

# ComRoPE: Scalable and Robust Rotary Position Embedding Parameterized by Trainable Commuting Angle Matrices

Hao Yu<sup>1</sup> Tangyu Jiang<sup>1†</sup> Shuning Jia<sup>1,2</sup> Shannan Yan<sup>1</sup> Shunning Liu<sup>1</sup>  
Haolong Qian<sup>1</sup> Guanghao Li<sup>1</sup> Shuting Dong<sup>1</sup> Huaisong Zhang<sup>1</sup> Chun Yuan<sup>1†</sup>

<sup>1</sup>Tsinghua University <sup>2</sup>Shenzhen University

longinyh@gmail.com, jiangtangyu, yuanc@sz.tsinghua.edu.cn

## Abstract

The Transformer architecture has revolutionized various fields since it was proposed, where positional encoding plays an essential role in effectively capturing sequential order and context. Therefore, Rotary Positional Encoding (RoPE) was proposed to alleviate these issues, which integrates positional information by rotating the embeddings in the attention mechanism. However, RoPE utilizes manually defined rotation matrices, a design choice that favors computational efficiency but limits the model’s flexibility and adaptability. In this work, we propose ComRoPE, which generalizes RoPE by defining it in terms of trainable commuting angle matrices. Specifically, we demonstrate that pairwise commutativity of these matrices is essential for RoPE to achieve scalability and positional robustness. We formally define the RoPE Equation, which is an essential condition that ensures consistent performance with position offsets. Based on the theoretical analysis, we present two types of trainable commuting angle matrices as sufficient solutions to the RoPE equation, which significantly improve performance, surpassing the current state-of-the-art method by 1.6% at training resolution and 2.9% at higher resolution on the ImageNet-1K dataset. Furthermore, our framework shows versatility in generalizing to existing RoPE formulations and offering new insights for future positional encoding research. To ensure reproducibility, the source code and instructions are available at <https://github.com/Longin-Yu/ComRoPE>.

## 1. Introduction

The Transformer architecture [35] has been widely adopted across various fields, including Natural Language Processing (NLP) [2, 6, 10, 25, 26] and Computer Vision (CV) [7]. Moreover, an increasing number of Transformer-based applications [3, 15, 16, 18, 19, 38–41] continue to demonstrate

their effectiveness across a wide range of domains. At the core of the Transformer model lies the attention mechanism, which enables the model to selectively focus on different parts of the input based on relevance, thereby effectively capturing long-range dependencies and contextual relationships.

However, since the attention mechanism is insensitive to the fundamental position information, it cannot inherently capture the order of elements in the data. In order to alleviate this issue, positional embeddings are added to the input representation, providing the model with the necessary positional information to process the sequence order and structure effectively.

A variety of positional encoding methods have been proposed, which can be divided into two categories: Absolute Positional Encoding (APE) and Relative Positional Encoding (RPE). APE explicitly encodes the absolute position of each token by generating fixed positional embeddings, which are directly added to the input embeddings at the beginning of training [35]. However, APE struggles with handling long sequences and exhibits high sensitivity to positional shifts, limiting the scalability and robustness of models [14]. In contrast, RPE does not directly modify input embeddings; instead, it incorporates relative positional information within the attention mechanism, enabling the model to effectively capture positional relationships between tokens. Among all the existing works, Rotary Position Embedding (RoPE) [32] has gained significant attention due to the advantage of applying a rotational transformation to token embeddings. It encodes relative positions by treating each pair of features as coordinates and rotating them by an angle proportional to their position, enabling all tokens to interact within the attention mechanism regardless of distance.

However, existing RoPE methods face several key challenges: i) The essential components (i.e., the RoPE matrices) of previous RoPE approaches rely on 2D rotation

<sup>†</sup>Corresponding author. This work was done when Shuning Jia was an intern at Tsinghua University.

groups, which simplify computations but consequently restrict their feature projection capabilities, especially in high-dimensional spaces [32]. ii) Moreover, the majority of the rotation matrices of RoPE require to be manually designed, leading to insufficient capability and suboptimal performance [24, 29]. iii) Finally, previous attempts [23] to extend the rotation group often prioritize design simplicity, making it difficult to consistently satisfy relative position dependency—a critical property of RoPE that ensures positional robustness against offsets.

**Our objectives.** This work aims to develop a novel RoPE method for Transformers that is both scalable and robust. Specifically, we seek to extend the rotation group from the existing 2D representation to a larger subgroup of the special orthogonal group, allowing for higher degrees of freedom while preserving consistent behavior with respect to position offsets. Unlike existing methods that rely on manually designed non-trainable rotation matrices, which suffer from limited expressiveness and reduced robustness, our approach is designed to offer richer feature representation capabilities. This framework addresses the scalability limitations of current approaches and enhances their robustness against positional transformations.

**Our contributions.** This work introduces ComRoPE, a novel framework that significantly enhances positional encoding in Transformers. ComRoPE leverages trainable angle matrices, extending the RoPE mechanism with higher scalability and robustness. We identify the pairwise commutativity of these matrices as a necessary and sufficient condition for effective positional encoding, thereby unifying various existing RoPE formulations under a single theoretical framework. The contributions are summarized as follows:

- We formally define the RoPE function parameterized by angle matrices and prove that pairwise commutativity is a necessary and sufficient condition, offering a unified theory that encompasses several existing RoPE variants.
- We introduce ComRoPE, a scalable and robust solution that leverages two types of trainable commuting angle matrix sets as sufficient solutions to the RoPE equation, capturing richer positional representations without the need for manual design.
- Our extensive experiments show that ComRoPE surpasses the current state-of-the-art LieRE, achieving a performance increase of 1.6% at training resolution and 2.9% at higher resolutions on the ImageNet-1K classification task while delivering strong results across other benchmarks.
- We explore further applications of ComRoPE, providing valuable insights for advancing position encoding techniques in future Transformer-based models.

## 2. Related work

### 2.1. Position information in attention

Transformers [35] utilize the attention mechanism to capture similarities within sequences. However, they lack inherent sequential information and cannot capture the positional information of each token. To address this limitation, positional encoding [6, 28, 30] was introduced. As research progressed, positional encoding generally evolved into two types: APE [6, 33, 35] and RPE [13, 21, 30, 32]. [35] first proposed using APE in the form of sine and cosine functions, effectively representing the positional relationships within the input sequence. This positional encoding method achieved remarkable results in natural language processing, becoming a foundational component for many NLP tasks. However, the fixed nature of positional encoding limited the model’s ability to generalize to longer input sequences. In response, subsequent research introduced learnable absolute positional encoding. [6] proposed further enhancing the model’s expressive power by incorporating learnable position embeddings, particularly excelling in tasks such as sentence alignment and context representation. Although APE provides positional information to enhance the model’s understanding of sequences, it shows limitations in handling long sequences and cross-sequence scenarios. Hierarchical ViT such as Swin Transformer [20], introduced Relative Position Bias (RPB) [20, 29, 30] to handle large numbers of tokens with limited positional embeddings [12]. Related research has explored alternative encoding methods, such as RPE, as a replacement for APE to better capture complex dependency structures. iRPE [36] proposed an improved RPB by combining relative position embedding.

### 2.2. Rotary position embedding

Building on these developments, RoFormer [32] combined the advantages of APE and RPE, proposing RoPE. Currently, RoPE is widely used in Large Language Models (LLMs), such as LLaMA [34] and Vicuna [4]. This approach enhances model performance on tasks involving long-text semantics and multi-turn dialogues, improving extrapolation capability. To better adapt to the characteristics of two-dimensional data such as images, researchers extended RoPE to two-dimensional sequences (2D-RoPE). LieRE [23] extends RoPE to a more generalized form by introducing a rotation-based positional encoding method grounded in Lie group theory. Unified-IO 2 [22] applies 2D-RoPE within its multimodal architecture; EVA-02 [8], FiT [8] these pioneering works used 2D RoPE with axial frequencies (2D Axial RoPE), but had limitations in processing in the diagonal direction. Therefore, RoPE for ViT [12] proposes to use mixed axial frequency for 2D RoPE, named RoPE-Mixed.

### 3. Method

In this section, we begin by introducing key definitions and reformulating the RoPE paradigm within multi-axial attention mechanisms, which we collectively refer to as the RoPE Equations. Next, we present our main theorem, which establishes the necessary and sufficient conditions for RoPE functions parameterized by angle matrices. Finally, we provide solutions to the RoPE Equations based on the propositions that outline the sufficient conditions.

