EulerFormer: Sequential User Behavior Modeling with Complex Vector Attention

Anonymous Author(s)

ABSTRACT

To capture user preference, transformer models have been widely applied to model sequential user behavior data. The core of transformer architecture lies in the self-attention mechanism, which computes the pairwise attention scores in a sequence. Due to the permutation-equivariant nature, positional encoding is used to enhance the attention between token representations. In this setting, the pairwise attention scores can be derived by both *semantic difference* and *positional difference*. However, prior studies often model the two kinds of difference measurements in different ways, which potentially limits the expressive capacity of sequence modeling.

To address this issue, this paper proposes a novel transformer variant with *complex vector attention*, named **EulerFormer**, which provides a unified theoretical framework to formulate both *semantic difference* and *positional difference*. The EulerFormer involves two key technical improvements. First, it employs a new transformation function for efficiently transforming the sequence tokens into polar-form complex vectors using Euler's formula, enabling the unified modeling of both semantic and positional information in a complex rotation form. Secondly, it develops a differential rotation mechanism, where the semantic rotation angles can be controlled by an adaptation function, enabling the adaptive integration of the semantic and positional information according to the semantic contexts. Furthermore, a phase contrastive learning task is proposed to improve the anisotropy of contextual representations in EulerFormer. Our theoretical framework possesses a high degree of completeness and generality (*e.g.,* RoPE can be instantiated as a special case). It is more robust to semantic variations and possesses more superior theoretical properties (*e.g.,* long-term decay) in principle. Extensive experiments conducted on four public datasets demonstrate the effectiveness and efficiency of our approach.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Complex Vector Attention, User Behavior Modeling

ACM Reference Format:

Anonymous Author(s). 2024. EulerFormer: Sequential User Behavior Modeling with Complex Vector Attention. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington D.C., USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

SIGIR '24, July 14–18, 2024, Washington D.C., USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

User behavior modeling [33, 34, 37, 41] has been a fundamental task in many online service systems, *e.g.,* Amazon and Netflix. It aims to accurately capture user preference from user's logged interaction data, so that the system can better predict future user behavior and provide high-quality service. For example, the performance of sequential recommendation can be largely improved if the underlying user preference model can be effectively learned [12, 13].

Typically, user's behavior data can be characterized as an interaction sequence of interested items in a chronological order. To model the sequential interaction patterns, the transformer architecture [32] has been widely applied for user behavior modeling [12, 13, 15, 27, 42]. For transformer architecture, the positional encoding module is an essential component, as the permutation-equivariant attention mechanism cannot capture the sequential order of the interacted items by user. Existing user behavior modeling studies [12, 13, 15, 27, 42] often integrate learnable *absolute* positional encoding (*i.e.,* each position is modeled with a unique embedding vector) for complementing the positional semantics of the contextual representations in transformer networks. Though effective to some extent, the learned positional embeddings can only model the observed positions, which cannot generalize to unseen positions in longer sequences. Another issue is that absolute position encoding cannot accurately capture the relative positional relations among items in a sequence, thereby less effective to model sequential dependencies.

Actually, these issues have been noted in the field of natural language processing (NLP), and a number of studies have been devoted to improving positional encoding of transformers, exemplified by rotary position embedding (RoPE) [23, 26, 28]. The basic idea of RoPE is to rotate the vector of each key or query (*e.g.,* $\boldsymbol{q}_m, \boldsymbol{k}_n$) by a specified angle (*e.g.,* α) based on their absolute positions (*e.g.,* m, n). According to the derivation in [26], the core formula of RoPE can be written as "$\boldsymbol{q}_m^\top \boldsymbol{k}_n^* \cdot \exp[i(m-n)\cdot\alpha]$[1]", in which it essentially captures two kinds of difference between query and key, namely *semantic difference* (denoted as "$\Delta \boldsymbol{S}$") that measures the semantic similarity between $\boldsymbol{q}_m$ and $\boldsymbol{k}_n$ in *dot product* and *positional difference* (denoted as "$\Delta \boldsymbol{P}$") that reflects their relative positions (*i.e.,* $m-n$) in *rotation angle*. Despite that the idea is very intuitive, a potential issue of RoPE is that it models the semantic difference and positional difference in a different way, thus likely having limited expressive capacities in user behavior modeling. From a general perspective, though dot product and rotation angle are similar as a distance or similarity measurement, they still possess different mathematical properties, and a straightforward integration of both parts is likely to lead to suboptimal capacity in sequence modeling.

[1] For simplicity, we omit some factors or terms from the original equation in [26].

More specifically, since users freely interact with interested items, the generated interaction sequence can be more complex than formal natural language text, which makes the semantic differences among items highly varied across different layers or sequences. Given the *highly varied* semantic difference $\Delta \boldsymbol{S}$ across different layers or users, RoPE still adopts a *pre-designed* relative positional encoding to capture positional difference $\Delta \boldsymbol{P}$, making it less flexible to adapt to varying contexts, *e.g.,* the two parts may have vastly different magnitude. Furthermore, it can be proven that when semantic difference is relatively large (*e.g.,* $\boldsymbol{q}_m$ and $\boldsymbol{k}_n$ have opposite directions), RoPE would become incapable of effectively modeling the relative positions [26]. Therefore, it is still challenging for existing methods to effectively capture the positional information in complicated interaction scenarios, thus leading to an inaccurate modeling of user preference.

Considering the above issues, our key point is to develop a more effective way to integrate the semantic difference and positional difference, thereby improving the capacity in user behavior modeling. To achieve this, a fundamental challenge is how to model the two kinds of differences in a unified mathematical form, so as to better aligning the two parts. Furthermore, the integration approach should also be adaptive to highly varied contexts, which can effectively adapt to various interaction scenarios.

To this end, in this paper, we develop a novel transformer variant with complex vector attention, named **EulerFormer**. The major contribution of EulerFormer is that it introduces a unified theoretical framework to formulate both semantic difference and positional difference in complex vector space, thus achieving a stronger expressive capacity in sequential behavior modeling. Technically, EulerFormer involves two key improvements. First, both semantic difference and positional difference are modeled via rotation angles in a complex vector space. Specifically, it employs a new transformation function, namely the *Euler transformation*, for efficiently transforming token embeddings into *polar-form* complex vectors using Euler's formula. With this function, the dot-product attention can be cast into a *complex rotation*, enabling the unified modeling of both semantic and positional difference in a rotation form (*i.e.,* "$\exp[i(\Delta \boldsymbol{S} + \Delta \boldsymbol{P})]$"). Secondly, it develops a differential rotation mechanism by introducing an adaptation function for adjusting semantic difference (Adapt$(\cdot)$) under different contexts. It can adaptively integrate the two kinds of difference in a weighted rotation form (*i.e.,* "$\exp[i(\text{Adapt}(\Delta \boldsymbol{S}) + \Delta \boldsymbol{P})]$"). Furthermore, based on the proposed framework, we introduce a phase contrastive task to enhance the anisotropy of contextual representations, which can further improve the effectiveness of sequential modeling.

The main contributions are summarized as follows:

• We develop a capable transformer variant, named EulerFormer, by introducing novel complex vector attention. It formulates the semantic and positional difference in a unified mathematical form, thus achieving a more expressive capacity in user behavior modeling. It can surpass the expressive limits of RoPE, *i.e.,* enhancing the model's robustness to semantic variations and possessing more superior theoretical properties (*e.g.,* long-term decay).

• We propose to enhance the complex vector attention with learnable adaptation function that can adaptively integrate semantic and positional differences under varied contexts. Further, we propose a phase contrastive learning approach for improving the anisotropy of sequence representations in EulerFormer.

• Experimental results on four public datasets demonstrate that EulerFormer can serve as a powerful substitute of transformer backbone in user behavior modeling, which largely improves the performance of several representative recommendation models, such as SASRec and BERT4Rec. Moreover, it consistently outperforms several strong positional encoding methods, showing both effectiveness and efficiency in user behavior modeling.

2 PRELIMINARY

In this section, we briefly introduce the task of sequential recommendation, the transformer-based sequential models, and Euler's formula, which will be used in our approach.

