (RoPE) [62, 18, 67]. RoPE rotates the query and key vectors with an angle proportional to their absolute positions before the dot product attention, which results in attention being a function of the relative distance between the tokens, capturing the relative positional information. Press et al. [52] and Kazemnejad et al. [33] found that RoPE-based language models have poor length generalization. To address this, positional interpolation (PI) [11] is proposed to extend the context window. Following the direction, there are LongLora [12], LongRope [24], YaRN [51] and CLEX [10]. Another popular direction is additive positional encoding. For most of these additive RPE methods, the computation of the (pre-softmax) attention logits can be unified using the following formula:

$$A_{\mathrm{RPE}}(X) = XW_Q(XW_K)^\top + B,$$

where the bias matrix $B \in \mathbb{R}^{n \times n}$ is induced by the position encoding function $b : \mathbb{N}^2 \to \mathbb{R}$ and the (i, j)-th entry of B is defined as $b(i, j)$. Different formulations and parameterizations of b lead to various RPE variants. Several methodologies that facilitate arbitrary sequence lengths include T5's RPE [56], Alibi [52], Kerple [13], Sandwich [14], and FIRE [41]. Currently, additive RPE delivers relatively robust performance in length extrapolation without necessitating additional operations. Alibi constructs the bias matrix B utilizing prior knowledge, resulting in a basis matrix devoid of learnable parameters [52]. Conversely, both Kerple [13] and Sandwich [14] incorporate two learnable parameters to facilitate the learning of a bias matrix while retaining some elements of priors. FIRE suggests adopting a learnable continuous function, such as MLPs, to convert input positions to biases [41]. Observing these developments, it becomes evident that the next generation of bias matrices will likely incorporate adaptivity and flexibility. Based on this insight, we propose our method DAPE, a semantically adaptive method.

3 Method

In this section, we formally introduce DAPE (data-adaptive positional encoding), a new relative positional encoding approach that further enhances transformer performance. Compared with previous works on static positional encoding methods, DAPE adopts semantically adaptive positional bias matrices depending on input context. We first demonstrate that most of the popular positional bias matrices are fixed once the training is finished, independent of the input sequences. To address this limitation, we then accordingly develop DAPE that captures the implicit relationships by MLPs and adjusts the bias matrices based on input context. Furthermore, we discuss a variant of DAPE with residual connections and its extensions to multi-head transformers.

3.1 Additive Relative Positional Encoding

For most additive RPE methods, the computation of pre-softmax attention logits can be unified under the following formula:

$$A_{\mathrm{RPE}}(X) = XW_Q(XW_K)^\top + B, \tag{1}$$

where the bias matrix $B \in \mathbb{R}^{n \times n}$ is induced by the position encoding function $b : \mathbb{N}^2 \to \mathbb{R}$ and the (i, j)-th entry of B is defined as $b(i, j)$. Various formulations and parameterizations of b give rise to different variants of RPE. Examples of additive RPE include: (1) Alibi: $b(i, j) = -r|i - j|$, with the scaler $r > 0$ as a hyper-parameter; (2) Kerple: $b(i, j) = -r_1 log(1 + r_2|i - j|)$ with r_1 and r_2 are two learnable parameters; (3) FIRE: $b(i, j) = f_\theta \left(\frac{\psi(i-j)}{\psi(\max\{L,i\})} \right)$, where the positional encoding function f_θ parameterized by θ is learned from data and ψ is a transformation function aimed at assigning more model capacity to local positions.

We observe from the formulation of those additive RPEs that they remain static once the training process is completed and depend solely on the positions, regardless of the input context. This inflexibility and lack of adaptivity can lead to performance degradation, especially in tasks involving long-context generation and reasoning. Intuitively, the learned static RPEs are optimal on average across all training samples. However, this means they are suboptimal when considering each individual instance, as they cannot adapt to specific tasks. To address these challenges and enhance model performance, it is essential to adopt an alternative approach using a semantically adaptive RPE that depends on input context.

3.2 Data-Adaptive Positional Encoding

For simplicity, we first consider the single-head case and the extension to the multi-head transformer will be discussed subsequently. The design of data-adaptive positional encodings in natural language tasks is motivated by the need to capture the intricate relationships between tokens. Arora et al. [5] reveals that associate recall accounts for most of the perplexity difference between transformer, RNN-based, and convolution models. For example, we consider a consistent pairing that "Hakuna" is always followed by "Matata" in a long paragraph. This pattern suggests a reduced reliance on positional information in favor of enhancing token embedding similarity, thus allowing for 'Hakuna' to be effectively linked with a preceding 'Matata'. Similarly, in tasks involving long-context understanding and search, semantic similarity should be prioritized in the attention mechanism rather than being overshadowed by positional encodings, which can be less relevant over long distances. Consequently, the transformer should preserve information without being influenced overly by positional distance. Instead, a satisfactory PE should integrate both semantic and positional information. Therefore, a semantically dependent positional encoding approach is preferable and expected to enhance model performances. Here, we use the attention $\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top$ to represent the semantic information and positional bias matrices $\boldsymbol{B}$ (e.g., Alibi and FIRE) to capture positional details. Then the context-adaptive PE is described by $f(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top, \boldsymbol{B})$, where $f(\cdot)$ is an implicit function that integrates both semantic and positional data into the desired positional encodings. Thus, the pre-softmax attention logit incorporated with DAPE is formulated as

$$\boldsymbol{A}_{\mathrm{DAPE}}(\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top + f(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top, \boldsymbol{B}). \tag{2}$$

Here, $f : \mathbb{R}^{T \times T} \times \mathbb{R}^{T \times T} \to \mathbb{R}^{T \times T}$ is an element-wise function. In practice, we utilize a two-layer *LeakyReLU* neural network to parameterize $f(\cdot)$ due to its universal approximability [36]. All parameters are learned directly from the data during the training process. This architecture allows $f(\cdot)$ to dynamically adjust positional embeddings based on the input context, ensuring that the encoding method is both adaptive and dependent on the input data.

Different from FIRE, which also models the implicit positional encoding by MLPs, our approach additionally integrates semantic information. This integration enables the adaptivity, flexibility, and context-dependency of the positional encodings. Significantly, our method is compatible with most additive RPE techniques, as these commonly involve positional bias matrices $\boldsymbol{B}$ that inherently contain positional relations. Unlike previous RPEs, which rely solely on absolute positional differences, our DAPE method, can be seen as utilizing multi-level positional bias matrices. Here, the bias matrices dynamically adjust based on the input context, offering a more reasonable and responsive encoding mechanism.

Expressiveness. Due to the universal approximability of (*LeakyReLU*) neural networks [36], $f(\cdot)$ is capable of capturing complex relationships between the desired positional encoding and both semantic and positional information. Regardless of the semantic component, when the relative position $i - j$ is used as input, DAPE can realize classical additive RPEs (e.g., Alibi and Kerple), according to [41]. This demonstrates the versatility of DAPE in accommodating traditional encoding schemes while also offering enhanced capabilities. There exists a fundamental trade-off between the expressiveness and computational costs. Wider hidden layers lead to higher expressiveness but also contribute to more computational costs. In practice, we find that two-layer neural networks with 32 hidden units per layer provide sufficient expressiveness to deliver satisfactory performance, balancing complexity and efficiency effectively.

Discussion. We can also interpret the proposed method from an alternative perspective. In the standard transformer architecture, the pre-softmax attention typically involves the key-query similarity and the positional encoding by either addition (in the form of $a + b$, e.g., Alibi and Kerple) or multiplication (in the form of $a * b$, e.g., RoPE). Here, we propose a unified approach by replacing them with

learnable MLPs, i.e., MLP(a, b). This configuration allows the model to learn the desired relationship between the pre-softmax attention, the key-query similarity and the positional encoding. It can also be regarded as a new transformer architecture that empower the transformer with additional MLPs on pre-softmax attentions.

A variant of DAPE with residual connections. It is well-known that deep neural networks may suffer from gradient vanishing. To further enhance the practical performances, we introduce the residual connection for positional information. Consequently, Equation 2 is modified as follows:

$$A_{\text{DAPE}}(\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top + \boldsymbol{B} + f(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top, \boldsymbol{B}). \tag{3}$$

In this reformulation, $f(\cdot)$ acts as an adaptive correction term to the traditionally fixed RPE, dynamically adjusting the positional bias matrices $\boldsymbol{B}$ based on both semantic and positional inputs. In Section 4, we empirically explore the impact of residual connections in DAPE. Our observations reveal that for well-behaved bias matrices $\boldsymbol{B}$, the DAPE model with residual connections, as specified in Equation 3, is preferable. Conversely, if the bias matrix is underperforming but still conveys positional information, the original implementation in Equation 2 is more effective.

Multi-head DAPE. In its simplest form, DAPE is considered for a single-head case as described in Equation 2 and Equation 3. However, adopting a multi-head mechanism significantly enhances model capabilities. To effectively combine both semantic and positional information, the DAPE in a multi-head setup processes the key-query similarities and bias matrices from all heads. Specifically, for an h-head layer, the function $f(\cdot)$ inputs a $2h$-dimensional concatenation of key-query similarities and positional biases. It then outputs h-dimensional vectors, where each element corresponds to the DAPE for the respective head. We have shown the code implementation in Appendix J. Importantly, semantic and positional information across different heads are processed simultaneously within the same MLPs, rather than sequentially. This approach not only improves computational efficiencies through parallel processing but also capitalizes on the richer semantic information available across all heads. Compared to the key-query similarity derived from a single head, the comprehensive attention from all heads yields more substantial semantic information.

Computational costs analysis. Here, we evaluate the additional computational costs introduced by the DAPE method, compared with the classical positional encoding methods (e.g., Alibi and Kerple). We consider a transformer model with h heads and assume a sequence length of N and all hidden dimensions in the attention layer being d. Then the total computational cost for a standard transformer equipped with classical PEs is $\mathcal{O}\left(hN^2d + hNd^2\right)$. When incorporating the proposed DAPE, which employs two-layer MLPs with hidden dimension D_{DAPE}, the additional computational costs are $\mathcal{O}\left(hN^2D_{\text{DAPE}}\right)$. If the hidden dimensions $D_{\text{DAPE}} \ll d$, the incremental computational cost introduced by DAPE is not significant.

4 Experiment

Baselines. We evaluate the proposed DAPE against a range of established baselines, including NoPE [33], RoPE [62], YaRN [51], Randomized RoPE [57, 31], T5's Bias [56], Alibi [52], Kerple [13], and FIRE [41]. For RoPE, the randomized positional encoding [57, 31] is applied to enhance the model performance, extending the randomized length to four times that of the training length.

Datasets. Our analysis involves training language models on the Arxiv and Books3 datasets, which are frequently used benchmarks for evaluating model performance [52, 13, 41, 24]. We start our evaluation by comparing the last 256 tokens' zero-shot perplexity across different input lengths. Besides perplexity as evaluation metrics, we also employ the downstream datasets in randomized positional encoding [57] to evaluate DAPE, where details are included in Appendix D.

Experiment settings. Initially, we compare DAPE with other baselines at training lengths of 128, 512, and 1024, with model size 125M decoder-only Transformers [9], whose configuration is shown in Appendix B. Subsequently, we evaluate the performance of larger model size 350M, DAPE variants and explore the impact of hidden dimension of MLPs D_{DAPE}. We also examine the computational efficiency of DAPE, focusing on processing times. Additionally, we provide visualizations of the DAPE bias in the Appendix I. Finally, we also evaluate DAPE on algorithmic reasoning datasets via accuracy metrics.
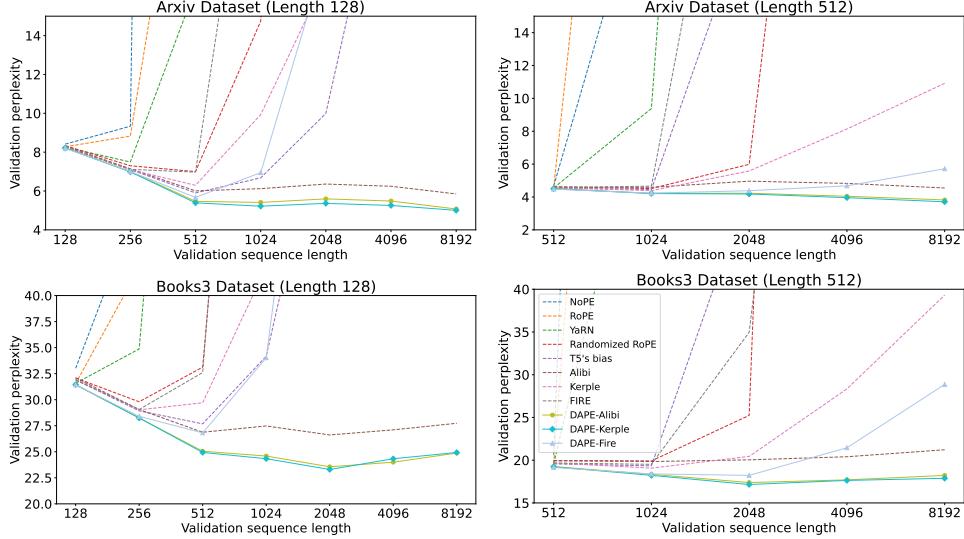
Figure 2: **Comparisons with baselines:** performance with training lengths 128 and 512 on Arxiv and Books3 datasets.

4.1 Comparisons with Baselines

DAPE's superior performance within training length and beyond training length, compared to all baselines. As shown in Figure 2 and Table 5, DAPE consistently outperforms established baselines such as RoPE, Alibi, and Kerple across various settings. Notably, DAPE-Kerple (the positional information in DAPE comes from Kerple bias matrices) outstands in both short and long training lengths (128 and 512), compared to previous RoPE, T5's bias, and so on. It demonstrates that the semantic adaptivity of DAPE significantly enhances its state-of-the-art performance against all other static positional encoding methods.