#### 3.1. Preliminaries

We use  $\mathbf{R}(\cdot) : \mathbb{R}^N \rightarrow \mathbb{R}^{d \times d}$  to denote a matrix-value function. We denote the vectors and matrices as the lower and uppercase bold font, respectively. We first recall two fundamental definitions of matrices as follows:

**Definition 1** (Matrix Exponential). *The exponential of a square matrix  $\mathbf{A}$ , denoted as  $e^{\mathbf{A}}$  or  $\exp(\mathbf{A})$ , is defined using the matrix exponential series such that:*

$$e^{\mathbf{A}} = \exp(\mathbf{A}) = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!} = \mathbf{I} + \mathbf{A} + \frac{\mathbf{A}^2}{2!} + \frac{\mathbf{A}^3}{3!} + \cdots,$$

*This series converges for any square matrix  $\mathbf{A}$ .*

**Definition 2** (Commuting Matrices). *Two square matrices  $\mathbf{A}_1$  and  $\mathbf{A}_2$  are said to **commute** if their product is independent of the order of multiplication, i.e.,*

$$\mathbf{A}_1 \mathbf{A}_2 = \mathbf{A}_2 \mathbf{A}_1.$$

*A set of square matrices  $\{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_N\}$  is said to **pairwise commute** if every pair of matrices within the set commutes with each other. That is, for all  $i, j$  such that  $1 \leq i, j \leq N$ ,*

$$\mathbf{A}_i \mathbf{A}_j = \mathbf{A}_j \mathbf{A}_i.$$

To clarify and better illustrate the main theorems presented in the following sections, we first reformulate and unify the definitions of the RoPE paradigm, specifically in the context of multi-axial attention mechanisms. RoPE was initially proposed by Su et al. [32] as a positional encoding method based on relative position dependencies. However, previous work provided only conceptual and descriptive descriptions of RPE and RoPE without offering a rigorous formal definition. In this work, we provide the formal definitions of both RPE and RoPE.

**Definition 3** (RPE Equation). *Let  $f : \mathbb{R}^d \times \mathbb{R}^N \rightarrow \mathbb{R}^d$  be a positional encoding function, and  $\rho : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$  be a similarity function.  $f$  is said to be a **RPE function** if and only if there exists a function  $g : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^N \rightarrow \mathbb{R}$  such that the following conditions hold for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$  and  $\mathbf{q}, \mathbf{k} \in \mathbb{R}^d$ :*

$$g(\mathbf{q}, \mathbf{k}, \mathbf{x} - \mathbf{y}) = \rho(f(\mathbf{q}, \mathbf{x}), f(\mathbf{k}, \mathbf{y})), \quad (1)$$

*We refer to Eq. (1) as the **RPE Equation**.*

**Definition 4** (RoPE Equation). *Let  $f : \mathbb{R}^d \times \mathbb{R}^N \rightarrow \mathbb{R}^d$  be an RPE function.  $f$  is said to be a **RoPE function** if and only if there exists a matrix-valued function  $\mathbf{R}_f(\cdot)$  such that the following conditions hold for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$  and  $\mathbf{q}, \mathbf{k} \in \mathbb{R}^d$  in RPE Equation:*

$$\begin{cases} f(\mathbf{q}, \mathbf{x}) = \mathbf{R}_f(\mathbf{x})\mathbf{q} \\ \rho(\mathbf{q}, \mathbf{k}) = \mathbf{q}^\top \mathbf{k} \\ g(\mathbf{q}, \mathbf{k}, \mathbf{x} - \mathbf{y}) = \mathbf{q}^\top \mathbf{R}_f(\mathbf{y} - \mathbf{x}) \mathbf{k} \end{cases} \quad (2)$$

*We refer to Eq. (2) as the **RoPE Equation**.*

By substituting the RoPE equation into the RPE equation, we obtain that the RoPE function satisfies the following property.

**Proposition 1.**  *$f$  is said to be a RoPE function if and only if the matrix-valued function  $\mathbf{R}_f(\cdot)$  satisfies:*

$$\mathbf{R}_f(\mathbf{x})^\top \mathbf{R}_f(\mathbf{y}) = \mathbf{R}_f(\mathbf{y} - \mathbf{x}),$$

*for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ .*

Note that the definition of RoPE Equation demonstrates that the position encoding should only be dependent on the relative position of the tokens, which thus be robust against the offset operations. We further expand the definition of the RoPE function to a parameterized one (i.e., Definition 6) via rotation matrices (i.e., Definition 5) as follows<sup>1</sup>.

**Definition 5** (Parameterized Rotation Matrix). *Let  $\mathbf{R}(\cdot; \mathcal{A}) : \mathbb{R}^N \rightarrow \mathbb{R}^{d \times d}$  be a matrix-valued function parameterized by  $N$  skew-symmetric matrices  $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_N\}$ . We say that  $\mathbf{R}(\cdot; \mathcal{A})$  is a **rotation matrix function** parameterized by angle matrices  $\mathcal{A}$  if it can be expressed as:*

$$\mathbf{R}(\mathbf{x}; \mathcal{A}) = \exp\left(\sum_{i=1}^N \mathbf{A}_i x_i\right), \quad (3)$$

*where  $\mathbf{x} = (x_1, x_2, \dots, x_N) \in \mathbb{R}^N$  and  $\exp(\cdot)$  denotes the matrix exponential.*

**Definition 6** (Parameterized RoPE Function). *Let  $f : \mathbb{R}^d \times \mathbb{R}^N \rightarrow \mathbb{R}^d$  be a RoPE function.  $f$  is said to be **parameterized by angle matrices** if and only if there exists a rotation matrix function  $\mathbf{R}_f(\cdot; \mathcal{A})$  parameterized by angle matrices  $\mathcal{A}$  such that the RoPE Equation (i.e., Eq. (2)) holds.*

*In this case,  $\mathbf{R}_f(\cdot; \mathcal{A})$  is referred to as the **rotation matrix function of RoPE function**  $f(\cdot; \mathcal{A})$  **parameterized by angle matrices**  $\mathcal{A}$ . For simplicity, we slightly abuse the notation and refer to  $\mathbf{R}$  as the rotation matrix of RoPE function  $f$ .*

<sup>1</sup>For clarification, the definitions of the rotation matrix and its exponential representation can be found in Appendix A.

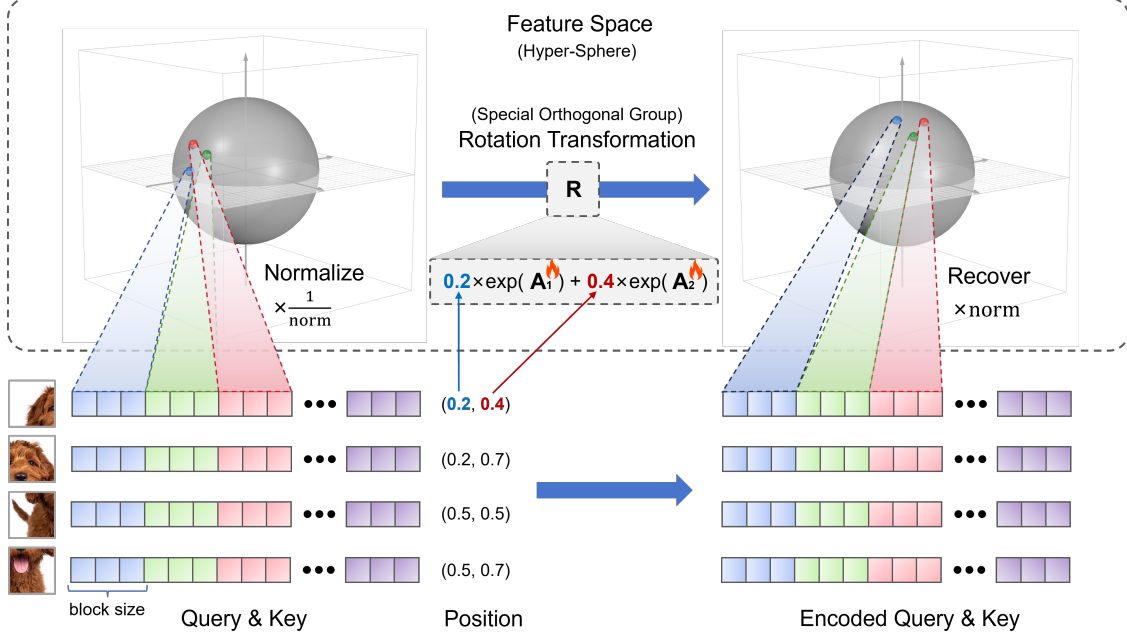


Figure 1. Overview of ComRoPE. Features are arranged into several blocks, each representing a distinct point in the feature space. The positions, along with the angle matrices, define the rotation matrix, which is an element of the special orthogonal group. The rotation transformation projects a feature point onto another point on the surface of the same hypersphere.