Sequential Recommendation. In this work, we consider the task of sequential recommendation for examining the performance of user behavior models. The setting of sequential recommendation typically involves a set of users (*i.e.,* $\mathcal{U}$) and items (*i.e.,* $\mathcal{V}$), where $u \in \mathcal{U}$ denotes a user and $v \in \mathcal{V}$ denotes an item. In sequential recommendation with implicit feedback, each user u has a chronologically-ordered historical interaction sequence, denoted by $v_{1:N} := \{v_1, v_2, \cdots, v_N\}$, where N is the number of interactions and v_t is the t-th item that the user has interacted with. Formally, given the historical interaction sequence $v_{1:N}$, sequential recommendation aims to predict the next item that the user is most likely to interact with at the $N+1$-th step, *i.e.,* $p(v_{N+1}|v_{1:N})$.

Transformer-based Sequential Models. The core of transformer is the self-attention mechanism. Given the inputs $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, where d is the hidden dimension, it is first mapped into the query, key and value spaces as $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ respectively. As such, the attention scores can be given by $\text{Softmax}(\boldsymbol{Q}\boldsymbol{K}^\top/\sqrt{d})$. Since the attention score is independent of the positional order of inputs, most models use a set of positional embeddings $\boldsymbol{P} = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_N\}$ to model the positions. As such, given the behavior sequence $\{v_1, \cdots, v_N\}$ with its embeddings $\boldsymbol{E} = \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_N\}$, the inputs are obtained by adding the item embedding to its positional embedding, *i.e.,* $\boldsymbol{X} = \boldsymbol{P} + \boldsymbol{E}$. Besides the above absolute encoding, RoPE [26] proposes a *rotary encoding* to model absolute and relative positional information. Formally, given the query $\boldsymbol{q}_m$ and key $\boldsymbol{k}_n$, its core formula for calculating the attention score is written as:

$$\boldsymbol{A} = \text{Re}\Big[\boldsymbol{q}_m^\top \boldsymbol{k}_n^* \exp\Big(i(m-n)\cdot\alpha\Big)\Big], \tag{1}$$

where $\boldsymbol{k}_m^*$ is the conjugate vector [26] of $\boldsymbol{k}_n$, $\text{Re}[\cdot]$ is the real part of a complex vector, and α is a specified angle. In this framework, the semantic difference is modeled by the dot-product (*i.e.,* $\boldsymbol{q}_m^\top \boldsymbol{k}_n^*$), which is different from the way (*i.e.,* "*rotation*") to model the positional difference (*i.e.,* $(m-n)\cdot\alpha$). As we can see, semantic and positional difference are essentially modeled in different mathematical forms, which potentially limits the expressive capacity of transformer models, especially under varied interaction scenarios.

Euler's Formula. It is a mathematical equation describing the relation among different forms of complex vectors. Formally, given a complex vector in the rectangular form $\boldsymbol{r} + i\boldsymbol{s}$, where $\boldsymbol{r}, \boldsymbol{s}$ are the

real and imaginary parts respectively, and i is the imaginary unit that satisfies $i^2 = -1$, the Euler's formula can be written as:

$$\boldsymbol{r} + i\boldsymbol{s} \text{ (rectangular form)} = \boldsymbol{\lambda} e^{i\boldsymbol{\theta}} \text{ (polar form)}, \tag{2}$$

where $\boldsymbol{\lambda} e^{i\boldsymbol{\theta}}$ is the representation of a complex vector in the polar form, $\boldsymbol{\lambda} = \sqrt{\boldsymbol{r}^2 + \boldsymbol{s}^2}$ and $\boldsymbol{\theta} = \text{atan2}(\boldsymbol{s}, \boldsymbol{r})$ are the *modulus* and *phase* of a complex vector, and $\text{atan2}(\boldsymbol{y}, \boldsymbol{x})$ is the arctangent function.

3 METHODOLOGY

In this section, we introduce the proposed transformer model with complex vector attention, named **EulerFormer**. Next, we first present a general introduction to our model.

3.1 Overview of EulerFormer

As shown in Figure 1, EulerFormer extends the vanilla transformer with a novel attention mechanism in complex vector space. Compred with prior work (*i.e.,* RoPE [26]), the major novelty of our approach lies in the unified theoretical framework to formulate both *semantic difference* and *positional difference* when modeling the sequential dependencies in the attention module, thus achieving a stronger expressive capacity in user behavior modeling.

Specially, it introduces two key technical improvements: (i) both semantic difference $\Delta\boldsymbol{S}$ and positional difference $\Delta\boldsymbol{P}$ are modeled via rotation angles in a complex vector space. With Euler transformation, it can efficiently transform token embeddings into *polar-form* complex vectors. As such, it casts the dot-product attention into a *complex vector rotation*, and enables it to model the semantic difference and positional difference in a unified rotation form (*i.e.,* "$\exp[i(\Delta\boldsymbol{S} + \Delta\boldsymbol{P})]$", Section 3.2.1); (ii) it proposes an adaptation function, which can adaptively integrate the semantic difference and positional difference according to the semantic contexts. By introducing learnable adaptation function $\text{Adapt}(\cdot)$, the semantic difference can be adjusted to be well aligned with positional difference (*i.e.,* "$\exp[i(\text{Adapt}(\Delta\boldsymbol{S}) + \Delta\boldsymbol{P})]$", Section 3.2.2). Furthermore, based on this framework, a phase contrastive task is introduced to improve the anisotropy of contextual representations, which can further improve the effectiveness of EulerFormer.

Next, we will introduce the complex space attention mechanism with unified formulation and adaptation function (Section 3.2), and then present the phase contrastive learning strategy (Section 3.3).

3.2 Attention Modeling in Complex Space

In our approach, we propose a novel complex vector attention mechanism, which can effectively integrate semantic and positional information. Specifically, we focus on attention modeling with both *absolute* and *relative* positional encoding.

3.2.1 Absolute Positional Encoding via Complex Rotation. Different from the original transformer [32], we formulate the self-attention mechanism in a complex vector space, where the semantic information is expressed by the phase and positional information is modeled by the complex rotation. Such a formulation enables the modeling of positional and semantic information in a unified mathematical form. Our approach first transforms the tokens into the complex vector space, and then models the self-attention mechanism in the complex vector space. Next, we introduce the details for each step.

Euler Transformation. In order to improve the expressiveness of contextual representations, we develop a transformation function, *i.e., Euler transformation*, to transform the input from original *real vector space* into a *complex vector space*. Our idea is inspired by a recent work [30], which finds that complex vector space inherently possesses some inherent advantages when performing vector rotation. However, different from the work [30], to maintain the consistency of the dot-product results before and after the transformation, we perform a split operation on a given token embedding $\boldsymbol{x}_j \in \mathbb{R}^d$, to transform it into the complex vector space. In this split, the first half of and the second half of the token embedding serve as the real and imaginary parts, respectively. As such, we can utilize Euler's formula (See in Eq. (2)) to obtain its polar-form representation. The transformation process is formulated as:

$$\tilde{\boldsymbol{x}}_j = \mathcal{F}(\boldsymbol{x}_j) = \mathcal{F}([\boldsymbol{r}_j; \boldsymbol{s}_j]) = \boldsymbol{\lambda}_j \exp(i\boldsymbol{\theta}_j), \tag{3}$$

where $\boldsymbol{r}_j = \boldsymbol{x}_j[1 : \frac{d}{2}]$ and $\boldsymbol{s}_j = \boldsymbol{x}_j[\frac{d}{2} + 1 : d]$ are the two vectors obtained after splitting $\boldsymbol{x}_j$, serving as the real and imaginary parts of the transformed complex vector $\tilde{\boldsymbol{x}}_j$ accordingly, and $\boldsymbol{\lambda}_j = \sqrt{\boldsymbol{r}_j^2 + \boldsymbol{s}_j^2}$ and $\boldsymbol{\theta}_j = \text{atan2}(\boldsymbol{s}_j, \boldsymbol{r}_j)$ are the polar-form representations of $\tilde{\boldsymbol{x}}_j$ using Euler's formula (See in Eq. (2)). The above Euler transformation $\mathcal{F} : \mathbb{R}^d \to \mathbb{C}^{d/2}$ provides a simple yet flexible approach to formulate the token embeddings and self-attention mechanism in the complex vector space. In this form, the dot-product operation of two token embeddings is cast into a complex rotation multiplied with their corresponding modulus:

$$\begin{aligned} \boldsymbol{x}_j^\top \boldsymbol{x}_k &= \boldsymbol{r}_j^\top \boldsymbol{r}_k + \boldsymbol{s}_j^\top \boldsymbol{s}_k = \text{Re}[\tilde{\boldsymbol{x}}_j^\top \tilde{\boldsymbol{x}}_k^*] \\ &= (\boldsymbol{\lambda}_j \odot \boldsymbol{\lambda}_k)^\top \text{Re}\left[\exp\left(i(\boldsymbol{\theta}_j - \boldsymbol{\theta}_k)\right)\right]. \end{aligned} \tag{4}$$

where $\tilde{\boldsymbol{x}}_k^* = \boldsymbol{\lambda}_k \exp(-i\boldsymbol{\theta}_k)$ is the conjugate vector of $\tilde{\boldsymbol{x}}_k$. As we can see, the original semantic difference between two vectors measured in dot product can be converted into a form of angular difference of complex vectors, which is the key to unify the modeling of semantic and positional difference (as discussed below).