The performance on longer training length 1024. As shown in Figure 3, the proposed method consistently delivers state-of-the-art performance for the training length of 1024. When the evaluation extends to 2048, both DAPE-Kerple and DAPE-FIRE achieve notable results, recording performances of 3.91 and 3.93 perplexity scores, respectively. Remarkably, DAPE-FIRE behaves well at the longer evaluation length of 8192, achieving a performance of 3.91 scores and surpassing Alibi's score of 4.28. These findings reveal that DAPE sustains robust performance with a longer training length of 1024.
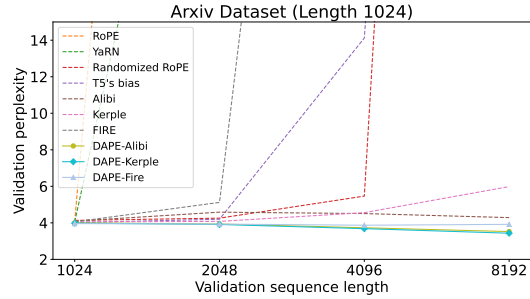

Figure 3: **Results on the training length 1024.**

DAPE enhances intra-length performance, indicating that its lower perplexity may come from thorough utilization of entire sentences but not disregarding long-distance information (Also proved in Figure 1). Compared to Alibi, Kerple, and FIRE, the adapted versions DAPE-Alibi, DAPE-Kerple, and DAPE-FIRE demonstrate consistently and significantly better intra-length performance. With the growing sequence length, the Alibi tends to transition from full attention to almost local attention, and this is why Alibi is worse than most baselines within training length but better beyond training lengths. The results (as shown in Table 5) indicate that the superior intra-length performance of DAPE is statistically significant, with a p-value less than 0.05. Therefore, the consistent intra-length performances across various training lengths indicate that the lower perplexity of DAPE results from effectively utilizing the entire sequence, rather than focusing on local parts and neglecting long-distance information.

6

DAPE significantly improves length extrapolation performance, compared to ALibi, Kerple, and FIRE. DAPE-Kerple significantly surpasses competitors like vanilla Kerple when training and evaluating at different lengths. On the Arxiv dataset trained at a length of 128, DAPE-Kerple achieves a remarkably low perplexity of 5.00 at an evaluation length of 8192, in stark contrast to Kerple's 31.93. Similarly, on the Books3 dataset with a training length of 512, DAPE-Kerple records a perplexity of 17.88 at the same extended evaluation length, far outperforming Kerple's 39.31. These results affirm that DAPE, through its semantic adaptivity and flexibility, consistently enhances performance beyond training lengths, eclipsing static positional encoding methods.

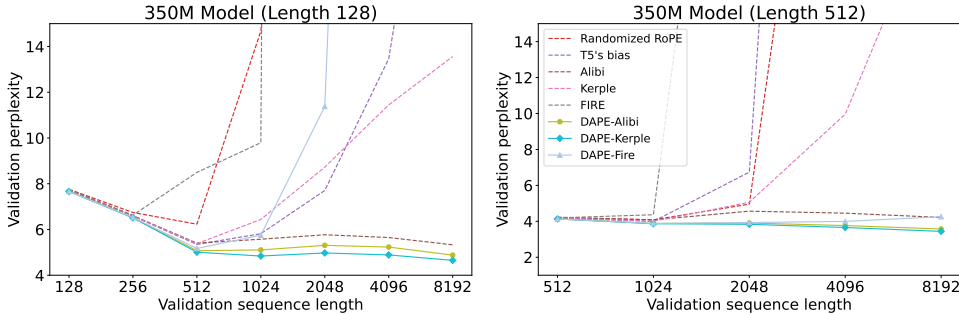## 4.2 The Effect of Model Size

Figure 4: **The effect of model size:** for the 350M model, the performance with training lengths 128 and 512 on the Arxiv dataset.

DAPE enhances performance with increasing model sizes. As the model size grows (as shown in Figure 4), DAPE consistently demonstrates an improvement in performance metrics. When the model size is augmented from 125M to 350M, the perplexity at an evaluation sequence length of 8192 (with a training length of 512) for DAPE-Alibi shows a notable decrease from 3.82 to 3.57. These numbers are appreciably smaller than those recorded for original Alibi, which decreases from 4.54 to 4.21 in perplexity, indicating a robust performance improvement. Additionally, DAPE-Kerple significantly reduces the perplexity for Kerple, bringing it down from an initial 22.76 to an impressive 3.43. These results confirm that DAPE retains its efficacy and continues to perform well even as the model size is increased, mainly due to the adoption of semantically adaptive PEs.

DAPE methods almost are ranked top-3 with large model size. With the incremental model size, DAPE-FIRE begins to match, and nearly approach, the performance levels of Alibi. Initially, at a model size of 125M and a training length of 512, DAPE-FIRE achieves a perplexity of 5.71 at an evaluation sequence length of 8192, while Alibi stands at a perplexity of 4.54. However, as the model size is increased to 350M, the performance gap significantly narrows. Specifically, DAPE-FIRE outperforms Alibi regarding the perplexity scores when the evaluation length is smaller than 4096, as the model size grows for evaluation. In conclusion, as shown in Figure 4, we observe that the DAPE methods almost win the top-3 among all positional encoding methods. This trend underlines the scalability and adaptability of DAPE, emphasizing its potential to handle more substantial computation challenges.

4.3 Different Variants of DAPE

In this section, we evaluate the performance of DAPE across its various forms. Our analysis focuses on DAPE-Kerple. Notably, as shown in Figure 5, all variants of DAPE surpass the baseline performance of Kerple. The *Addition_Residual* variant of DAPE, while requiring less computational effort, delivers relatively inferior results. As illustrated in Figure 5, concatenation methods (either *Concat* or *Concat_Residual*) outperform the *Addition_Residual* approach, for both the training length of 128 and 512. Furthermore, both *Concat* and *Concat_Residual* exhibit comparable performance metrics. Specifically, at a training length of 128, *Concat_Residual* records a score of 5.00 and *Concat* scores 5.03 at an evaluation length of 8192, whereas *Add_Residual* posts a 5.17 perplexity score. With a training length of 512, *Concat_Residual* achieves a score of 3.70, and *Concat* scores 3.69 at an evaluation length of 8192, compared to *Add_Residual*'s 3.75. Based on the current observation, the different variants of DAPE show comparable performances, compared to baselines.
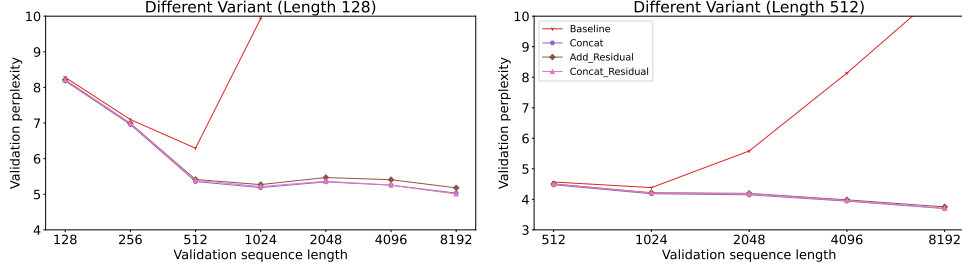
Figure 5: **Different variants of DAPE:** the DAPE-Kerple performance under different variants. (1) *Add_Residual*: $XW_Q(XW_K)^\top + B + f(XW_Q(XW_K)^\top + B)$; (2) *Concate*: $XW_Q(XW_K)^\top + f(XW_Q(XW_K)^\top, B)$; (3) *Concate_Residual*: $XW_Q(XW_K)^\top + B + f(XW_Q(XW_K)^\top, B)$.

4.4 The Effect of the Hidden Dimension $D_{\textbf{DAPE}}$

Even small $D_{\textbf{DAPE}}$ can improve the performance. The experiments are conducted with Alibi and DAPE-Alibi. As shown in Appendix Figure 6, when considering the training length 128 and D_{DAPE} is set as 4, the DAPE-Alibi achieves 8.25 at evaluation length 128 and 5.67 at length 8192, which is better than Alibi's 8.31 and 5.85. Whatever D_{DAPE} is 4, 16 32, or 64, the performance is always better than the original Alibi at all evaluation lengths. This suggests the effectiveness of DAPE, even with smaller D_{DAPE}.

The choice of $D_{\textbf{DAPE}}$. Based on the experiment, overly small values of D_{DAPE} can degrade performance, although they still perform better than the baseline. Conversely, larger values of D_{DAPE} increase computational costs. The function $f(\cdot)$ is implemented as a two-layer MLP, where the input dimension is either the head number or twice the head number, and the output dimension is the head number. Therefore, we recommend setting the hidden dimension to the head number to prevent information loss and ensure the capacity of $f(\cdot)$.

4.5 The Time Cost

Table 1: The time cost (millisecond) under different testing lengths, with $D_{\textbf{DAPE}}$ as 32 and default batch size 1, with training length 512.

Method	350M Total	Ratio	2.7B Total	Ratio	6.7B Total	Ratio
RoPE [62]	210.01	0.9366	472.63	1.1187	635.57	0.8858
T5's bias [56]	355.16	1.5839	537.62	1.2725	808.85	1.1273
ALiBi [52]	172.60	0.7697	325.95	0.7715	596.77	0.8317
Kerple [13]	189.91	0.8469	370.32	0.8765	661.82	0.9224
FIRE [41]	248.13	1.1066	432.63	1.0240	797.68	1.1118
DAPE-Kerple	224.22	1.0000	422.48	1.0000	717.46	1.0000

Practical additional time cost. The additional training ratio will gradually decrease with a larger model size, compared to baseline Kerple.. The cost of Feed-Forward Network is: $O(Nd_{head}^2 d_{hidden}^2)=aNd_{head}^2 d_{hidden}^2$, where a is a constant, N is the sequence length, d_{head} is the attention head number and d_{hidden} is the dimension for attention calculation. The cost of Attention: $O(N^2 d_{head} d_{hidden})=bN^2 d_{head} d_{hidden}$, where b is a constant. The additional cost of DAPE: $O(N^2 d_{head} d_{DAPE})=cN^2 d_{head} d_{DAPE}$, where c is a constant. The cost ratio is $\frac{aNd_{head}^2 d_{hidden}^2 + bN^2 d_{head} d_{hidden}}{aNd_{head}^2 d_{hidden}^2 + bN^2 d_{head} d_{hidden} + cN^2 d_{head} d_{DAPE}} = \frac{ad_{head} d_{hidden}^2 + bNd_{hidden}}{ad_{head} d_{hidden}^2 + bNd_{hidden} + cNd_{DAPE}}$. Therefore, with the fixed sequence length and d_{DAPE}, with the model becomes larger (with bigger d_{head} and d_{hidden}), the additional cost ratio of DAPE will greatly become smaller. Also, we have shown in Figure 6 that d_{DAPE} still works well with very small value, such as 4.

Table 2: Train on length 40 with 200k steps, and test from lengths 41 to 500. The random accuracy is 50%, except for MODULAR ARITHMETIC (SIMPLE), CYCLE NAVIGATION, BUCKET SORT, SOLVE EQUATION and MODULAR ARITHMETIC, where it is 20%. ††† denotes permutation-invariant tasks, which are expected to be solved without positional information.

| Level | Task | Randomized | | | | | | | DAPE (Ours) | | |
		Learned	sin / cos	RoPE	Relative [19]	ALiBi	Kerple	FIRE	Alibi	Kerple	FIRE
R	EVEN PAIRS	50.04	91.27	99.98	96.60	73.52	57.50	73.86	99.99	99.58	**100**
	MODULAR ARITHMETIC (SIMPLE)	19.95	20.39	21.35	20.84	20.02	21.79	21.09	23.58	**24.47**	24.46
	PARITY CHECK†††	50.14	50.52	50.05	50.09	50.09	50.07	**50.97**	50.30	50.07	50.04
	CYCLE NAVIGATION†††	24.97	25.37	27.63	26.95	24.64	29.47	28.41	22.99	**34.53**	27.54
DCF	STACK MANIPULATION	59.92	65.92	61.49	64.73	66.42	66.93	69.33	68.18	**72.04**	70.90
	REVERSE STRING	52.76	67.28	65.23	65.59	71.09	71.54	65.89	73.37	70.74	**76.40**
	MODULAR ARITHMETIC	31.00	30.70	31.25	31.74	30.56	24.79	30.92	31.34	**32.37**	31.50
	SOLVE EQUATION	20.00	19.97	21.85	**22.93**	19.92	21.15	22.06	20.03	22.49	22.42
CS	DUPLICATE STRING	52.77	65.44	64.97	67.66	65.13	66.72	69.03	70.84	**72.95**	72.71
	MISSING DUPLICATE	50.38	49.78	63.37	72.34	74.21	79.06	79.27	83.41	87.57	**89.17**
	ODDS FIRST	52.77	58.61	61.00	61.57	59.88	62.59	63.28	63.78	**67.08**	66.34
	BINARY ADDITION	54.63	55.78	55.59	56.96	54.72	56.35	55.70	59.71	**60.88**	56.62
	COMPUTE SQRT	50.47	51.11	51.88	51.63	50.63	51.11	50.80	51.64	51.33	**52.46**
	BUCKET SORT†††	98.32	98.92	98.12	99.31	98.45	99.38	**99.57**	99.38	98.81	99.37

4.6 The Visualization of DAPE

In this subsection, we present the visualization of learned positional encoding biases from a DAPE-Kerple model pretrained on Arxiv (training length is 512). We plot the learned positional encoding bias for the query token at the 8192th position, for all the attention heads from selected layers in Figure 1. We would like to highlight two features of DAPE. First, in different attention heads, the bias matrix of DAPE learns both local and "anti-local" attention patterns that emphasize more on far-away keys (just like FIRE), compared to a fixed local inductive bias (such as Kerple and Alibi). Secondly, the bias matrix can be dynamically adjusted with different attention values, compared to the static bias fixed for all attentions. We have shown more examples, including different layers and different samples, in Appendix I.