### 3.2. Main theorems

Based on the formal definitions above, we present our key theoretical results in the following theorem.

**Theorem 1.** Let  $\mathbf{R}(\cdot; \mathcal{A}) : \mathbb{R}^N \rightarrow \mathbb{R}^{d \times d}$  ( $N > 1$ ) be a rotation matrix function parameterized by angle matrices  $\mathcal{A}$ . The rotation difference  $\mathbf{R}(\mathbf{x})^\top \mathbf{R}(\mathbf{y})$  can be represented by the location difference  $\mathbf{y} - \mathbf{x}$  if and only if  $\mathcal{A}$  pairwise commute.

The proof of Theorem 1 is shown in Appendix A.

Theorem 1 together with Proposition 1 demonstrate that a function  $f$  is a RoPE function parameterized by angle matrices if and only if the angle matrices  $\mathcal{A}$  pairwise commute. In other words, to construct a relative positional encoding method for an attention mechanism that is robust to offset, it suffices to establish an angle matrix set that pairwise commutes. Thus, in the following section, we focus on the construction of  $\mathcal{A}$  to satisfy this particular requirement. We call our method ComRoPE to indicate the **commutativity** of angle matrices in RoPE function. The overview of ComRoPE is shown in Figure 1.

**Remark 1.** Among all the previous RoPE methods, LieRE [23] is the most related one to ours which solves Equation 3 by directly training the skew-symmetric matrix set  $\mathcal{A}$ . However, it is worth noting that in the following equation of their implementation

$$(\mathbf{R}(\mathbf{x}_q; \mathcal{A})\mathbf{q})^\top (\mathbf{R}(\mathbf{x}_k; \mathcal{A})\mathbf{k}) = \mathbf{q}^\top \mathbf{R}(\mathbf{x}_q; \mathcal{A})^\top \mathbf{R}(\mathbf{x}_k; \mathcal{A})\mathbf{k}, \quad (4)$$

sometimes  $\mathbf{R}(\mathbf{x}_q; \mathcal{A})^\top \mathbf{R}(\mathbf{x}_k; \mathcal{A}) \neq \mathbf{R}(\mathbf{x}_q - \mathbf{x}_k; \mathcal{A})$  because the probability that two random matrices commute is small, where  $\mathbf{x}_q$  and  $\mathbf{x}_k$  are two arbitrary coordinates. Thus,  $\mathbf{R}$  proposed by LieRE does not consistently satisfy the requirements of the RoPE Equation.

### 3.3. Construction of pairwise commuting matrices

In this section, we elaborate on concrete ways to construct the pairwise commuting matrices to solve the RoPE Equation parameterized by angle matrices. Note that if two matrices are both block diagonal with the same block sizes, where the corresponding blocks are commutative, then these two matrices are commutative. Formally speaking, for two matrices  $\mathbf{M}, \mathbf{N} \in \mathbb{R}^{d \times d}$ , they commute if

$$\begin{cases} \mathbf{M} = \text{diag}(\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_b) \\ \mathbf{N} = \text{diag}(\mathbf{N}_1, \mathbf{N}_2, \dots, \mathbf{N}_b) \\ \mathbf{M}_i, \mathbf{N}_i \in \mathbb{R}^{b \times b} \quad \forall i \in \{1, 2, \dots, m\} \\ \mathbf{M}_i \mathbf{N}_i = \mathbf{N}_i \mathbf{M}_i \quad \forall i \in \{1, 2, \dots, m\} \end{cases} \quad (5)$$

where  $b$  denotes block size that is a factor of  $d$  and  $m = \frac{d}{b}$ .

Thus, to present solutions to RoPE Equation parameterized by angle matrices, it suffices to partition the angle matrices  $\mathbf{A}_i$  in Eq. (3) into  $m$  blocks  $\mathbf{B}_{i1}, \mathbf{B}_{i2}, \dots, \mathbf{B}_{im}$  where:

$$\begin{cases} \mathbf{A}_i = \text{diag}(\mathbf{B}_{i1}, \mathbf{B}_{i2}, \dots, \mathbf{B}_{im}) \\ \mathbf{B}_{ij} \in \mathbb{R}^{b \times b} \quad \forall j \in \{1, 2, \dots, m\} \end{cases} \quad (6)$$



Defining  $\mathcal{B}_j = \{\mathbf{B}_{1j}, \mathbf{B}_{2j}, \dots, \mathbf{B}_{Nj}\}$ , if  $\mathcal{B}_j$  pairwise commutes for all  $j \in \{1, 2, \dots, m\}$ , then  $\mathcal{A}$  pairwise commutes. Note that  $\mathcal{B}_1, \mathcal{B}_2, \dots, \mathcal{B}_m$  are equivalent. Thus, without causing confusion, we will uniformly use  $\mathcal{B} = \{\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_N\}$  to represent  $\mathcal{B}_j = \{\mathbf{B}_{1j}, \mathbf{B}_{2j}, \dots, \mathbf{B}_{Nj}\}$  in the following.

Currently, there is no general method that provides the necessary and sufficient conditions for ensuring that arbitrary trainable skew-symmetric matrices commute. Therefore, we aim to establish sufficient conditions for enforcing this constraint. More concretely, we present Proposition 2 and Proposition 3 to construct two different variants of matrices, respectively.

**Proposition 2.** A set of matrices  $\mathcal{B} = \{\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_N\}$  pairwise commutes if all but one of them are zero matrices, i.e.,  $\exists k \in \{1, 2, \dots, N\}$  s.t.

$$\mathbf{B}_i = \mathbf{O} \ (\forall i \neq k) \quad (7)$$

Based on Proposition 2, we propose Trainable RoPE parameterized by **Axial-Partition Angle Matrices (ComRoPE-AP)**. We divide the  $m$  blocks into  $N$  parts, such that each part is responsible for the rotation of a specific axis. In this case,  $m$  should be a multiple of  $N$ . Specifically,

$$\mathbf{B}_{ij} = \begin{cases} \mathbf{P}_j - \mathbf{P}_j^\top, & \text{if } j \equiv i \pmod{N} \\ \mathbf{O}, & \text{otherwise} \end{cases} \quad (8)$$

where  $\{\mathbf{P}_j\}_{j=1}^m$  represents a set of trainable matrices,  $\mathbf{O}$  denotes a zero matrix, and  $\equiv$  indicates congruence modulo.

**Proposition 3.** A set of matrices  $\mathcal{B} = \{\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_N\}$  pairwise commutes if they pairwise linearly dependent, i.e.,  $\exists \lambda_1, \lambda_2, \dots, \lambda_N \in \mathbb{R}$  s.t.

$$\lambda_1 \mathbf{B}_1 = \lambda_2 \mathbf{B}_2 = \dots = \lambda_N \mathbf{B}_N \quad (9)$$

Based on Proposition 3, we propose Trainable RoPE parameterized by **Linearly-Dependent Angle Matrices (ComRoPE-LD)**. Specifically, We train a base matrix  $\mathbf{P}$  with scaling factors  $\{\theta_i\}_{i=1}^N$ . Then we obtain:

$$\mathcal{B} = \{\mathbf{B}_i = \theta_i(\mathbf{P} - \mathbf{P}^\top) \mid i = 1, 2, \dots, N\} \quad (10)$$

### 3.4. Implementation details of coordinates and improvements

#### 3.4.1 Relative scaling and center offset

**Relative scaling.** In language models, positions are typically treated as discrete. Additionally, due to the relationship between tokens and the inherent uncertainty in sequence length, there is no need to scale these positions into a specified range. However, in the case of images, positions are continuous. When using different patch sizes, it

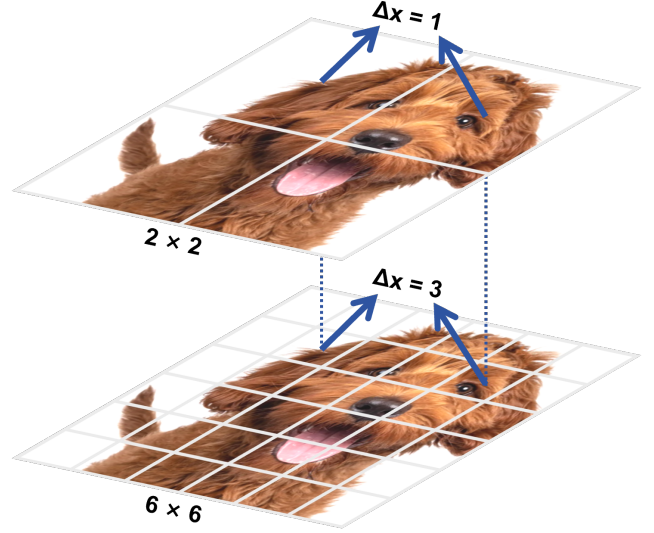


Figure 2. Different patch sizes result in different relative relationships.

becomes unreasonable to define their relative relationships solely based on the span of patches. As shown in Figure 2, the same locations with varying patch sizes can result in significantly different relative relationships.