Rotation-based Absolute Positional Encoding. Based on the Euler transformation, we next discuss how to integrate the positional information into the complex representations. Formally, given the user interaction sequence $\{v_1, \cdots, v_N\}$, the input of the traditional Transformers is given by $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\} \in \mathbb{R}^{N \times d}$, where $\boldsymbol{x}_j = \boldsymbol{e}_j + \boldsymbol{p}_j$, $\boldsymbol{e}_j$ is the embedding of item v_j and $\boldsymbol{p}_j$ is the embedding of j-th position. To further improve the effectiveness of position embedding, we perform Euler transformation and incorporate a set of *rotary embeddings* (*i.e.,* $\{\boldsymbol{\psi}_1, \cdots, \boldsymbol{\psi}_N\} \in \mathbb{R}^{N \times d/2}$):

$$\tilde{\boldsymbol{x}}'_j = \tilde{\boldsymbol{x}}_j \odot \exp(i\boldsymbol{\psi}_j) = \boldsymbol{\lambda}_j \exp(i(\boldsymbol{\theta}_j + \boldsymbol{\psi}_j)), \tag{5}$$

where $\tilde{\boldsymbol{x}}_j$ is the transformed representation of $\boldsymbol{x}_j$ (See in Eq.(3)). In this way, the token embeddings are rotated based on their absolute position (*i.e.,* $\boldsymbol{\theta}'_j = \boldsymbol{\theta}_j + \boldsymbol{\psi}_j, \boldsymbol{\lambda}'_j = \boldsymbol{\lambda}_j$). Since the above representations are still in polar space, which cannot directly perform aggregation operation (*e.g.,* vector addition) in the self-attention formulation, we further conduct the *inverse transformation* (*i.e.,* $\mathcal{F}^{-1}$) to transform them into the real vector space, shown as:

$$\boldsymbol{x}'_j = \mathcal{F}^{-1}(\tilde{\boldsymbol{x}}'_j) = \mathcal{F}^{-1}\left(\boldsymbol{\lambda}'_j \exp(i\boldsymbol{\theta}'_j)\right) = [\boldsymbol{\lambda}'_j \cos\boldsymbol{\theta}'_j; \boldsymbol{\lambda}'_j \sin\boldsymbol{\theta}'_j], \tag{6}$$

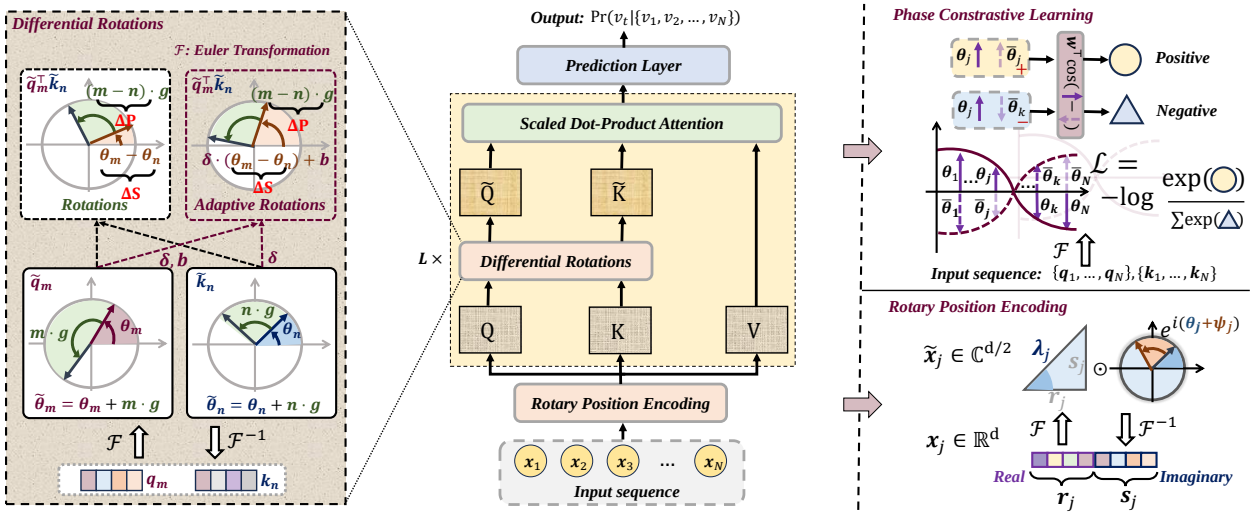


Figure 1: The overall architecture of EulerFormer.

where '[;]" denotes the concatenation operation. In this way, we obtain a set of transformed token embeddings (*i.e.,* $\{\boldsymbol{x}'_1, \cdots, \boldsymbol{x}'_N\}$), with positional information modeled by the additive embedding (*e.g.,* $\boldsymbol{p}_j$) and the rotary embedding (*e.g.,* $\boldsymbol{\psi}_j$). Further, as previous work shows [30], these two kinds of embeddings mutually enhance each other, which can jointly improve the position modeling capacity of transformers. Thus, we use the embeddings $\boldsymbol{X}' = \{\boldsymbol{x}'_1, \cdots, \boldsymbol{x}'_N\} \in \mathbb{R}^{N\times d}$ in the following self-attention formulation.

3.2.2 Differential Rotation for Relative Positional Encoding. Considering that the above transformations mainly integrate positional information in an absolute way, it cannot learn the relative positional relationship, which limits the sequential modeling capacity. To address this issue, we propose a differential rotation mechanism, which enables the model to better capture the sequential dependencies. As shown in Figure 1, it takes as input a set of queries and keys, and outputs a set of rotated queries and keys. In this way, we can integrate it within each transformer layer. Next, we will discuss the encoding process within a single layer.

Rotation-based Relative Position Encoding. We follow the basic process of the self-attention mechanism (See Section 2). Formally, given a token embedding sequence, we first project it into a set of queries $\boldsymbol{Q} = \{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_N\} \in \mathbb{R}^{N\times d}$ and keys $\boldsymbol{K} = \{\boldsymbol{k}_1, \cdots, \boldsymbol{k}_N\} \in \mathbb{R}^{N\times d}$. As such, the differential rotation mechanism is shown as:

$$\tilde{\boldsymbol{q}}_j = \mathcal{F}(\boldsymbol{q}_j) \odot \exp(i\boldsymbol{\alpha}_j) = \boldsymbol{\lambda}^Q_j \exp\Big(i(\boldsymbol{\theta}^Q_j + \boldsymbol{\alpha}_j)\Big), \tag{7}$$

$$\tilde{\boldsymbol{k}}_j = \mathcal{F}(\boldsymbol{k}_j) \odot \exp(i\boldsymbol{\alpha}_j) = \boldsymbol{\lambda}^K_j \exp\Big(i(\boldsymbol{\theta}^K_j + \boldsymbol{\alpha}_j)\Big), \tag{8}$$

$$\boldsymbol{\alpha}_j = j \cdot \boldsymbol{g},\ g_t = 10000^{-\frac{2t}{d}}. \tag{9}$$