4.7 Experiments on CHE Benchmark

Besides employing perplexity as an evaluation metric, we also evaluated DAPE on downstream Chomsky Hierarchy Evaluation Benchmark (CHE) [21] (need to utilize the whole sentence information to generate correct answers) to further discuss its effects. The experimental setup follows randomized positional encodings [57], detailed in Table 4, with the experiment setting shown in Appendix Section D. Overall, FIRE outperforms Kerple in 9 out of 14 tasks, while Kerple outperforms Alibi in 11 out of 14 tasks. This observation aligns with findings in [41], suggesting that the experiments in Table 2 are reliable and reflect the performance of positional encoding in downstream tasks.

DAPE works better on permutation-variant tasks. DAPE (with Kerple and FIRE) presented the best performance in 10 out of 11 permutation-variant tasks (which require positional information), achieving the second-best performance in the SOLVE EQUATION task. This underscores the efficacy of DAPE with semantic adaptivity in handling permutation-variant challenges.

DAPE's performance on permutation-invariant tasks. In tasks that are permutation-invariant, where positional information is non-critical, DAPE demonstrated comparable performance. Notably, DAPE-Alibi achieved scores of 50.30 on PARITY CHECK and 99.38 on BUCKET SORT tasks, compared to the highest scores of 50.97 and 99.57, respectively, demonstrating competitive performances.

Comparative performance improvements. DAPE consistently enhanced performance across various tasks, especially on permutation-variant tasks. Specifically, DAPE improved upon Alibi and FIRE's results in all 11 tested permutation-invariant tasks. Similarly, it outperformed Kerple in 10 of these tasks. These results highlight the effectiveness of DAPE over static positional encoding methods like Alibi, Kerple, and FIRE, resulting from its dynamic adaptivity.

5 Evaluation Protocol

In this work, we initially utilize the model to process the entire input sentence and subsequently select the final 256 tokens for perplexity computation. This approach contrasts with a variety of other studies, which employ methods that process the full sequence during perplexity calculations [51, 31, 12, 24, 10, 41]. As a result, our reported baseline perplexity is comparatively higher than the results presented in ALiBi [52], which adopt a non-overlapping evaluation strategy. This method divides sequences longer than L into multiple segments of length L, thereby yielding lower perplexity figures.

Though ALiBi [52] and Kerple all claim that they use non-overlapping evaluations, their reported results are different. For example, in the ALiBi paper Table 2, the sinusoidal position encoding perplexity increases from 20.05 (evaluation length 512) to 406.01 (evaluation length (evaluation length 15512), while in Kerple paper Table 3 the sinusoidal position encoding perplexity from 33 to 30046. This may be caused by **different evaluation protocols and training strategies.**

Recommended Protocols. We strongly recommend that researchers process the entirety of the sequence before selecting the last K tokens for the purpose of calculating perplexity. The rationale behind processing the whole sentence is that it provides a comprehensive evaluation of the model's capability to handle long-context dependencies, thus offering a more accurate reflection of its performance. Following this step, we advocate for the selection of the last K tokens to compute perplexity, ensuring that the same number of tokens is used across different evaluation lengths, which promotes consistency and comparability in the results.

Release this work's code for future work. In light of this methodology, we have made our code publicly available to other researchers in the field. This initiative aims to facilitate a standardized comparison and evaluation of their respective methods, thereby advancing the collective understanding of model performance in relation to perplexity calculations.

6 Conclusion

In this paper, we propose the data-adaptive positional encoding (DAPE) by incorporating both the semantic and the positional information to improve the model performance. We show that the additional computation introduced by DAPE is not significant under proper choices of hyperparameters. We conduct comprehensive experiments on Arxiv, Books3, and CHE to validate the effectiveness of the proposed method, revealing that the adaptive PE method has advantages over static PEs. We believe that the DAPE could benefit the whole community, especially on length generalization tasks.

References

[1] Muhammad Adnan, Akhil Arunkumar, Gaurav Jain, Prashant J Nair, Ilya Soloveychik, and Purushotham Kamath. Keyformer: KV cache reduction through key tokens selection for efficient generative inference. *arXiv preprint arXiv:2403.09054*, 2024.

[2] Devanshu Agrawal, Shang Gao, and Martin Gajek. Can't remember details in long documents? you need some r&r. *arXiv preprint arXiv:2403.05004*, 2024.

[3] Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontanon, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, et al. CoLT5: Faster long-range transformers with conditional computation. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[4] Chenxin An, Fei Huang, Jun Zhang, Shansan Gong, Xipeng Qiu, Chang Zhou, and Lingpeng Kong. Training-free long-context scaling of large language models. *arXiv preprint arXiv:2402.17463*, 2024.

[5] Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. *arXiv preprint arXiv:2312.04927*, 2023.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations*, 2015.

[7] Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended long short-term memory. *arXiv preprint arXiv:2405.04517*, 2024.

[8] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[10] Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. CLEX: Continuous length extrapolation for large language models. In *International Conference on Learning Representations*, 2023.

[11] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023.

[12] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. LongLoRA: Efficient fine-tuning of long-context large language models. *International Conference on Learning Representations*, 2023.

[13] Ta-Chung Chi, Ting-Han Fan, Peter J Ramadge, and Alexander Rudnicky. KERPLE: Kernelized relative positional embedding for length extrapolation. *Advances in Neural Information Processing Systems*, 35:8386–8399, 2022.

[14] Ta-Chung Chi, Ting-Han Fan, Alexander Rudnicky, and Peter Ramadge. Dissecting transformer length extrapolation via the lens of receptive field analysis. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13522–13537, 2023.

[15] Ta-Chung Chi, Ting-Han Fan, and Alexander I Rudnicky. Attention alignment and flexible positional embeddings improve transformer length extrapolation. *arXiv preprint arXiv:2311.00684*, 2023.

[16] Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, 1956.

[17] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021.

[18] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[19] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, 2019.

[20] Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

[21] Gregoire Deletang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. Neural networks and the chomsky hierarchy. In *International Conference on Learning Representations*, 2022.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[23] Hantian Ding, Zijian Wang, Giovanni Paolini, Varun Kumar, Anoop Deoras, Dan Roth, and Stefano Soatto. Fewer truncations improve language modeling. *arXiv preprint arXiv:2404.10830*, 2024.

[24] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. LongRoPE: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753*, 2024.

[25] Zafeirios Fountas, Martin A Benfeghoul, Adnan Oomerjee, Fenia Christopoulou, Gerasimos Lampouras, Haitham Bou-Ammar, and Jun Wang. Human-like episodic memory for infinite context llms. *arXiv preprint arXiv:2407.09450*, 2024.

[26] Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.

[27] Yihang Gao, Chuanyang Zheng, Enze Xie, Han Shi, Tianyang Hu, Yu Li, Michael K Ng, Zhenguo Li, and Zhaoqiang Liu. On the expressive power of a variant of the looped transformer. *arXiv preprint arXiv:2402.13572*, 2024.

[28] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[29] Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. LongT5: Efficient text-to-text transformer for long sequences. *Findings of the Association for Computational Linguistics: NAACL*, 2022.

[30] Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. Transformer language models without positional encodings still learn positional information. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1382–1390, 2022.

[31] Zhenyu He, Guhao Feng, Shengjie Luo, Kai Yang, Di He, Jingjing Xu, Zhi Zhang, Hongxia Yang, and Liwei Wang. Two stones hit one bird: Bilevel positional encoding for better length extrapolation. *arXiv preprint arXiv:2401.16421*, 2024.

[32] Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. LLM maybe LongLM: Self-extend LLM context window without tuning. *arXiv preprint arXiv:2401.01325*, 2024.

[33] Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

[34] Guolin Ke, Di He, and Tie-Yan Liu. Rethinking positional encoding in language pre-training. In *International Conference on Learning Representations*, 2020.

[35] Shun Kiyono, Sosuke Kobayashi, Jun Suzuki, and Kentaro Inui. SHAPE: Shifted absolute position embedding for transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3309–3321, 2021.

[36] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[37] Jingyao Li, Pengguang Chen, Zexin He, Shaozuo Yu, Shu Liu, and Jiaya Jia. Rethinking out-of-distribution (OOD) detection: Masked image modeling is all you need. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11578–11589, June 2023.

[38] Jingyao Li, Pengguang Chen, and Jiaya Jia. Motcoder: Elevating large language models with modular of thought for challenging programming tasks, 2024.

[39] Jingyao Li, Pengguang Chen, Shengju Qian, and Jiaya Jia. Tagclip: Improving discrimination ability of open-vocabulary semantic segmentation, 2023.

[40] Jingyao Li, Pengguang Chen, Shaozuo Yu, Shu Liu, and Jiaya Jia. Bal: Balancing diversity and novelty for active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3653–3664, 2024.

[41] Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. Functional interpolation for relative positions improves long context transformers. In *International Conference on Learning Representations*, 2023.

[42] Tatiana Likhomanenko, Qiantong Xu, Gabriel Synnaeve, Ronan Collobert, and Alex Rogozhnikov. CAPE: Encoding relative positions with continuous augmented positional embeddings. *Advances in Neural Information Processing Systems*, 34:16079–16092, 2021.

[43] Bin Lin, Tao Peng, Chen Zhang, Minmin Sun, Lanbo Li, Hanyu Zhao, Wencong Xiao, Qi Xu, Xiafei Qiu, Shen Li, et al. Infinite-LLM: Efficient LLM service for long context with distattention and distributed kvcache. *arXiv preprint arXiv:2401.02669*, 2024.

[44] Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*, 2024.

[45] Jiaheng Liu, Zhiqi Bai, Yuanxing Zhang, Chenchen Zhang, Yu Zhang, Ge Zhang, Jiakai Wang, Haoran Que, Yukang Chen, Wenbo Su, et al. Eˆ2-LLM: Efficient and extreme length extension of large language models. *arXiv preprint arXiv:2401.06951*, 2024.

[46] Xiaoran Liu, Hang Yan, Chenxin An, Xipeng Qiu, and Dahua Lin. Scaling laws of RoPE-based extrapolation. In *International Conference on Learning Representations*, 2023.

[47] Xuanqing Liu, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. Learning to encode position for transformer with continuous dynamical model. In *International Conference on Machine Learning*, pages 6327–6335. PMLR, 2020.

[48] Shengjie Luo, Shanda Li, Tianle Cai, Di He, Dinglan Peng, Shuxin Zheng, Guolin Ke, Liwei Wang, and Tie-Yan Liu. Stable, fast and accurate: Kernelized attention with relative positional encoding. *Advances in Neural Information Processing Systems*, 34:22795–22807, 2021.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

[50] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, et al. RWKV: Reinventing RNNs for the transformer era. *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

[51] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *International Conference on Learning Representations*, 2023.

[52] Ofir Press, Noah Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2021.

[53] Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Lightning attention-2: A free lunch for handling unlimited sequence lengths in large language models. *arXiv preprint arXiv:2401.04658*, 2024.

[54] Zhen Qin, Yiran Zhong, and Hui Deng. Exploring transformer extrapolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18897–18905, 2024.

[55] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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

[57] Anian Ruoss, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Róbert Csordás, Mehdi Bennani, Shane Legg, and Joel Veness. Randomized positional encodings boost length generalization of transformers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1889–1903, 2023.

[58] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, 2018.

[59] Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Victoria Lin, Noah A Smith, Luke Zettlemoyer, Scott Yih, and Mike Lewis. In-context pretraining: Language modeling beyond document boundaries. *International Conference on Learning Representations*, 2023.

[60] Konrad Staniszewski, Szymon Tworkowski, Sebastian Jaszczur, Henryk Michalewski, Łukasz Kuciński, and Piotr Miłoś. Structured packing in LLM training improves long context utilization. *arXiv preprint arXiv:2312.17296*, 2023.

[61] Jianlin Su, Murtadha Ahmed, Luo Ao, Mingren Zhu, Yunfeng Liu, et al. Naive bayes-based context extension for large language models. *arXiv preprint arXiv:2403.17552*, 2024.

[62] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[63] Jiankai Sun, Chuanyang Zheng, Enze Xie, Zhengying Liu, Ruihang Chu, Jianing Qiu, Jiaqi Xu, Mingyu Ding, Hongyang Li, Mengzhe Geng, et al. A survey of reasoning with foundation models. *arXiv preprint arXiv:2312.11562*, 2023.

[64] Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. A length-extrapolatable transformer. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14590–14604, July 2023.

[65] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.

[66] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR, 2020.

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

[68] Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36, 2024.

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[70] Benyou Wang, Lifeng Shang, Christina Lioma, Xin Jiang, Hao Yang, Qun Liu, and Jakob Grue Simonsen. On position embeddings in BERT. In *International Conference on Learning Representations*, 2020.

[71] Jie Wang, Tao Ji, Yuanbin Wu, Hang Yan, Tao Gui, Qi Zhang, Xuanjing Huang, and Xiaoling Wang. Length generalization of causal transformers without position encoding. *arXiv preprint arXiv:2404.12224*, 2024.