As a result, applying relative coordinates is a better method for measuring relative relationships in images. For an image with shape  $H \times W$ , we scale both the height  $H$  and the width  $W$  to 1. Therefore, for a pixel located at  $(h, w)$  in the raw image, its coordinate is treated as  $(\frac{h}{H}, \frac{w}{W})$ . For high-dimensional coordinates, we perform the same operation, i.e., transform a raw coordinate  $(x_1, x_2, \dots, x_N)$  in multi-axial canvas with shape  $(X_1, X_2, \dots, X_N)$  into  $(\frac{x_1}{X_1}, \frac{x_2}{X_2}, \dots, \frac{x_N}{X_N})$ .

**Center offset.** When we project a patch to a feature tensor with a coordinate, we aggregate all information of the patch into a specified location. Simply, we adopt the center point of a patch as the aggregation location.

#### 3.4.2 Position perturbation

To achieve better robustness and excellent performance across different scales during inference, we add perturbations to the coordinates of the patches. Specifically, for a patch with center  $(x_1, \dots, x_N)$  and size  $(\Delta X_1, \dots, \Delta X_N)$ , during training, we formulate its location as:

$$\mathcal{N}\left((x_1, \dots, x_N)^\top; \text{diag}(\sigma \Delta X_1, \dots, \sigma \Delta X_N)^2\right) \quad (11)$$

where  $\sigma$  is a hyper-parameter called perturbation intensity. Additionally, we truncate the location within the patch area, i.e.,  $[x_k - \frac{\Delta X_k}{2}, x_k + \frac{\Delta X_k}{2}]$ .

Position Encoding Method	Perturbation Intensity	Evaluation Resolution								
		112	128	192	224	256	320	384	448	512
APE	1	30.04	38.69	56.4	58.76	60.02	59.27	57.04	54.10	50.99
	0	19.71	33.43	55.97	58.62	55.31	52.63	49.68	46.39	42.66
Vanilla RoPE	1	36.94	45.48	60.72	63.09	63.24	62.12	59.24	55.51	51.11
	0	36.41	44.48	59.97	62.03	62.54	61.36	58.56	54.79	50.58
LieRE	1	38.03	46.97	62.22	64.36	64.99	63.78	61.15	57.92	53.74
	0	38.22	46.85	62.01	64.07	64.54	63.46	60.74	56.89	52.51
ComRoPE-AP (ours)	1	35.75	46.18	62.82	<u>65.32</u>	<u>65.83</u>	<u>64.78</u>	<u>61.88</u>	<u>58.21</u>	<u>54.02</u>
	0	37.17	46.95	62.63	64.76	65.11	64.35	61.62	58.10	53.81
ComRoPE-LD (ours)	1	<b>38.30</b>	<b>47.28</b>	<b>63.53</b>	<b>65.49</b>	<b>65.95</b>	<b>65.27</b>	<b>62.62</b>	<b>59.11</b>	<b>55.29</b>
	0	36.88	46.54	<u>62.89</u>	65.27	65.66	64.27	61.83	57.87	53.64

Table 1. Accuracy of 2D classification on ImageNet. Models are trained at a resolution of  $224 \times 224$  and evaluated at varying resolutions.

## 4. Experiments

In this section, we evaluate the performance of various positional encoding methods on classic vision tasks. We first assess their scalability in 2D image classification across different resolutions. Additionally, we conduct object detection experiments to demonstrate the generalizability of our approach. To further examine the ability to handle higher-dimensional data, we perform 3D classification experiments, which are detailed in Appendix B.

### 4.1. 2D classification

#### 4.1.1 Setup

**Baselines and model architecture.** We evaluate our proposed methods (ComRoPE-LD and ComRoPE-AP) against APE, vanilla RoPE (as introduced by RoFormer), and LieRE. To isolate the effects of positional encoding, we utilize a standard Vision Transformer (ViT-B/16) architecture with minimal modifications. The APE codebook is removed for methods that do not employ APE, and self-attention layers are replaced with RoPE self-attention parameterized by angle matrices. This design highlights the performance differences among various positional encoding methods. A block size of 8 is used in practice. More details are provided in Appendix D.

**Training and evaluation protocol.** All models are trained at a standard resolution of  $224 \times 224$  and evaluated across multiple resolutions to test their robustness and scalability on the ImageNet-1K dataset [5]. The models are trained from scratch using randomly initialized parameters, ensuring no influence from pre-trained weights or external priors. To maintain fairness and reproducibility, we apply only basic data augmentation techniques, such as resizing and random cropping, focusing on relative performance comparisons rather than achieving absolute accuracy. The primary evaluation metric is accuracy on the test set across

various resolutions. Since APE is inherently fixed and discrete, bilinear interpolation is applied to adapt it to different resolutions during evaluation. Optimization methods and hyper-parameters are detailed in Appendix C.

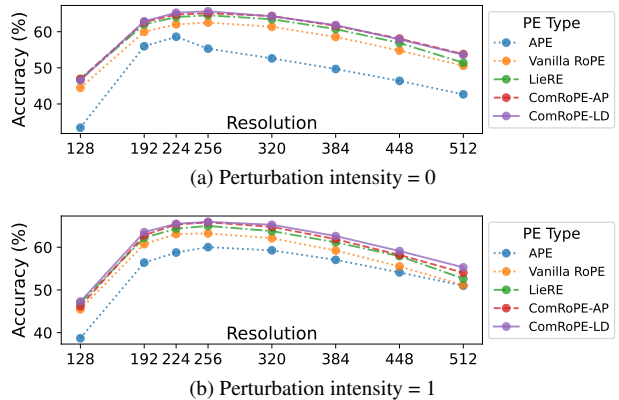


Figure 3. Accuracy on ImageNet-1K for various positional encoding methods. The results for the same perturbation intensity are presented together for better comparison. For better visualization, evaluation resolution at  $112 \times 112$  is not included in these figures.

#### 4.1.2 Main results

The accuracy metrics for each positional encoding method across various resolutions are summarized in Table 1. Additionally, Figure 3 visually compares performance under different levels of perturbation intensity separately.

**Overall performance.** Across all evaluations, APE consistently exhibits the lowest accuracy, corroborating previous findings regarding its limitations in dynamic contexts. The vanilla RoPE shows a modest improvement over APE but remains less effective. In contrast, the trainable angle matrices, namely LieRE, ComRoPE-AP, and ComRoPE-LD,

demonstrate significantly higher accuracy across all resolutions. Notably, ComRoPE-LD achieves the best performance among the three, suggesting that its inherent linear dependencies may enhance flexibility and structural learning capabilities.

**Accuracy at training resolution.** At the training resolution of  $224 \times 224$ , all three methods with trainable angle matrices (ComRoPE-LD, ComRoPE-AP, and LieRE) achieve comparable accuracy, significantly outperforming both APE and the standard RoPE, which underscores the effectiveness of RoPE parameterized by trainable angle matrices. Notably, ComRoPE-LD surpasses the current state-of-the-art LieRE by 1.6%<sup>2</sup>.

**Scaling to higher resolution.** At resolutions beyond the training size, LieRE shows the steepest decline in accuracy among the three trainable RoPE variants, indicating greater sensitivity to resolution changes. In contrast, ComRoPE-LD and ComRoPE-AP exhibit a more gradual decrease in performance, thanks to their commutative properties that enhance positional robustness. Specifically, ComRoPE-LD outperforms LieRE by 2.9% at a resolution of  $512 \times 512$ .

**Significance of commutativity.** These findings illustrate the effectiveness of trainable commutative angle matrices, particularly ComRoPE-LD, in maintaining accuracy and scalability across diverse resolutions. The results underscore the importance of commutativity in ensuring robust RoPE parameterized by trainable angle matrices for vision tasks. Furthermore, to better understand the role of commutativity, we conduct additional experiments by introducing coordinate offsets in LieRE (see Section 4.3.1 for details).