In such a transformation, each query or key token is first transformed into a complex vector space (*e.g.,* $\mathcal{F}(\boldsymbol{q}_j) = \boldsymbol{\lambda}^Q_j \exp(i\boldsymbol{\theta}^Q_j)$), and then rotated by a fixed angle (*i.e.,* $\boldsymbol{\alpha}_j \in \mathbb{R}^{d/2}$) according to its absolute position. Note that the differential rotation variable $\boldsymbol{\alpha}_j$ is pre-defined, which is not tuned during the training process. Here we follow the sinusoidal function used in transformer [32] that empirically sets the angle base to 10000. According to Eq. (4), the dot-product score between queries and keys can be formulated as:

$$\mathrm{Re}[\tilde{\boldsymbol{q}}^\top_m \tilde{\boldsymbol{k}}^*_n] = (\boldsymbol{\lambda}^Q_m \odot \boldsymbol{\lambda}^K_n)^\top \mathrm{Re}\left[\exp\left(i\Big(\underbrace{(\boldsymbol{\theta}^Q_m - \boldsymbol{\theta}^K_n)}_{\Delta S} + \underbrace{(\boldsymbol{\alpha}_m - \boldsymbol{\alpha}_n)}_{\Delta P}\Big)\right)\right]. \tag{10}$$

This equation achieves the unified modeling of both semantic difference and positional difference. Specifically, the first term of the rotation part is the *semantic difference* (*i.e.,* $\Delta S = \boldsymbol{\theta}^Q_m - \boldsymbol{\theta}^K_n$), which reflects the semantic information and constitutes the main part of dot-product result. The second term (*i.e.,* $\Delta P = \boldsymbol{\alpha}_m - \boldsymbol{\alpha}_n = (m-n)\cdot\boldsymbol{g}$) is the *positional difference* modeled by the relative position (*i.e.,* $m-n$).

Generalized Adaptive Rotation. In the above formulation, we have discussed how to model relative positions with complex rotation. However, in practice, the semantic contexts may greatly vary in different interaction scenarios, and the straightforward integration (Eq. (10)) cannot effectively integrate varied semantic difference and relatively fixed positional difference. To address this issue, we propose an adaptive rotation mechanism to generalize Eq. (10) by adapting to different semantic contexts. Formally, given the query $\boldsymbol{q}_j$ and key $\boldsymbol{k}_j$, we conduct Euler transformation to obtain their polar-form representations as $\boldsymbol{\lambda}^Q_j \exp(i\boldsymbol{\theta}^Q_j)$ and $\boldsymbol{\lambda}^K_j \exp(i\boldsymbol{\theta}^K_j)$ respectively. As such, we perform linear transformations on the phase of queries and keys accordingly:

$$\tilde{\boldsymbol{\theta}}^Q_j = \boldsymbol{\delta}^l \odot \boldsymbol{\theta}^Q_j + \boldsymbol{b}^l, \tag{11}$$

$$\tilde{\boldsymbol{\theta}}^K_j = \boldsymbol{\delta}^l \odot \boldsymbol{\theta}^K_j, \tag{12}$$

where $\boldsymbol{\delta}^l$ and $\boldsymbol{b}^l$ are the layer-specific *scale factor* and *differential bias* for adapting the distributions at the l-th layer, and $\tilde{\boldsymbol{\theta}}^Q_j$ and $\tilde{\boldsymbol{\theta}}^K_j$ are the rotated phases of the query and key. In this way, the dot

product between transformed queries and keys is given by:

$$\mathrm{Re}[\tilde{\boldsymbol{q}}_m^\top \tilde{\boldsymbol{k}}_n^*] = (\boldsymbol{\lambda}_m^Q \odot \boldsymbol{\lambda}_n^K)^\top \mathrm{Re}\left[\exp\left(i\Big((\tilde{\boldsymbol{\theta}}_m^Q - \tilde{\boldsymbol{\theta}}_n^K) + (\boldsymbol{\alpha}_m - \boldsymbol{\alpha}_n)\Big)\right)\right]$$

$$= (\boldsymbol{\lambda}_m^Q \odot \boldsymbol{\lambda}_n^K)^\top \mathrm{Re}\left[\exp\left(i\Big(\underbrace{\boldsymbol{\delta}^l \odot (\boldsymbol{\theta}_m^Q - \boldsymbol{\theta}_n^K) + \boldsymbol{b}^l}_{\mathrm{Adapt}(\Delta \boldsymbol{S})} + \underbrace{(m-n)\cdot \boldsymbol{g}}_{\Delta \boldsymbol{P}}\Big)\right)\right]. \tag{13}$$

This function is the core of EulerFormer for achieving adaptive positional encoding. Different from RoPE [26], the semantic difference can be adaptively adjusted in each transformer layer, enabling more effective integration of semantic and positional information. Note that another adaptive integration way is to incorporate learnable parameters to adjust the positional differences. However, it would potentially affect the abilities of positional encoding and extrapolation (See more discussions in Section 4.3.1).

3.3 Anisotropic Representation Learning

A major training issue of transformer-based models is that it often induces an isotropic representation space [18], which will become more severe in the recommendation scenario with magnitude more item tokens [13]. In this case, the semantic difference may be very small and makes the integration with positional difference less effective in Eq. (10). To alleviate this issue, we propose a phase contrastive learning task, which can enhance the anisotropy of contextual representations in EulerFormer.

Phase-Contrastive Learning. This training task aims to enhance the anisotropy among the representations of all the tokens in a given sequence. Previous work [11–13, 27] mainly learns the representations in the original real vector space. These approaches cannot effectively capture the orientation relationships of vectors since both the magnitude and orientation of the vectors are optimized as a coupled whole. Different from them, as we can decouple the semantic and positional differences explicitly (See in Eq. (10)), and it is feasible to only adjusts the orientation of token embeddings and keeps their magnitude during learning. This approach can effectively enhance the discriminability of different items, while not compromising the other semantics (*e.g., modulus*). For the queries in each layer, we only consider the phase of the transformed embeddings, *i.e.,* $\tilde{\boldsymbol{\Theta}} = \{\tilde{\boldsymbol{\theta}}_1^Q, \cdots, \tilde{\boldsymbol{\theta}}_N^Q\}$. As such, we conduct data augmentation by randomly masking the phase in the original sequence. Given the sequence, the augmented ones ($\bar{\boldsymbol{\theta}}_j^Q$) are considered as positives, while other within-sequence ones are considered as negatives. We consider a batch setting of B training samples $\boldsymbol{\Delta} = \{\tilde{\boldsymbol{\Theta}}_k\}_{k=1}^{|B|}$, where each sample contains a sequence of user interaction records. Then the phase contrastive task is formulated as:

$$\mathcal{L}_{query} = -\frac{1}{B}\sum_{\tilde{\boldsymbol{\Theta}}\in\boldsymbol{\Delta}}\sum_{j=1}^{N}\log\frac{\exp\left(\boldsymbol{w}^\top \cos(\tilde{\boldsymbol{\theta}}_j^Q - \bar{\boldsymbol{\theta}}_j^Q)/\tau\right)}{\sum_{j'=1}^{N}\exp\left(\boldsymbol{w}^\top \cos(\tilde{\boldsymbol{\theta}}_j^Q - \bar{\boldsymbol{\theta}}_{j'}^Q)/\tau\right)}, \tag{14}$$

where $\boldsymbol{w}$ is a learnable projection parameter and τ is the temperature parameter. Accordingly, we also build the phase contrastive task on the keys of each Transformer layer, *i.e.,* $\mathcal{L}_{key}$. The total loss is the fusion of both term:

$$\mathcal{L}_{Con} = \mathcal{L}_{query} + \mathcal{L}_{key}. \tag{15}$$

Prediction and Optimization. Since we take sequential recommendaiton as the performance test task, we follow the transformer-based recommenders [12, 13] to make predictions. Formally, given the output sequence representations of the last layer $\{\hat{\boldsymbol{x}}_1, \cdots, \hat{\boldsymbol{x}}_N\}$. With the first t items encoded, the next item is predicted based on $\hat{\boldsymbol{x}}_t$. Finally, the user's preference to an arbitrary item j is given by:

$$\Pr(j|\{v_1, v_2, \cdots, v_t\}) = \hat{\boldsymbol{x}}_t^\top \boldsymbol{e}_j. \tag{16}$$

Then we adopt the cross entropy loss to deliver the next-item prediction task:

$$\mathcal{L}_{CE} = -\log\frac{\exp(\hat{\boldsymbol{x}}_t^\top \boldsymbol{e}_j)}{\sum_{j'=1}^{|I|}(\exp \hat{\boldsymbol{x}}_t^\top \boldsymbol{e}_{j'})}. \tag{17}$$

The final training loss is designed by combining the polar-contrastive learning task and the next-item prediction task using the weight ϵ:

$$\mathcal{L} = \mathcal{L}_{CE} + \epsilon \cdot \mathcal{L}_{Con}. \tag{18}$$

3.4 Discussion

In this part, we highlight the advantages of EulerFormer compared to previous methods (*e.g.,* RoPE [26]) in the following two aspects.