[72] Suyuchen Wang, Ivan Kobyzev, Peng Lu, Mehdi Rezagholizadeh, and Bang Liu. Resonance RoPE: Improving context length generalization of large language models. *arXiv preprint arXiv:2403.00071*, 2024.

[73] Y Wang, D Ma, and D Cai. With greater text comes greater necessity: Inference-time training helps long text generation. *arXiv preprint arXiv:2401.11504*, 2024.

[74] BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

[75] Wenhao Wu, Yizhong Wang, Yao Fu, Xiang Yue, Dawei Zhu, and Sujian Li. Long context alignment with short instructions and synthesized positions. *arXiv preprint arXiv:2405.03939*, 2024.

[76] Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, Song Han, and Maosong Sun. InfLLM: Unveiling the intrinsic capacity of LLMs for understanding extremely long sequences with training-free memory. *arXiv preprint arXiv:2402.04617*, 2024.

[77] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

[78] Jing Xiong, Zixuan Li, Chuanyang Zheng, Zhijiang Guo, Yichun Yin, Enze Xie, Zhicheng Yang, Qingxing Cao, Haiming Wang, Xiongwei Han, et al. Dq-lore: Dual queries with low rank approximation re-ranking for in-context learning. *arXiv preprint arXiv:2310.02954*, 2023.

[79] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

[80] Kai Yang, Jan Ackermann, Zhenyu He, Guhao Feng, Bohang Zhang, Yunzhen Feng, Qiwei Ye, Di He, and Liwei Wang. Do efficient transformers really save computation? *International Conference on Machine Learning*, 2024.

[81] Howard Yen, Tianyu Gao, and Danqi Chen. Long-context language modeling with parallel context encoding. *arXiv preprint arXiv:2402.16617*, 2024.

[82] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[83] Zhenyu Zhang, Runjin Chen, Shiwei Liu, Zhewei Yao, Olatunji Ruwase, Beidi Chen, Xiaoxia Wu, and Zhangyang Wang. Found in the middle: How language models use long contexts better via plug-and-play positional encoding. *arXiv preprint arXiv:2403.04797*, 2024.

[84] Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*, 2023.

[85] Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua M. Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. In *International Conference on Learning Representations*, 2024.

[86] Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. *arXiv preprint arXiv:2402.09371*, 2024.

[87] Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. PoSE: Efficient context window extension of llms via positional skip-wise training. In *International Conference on Learning Representations*, 2023.

[88] Shiyi Zhu, Jing Ye, Wei Jiang, Qi Zhang, Yifan Wu, and Jianguo Li. CoCA: Fusing position embedding with collinear constrained attention for fine-tuning free context window extending. *arXiv e-prints*, pages arXiv–2309, 2023.

A   Broader impacts

Positive societal impacts.   We propose a method for length extrapolation, which will be helpful for transformer-based models to process long context.

Negative societal impacts.   This method may be abused for other potential long-context applications.

B   Model Configuration

All experiments are conducted on 8 x A800 GPUs. The 125M model configuration is the following.

Table 3: **Model Configurations.**

	125M	**350M**
Training sequence length	512	512
Batch size	32×8	32×8
Numer of iterations	50k	50k
Dropout prob.	0.0	0.0
Attention dropout prob.	0.0	0.0
Attention head	12	16
Feature dimension	768	1024
Layer number	12	24
Optimizer	Adam	Adam
Optimizer parameter betas	[0.9, 0.95]	[0.9, 0.95]
Learning rate	$6e-4$	$3e-4$
Precision	float16	float16

C   Appendix: The Effect of the Hidden Dimension $D_{\mathbf{DAPE}}$
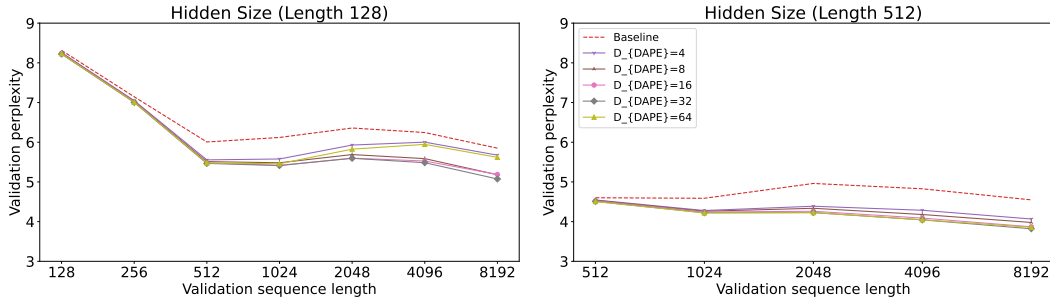
Figure 6: **The effect of $D_{\mathbf{DAPE}}$: the performance with training lengths 128 and 512 on the Arxiv dataset. The experiments are conducted with Alibi and DAPE-Alibi.**

D   Experiments on Chomsky Hierarchy Evaluation Benchmark (CHE)

Following the framework established by [21, 57], we conduct evaluations of our DAPE on a suite of tasks derived from the domain of formal language recognition. These tasks include modular arithmetic, reversing and duplicating strings, binary operations and bucket sort. Based on the Chomsky hierarchy [16], these tasks are categorized into distinct classes: Regular (R), Context-Free, Context-Sensitive (CS), and Recursively Enumerable. Each class aligns with specific computational models: Regular tasks are solvable using Finite-State Automata (FSA); Deterministic Context-Free tasks can be addressed by an FSA equipped with a deterministic stack; and Context-Sensitive tasks require an FSA complemented by access to a bounded tape.

Table 4: The example of different tasks. ††† denotes permutation-invariant tasks, which are expected to be solved without positional information. More explanation of tasks can be found in [21].

Level	Task	Example Input	Example Output
Regular	EVEN PAIRS	$aabba$	True
	MODULAR ARITHMETIC (SIMPLE)	$1 + 2 - 4$	4
	PARITY CHECK†††	$aaabba$	True
	CYCLE NAVIGATION†††	011210	2
DCF	STACK MANIPULATION	$abbaa$ POP PUSH a POP	$abba$
	REVERSE STRING	$aabba$	$abbaa$
	MODULAR ARITHMETIC	$-(1 - 2) \cdot (4 - 3 \cdot (-2))$	0
	SOLVE EQUATION	$-(x - 2) \cdot (4 - 3 \cdot (-2))$	1
CS	DUPLICATE STRING	$abaab$	$abaababaab$
	MISSING DUPLICATE	10011021	0
	ODDS FIRST	$aaabaa$	$aaaaba$
	BINARY ADDITION	$10010 + 101$	10111
	COMPUTE SQRT	100010	110
	BUCKET SORT†††	421302214	011222344

Problem Setting   Building upon the framework proposed by Ruoss et al. (2023) [57], we utilize the encoder-only configuration of the original sequence-to-sequence Transformer model, as delineated by Vaswani et al. (2017) [69]. In scenarios that necessitate a multi-token output sequence y, such as the task of string duplication, we extend the input sequence by appending $|y|$ placeholder tokens. Subsequently, we compute the entire Transformer output from this augmented sequence without resorting to autoregressive sampling techniques. Training is conducted on sequences whose lengths are uniformly distributed, sampled from $U(1, N)$, with N set to 40. Evaluation is performed on sequences that vary in length from $N + 1$ to M, where M equals 500. The implemented architecture comprises 5 layers, 8 attention heads, and a feature dimension of 256. The dimension D_{DAPE} is established at 64, while the maximum randomized position L is set to 2048.

E   The Error Bar and Significance Value

Table 5: **The perplexity performances on the Arxiv dataset when the training length is 512 and running with three random seeds.**

Method		512	1024	2048	4096	8192
RoPE	mean	4.5755	45.1974	134.1615	222.3333	265.4545
	std	0.0216	12.2241	21.8522	22.5955	20.3862
T5's bias	mean	4.5547	4.3730	12.1017	163.9289	595.0829
	std	0.0204	0.0747	6.5327	136.8294	306.3671
Alibi	mean	4.6146	4.5475	4.8693	5.0278	4.7679
	std	0.0278	0.0990	0.1246	0.1845	0.2196
DAPE-Alibi	mean	4.5262	4.1921	4.1068	4.1315	4.0013
	std	0.0270	0.0653	0.0885	0.0617	0.2708
Kerple	mean	4.5817	4.3504	5.4438	8.7511	13.3524
	std	0.0252	0.0526	0.3187	1.1066	2.4109
DAPE-Kerple	mean	4.5123	4.1716	4.0505	4.0033	3.8642
	std	0.0251	0.0637	0.0973	0.0333	0.2342
FIRE	mean	4.5741	4.6953	30.1164	165.5394	308.6173
	std	0.0227	0.1125	4.4317	41.2065	78.4652
DAPE-FIRE	mean	4.4879	4.1990	4.2826	4.7983	6.1623
	std	0.0206	0.0619	0.0709	0.1701	0.9226

According to Table 5, the performances of different methods are evaluated based on perplexity across various validation lengths ranging from 512 to 8192. The results indicate that DAPE-Alibi consistently outperforms the Alibi method, DAPE-Kerple surpasses Kerple, and DAPE-FIRE shows better performance than FIRE, all with p-values less than 0.05, suggesting significant improvements.

DAPE-Alibi demonstrates lower perplexity values compared to Alibi at all lengths, indicating more effective learning and generalization. Similarly, DAPE-Kerple shows significant improvements over Kerple, especially at larger lengths, where the gap in performance widens. The DAPE-FIRE method also shows notable enhancements over the FIRE method, particularly at higher lengths where standard FIRE struggles with increased perplexity.

Moreover, DAPE-Alibi, DAPE-Kerple, and DAPE-FIRE not only perform better than their respective original methods but also show superiority over RoPE and T5's bias across all validation lengths. This suggests that the modifications implemented in DAPE versions provide a more robust and generalized model, capable of maintaining lower perplexity and thus better performance on the Arxiv dataset.

In conclusion, the statistical analysis confirms that the proposed DAPE variations offer significant improvements in perplexity performance, thereby validating their effectiveness in comparison to their original counterparts and other baseline methods.

F   Data-Adaptive Related Position Encoding Performance Comparison

Table 6: The performance comparison between data-related position encoding, with dataset Books3 and training length 128.

Method	128	256	512	1024	2048	4096	8192
Transformer-XL	31.57	28.49	26.07	26.98	27.90	32.76	41.12
CoPE	31.61	28.41	25.79	27.96	33.80	54.08	90.66
DAPE-Kerple	31.49	28.27	24.93	24.31	23.34	24.38	25.01

According to the experiment, the transformer-xl achieves good performance on length extrapolation. Therefore, this also suggests that the position encoding should interact with attention/query/key to further improve the performance.

G Ablation Study on Bias Matrix

Table 7: Books Dataset Results: Train Length 512

	512	1024	2048	4096	8192
$QK^T + B$ (baseline)	19.68	19.06	20.44	28.34	39.31
$QK^T + B + f(B)$	19.64	18.83	18.49	20.62	23.49
$QK^T + B + f(QK^T, B)$	19.22	18.22	17.15	17.63	17.88

We further conduct an ablation study on the f, proving that f help enhance the bias matrix. DAPE improves the bias matrix for A_{final}, while the A_{final} is used to calculate $A_{final}K$. For the unseen position, the B partially could handle it (FIRE changes the problem to interpolation), but is not accurate enough so that DAPE helps enhance the bias matrix B via attention score. The experiment suggests two points: 1) The DAPE $f(QK^T, B)$ is better than naive $f(B)$, suggesting that the context-adaptive is important. 2) The $QK^T + B + f(B)$ is better than $QK^T + B$, suggesting that benefiting from improving the expressiveness of the bias matrix.

H The result on 2.7B and 6.7B

Table 8: Model Size and Method Comparison with Training Length 512

Model Size	Method	512	1024	2048	4096
2.7B	RoPE	21.01	25.00	48.13	160.59
	RPE	21.10	21.88	23.59	33.23
	Alibi	21.23	22.17	22.91	23.22
	Kerple	21.14	22.08	23.38	27.21
	DAPE-Kerple	20.52	21.01	20.23	19.57

Table 9: Model Size and Method Comparison with Training Length 512

Model Size	Method	512	1024	2048
6.7B	RoPE	20.86	22.27	28.01
	RPE	20.79	21.60	22.32
	Alibi	20.79	21.63	22.45
	Kerple	20.71	21.57	22.07
	DAPE-Kerple	20.09	20.54	19.83

According to the result, we can find that the proposed DAPE still works well, whatever the model size is 2.7B or 6.7B. With 2.7B model size, RoPE achieves 21.01 on evaluation length 512 and 160.50 on evaluation 4096, while our DAPE-Kerple achieves 20.52 and 19.57 respectively. Also, DAPE-Kerple achieves the best performance, whatever the model size is 2.7B or 6.7B from evaluation length 512 to 8192. This suggests that our proposed DAPE has great scalability.