## 4.2. Object detection

To demonstrate the generalizability of our approach, we conduct object detection experiments using the framework from [37]. We adopt ViT-S as our backbone and apply ComRoPE to the attention layers. To ensure consistency with the pre-trained model, we initialize the angle matrix to zero.

We evaluate ComRoPE-LD, LieRE, and APE on the MS COCO dataset [17]. As summarized in Table 2, both ComRoPE-LD and LieRE outperform APE, with ComRoPE-LD achieving slightly better performance than LieRE while only using nearly half the number of extra parameters.

To compare training efficiency, we plot the results for each epoch in Figure 4. Our findings indicate that both ComRoPE-LD and LieRE converge faster than APE, requiring 29% fewer iterations to achieve the same results as APE.

<sup>2</sup>Relative improvement is calculated as  $\text{target} = \text{baseline} \times (1 + \text{improvement})$ , where the improvement denotes the percentage increase over the baseline performance. This formula applies throughout the following sections.

PE Method	AP	AP <sup>50</sup>	AP <sup>75</sup>	AP <sup>s</sup>	AP <sup>m</sup>	AP <sup>l</sup>
APE	44.0	66.6	47.7	28.2	46.8	58.4
LieRE	44.5	67.3	48.4	29.0	46.9	58.7
ComRoPE-LD	<b>44.7</b>	<b>67.6</b>	<b>48.5</b>	<b>29.2</b>	<b>47.1</b>	<b>60.0</b>

Table 2. Results of object detection on MS COCO.

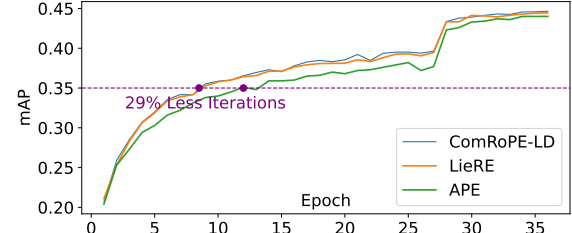


Figure 4. Results over the whole training procedure.

## 4.3. Ablation study

### 4.3.1 Impact of commutativity

To evaluate the significance of the commutativity of angle matrices, we conduct experiments on the LieRE by introducing a coordinate offset. Specifically, before inference, a random offset is applied uniformly across all coordinates within the image. The offset is sampled from a Gaussian distribution as follows:

$$\mathcal{N}(0; \rho^2 \cdot \mathbf{I}_{N \times N}) \quad (12)$$

where  $\rho$  represents the standard deviation of the random offset, and  $\mathbf{I}_{N \times N}$  is the identity matrix of size  $N \times N$ . It is important to note that applying the same offset to all coordinates does not influence the relative positional dependencies between the patches. The other experimental settings remain consistent with those in the 2D classification tasks.

The results shown in Figure 5 demonstrate that the baseline model’s performance deteriorates significantly as the standard deviation of the offset increases. In contrast, our proposed model, ComRoPE, maintains consistent performance across all offset values. This is due to the commutativity of the angle matrices, which allows the model to remain invariant to such coordinate shifts. The robustness of ComRoPE to this type of perturbation highlights its capacity to preserve relative positional information, even in the presence of modification introduced by coordinate offsets.

### 4.3.2 Impact of block size

In this section, we examine the impact of block size on 2D classification using the ImageNet dataset. We maintain the same experimental setup as in our primary experiments, varying the block size from 2 to 8. The results are presented in Figure 6. Our findings indicate that larger block sizes consistently improve performance by extending the rotation transformation space to a more significant subgroup of the

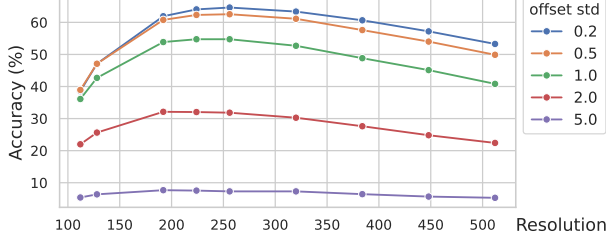


Figure 5. Effect of coordinate offset on LieRE. As the standard deviation of the offset increases, the performance of the baseline model deteriorates, while ComRoPE remains unaffected (un-painted).

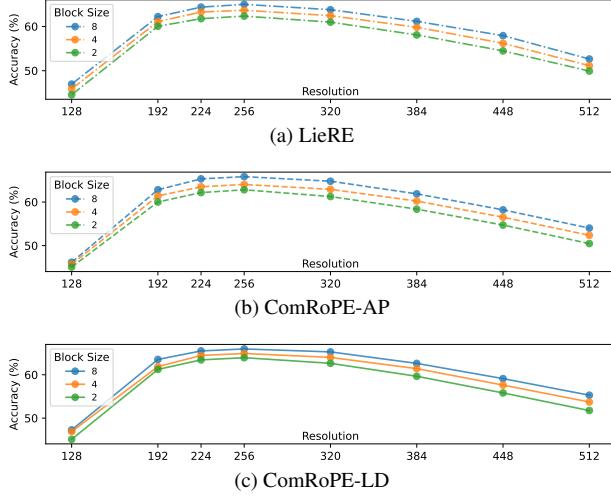


Figure 6. Accuracy on ImageNet for various block sizes. Larger block size results in better performance.

particular orthogonal group by introducing additional parameters and computation time. When the block size is small, the associated costs are minimal. However, as the block size increases, the primary term of time complexity grows to  $O(Lndb^2)$  from  $O(Lnd \cdot \frac{d}{h})$ , which becomes significant. Therefore, we limit the block size to a maximum of 8 to balance performance with additional costs. In other words, we select a block size of 8 to optimize performance while keeping the extra computational cost manageable.

### 4.3.3 Utility of position perturbation

In this section, we explore the impact of positional perturbations. We conducted experiments on ComRoPE-LD and APE using the ImageNet dataset, with the results presented in Figure 7.

As shown in Figure 7a, APE is highly sensitive to positional perturbations, leading to significant performance improvement (+19.5% when increasing intensity from 0 to 1) when these perturbations are introduced. For RoPE with angle matrices shown in Figure 7b, positional perturbations also resulted in some performance gains (+2.9%), though

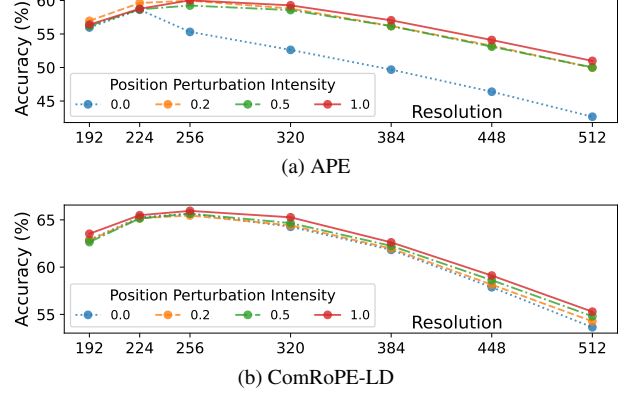


Figure 7. Accuracy on ImageNet over varying position perturbation intensity.

the improvement was relatively modest. This is likely due to the inherent robustness of the RoPE design with angle matrices, which is already well-equipped to handle variations in position.

## 4.4. Applications

In our approach, when the angle matrix is an all-zero matrix, the rotation matrix becomes the identity matrix, causing RoPE Attention to reduce to the standard Attention mechanism. When the block size of the angle matrix is set to 2, ComRoPE-AP effectively reduces to the commonly used RoPE Attention in language models. This demonstrates that our method can represent standard Attention and various common variants of RoPE Attention. Therefore, during the fine-tuning stage, we can replace standard Attention with our method, load the pre-trained weights, and fine-tune them under the new paradigm. In other words, ComRoPE can be seamlessly integrated into the fine-tuning process, even if it was not applied during pre-training. Additional experiments can be found in Appendix B.

## 5. Conclusion

In this work, we proposed ComRoPE, a novel and adaptive framework for Rotary Position Embedding (RoPE) parameterized by trainable angle matrices. We rigorously formulate the RoPE Equation and then establish a necessary and sufficient condition for its solution. Our approach effectively overcomes the scalability and robustness limitations of existing RoPE methods by eliminating the need for manually designed rotation matrices and introducing a more flexible, scalable solution. Extensive experimental results show that ComRoPE outperforms the existing positional encoding methods across various tasks. Furthermore, our framework generalizes existing RoPE formulations and demonstrates the potential for broader application in Transformer models, offering insights and a solid foundation for future research in positional encoding techniques.