• *Theoretical Superiority.* The theoretical framework of EulerFormer possess a high degree of *completeness* and *generality*. Firstly, it enables the learning of *semantic angles*, *modulus*, and other important factors related to the complex vectors, compatible with typical training methods (*e.g.,* contrastive learning) and supporting diverse data types (*e.g.,* complex sequences). Secondly, prior positional encoding methods [23, 26, 28] can be instantiated in our framework. For example, RoPE [26] can be viewed as a special case of EulerFormer, by specifying $\boldsymbol{\delta} = \mathbf{1}, \boldsymbol{b} = \mathbf{0}$ in Eq (13). Thus, in principle, EulerFormer would be potentially more powerful than previous methods [4, 26, 32]. Specially, one representative advantage of EulerFormer lies in the *flexibility* for controlling the *gradient* of *long-term decay* [26], referring to the property that the attention score decreases as relative distances increase. Formally, based on Eq. (13), given the query $\boldsymbol{q}_m$ and key $\boldsymbol{k}_n$ with their *distance* $D = m - n$, the gradient of the attention score can be written as:

$$\frac{\partial(\boldsymbol{q}_m^T \boldsymbol{k}_n)}{\partial D} = -(\boldsymbol{\lambda}_m^Q \odot \boldsymbol{\lambda}_n^K \odot \boldsymbol{g})^\top \sin\Big(\boldsymbol{\delta} \odot (\boldsymbol{\theta}_m^Q - \boldsymbol{\theta}_n^K) + \boldsymbol{b} + D \cdot \boldsymbol{g}\Big),$$
$$= -\boldsymbol{C}^\top \sin(\boldsymbol{\delta} \odot \Delta \boldsymbol{S} + \boldsymbol{b} + \Delta \boldsymbol{P}).$$

Here $\boldsymbol{C} = \boldsymbol{\lambda}_m^Q \odot \boldsymbol{\lambda}_n^K \odot \boldsymbol{g}$ (*non-negative*). It is evident that RoPE [26] is less capable of adjusting such gradient (*i.e.,* "$-\boldsymbol{C}^\top \sin(\Delta \boldsymbol{S} + \Delta \boldsymbol{P})$"), which may result in a *positive gradient* when $\Delta \boldsymbol{S} + \Delta \boldsymbol{P} \in [\pi + 2k\pi, (2k+2)\pi], k \in \mathbb{Z}$, even leading to an increase in attention score with increasing distance D. In comparison, EulerFormer can adaptively integrate $\Delta \boldsymbol{S}$ and $\Delta \boldsymbol{P}$, allowing for suitable $\boldsymbol{\delta}$ and $\boldsymbol{b}$ to maintain the negative gradient, thereby ensuring the properties of long-term decay. Besides, we can also set $\boldsymbol{\delta} = -\Delta P/\Delta S, \boldsymbol{b} = \boldsymbol{\pi}/2$ to increase the decay rate. In general, our framework provides a capable solution to model complicated sequential patterns and also

has great potential for application in other fields (*e.g.,* NLP). The comparison of transformer variants is presented in Table 1.

• *Efficiency.* EulerFormer has inherently good properties to ensure its efficiency. Specifically, the designed Euler transformation $\mathcal{F}$ is *reversible*, and the inner product of real vectors and transformed complex vectors is *consistent*. Based on this property, EulerFormer employs an efficient *inverse transformation* ($\mathcal{F}^{-1}$) for mapping the rotated complex tokens into the real vector space, thus avoiding the time consumption of complex inner products. In this way, the cost of Euler transformation $\mathcal{F}$ and its inverse transformation $\mathcal{F}^{-1}$ is $O(d)$. Thus given N queries and keys, the complexity of absolute positional encoding is $O(Nd)$. Besides, the cost of performing adaptive rotation (Eq. (11)) is also $O(d)$ and the complexity of relative positional encoding is $O(Nd)$. Therefore, the total complexity of positional encoding module is $O(Nd)$. Generally, EulerFormer could achieve linear encoding complexity with sequence length N and hidden dimension d, with no need of any matrix multiplication, and is comparable to many efficient methods (See in Table 4).

Table 1: Comparison of different transformer variants. "Capable" denotes whether it has the ability of long-term decay, and "Flexible" denotes whether it can control the decay gradient.

Methods	Positional Encoding			Long-term Decay		Complexity
	Absolute	Relative	Adaptive	Capable	Flexible	
Sinusoidal [32]	✔	✗	✗	✔	✗	$O(Nd)$
XLNet [4]	✗	✔	✗	✗	✗	$O(N^2d^2)$
RoPE [26]	✔	✔	✗	✔	✗	$O(Nd)$
EulerFormer (ours)	✔	✔	✔	✔	✔	$O(Nd)$

4 EXPERIMENTS

To verify the effectiveness of the proposed EulerFormer, we conduct extensive experiments and report detailed analysis results.

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Metrics. To evaluate the performance of EulerFormer, we conduct experiments on four publicly available datasets: **(1) MovieLens** [8] contains users' ratings on movies, which is a popular dataset for sequential recommendations. We adopt two widely used versions: **ML-1M** and **ML-20M**. **(2) Yelp** is a business dataset containing user reviews for various restaurants. We adopt its latest version, which was released in 2022 (**Yelp2022**). **(3) Amazon_Books** [22] is part of the Amazon review datasets, covering rich user interactions with an extensive range of books. Following existing work [7, 11, 13], we group user interactions in chronological order across all datasets and split the data using the leave-one-out strategy. Moreover, we use three common evaluation metrics, including Recall, MRR, and NDCG, for the evaluation of top-k ($k = 10$) recommendations. We rank the ground-truth item of each sequence among all other items for evaluation on the test set and finally report the average score of all test users.

4.1.2 Baselines. To show the effectiveness of our model, we integrate EulerFormer into existing state-of-the-art sequential recommenders, including: **(1) SASRec** [15] adopts the transformer for obtaining the sequence representation. **(2) BERT4Rec** [27] adapts the BERT [5] model by using the cloze task to enhance sequence representations. **(3) CORE** [12] proposes to unify the representation space for generating better representations. **(4) SASRecF** [15] improves SASRec [15] by introducing item features as additional contextual information. **(5) FDSA** [42] equips multiple transformers to learn item-wise and feature-wise representations. Further, we also compare our model with the state-of-the-art positional encoding methods in NLP, including: **(1) Sinusoidal** [32] uses sinusoidal functions to generate positional embeddings. **(2) XLNet** [4] explicitly employs relative position embeddings to learn relative position information. **(3) RoPE** [26] sets specific rotatory matrices based on the position of each key or query, enabling the computation of attention scores using relative positional information.

Table 2: The statistics of datasets.

Dataset	# Users	# Items	# Interactions	# Sparsity	# Avg. N
ML-1M	6,034	3,124	834,449	95.57%	138.31
Yelp2022	9,370	24,627	811,573	99.65%	86.62
Amazon_Books	543,598	276,863	15,035,247	99.99%	27.66
ML-20M	137,524	14,259	16,447,894	99.16%	119.60

4.1.3 Implementation Details. We implement EulerFormer using a popular recommendation library RecBole [43]. We use the Adam [17] optimizer and carefully search the hyper-parameters of the compared methods. The dimension of the latent vectors is fixed to 64, and each sequence is truncated within a maximum length of 250. Following the original papers, we set the hyper-parameters of Transformer for all sequential models, including the number of layers as $L = 2$, attention head as $h = 2$, inner size of feed-forward layers as 256, and dimension size as $d = 64$. For EulerFormer, we search the temperature parameters τ (See in Eq. (14)) among $\{0.5, 1, 5, 10\}$ and loss weight ϵ (See in Eq. (18)) among $\{1e-4, 1e-5, 1e-6\}$. Our code will be public after acceptance.