I DAPE Visualization

The model is trained with DAPE-Kerple on length 512.
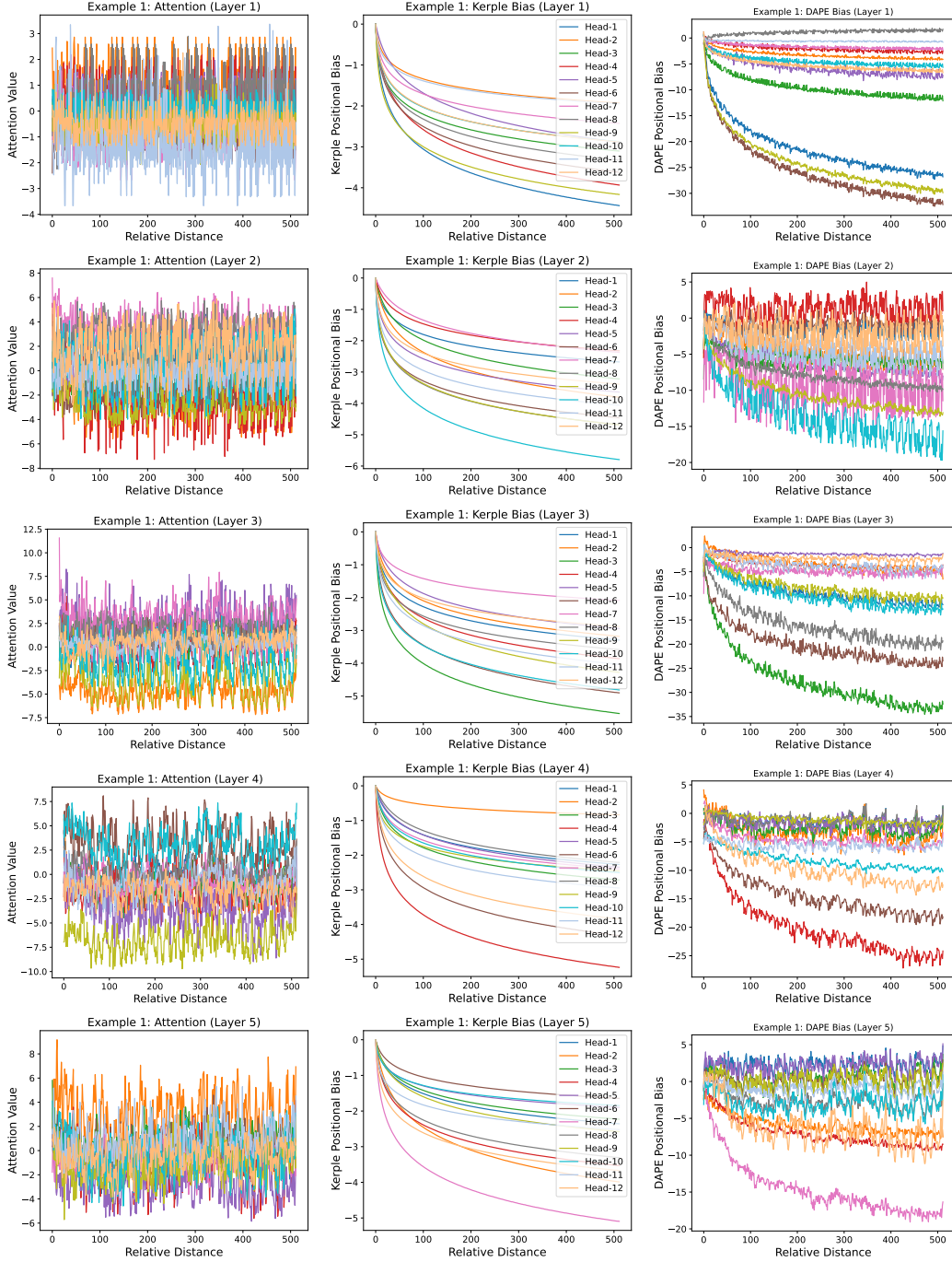
I.1 Visualization on length 512
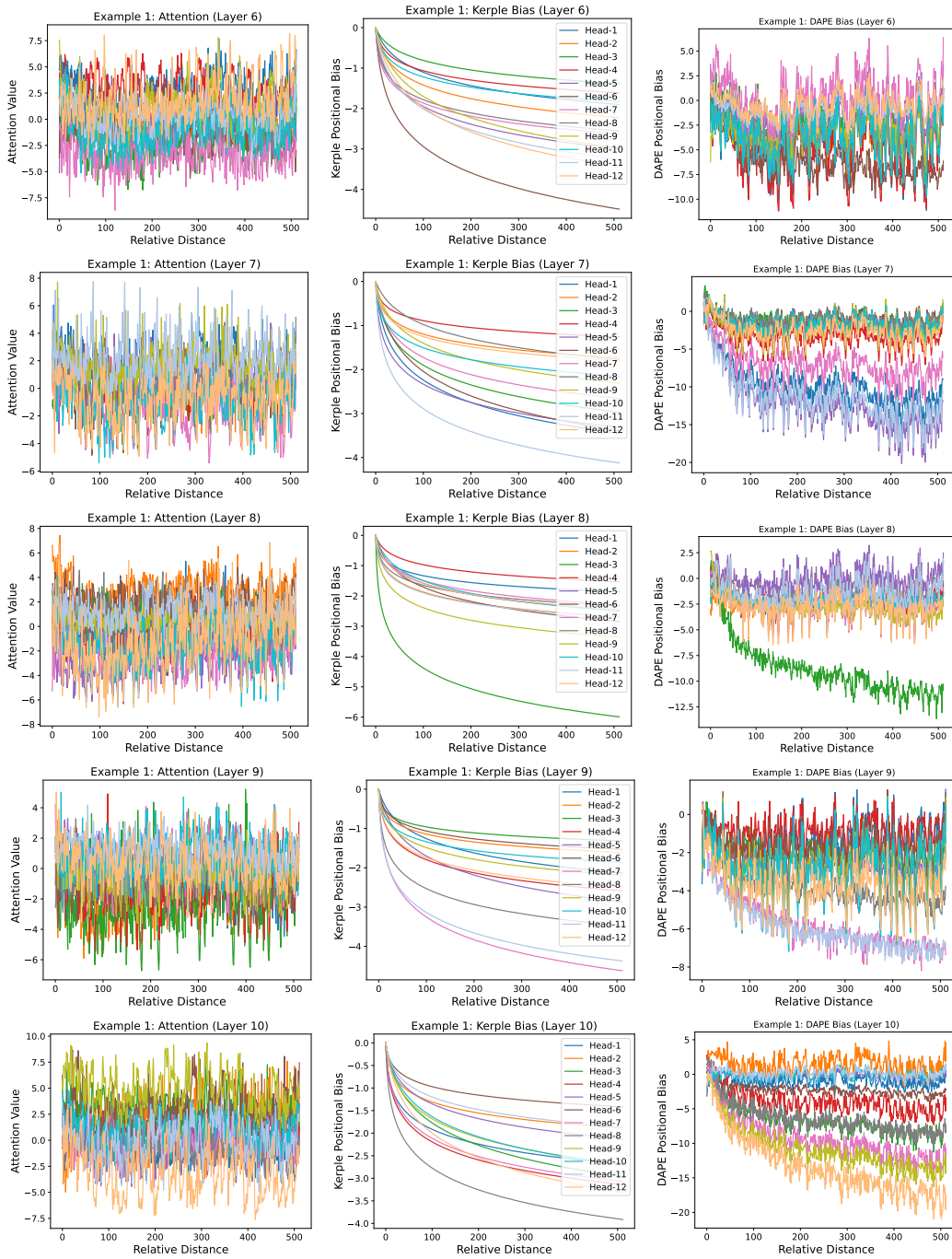

Figure 7: **Evaluation Length 512 Example 1: Part 1**

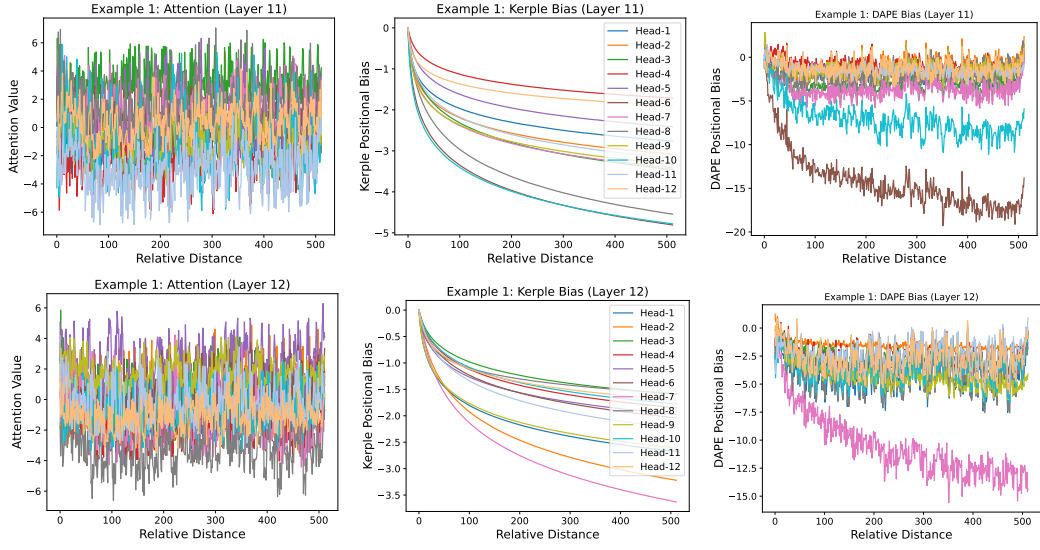Figure 8: **Evaluation Length 512 Example 1: Part 2**
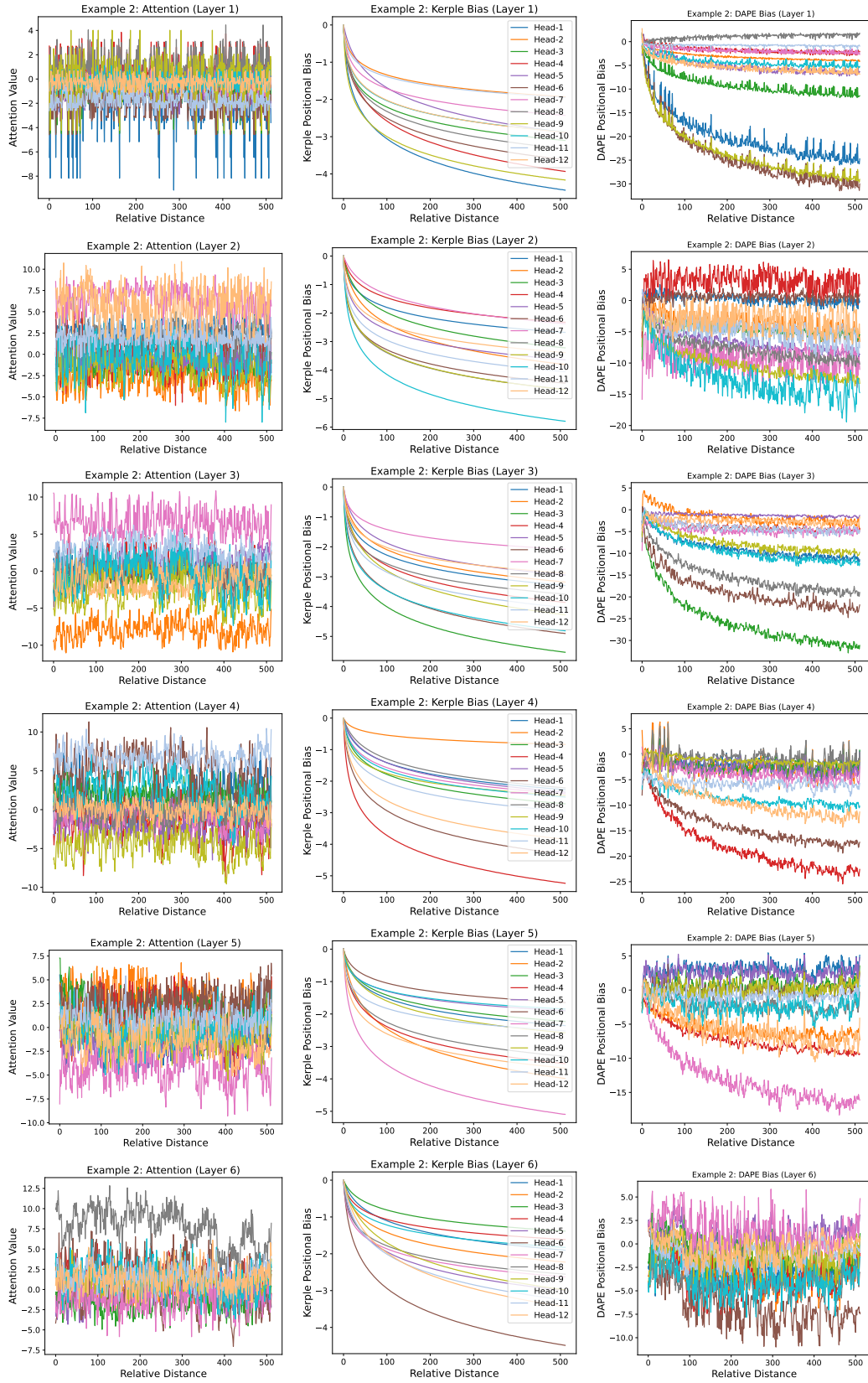
Figure 9: **Evaluation Length 512 Example 1: Part 3**
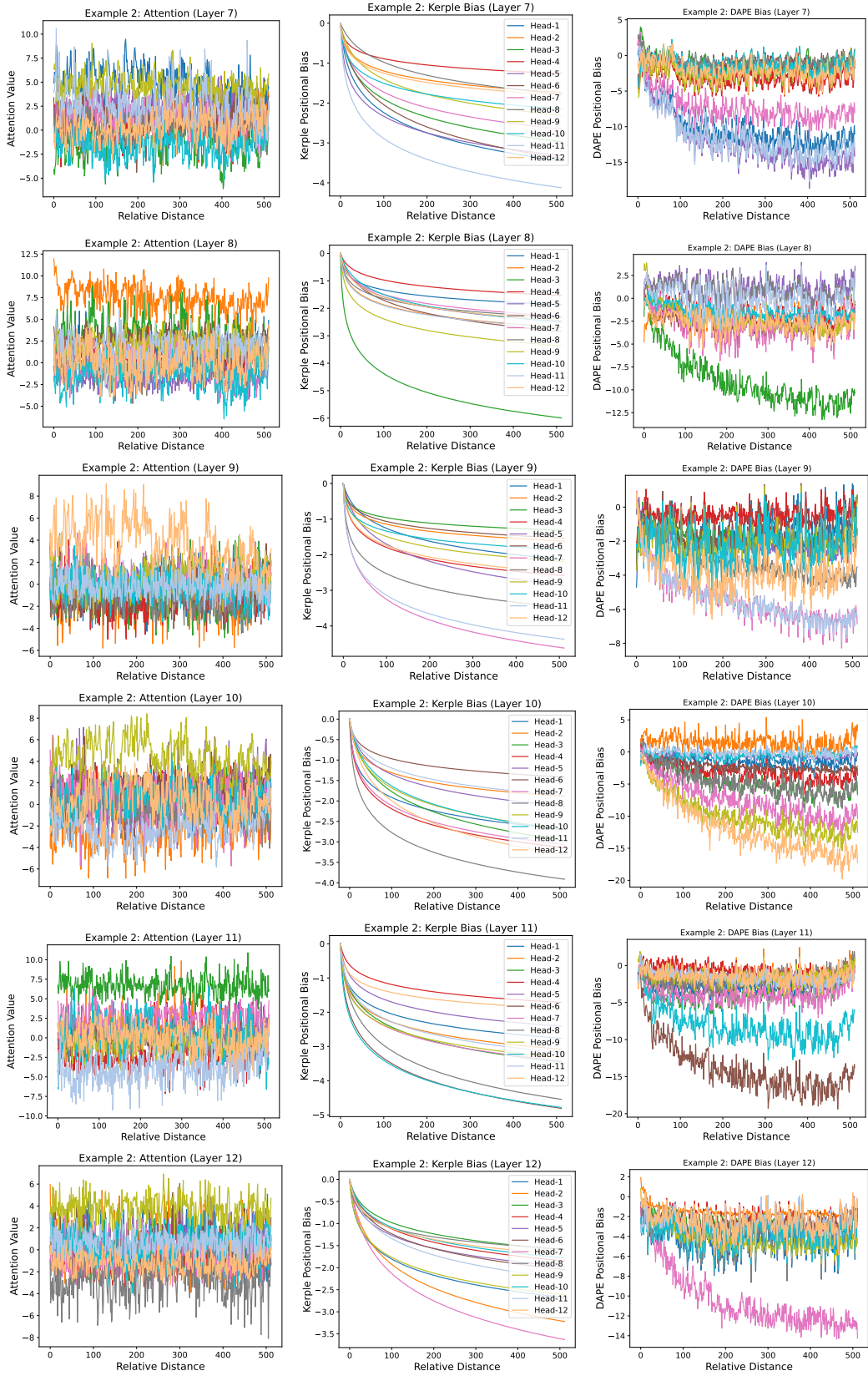
Figure 10: **Evaluation Length 512 Example 2: Part 1**