## Acknowledgment

This work is supported by the National Key R&D Program of China (2022YFB4701400/4701402), SSTIC Grant (KJZD20230923115106012, KJZD20230923114916032, GJHZ20240218113604008) and Beijing Key Lab of Networked Multimedia.

## References

- [1] Philipp Bader, Sergio Blanes, and Fernando Casas. Computing the matrix exponential with an optimized taylor polynomial approximation. *Mathematics*, 7:1174, 2019. 4
- [2] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1
- [3] Yirui Chen, Xudong Huang, Quan Zhang, Wei Li, Mingjian Zhu, Qiangyu Yan, Simiao Li, Hanting Chen, Hailin Hu, Jie Yang, et al. Gim: A million-scale benchmark for generative image manipulation detection and localization. *arXiv preprint arXiv:2406.16531*, 2024. 1
- [4] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%\* chatgpt quality, march 2023. URL <https://lmsys.org/blog/2023-03-30-vicuna>, 3(5), 2023. 2
- [5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. 1, 2
- [7] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1
- [8] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 149:105171, 2024. 2
- [9] Jean H. Gallier. Basics of classical lie groups: The exponential map, lie groups, and lie algebras. 2001. 3
- [10] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024. 1
- [11] Larry C. Grove. Classical groups and geometric algebra. 2001. 2
- [12] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. *arXiv preprint arXiv:2403.13298*, 2024. 2
- [13] Max Horn, Kumar Shridhar, Elrich Groenewald, and Philipp FM Baumann. Translational equivariance in kernelizable attention. *arXiv preprint arXiv:2102.07680*, 2021. 2
- [14] 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: 24892–24928, 2023. 1
- [15] Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, et al. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5295–5306, 2024. 1
- [16] Hongyu Li, Jinyu Chen, Ziyu Wei, Shaofei Huang, Tianrui Hui, Jialin Gao, Xiaoming Wei, and Si Liu. Llava-st: A multimodal large language model for fine-grained spatial-temporal understanding. *arXiv preprint arXiv:2501.08282*, 2025. 1
- [17] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014. 7
- [18] Xiao Liu, Hanyu Lai, Hao Yu, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. Webglm: towards an efficient web-enhanced question answering system with human preferences. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pages 4549–4560, 2023. 1
- [19] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations*, 2024. 1
- [20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2
- [21] Antoine Liutkus, Ondřej Cifka, Shih-Lun Wu, Umut Simsekli, Yi-Hsuan Yang, and Gael Richard. Relative positional encoding for transformers with linear complexity. In *International Conference on Machine Learning*, pages 7067–7079. PMLR, 2021. 2
- [22] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26439–26455, 2024. 2
- [23] Sophie Ostmeier, Brian Axelrod, Michael E Moseley, Akshay Chaudhari, and Curtis Langlotz. Liere: Generalizing rotary position encodings. *arXiv preprint arXiv:2406.10322*, 2024. 2, 4
- [24] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021. 2
- [25] A Radford. Improving language understanding by generative pre-training. 2018. 1

- [26] 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. [1](#)
- [27] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. *International Conference on Machine Learning*, 2021. [3](#)
- [28] 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. [2](#)
- [29] 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. [2](#)
- [30] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018. [2](#)
- [31] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv: 1212.0402*, 2012. [3](#)
- [32] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. [1](#), [2](#), [3](#)
- [33] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015. [2](#)
- [34] 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. [2](#)
- [35] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. [1](#), [2](#)
- [36] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10033–10041, 2021. [2](#)
- [37] Chunlong Xia, Xinliang Wang, Feng Lv, Xin Hao, and Yifeng Shi. Vit-comer: Vision transformer with convolutional multi-scale feature interaction for dense predictions. *Computer Vision and Pattern Recognition*, 2024. [7](#)
- [38] Quan Zhang and Yuxin Qi. Can mllms guide weakly-supervised temporal action localization tasks? *arXiv preprint arXiv:2411.08466*, 2024. [1](#)
- [39] Quan Zhang, Xiaoyu Liu, Wei Li, Hanting Chen, Junchao Liu, Jie Hu, Zhiwei Xiong, Chun Yuan, and Yunhe Wang. Distilling semantic priors from sam to efficient image restoration models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25409–25419, 2024.
- [40] Quan Zhang, Yuxin Qi, Xi Tang, Jinwei Fang, Xi Lin, Ke Zhang, and Chun Yuan. IMDPrompter: Adapting SAM to image manipulation detection by cross-view automated prompt learning. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [41] Quan Zhang, Yuxin Qi, Xi Tang, Rui Yuan, Xi Lin, Ke Zhang, and Chun Yuan. Rethinking pseudo-label guided learning for weakly supervised temporal action localization from the perspective of noise correction. *arXiv preprint arXiv:2501.11124*, 2025. [1](#)

# ComRoPE: Scalable and Robust Rotary Position Embedding Parameterized by Trainable Commuting Angle Matrices

## Supplementary Material

### A. Theorems and proofs

#### A.1. Proof of the main theorem

To prove our main theorem (i.e., Theorem 1), we first propose some lemmas and prove them.

**Lemma 1.** *Matrices  $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$  commute if and only if  $e^{\mathbf{A}x}e^{\mathbf{B}y} = e^{\mathbf{A}x+\mathbf{B}y}$  for all  $x, y \in \mathbb{R}$ .*

*Proof.*

**1) Necessity ( $\Rightarrow$ ).** By the definition of  $e^{\mathbf{A}}$ , we have:

$$\begin{aligned} e^{\mathbf{A}+\mathbf{B}} &= \sum_{n=0}^{\infty} \frac{(\mathbf{A}+\mathbf{B})^n}{n!} \\ &= \sum_{n=0}^{\infty} \frac{\sum_{k=0}^n \binom{n}{k} \mathbf{A}^k \mathbf{B}^{n-k}}{n!} \\ &= \sum_{n=0}^{\infty} \sum_{k=0}^n \frac{\mathbf{A}^k \mathbf{B}^{n-k}}{k!(n-k)!} \\ &= \left( \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!} \right) \left( \sum_{m=0}^{\infty} \frac{\mathbf{B}^m}{m!} \right) \\ &= e^{\mathbf{A}}e^{\mathbf{B}}. \end{aligned} \quad (13)$$

Substituting  $\mathbf{A}, \mathbf{B}$  with  $\mathbf{A}x, \mathbf{B}y$ , we obtain:

$$e^{\mathbf{A}x+\mathbf{B}y} = e^{\mathbf{A}x}e^{\mathbf{B}y}. \quad (14)$$

**2) Sufficiency ( $\Leftarrow$ ).** We have:

$$\begin{aligned} e^{\mathbf{A}t}e^{\mathbf{B}t} &= \left( \sum_{n=0}^{\infty} \frac{t^n \mathbf{A}^n}{n!} \right) \left( \sum_{m=0}^{\infty} \frac{t^m \mathbf{B}^m}{m!} \right) \\ &= \mathbf{I} + t(\mathbf{A} + \mathbf{B}) + t^2 \cdot \frac{\mathbf{A}^2 + 2\mathbf{A}\mathbf{B} + \mathbf{B}^2}{4} + o(t^2), \end{aligned} \quad (15)$$

and

$$\begin{aligned} e^{(\mathbf{A}+\mathbf{B})t} &= \sum_{n=0}^{\infty} \frac{((\mathbf{A}+\mathbf{B})t)^n}{n!} \\ &= \mathbf{I} + t(\mathbf{A} + \mathbf{B}) + t^2 \cdot \frac{(\mathbf{A} + \mathbf{B})^2}{4} + o(t^2). \end{aligned} \quad (16)$$

Let  $t^2 f(t)$  be the difference between the two expressions

above. Thus, we obtain:

$$\begin{aligned} f(t) &= \frac{e^{\mathbf{A}t}e^{\mathbf{B}t} - e^{(\mathbf{A}+\mathbf{B})t}}{t^2} \\ &= \frac{\mathbf{A}^2 + 2\mathbf{A}\mathbf{B} + \mathbf{B}^2}{4} - \frac{(\mathbf{A} + \mathbf{B})^2}{4} + o(1) \\ &= \frac{\mathbf{A}\mathbf{B} - \mathbf{B}\mathbf{A}}{4} + o(1). \end{aligned} \quad (17)$$

Taking the limit as  $t \rightarrow 0$ , we have:

$$\lim_{t \rightarrow 0} f(t) = \frac{\mathbf{A}\mathbf{B} - \mathbf{B}\mathbf{A}}{4}. \quad (18)$$

Since  $e^{\mathbf{A}x}e^{\mathbf{B}y} = e^{\mathbf{A}x+\mathbf{B}y}$ , we have  $f(t) = 0$ , which implies  $\mathbf{A}\mathbf{B} = \mathbf{B}\mathbf{A}$ . ■

**Lemma 2.** *Matrices  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_m \in \mathbb{R}^{n \times n}$  ( $m > 1$ ) pairwise commute if and only if:*

$$e^{\mathbf{A}_1 x_1} e^{\mathbf{A}_2 x_2} \dots e^{\mathbf{A}_m x_m} = e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_m x_m} \quad (19)$$

for all  $x_1, x_2, \dots, x_m \in \mathbb{R}$ .