4.2 Overall Performance

4.2.1 Effectiveness in Sequential Recommender Systems. To demonstrate the effectiveness of EulerFormer in improving recommendation performance, we integrate it into representative Transformer-based recommender models. We report the main experimental results in Table 3 and make the following observations:

(1) For all methods, equipped with EulerFormer, the model performs better than the original ones in all cases. Notably, integrating EulerFormer into SASRec [15] (EulerFormer+SASRec) exhibits improvements of 10.46% and 8.30% with respect to the NDCG metric on ML-1M and Yelp2022 dataset respectively. It demonstrates the effectiveness of our approach for enhancing the positional encoding capacity of transformer architecture.

(2) The approach EulerFormer+BERT4Rec [27] achieves the best performance on ML-1M and ML-20M datasets, suggesting the powerful capacity of BERT model in sequential recommendations. Further, EulerFormer+CORE [12] has notable improvement on Yelp2022 and Amazon_Books datasets, which shows that EulerFormer can effectively enhance CORE's performance in sparse datasets.

(3) Our method shows a smaller improvement for SASRecF [15] and FDSA [42] than for SASRec [15]. It may be caused by the fact that item features will enhance the isotropy of the representations, leading to the model being less sensitive to positional information.

Table 3: Performance (%) of backbone models (w): use EulerFormer and (w/o): use original Transformer. All improvements are statistically significant (*i.e.,* two-sided t-test with p < 0.01) over original models.

Dataset	Metric	SASRec			BERT4Rec			CORE			SASRecF			FDSA		
		w/o	w	*Improv.*	w/o	w	*Improv.*	w/o	w	*Improv.*	w/o	w	*Improv.*	w/o	w	*Improv.*
ML-1M	Recall@10	23.70	25.31	+6.79%	27.15	**28.11**	+3.54%	15.33	15.58	+1.63%	24.08	25.44	+5.64%	23.59	24.42	+3.51%
	MRR	8.88	10.07	+13.40%	10.99	**11.59**	+5.46%	3.86	3.93	+1.81%	9.79	10.34	+5.61%	9.02	9.53	+5.65%
	NDCG@10	12.33	13.62	+10.46%	14.74	**15.42**	+4.95%	6.49	6.60	+1.69%	13.12	13.86	+5.64%	12.40	12.99	+4.75%
Yelp2022	Recall@10	5.38	5.52	+2.60%	5.06	5.17	+2.17%	5.72	**6.02**	+5.24%	4.75	4.92	+3.58%	5.81	5.85	+0.69%
	MRR	1.84	**2.06**	+11.95%	1.71	1.77	+3.51%	1.93	2.05	+6.22%	1.50	1.56	+4.00%	1.89	1.90	+0.53%
	NDCG@10	2.65	2.87	+8.30%	2.48	2.56	+3.22%	2.80	**2.96**	+5.71%	2.25	2.33	+3.55%	2.79	2.81	+0.72%
Amazon_Books	Recall@10	11.34	**11.60**	+2.29%	10.15	10.18	+0.30%	8.07	8.86	+9.79%	11.25	11.41	+1.42%	9.19	9.23	+0.44%
	MRR	5.84	6.25	+7.02%	5.36	5.56	+3.73%	2.72	3.08	+13.24%	6.46	**6.56**	+1.55%	4.84	4.89	+1.03%
	NDCG@10	7.13	7.50	+5.19%	6.48	6.64	+2.47%	3.97	4.44	+11.84%	7.58	**7.70**	+1.58%	5.85	5.91	+1.03%
ML-20M	Recall@10	20.26	20.45	+0.94%	21.92	**22.58**	+3.01%	11.65	11.89	+2.06%	20.61	20.82	+1.02%	18.52	18.80	+1.51%
	MRR	8.36	8.68	+3.83%	9.46	**9.79**	+3.49%	3.01	3.09	+2.66%	8.73	8.90	+1.95%	7.77	7.78	+0.13%
	NDCG@10	11.13	11.42	+2.61%	12.37	**12.78**	+3.31%	5.00	5.11	+2.20%	11.50	11.68	+1.57%	10.27	10.35	+0.79%

4.2.2 Comparison of Different Positional Encoding Approaches. To demonstrate the superiority of EulerFormer over the state-of-the-art positional encoding models (See in Section 4.1.2), we take the SASRec as the base model and report the experimental results in Table 4. Accordingly, we have the following observations:

(1) XLNet [4] outperforms Sinusoidal [32] on the Yelp2022, Amazon_Books and ML-20M datasets, which shows the superiority of relative positional encoding in handling sparse datasets.

(2) RoPE [26] performs very well on the ML-1M, Yelp2022 and Amazon_Books datasets. It indicates the rotary encoding is more capable of modeling positional information. On the other hand, the base model (learned position embeddings) demonstrates competitive performance across all datasets, which highlights the significance of absolute positional encoding in recommendation tasks.

(3) Our proposed EulerFormer consistently outperforms all the other baseline models on all four datasets. It shows the effectiveness of adaptively modeling both the absolute and relative positional information via the proposed adaptive rotation mechanism.

As for the efficiency, we can observe that the base model (learned position embedding [32]) is the most efficient since its structure is simple and does not require additional calculations. Compared to Sinusoidal [32], RoPE [26] requires the reversal of embedding dimensions, and thus it has higher latency. The latency of XLNet [4] is much larger due to the complicated positional encoding strategy. As a comparison, the latency of EulerFormer is much less than RoPE [26] and XLNet [4], and it is comparable to the efficient algorithm Sinusoidal [32]. With the highest accuracy and linear complexity, EulerFormer has a great potential to be applied into large-scale recommender systems.

Table 4: Performance (%) of different positional encoding methods with the base model as SASRec. "*" denotes that the improvements are significant at the level of 0.01 with paired t-test.

Dataset	Model	Recall@10	MRR	NDCG@10	Latency
ML-1M	Base	23.70	8.88	12.33	1.85 s
	Sinusoidal	23.54	8.94	12.34	1.94 s
	XLNet	23.50	8.88	12.28	5.12 s
	RoPE	23.85	8.91	12.39	2.47 s
	EulerFormer	**25.31***	**10.07***	**13.62***	2.14 s
	Improv.	+6.12%	+12.64%	+9.93%	–
Yelp2022	Base	5.38	1.84	2.65	3.07 s
	Sinusoidal	4.61	1.58	2.28	3.54 s
	XLNet	5.42	1.89	2.71	8.77 s
	RoPE	5.44	2.02	2.81	4.58 s
	EulerFormer	**5.52***	**2.06***	**2.87***	3.56 s
	Improv.	+1.47%	+1.98%	+2.14%	–
Amazon_Books	Base	11.34	5.84	7.13	14.88 s
	Sinusoidal	10.85	5.65	6.87	15.03 s
	XLNet	11.26	5.84	7.11	24.39 s
	RoPE	11.34	5.89	7.17	16.63 s
	EulerFormer	**11.60***	**6.25***	**7.50***	15.18 s
	Improv.	+2.29%	+6.11%	+4.60%	–
ML-20M	Base	20.26	8.36	11.13	2.32 s
	Sinusoidal	19.66	8.00	10.72	2.59 s
	XLNet	19.75	8.12	10.84	5.64 s
	RoPE	19.96	8.14	10.90	3.08 s
	EulerFormer	**20.45***	**8.68***	**11.42***	2.54 s
	Improv.	+0.94%	+3.83%	+2.61%	–

4.3 Further Analysis of EulerFormer

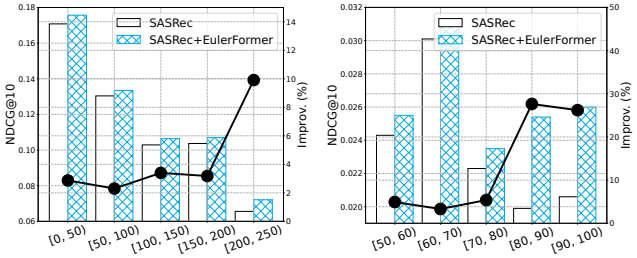
In this section, we further perform a series of detailed analysis on the proposed EulerFormer to confirm its effectiveness. Due to the limited space, we only report the results on ML-1M and Yelp2022 datasets, and the observations are similar on other datasets.