Figure 11: **Evaluation Length 512 Example 2: Part 2**
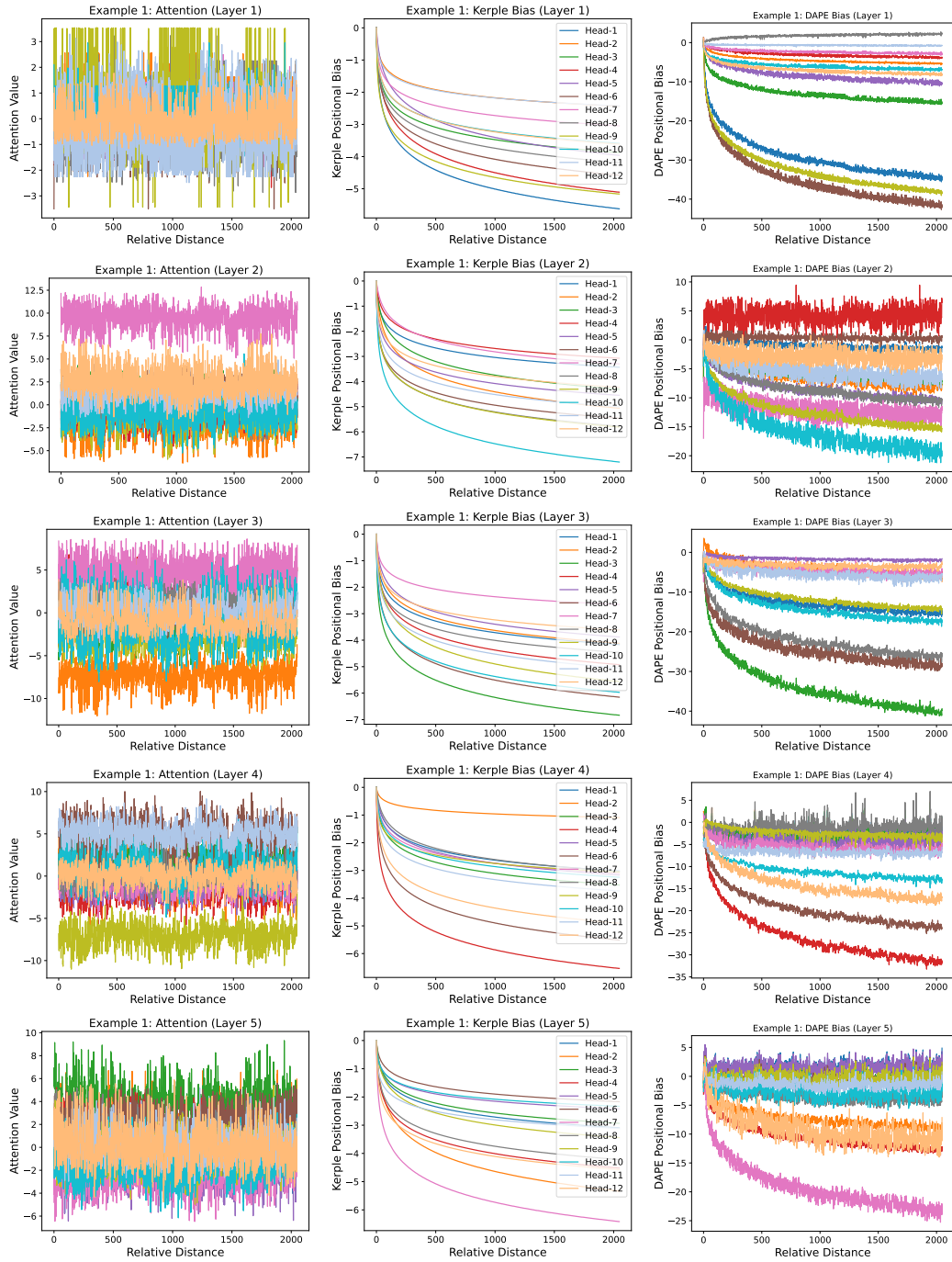
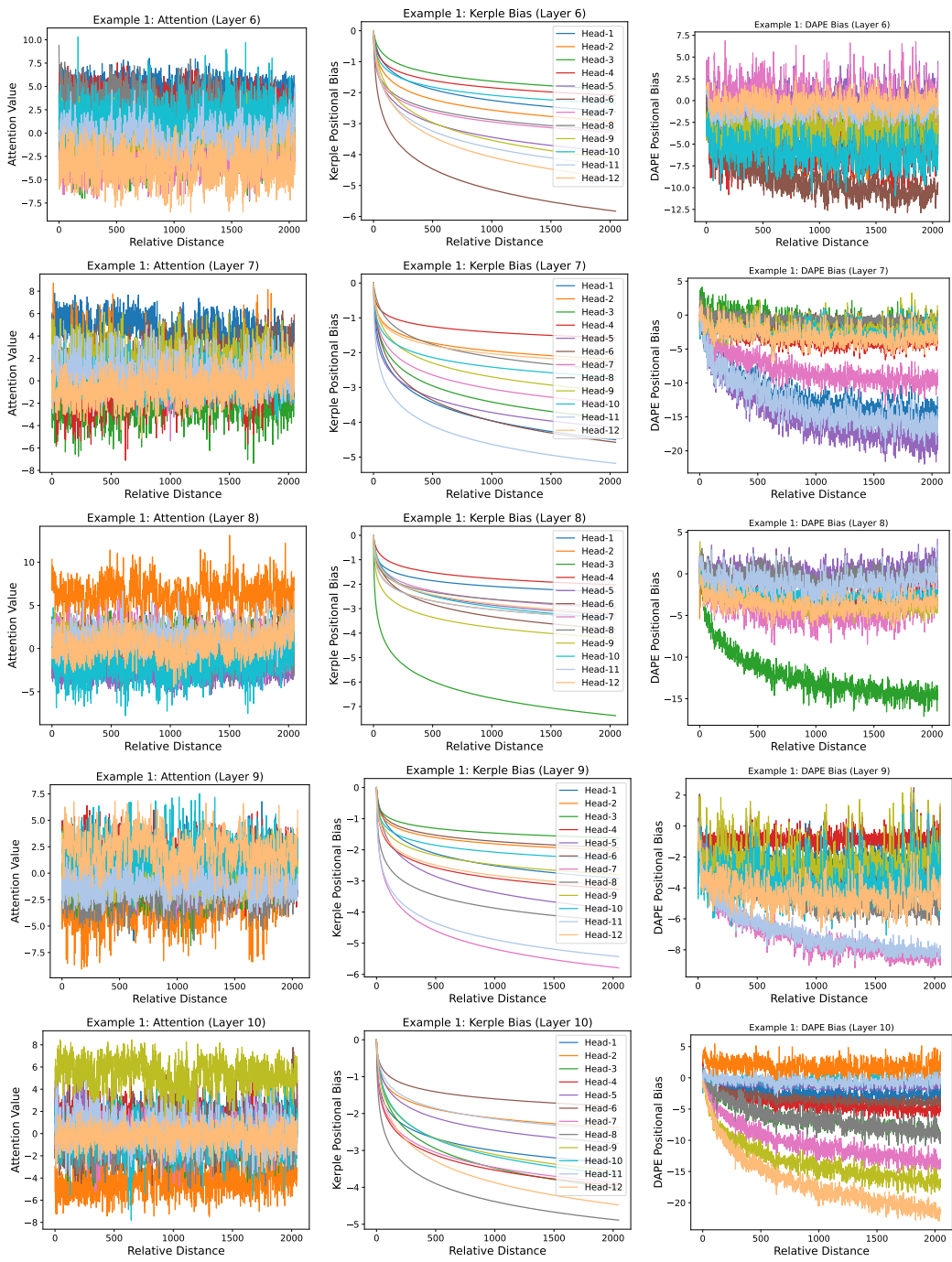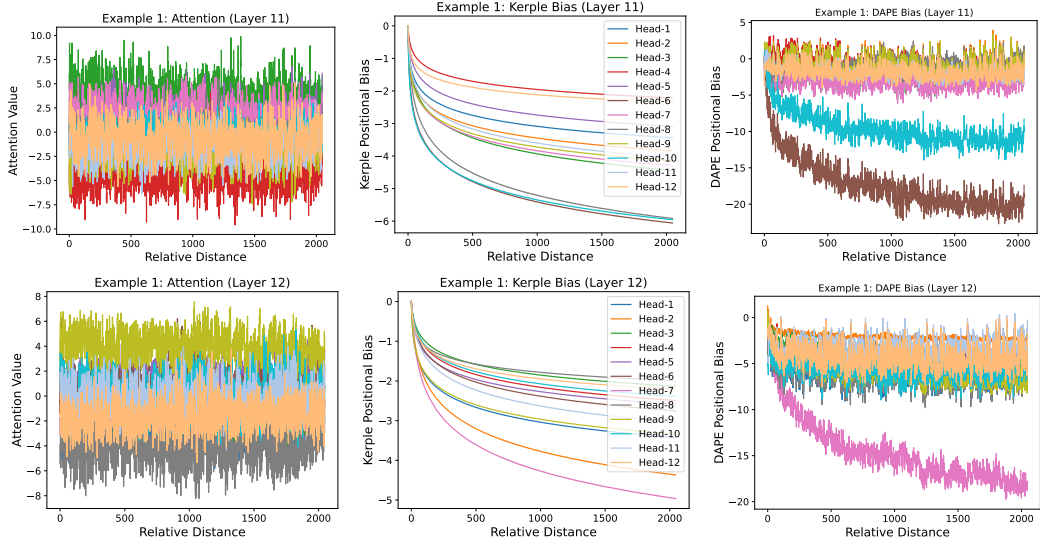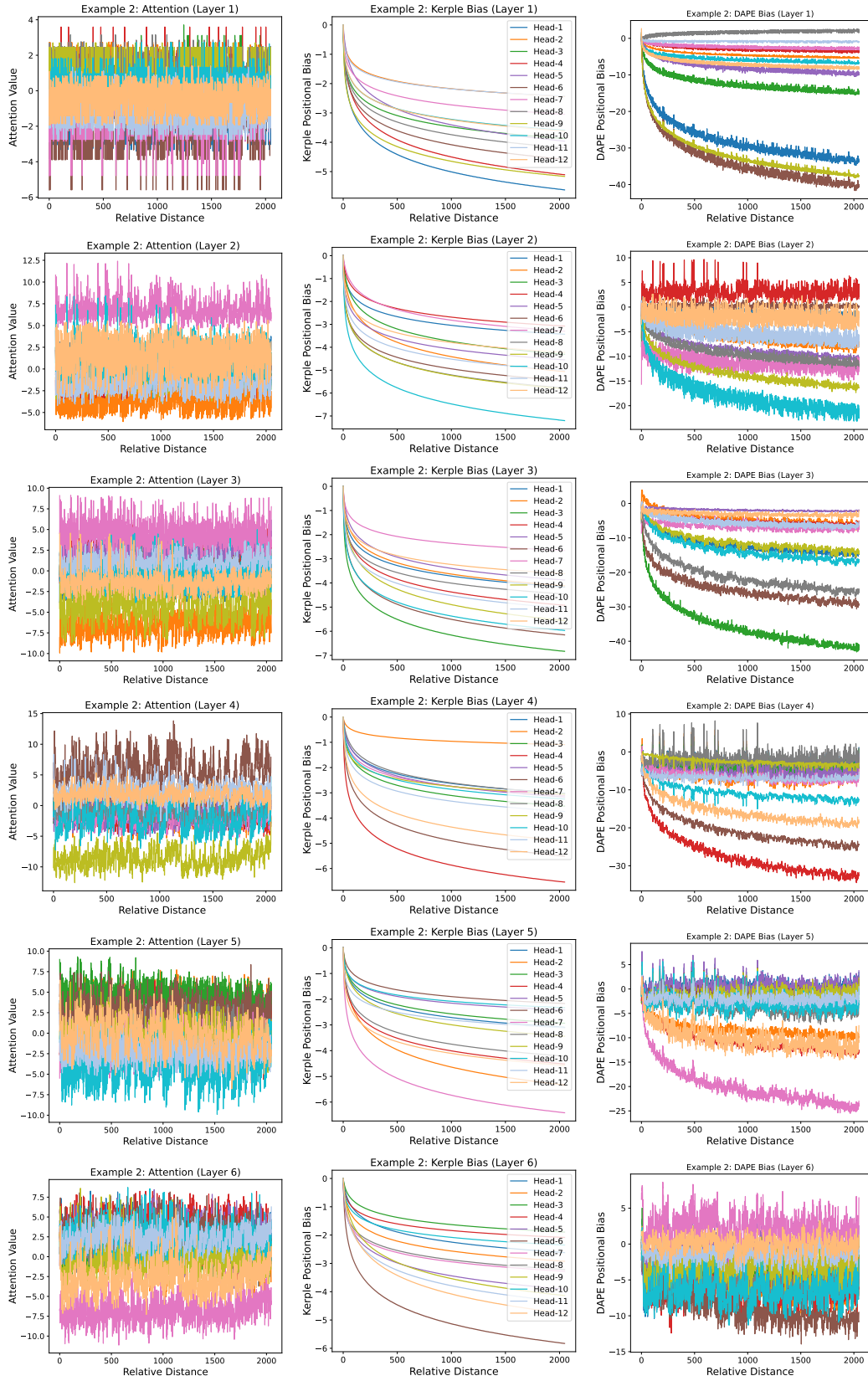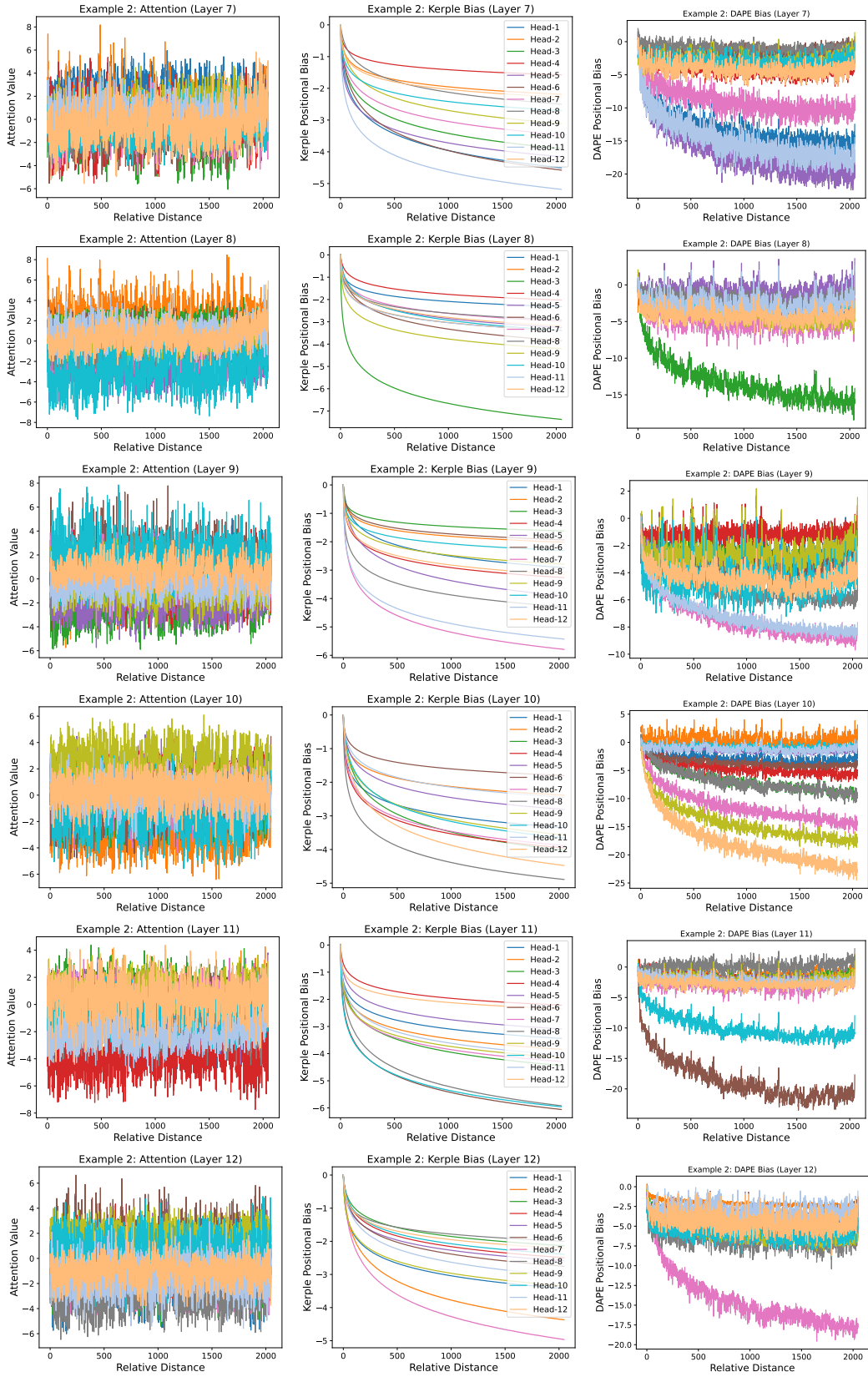## I.2 Visualization on length 2048