*Proof.* For  $m = 2$ , the theorem holds by Lemma 1. Suppose the theorem holds for all  $2 \leq m \leq k$ . We prove it for  $m = k + 1$ .

**1) Necessity ( $\Rightarrow$ ).** Assuming:

$$e^{\mathbf{A}_1 x_1} e^{\mathbf{A}_2 x_2} \dots e^{\mathbf{A}_k x_k} = e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_k x_k}, \quad (20)$$

we split  $\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_{k+1} x_{k+1}$  into two parts:

$$\begin{aligned} &\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_{k+1} x_{k+1} \\ &= (\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_k x_k) + (\mathbf{A}_{k+1} x_{k+1}). \end{aligned} \quad (21)$$

Since  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_{k+1}$  commute in pairs,  $\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_k x_k$  and  $\mathbf{A}_{k+1} x_{k+1}$  also commute. Thus:

$$\begin{aligned} &e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_{k+1} x_{k+1}} \\ &= e^{(\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_k x_k)} e^{\mathbf{A}_{k+1} x_{k+1}} \\ &= e^{\mathbf{A}_1 x_1} e^{\mathbf{A}_2 x_2} \dots e^{\mathbf{A}_{k+1} x_{k+1}}. \end{aligned} \quad (22)$$

**2) Sufficiency ( $\Leftarrow$ ).** Let  $x_{k+1} = 0$ . Then:

$$e^{\mathbf{A}_1 x_1} e^{\mathbf{A}_2 x_2} \dots e^{\mathbf{A}_k x_k} = e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_k x_k}, \quad (23)$$

implying that  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_k$  commute in pairs.

Position Encoding Method	Perturbation Intensity	Evaluation Resolution								
		112	128	192	224	256	320	384	448	512
APE	1	48.10	55.25	76.50	93.10	76.70	71.48	74.36	62.23	53.18
	0	45.54	55.18	76.48	92.87	75.91	70.70	73.70	60.43	48.79
Vanilla RoPE	1	47.28	54.96	75.69	93.79	77.66	72.53	74.72	65.19	57.34
	0	48.12	55.21	76.47	94.12	76.72	71.59	74.47	62.28	53.99
LieRE	1	48.97	56.15	77.33	94.43	78.35	73.20	77.33	65.74	58.23
	0	48.75	55.46	78.16	94.24	78.91	72.92	76.86	65.35	56.85
ComRoPE-AP	1	<b>50.14</b>	55.63	77.47	94.37	79.27	73.56	76.66	<b>67.68</b>	<u>59.34</u>
	0	48.06	55.63	75.72	94.26	75.75	70.93	74.72	64.91	57.98
ComRoPE-LD	1	<u>49.89</u>	<b>56.60</b>	<b>79.21</b>	94.24	<b>80.27</b>	<b>74.22</b>	<b>78.60</b>	<u>67.46</u>	<b>60.39</b>
	0	48.70	<u>56.46</u>	<u>78.30</u>	<b>94.48</b>	<u>79.27</u>	<u>74.58</u>	76.86	<u>66.02</u>	57.68

Table 3. Accuracy of 3D classification on UCF-101. Models are trained at a resolution of  $224 \times 224$  and evaluated at varying resolutions.

For any  $p \in \{1, 2, \dots, k\}$ , set all  $x_i = 0$  except for  $x_p$  and  $x_{k+1}$ . This yields:

$$e^{\mathbf{A}_p x_p} e^{\mathbf{A}_{k+1} x_{k+1}} = e^{\mathbf{A}_p x_p + \mathbf{A}_{k+1} x_{k+1}}, \quad (24)$$

which implies  $\mathbf{A}_p$  and  $\mathbf{A}_{k+1}$  commute. Thus,  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_{k+1}$  commute in pairs. ■

**Lemma 3.** *Matrices  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_m \in \mathbb{R}^{n \times n}$  ( $m > 1$ ) pairwise commute if and only if there exists a function  $f : \mathbb{R}^m \rightarrow \mathbb{R}^{n \times n}$  such that:*

$$f(x_1 + y_1, x_2 + y_2, \dots, x_m + y_m) = e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_m x_m} e^{\mathbf{A}_1 y_1 + \mathbf{A}_2 y_2 + \dots + \mathbf{A}_m y_m} \quad (25)$$

for all  $x_1, y_1, x_2, y_2, \dots, x_m, y_m \in \mathbb{R}$ .

*Proof.* **1) Necessity ( $\Rightarrow$ ).** By Lemma 2, we can easily verify that the following  $f$  satisfies the condition:

$$f(x_1 + y_1, x_2 + y_2, \dots, x_m + y_m) = e^{\mathbf{A}_1(x_1+y_1) + \mathbf{A}_2(x_2+y_2) + \dots + \mathbf{A}_m(x_m+y_m)}. \quad (26)$$

**2) Sufficiency ( $\Leftarrow$ ).** From Equation 31, let  $x_k$  be replaced with  $x_k + y_k$  and  $y_k$  with 0. We obtain:

$$\begin{aligned} f(x_1 + y_1, x_2 + y_2, \dots, x_m + y_m) &= e^{\mathbf{A}_1(x_1+y_1) + \dots + \mathbf{A}_m(x_m+y_m)} e^{\mathbf{A}_1 \cdot 0 + \dots + \mathbf{A}_m \cdot 0} \\ &= e^{\mathbf{A}_1(x_1+y_1) + \dots + \mathbf{A}_m(x_m+y_m)}. \end{aligned} \quad (27)$$

Comparing this with Equation 31, we get:

$$\begin{aligned} e^{\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \dots + \mathbf{A}_m x_m} e^{\mathbf{A}_1 y_1 + \mathbf{A}_2 y_2 + \dots + \mathbf{A}_m y_m} \\ = e^{\mathbf{A}_1(x_1+y_1) + \dots + \mathbf{A}_m(x_m+y_m)}. \end{aligned} \quad (28)$$

For any  $i, j \in \{1, 2, \dots, m\}$ , set  $x_k = 0$  for all  $k \neq i$  and  $y_k = 0$  for all  $k \neq j$ . This leads to:

$$e^{\mathbf{A}_i x_i} e^{\mathbf{A}_j y_j} = e^{\mathbf{A}_i x_i + \mathbf{A}_j y_j}. \quad (29)$$

By Lemma 1, this implies that  $\mathbf{A}_i$  and  $\mathbf{A}_j$  commute. Therefore, matrices  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_m \in \mathbb{R}^{n \times n}$  ( $m > 1$ ) pairwise commute. ■

*Proof of Theorem 1.* Recall that  $e^{\mathbf{A}}$  is an orthogonal matrix if  $\mathbf{A}$  is skew-symmetric, which implies  $\mathbf{R}(\mathbf{x}; \mathcal{A})^\top = \mathbf{R}(\mathbf{x}; \mathcal{A})^{-1} = \mathbf{R}(-\mathbf{x}; \mathcal{A})$ . Thus, we have:

$$\begin{aligned} \mathbf{R}(\mathbf{x}; \mathcal{A})^\top \mathbf{R}(\mathbf{y}; \mathcal{A}) \\ = e^{-\mathbf{A}_1 x_1 - \mathbf{A}_2 x_2 - \dots - \mathbf{A}_N x_N} e^{\mathbf{A}_1 y_1 + \mathbf{A}_2 y_2 + \dots + \mathbf{A}_N y_N}. \end{aligned} \quad (30)$$

By Lemma 3 and Equation 30,  $\mathcal{A}$  pairwise commute if and only if there exists a function  $f : \mathbb{R}^N \rightarrow \mathbb{R}^{d \times d}$  such that:

$$\begin{aligned} f(y_1 - x_1, y_2 - x_2, \dots, y_N - x_N) \\ = \mathbf{R}(\mathbf{x}; \mathcal{A})^\top \mathbf{R}(\mathbf{y}; \mathcal{A}). \end{aligned} \quad (31)$$

Therefore, the theorem holds. ■

## A.2. Explanation of rotation matrix and its exponential representation

Following the definition in [11], we first demonstrate the definition of rotation group and rotation matrix:

**Definition 7** (Rotation Group and Rotation Matrix). *A special orthogonal group in  $\mathbb{R}^n$ , denoted  $SO(n)$ , is the set of all  $n \times n$  orthogonal matrices with determinant 1, i.e.,*

$$SO(n) = \{\mathbf{R} \in \mathbb{R}^{n \times n} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{I}, \det(\mathbf{R}) = 1\}.$$



We use the terms **rotation group** and **special orthogonal group** interchangeably. Any matrix in the rotation group is called a **rotation matrix**.