4.3.1 Ablation Study. We analyze how our proposed components influence the final encoding performance. We prepare four variants of the proposed EulerFormer model for comparisons, including: (1) *w/o adaptation function* that removes the weights $\boldsymbol{\delta}$ and bias $\boldsymbol{b}$ in Eq. (13), *i.e.,* directly uses Eq. (10) for positional encoding; (2) *w learnable positional encoding* that removes the adaptive weight $\boldsymbol{\delta}$ and $\boldsymbol{b}$ and sets $\boldsymbol{g}$ in Eq. (10) as a learnable parameter; (3) *w/o differential rotation* that removes the relative positional encoding module (*i.e.,* $\Delta\boldsymbol{P}$ in Eq. (10)); (4) *w/o rotary embedding* that removes the rotary embedding in Eq. (5); (5) *w/o phase-contrastive learning* that removes the phase contrastive learning in the training process. The results are shown in Table 5. We can observe that all the proposed methods are useful to improve the performance. Specifically, the variant (1) and (2) show a great decrease, which indicates that our proposed adaptive rotation mechanism is more suitable to model the positional information in the self-attention formulation.

Table 5: Ablation study on the ML-1M and Yelp2022 datasets.

Dataset	Variant	Recall@10	MRR	NDCG@10
ML-1M	(0): EulerFormer	**25.31**	**10.07**	**13.62**
	(1): w/o adaptation function	24.61	9.60	13.11
	(2): w learnable positional encoding	23.79	8.93	12.39
	(3): w/o differential rotation	24.68	9.42	12.96
	(4): w/o rotary embedding	24.76	9.57	13.11
	(5): w/o phase-contrastive learning	24.76	9.44	13.02
Yelp2022	(0): EulerFormer	**5.52**	**2.06**	**2.87**
	(1): w/o adaptation function	5.37	1.98	2.76
	(2): w learnable positional encoding	5.23	1.96	2.72
	(3): w/o differential rotation	5.41	1.85	2.67
	(4): w/o rotary embedding	5.37	1.95	2.74
	(5): w/o phase-contrastive learning	5.31	1.93	2.78

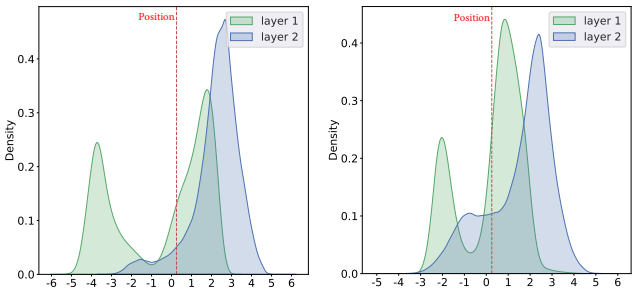
4.3.2 Performance w.r.t. Different Sequence Lengths. In recommendation scenarios, the sequential length is often very short. Typically, when the length satisfies $N > 1.5d$ (*e.g.,* 96), it is considered as a "long" sequence [20]. One advantage of adaptive positional encoding in EulerFormer lies in its capacity for handling the long-sequence data. To verify this, we split the test data into different groups according to the sequence length, and then compare the improved ratio of NDCG@10 score (*w.r.t.* SASRec) in each group. We cluster them by setting the interval to 50 on ML-1M dataset and 10 on Yelp2022 dataset for better presentation. From Figure 2, we can observe that our proposed EulerFormer can improve the performance in all cases, especially when the sequence length is long, *e.g.,* group $[200, 250)$ on the ML-1M dataset and group $[90, 100)$ on the Yelp2022 dataset. These results show that our approach can effectively enhance the model's capabilities in dealing with the long-term user's behavior, thereby improving the performance.



(a) ML-1M (b) Yelp2022

Figure 2: Performance analysis for different sequence lengths.

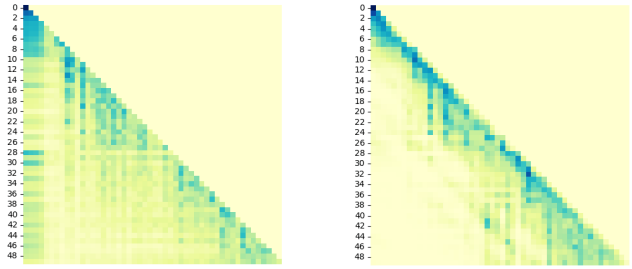
4.3.3 Analysis of Adaptive Rotations. As introduced in Section 3.2.2, EulerFormer uses an adaptive rotation mechanism to adapt semantic information in each layer, thereby better aligning with positional information. To verify this, we plot the distributions of semantic angular differences in different layers of EulerFormer with vanilla rotation (See Eq.(10)) and adaptive rotation (See Eq.(13)) in Figure 3 using Gaussian kernel density estimation (KDE [35]). We can see that in the vanilla rotation model, the distributions of semantic angular differences vary significantly, and they differ greatly from the anglular difference of position encoding. Whereas the distributions in the adaptive rotation model are closer, and it aligns more with the positional encoding angle. Flexibly adapting the semantic angular distributions can effectively help the model capture positional information, enhancing model's robustness to semantic variations.



(a) Vanilla Rotations (b) Adaptive Rotations

Figure 3: Visualization of token difference distributions at a certain position within different Transformer layers.

4.3.4 Visualization of Attention Scores. To further understand how EulerFormer helps the models deal with sequential data, we conduct a case study to visualize the attention scores. In Figure 4, we show a user's attention score map from the ML-1M dataset, where the lighter the color is, the higher the attention score is. Comparing the heatmaps generated by SASRec and SASRec+EulerFormer, we can observe that the attention of SASRec is relatively dispersed, hindering its ability to effectively capture useful information from the sequence. Equipped with EulerFormer, SASRec tends to assign larger attention scores on recent items, which indicates the improved capacity of capturing the relative position information of the sequential patterns. This is consistent with prior work [26]. It indicates the properties of EulerFormer in reducing the inter-token dependency as relative distances increase, which enables the model to focus on tokens that are closer in distance, thereby making it more capable of handling long-sequence data.

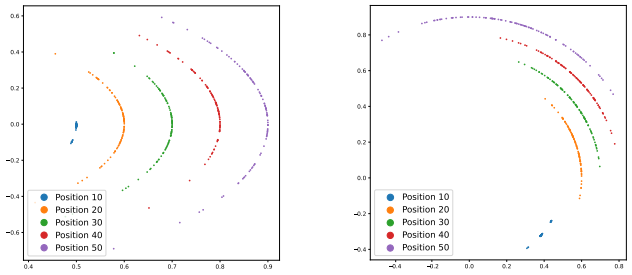


SASRec SASRec + EulerFormer

Figure 4: Heatmap of attention scores of different models.

4.3.5 Visualization of the Representations. As introduced in Section 3.3, we introduce a phase-contrastive task to enhance the anisotropy between different tokens. To verify this, we visualize the phase distribution of sequence tokens at different positions on the ML-1M dataset. As shown in Figure 5, for ease of illustration, we draw the phases located at different positions on different circular orbits, and the phases are displayed on those circles. We can observe that, without the phase-contrastive learning task, the phase distributions at different positions are very close. After undergoing phase contrastive learning, the differences in phase distributions at different positions become significant. These results demonstrate that our method can effectively enhance the anisotropy of item representations at different positions. This enhancement enables the

model to effectively distinguish items at various positions, thereby improving the effectiveness of positional encoding.

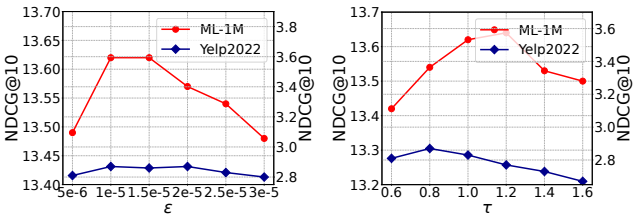


(a) without PCL (b) with PCL

Figure 5: Visualization of the phase distributions with (w) and without (w/o) phase constrastive learning (PCL).