Figure 12: **Evaluation Length 2048 Example 1: Part 1**

Figure 13: **Evaluation Length 2048 Example 1: Part 2**

Figure 14: **Evaluation Length 2048 Example 1: Part 3**

Figure 15: **Evaluation Length 2048 Example 2: Part 1**

Figure 16: **Evaluation Length 2048 Example 2: Part 2**
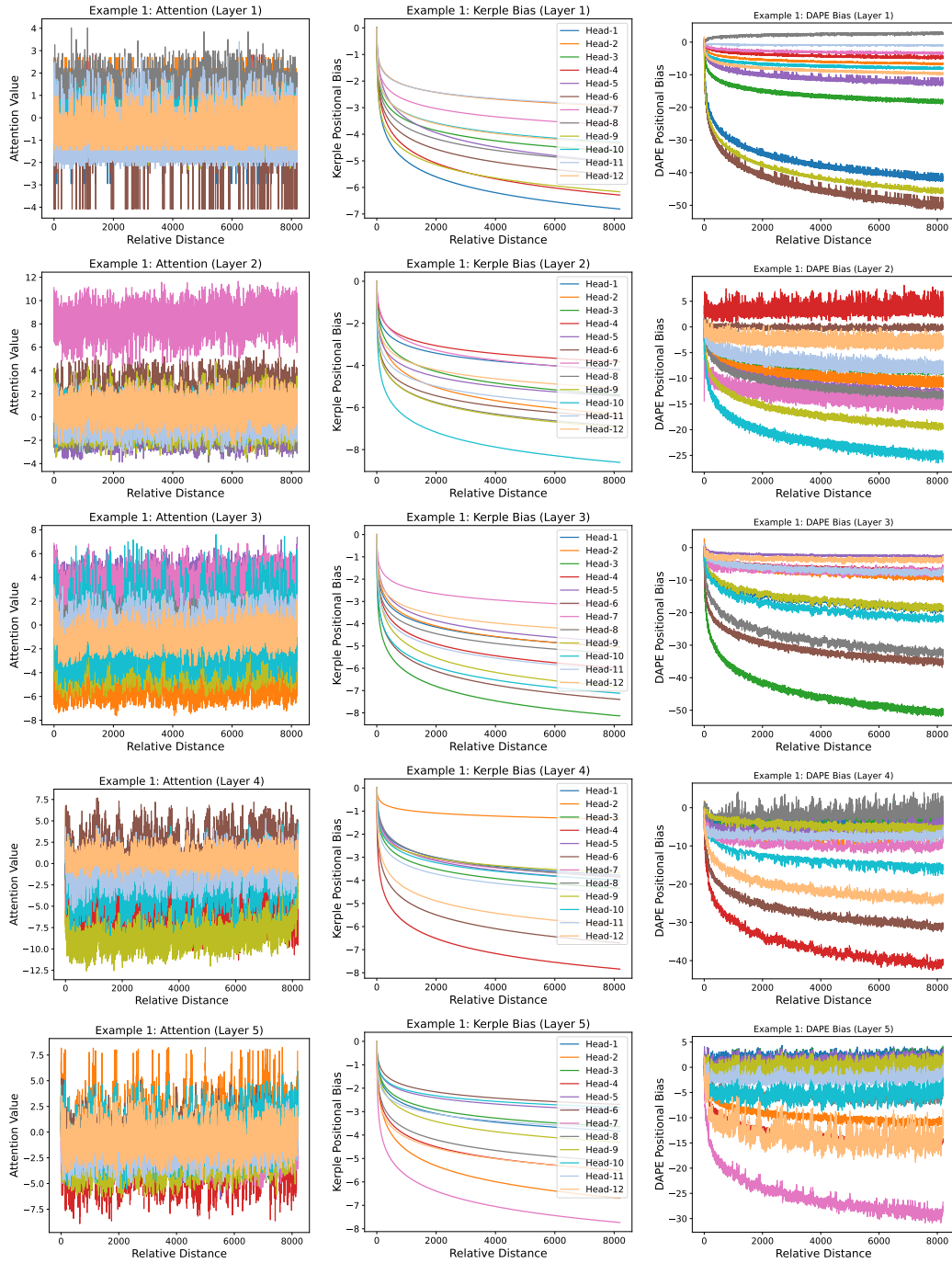
I.3 Visualization on length 8192

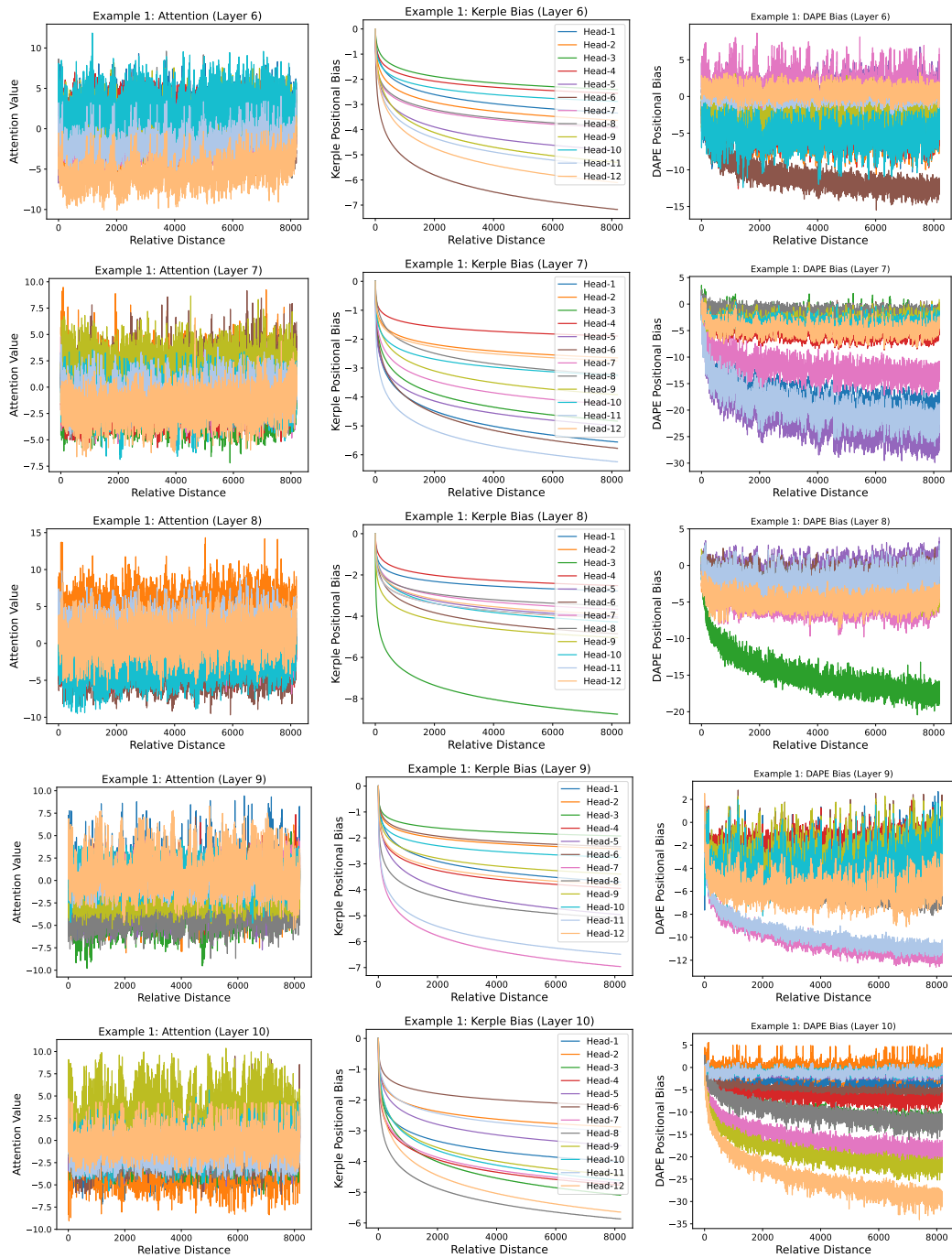Figure 17: **Evaluation Length 8192 Example 1: Part 1**

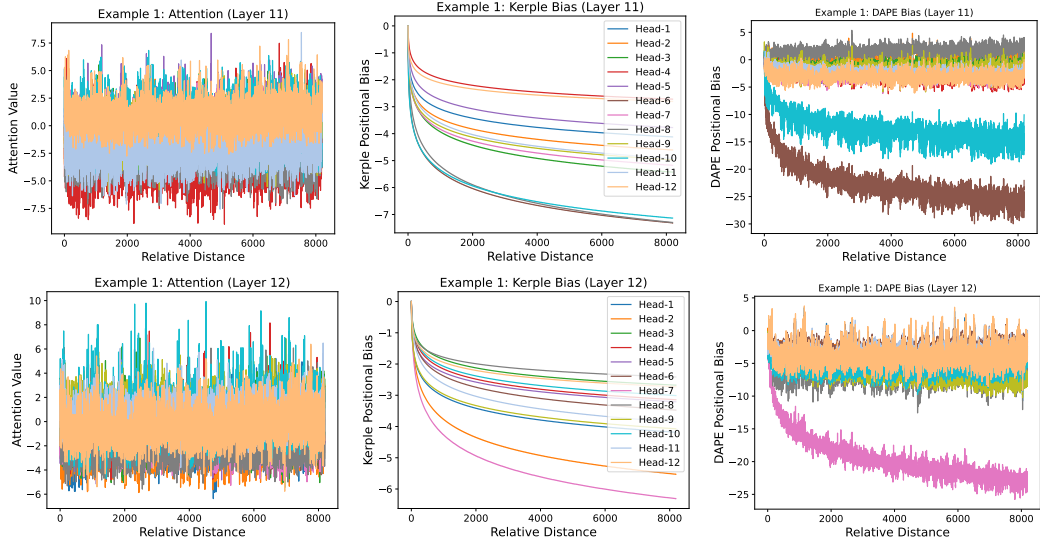Figure 18: **Evaluation Length 8192 Example 1: Part 2**

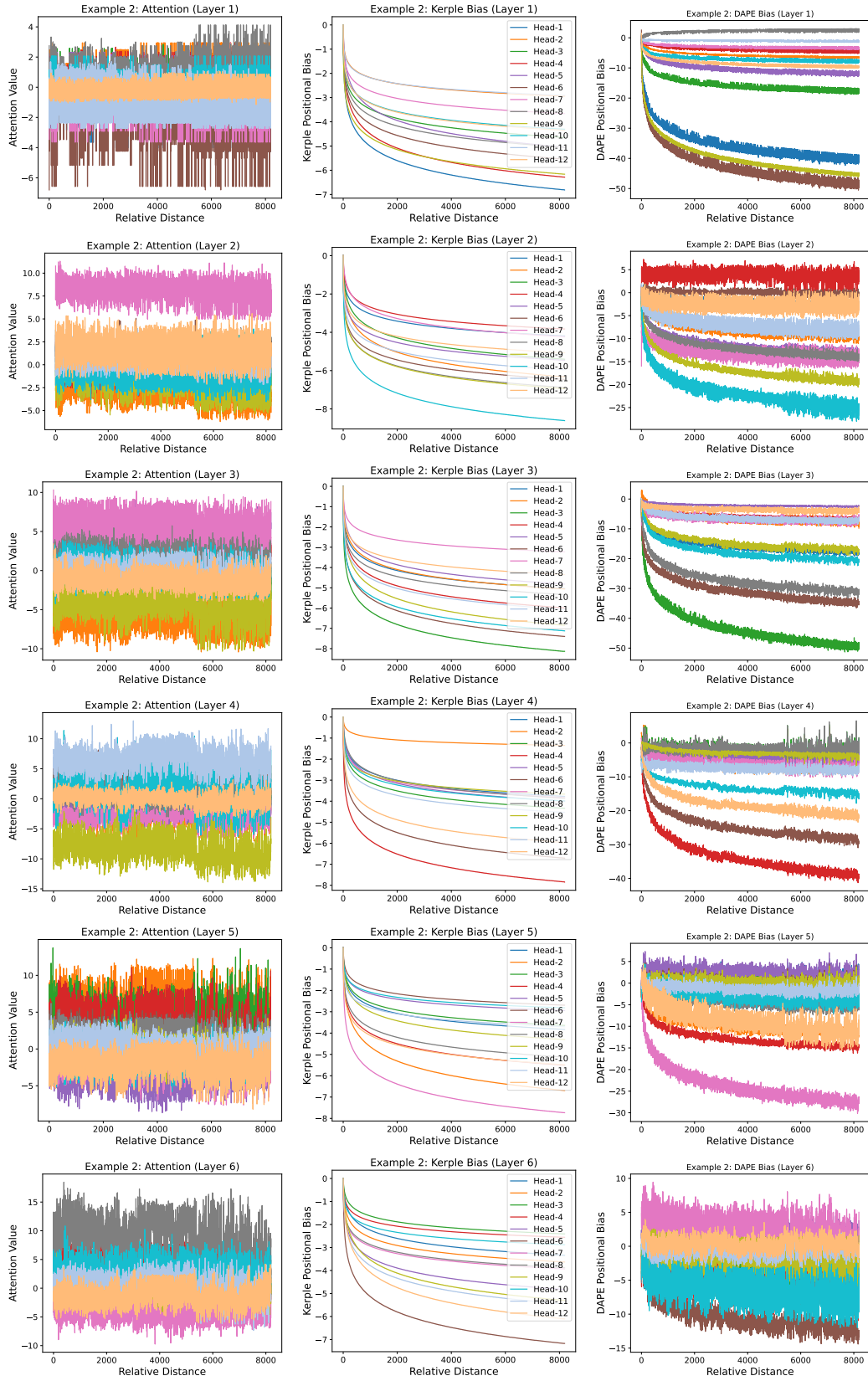Figure 19: **Evaluation Length 8192 Example 1: Part 3**

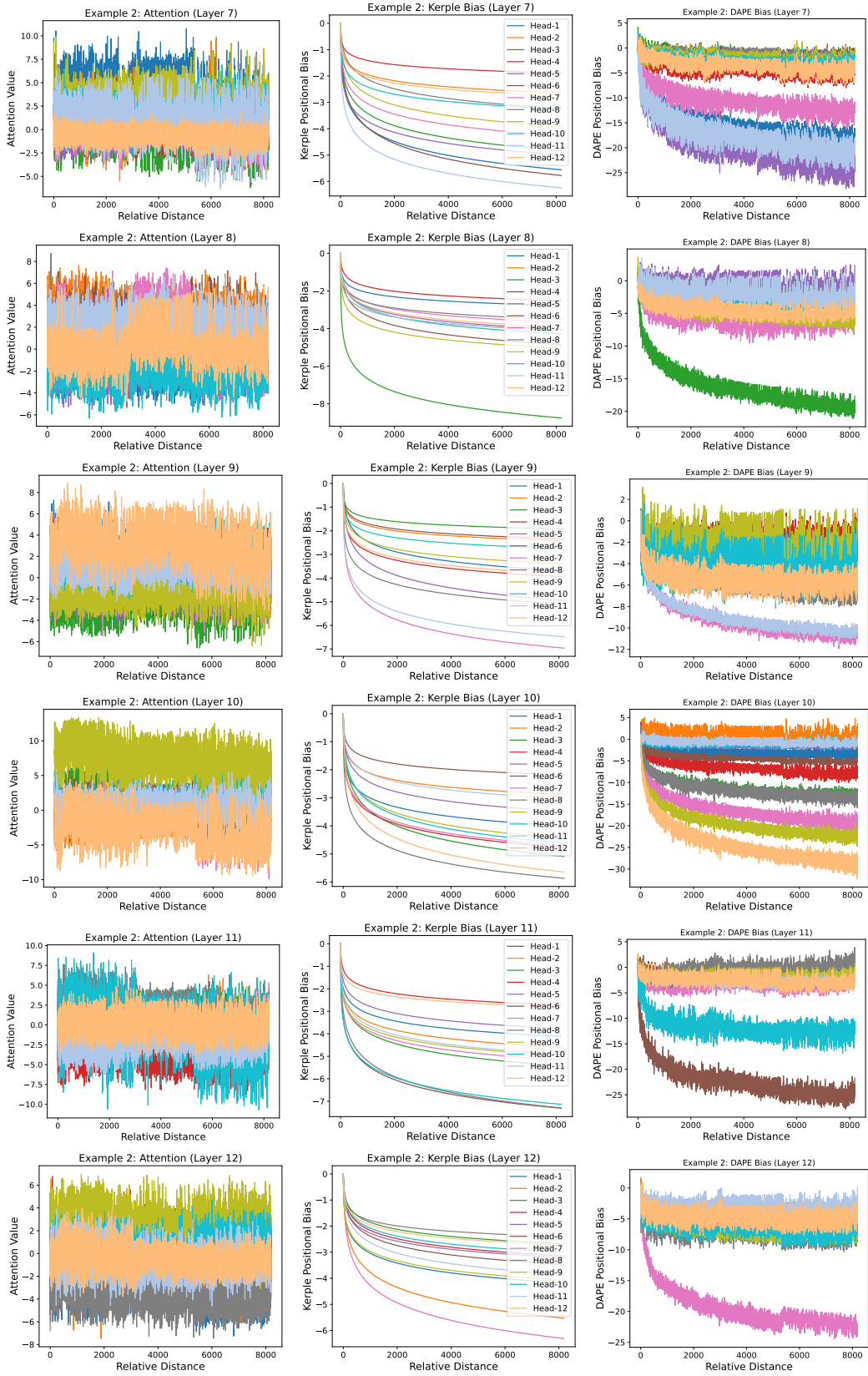Figure 20: **Evaluation Length 8192 Example 2: Part 1**

Figure 21: **Evaluation Length 8192 Example 2: Part 2**

J Implementation

In this section, we present the implementation of the proposed DAPE module in `PyTorch` [49].

```python
import torch
import torch.nn as nn

class DAPE(nn.Module):
  def __init__(self, head_number=12, mlp_width=32):
    """
    DAPE attention bias module.

    Args:
      num_heads: number of attention heads.
      mlp_width: Width of MLP.
    """
    super(DAPE, self).__init__()


    self.mlp = nn.Sequential(
      nn.Linear(2*head_number, mlp_width),
      nn.LeaklyReLU(),
      nn.Linear(mlp_width, num_heads)
    )

  def forward(self, attention: torch.Tensor, bias: torch.Tensor):
    """
    Args:
      attention: input sequence, which is q^T * k,
          shape [bsz, num_heads, seq_len, seq_len]
      bias: bias matrix, which can be generated by Alibi, Kerple
      FIRE or other additive position encodings
          shape [1,num_heads, seq_len, seq_len]

    Returns:
      attention with DAPE,
      shape [bsz, num_heads, seq_len, seq_len]
    """
    bias_tile = repeat(bias, '1 h T T -> b h T T', b=attention.shape
        [0])

    # Concatenate attention and bias
    attention_bias_concat = torch.cat((attention, bias_tile), dim=1)

    # Rearrange the dimensions for MLP processing
    attention_bias_concat = rearrange(attention_bias_concat, 'b h T
        T -> b T T h')

    # Apply the MLP
    attention_bias_concat = self.mlp(attention_bias_concat)

    # Rearrange back to original dimensions
    attention_bias_concat = rearrange(attention_bias_concat, 'b T T
        h -> b h T T')

    return attention + bias + attention_bias_concat
```

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 have stated the claim in Abstract and Introduction part (Section 1). We also conduct extensive experiments in Experiment part (Section 4)

   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: The we discuss the limitation in the Time Cost part (Section 4.5) of Experiment.

   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: We have prove the expressiveness of our method in the Method part.

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: Yes, we have proved the implementation in Method (Section 3) part and model configuration in Appendix Model Configuration.

   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: We have provided the pytorch implementation code in Appendix Section J.

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: We have provided the experiment setting in the Experiment (Section 4) part.

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: [Yes]

Justification: In the paper, we report the mean perplexity values and standard variance Appendix Section E. Additionally, we also report that the proposed method shows statistically significant improvements over other static PE methods (according to the p-value less than 0.05).

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: We have provided the compute resources in the Time Cost part of Experiment (Section 4) and Appendix B.

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: We have followed the guidelines.

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: [Yes]

Justification: We have discussed it in the Broad Impact section, which is Appendix Section A.

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: The paper poses no such risks.

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: Yes, we have cited the related works.

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 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: The paper does not release 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: The 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: The 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.
- 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.