4.3.6 Impact of the Coefficient ϵ. In the phase contrastive learning loss defined in Eq. (18), the coefficient ϵ can balance the two losses for sequential recommendations. To analyze the influence of ϵ, we vary ϵ from 5e-6 to 3e-5 and report the results in Figure 6(a). It demonstrates that an appropriate ϵ can effectively improve the recommendation performance of the EulerFormer. Specifically, setting the hyper-parameter ϵ to approximately 1e-5 results in enhanced performance on both ML-1M and Yelp2022 datasets.

4.3.7 Impact of the Temperature τ. To analyze the impact of temperature (See in Eq. (14)) on EulerFormer, we vary τ in the range of 0.6 to 1.6 and show the results in Figure 6(b). We can observe that too large a value of τ will cause poor performance, which is consistent with the experimental results reported in prior work [19, 39]. In addition, the optimal temperature for the Yelp2022 dataset is lower, indicating that the temperature setting for EulerFormer should be reduced for more sparse datasets. Generally, a temperature in the range of 0.8 to 1.2 can lead to good recommendation performance.



(a) ϵ (b) τ

Figure 6: Performance comparison *w.r.t.* different ϵ and τ.

5 RELATED WORK

Transformer-based Recommenders. The sequence recommendation [6, 7, 10, 15, 21, 27, 29] aims to predict the next item a user will interact with based on their historical interaction records. In the literature, transformer models have shown strong capabilities in sequential recommendations [10, 19, 21, 27, 29]. Most approaches [10, 19, 21, 27, 29] borrow ideas from NLP and directly use transformers to obtain sequential representations. Specifically, SASRec [15] directly employs the self-attention mechanism to learn the representations of next item; BERT4Rec [27] models the sequential recommendations as a cloze task to discover the latent features in the interaction sequence. In addition, a number of approaches [2, 7, 12, 36, 40, 42] have been proposed to expand the framework of transformer architectures for better enhancing the representation learning of transformers. For example, CORE [12] employs a representation-consistent encoder and an enhanced distance measurement method, augmenting the capabilities of traditional sequential encoders. However, these approaches are unable to effectively model positional information, which severely constrains the performance of the models.

Positional Encoding in Transformers. Learning effective positional information is crucial for enhancing the performance of transformers [1, 16, 24, 32, 44, 45]. Generally, positional encoding methods [4, 23–25, 38] are divided into two types: absolute positional encoding [32] and relative positional encoding [4, 23, 25, 38]. Specifically, the absolute positional encoding used in the original Transformer [32] typically has two variants: sinusoidal and learned position embeddings, with the latter being commonly used in existing recommender systems. Unlike absolute positional embeddings, relative positional embeddings [4, 9, 14, 23, 25, 25, 38] are often generated based on the offsets between keys and queries. For example, XLNet [4, 38] involves modifying the calculation of attention scores between keys and queries to incorporate learnable embeddings that correspond to relative positions. As a promising approach, a number of rotary position embedding based models [23, 26, 28] have been proposed, which set specific rotary matrices based on the position of each key or query. This enables the learning of both absolute and relative positions within the self-attention mechanism. Due to its superior performance, it is widely applied in the lastest LLMs, such as LLaMA [31], Palm [3] and T5 [25]. However, these approaches integrate the semantic and positional difference in different ways, which potentially limits the expressive capacity of transformer models, especially under varied interaction scenarios.

6 CONCLUSION

In this paper, we proposed a novel transformer variant **EulerFormer**. As the core contribution, EulerFormer provided a unified theoretical framework to formulate both semantic and positional information, thus possessing a stronger expressive capacity in sequential modeling. Specially, in EulerFormer, both semantic difference and positional difference among tokens can be directly modeled in a unified rotation form of complex vector. To adapt to varied contexts, we further developed an adaptation function that adaptively adjust the semantic difference for effective integration with positional difference. Furthermore, we introduced a phase contrastive learning task, which takes in-batch phase pairs as the contrastive objective, for enhancing the effectiveness of positional encoding. Compared with prior methods (*e.g.,* RoPE), EulerFormer is more robust to semantic variations and have more superior theoretical properties (*e.g.,* long-term decay). As future work, we will further test the capacity of EulerFormer in more recommendation scenarios, *e.g.,* cross-domain and cold-start recommendation. Since EulerFormer can effectively serve as the powerful substitute of transformer backbone, we will also consider applying it to model other types of sequence data, such as text and time-series data.

REFERENCES

[1] Pu-Chin Chen, Henry Tsai, Srinadh Bhojanapalli, Hyung Won Chung, Yin-Wen Chang, and Chun-Sung Ferng. 2021. A simple and effective positional encoding for transformers. *arXiv preprint arXiv:2104.08698* (2021).

[2] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st international workshop on deep learning practice for high-dimensional sparse data*. 1–4.

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

[4] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Xinyan Fan, Jianxun Lian, Wayne Xin Zhao, Zheng Liu, Chaozhuo Li, and Xing Xie. 2022. Ada-Ranker: A Data Distribution Adaptive Ranking Paradigm for Sequential Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1599–1610.

[7] Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. 2021. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 1733–1737.

[8] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[9] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654* (2020).

[10] Ruining He, Wang-Cheng Kang, Julian J McAuley, et al. 2018. Translation-based Recommendation: A Scalable Method for Modeling Sequential Behavior.. In *IJCAI*. 5264–5268.

[11] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*. 1162–1171.

[12] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. 2022. Core: simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1796–1801.

[13] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 585–593.

[14] Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. 2020. Improve transformer models with better relative position embeddings. *arXiv preprint arXiv:2009.13658* (2020).

[15] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[16] Guolin Ke, Di He, and Tie-Yan Liu. 2020. Rethinking positional encoding in language pre-training. *arXiv preprint arXiv:2006.15595* (2020).

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864* (2020).

[19] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM Web Conference 2022*. 2320–2329.

[20] Langming Liu, Liu Cai, Chi Zhang, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Yifu Lv, Wenqi Fan, Yiqi Wang, Ming He, et al. 2023. Linrec: Linear attention mechanism for long-term sequential recommender systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 289–299.

[21] Ziru Liu, Jiejie Tian, Qingpeng Cai, Xiangyu Zhao, Jingtong Gao, Shuchang Liu, Dayou Chen, Tonghao He, Dong Zheng, Peng Jiang, et al. 2023. Multi-Task Recommendations with Reinforcement Learning. In *Proceedings of the ACM Web Conference 2023*. 1273–1282.

[22] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[23] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071* (2023).

[24] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[25] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

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

[27] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[28] Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A length-extrapolatable transformer. *arXiv preprint arXiv:2212.10554* (2022).

[29] Jiaxi Tang, Francois Belletti, Sagar Jain, Minmin Chen, Alex Beutel, Can Xu, and Ed H. Chi. 2019. Towards neural mixture recommender for long range dependent user sequences. In *The World Wide Web Conference*. 1782–1793.

[30] Zhen Tian, Ting Bai, Wayne Xin Zhao, Ji-Rong Wen, and Zhao Cao. 2023. EulerNet: Adaptive Feature Interaction Learning via Euler's Formula for CTR Prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1376–1385.

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

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

[33] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*. 403–412.

[34] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z Sheng, and Mehmet Orgun. 2019. Sequential recommender systems: challenges, progress and prospects. *arXiv preprint arXiv:2001.04830* (2019).

[35] Stanisław Węglarczyk. 2018. Kernel density estimation and its application. In *ITM web of conferences*, Vol. 23. EDP Sciences, 00037.

[36] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential recommendation via personalized transformer. In *Proceedings of the 14th ACM Conference on Recommender Systems*. 328–337.

[37] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE, 1259–1273.

[38] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* 32 (2019).

[39] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems* 33 (2020), 5812–5823.

[40] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. 2018. Next item recommendation with self-attention. *arXiv preprint arXiv:1808.06414* (2018).

[41] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 367–377.

[42] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation.. In *IJCAI*. 4320–4326.

[43] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. 2021. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4653–4664.

[44] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[45] Jianqiao Zheng, Sameera Ramasinghe, and Simon Lucey. 2021. Rethinking positional encoding. *arXiv preprint arXiv:2107.02561* (2021).