To establish Definition 5, there is a necessary proposition to ensure the correctness of the exponential representation of a rotation matrix:

**Proposition 4.** Any rotation matrix  $\mathbf{R}$  can be represented by  $\exp(\mathbf{A})$  where  $\mathbf{A}$  is a skew-symmetric matrix.

Proposition 4 is a well-known result in Lie theory, as detailed in [9]. Specifically, the matrix  $\mathbf{R}$  in Proposition 4 belongs to the Lie group  $SO(n)$ . The associated Lie algebra of this group is  $\mathfrak{so}(n)$ , within which the skew-symmetric matrix  $\mathbf{A}$  resides.

## B. More experiments

### B.1. 3D classification

To assess the ability to handle higher dimensions beyond 2D, we conduct a 3D classification task on UCF-101 [31]. The details of the model and configuration can be found in Appendix C.

The results shown in Table 3 demonstrate similar results in 2D experiments, that ComRoPE performs best when resolution increases beyond the training resolution, displaying the resolution robustness of ComRoPE.

### B.2. Fine-tune on pre-trained model

Recall that we represent the RoPE function parameterized by angle matrices as defined in Equation 3. If all elements in  $\mathcal{A} = \{\mathbf{A}_i\}_{i=1}^N$  are initialized as zero matrices (i.e.,  $\forall i, \mathbf{A}_i = \mathbf{O}$ ), the behavior of this RoPE function degenerates into a standard attention mechanism. This is because, in this case,  $\mathbf{R}(\mathbf{x}; \mathcal{A}) = \exp(\mathbf{O}) = \mathbf{I}$  for any input  $\mathbf{x}$ .

On the other hand, if  $\mathcal{A} = \{\mathbf{A}_i\}_{i=1}^N$  is initialized as described in Appendix D, the RoPE function reduces to the vanilla RoPE formulation.

These observations demonstrate that our method can represent both the standard attention mechanism and various common RoPE attention variants. Therefore, during fine-tuning, standard attention or vanilla RoPE can be replaced with our method. Pre-trained weights can be loaded and fine-tuned under this new paradigm seamlessly, even if ComRoPE was not applied during the pre-training phase.

As an example, we fine-tune the Vision Transformer pre-trained in CLIP [27] on ImageNet by simply replacing the standard attention mechanism with each RoPE method. Specifically, we fine-tune the model for 4 epochs using a batch size of 3456 and a learning rate of  $3 \times 10^{-4}$ .

The results, presented in Table 4, show that ComRoPE-LD achieves the best performance. An interesting observation is that vanilla RoPE exhibits the lowest accuracy among

all five methods. This is likely because its fixed and manually defined parameters cannot be loaded seamlessly. In other words, it must adapt the pre-trained latent space during fine-tuning to effectively complete the task, which may result in suboptimal performance.

Method	Accuracy
APE	79.91
Vanilla RoPE	79.82
LieRE	80.12
ComRoPE-AP (ours)	80.11
ComRoPE-LD (ours)	<b>80.17</b>

Table 4. Accuracy of fine-tuned models with different positional encoding methods on ImageNet.

## C. Details of configuration

### C.1. Configuration of 2D classification

Configuration of 2D classification task is shown in Table 5.

Key	Value
Layers	12
Image Size	224
Patch Size	16
Hidden Dimension	768
Attention Heads	12
Batch Size	6144
Optimizer	AdamW
Weight Decay	0.01
Learning Rate	$10^{-3}$
LR Scheduler	cosine
Warmup Ratio	0.02
Epochs	200

Table 5. Model and training configuration of 2D classification experiments.

### C.2. Configuration of 3D classification

Because the vanilla RoPE and ComRoPE-AP require that the head dimension be a multiple coordinate dimension, standard ViT-Base is not applicable. We modified the model parameters to make it possible to conduct experiments on all of the five positional encoding methods. Besides, because the data size of UCF-101 is not too large, using a smaller model is more appropriate. All the details are shown in Table 6.

Key	Value
Layers	8
Image Size	224
Frame Count	8
Patch Size	16
Hidden Dimension	384
Attention Heads	8
Batch Size	768
Optimizer	AdamW
Weight Decay	0.01
Learning Rate	$1.2 \times 10^{-4}$
LR Scheduler	cosine
Warmup Ratio	0.02
Epochs	80

Table 6. Model and training configuration of 3D classification experiments.

## D. Reformulation of baseline RoPE methods in detail

### D.1. Vanilla RoPE

Firstly, note that we can represent a 2D rotation matrix in the exponential form:

$$\begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} = \exp\left(\begin{pmatrix} 0 & -\alpha \\ \alpha & 0 \end{pmatrix}\right) \quad (32)$$

The solution proposed by RoFormer, which we call vanilla RoPE here, can be regarded as a special type of ComRoPE-AP with block size 2 and non-trainable  $\mathbf{P}_j$  in Equation 8 where:

$$\begin{aligned} \mathbf{P}_j &= \begin{pmatrix} \cos(m\theta \frac{2N}{d} \cdot j) & -\sin(m\theta \frac{2N}{d} \cdot j) \\ \sin(m\theta \frac{2N}{d} \cdot j) & \cos(m\theta \frac{2N}{d} \cdot j) \end{pmatrix} \\ &= \exp(m\theta \frac{2N}{d} \cdot j \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}) \end{aligned} \quad (33)$$

In practice, RoFormer adopts  $\theta = 10000^{-1}$  as the hyperparameter of the rotation base.

### D.2. LieRE

For LieRE, the blocks are independent and trainable. Hence, we directly define  $B_{ij}$  in Equation 6 as:

$$\mathbf{B}_{ij} = \mathbf{P}_{ij} - \mathbf{P}_{ij}^\top, \quad (34)$$

where  $\mathbf{P}_{ij}$  is a trainable matrix.

## E. Analysis and comparison of complexity and extra consumption

Table 7 presents an overview of the properties of the positional encoding methods evaluated in this work. Specifi-

cally, the table highlights their commutativity (*i.e.*, the commutativity of angle matrices when represented in the RoPE form parameterized by angle matrices), the number of additional parameters, and the extra time complexity introduced by the positional encoding module.

### E.1. APE

For a Transformer that takes  $n$  embeddings with  $d$  features as inputs, the extra parameters of position encoding are the tensors in the position code book, *i.e.*,  $n \times d$ . The extra computation is to add position embeddings onto the original features. Therefore, the extra time complexity is  $O(n \times d)$ .

### E.2. RoPE parameterized by angle matrices

We unify RoPE with angle matrices whose rotation process is presented in Algorithm 1, where  $n, h, d, b, N$  represents sequence length, number of heads, dimension of hidden states, block size, and number of axes respectively. In this part, we focus on extra parameters and time complexity on each layer.

---

#### Algorithm 1 Rotation of query and key matrices

---

**In 1:** query matrix  $\mathbf{Q}$  with shape  $(n, h, \frac{d}{h})$   
**In 2:** key matrix  $\mathbf{K}$  with shape  $(n, h, \frac{d}{h})$   
**In 3:** angle base matrix  $\mathbf{A}$  with shape  $(N, h, \frac{d}{hb}, b, b)$   
**In 4:** patch positions  $\mathbf{P}$  with shape  $(n, N)$   
**Out:** rotated query and key matrices  $\hat{\mathbf{Q}}, \hat{\mathbf{K}}$

**for**  $axis = 1$  to  $N$  **do**  
     $\mathbf{M}_{axis} \leftarrow \mathbf{A}_{axis} \odot \mathbf{P}_{axis}$   
**end for**  
 $\mathbf{M} \leftarrow \sum \mathbf{M}_{axis}$  where  $\mathbf{M}$  has a shape of  $(n, h, \frac{d}{hb}, b, b)$   
 $\mathbf{R} \leftarrow \text{diag}(e^{\mathbf{M}}, \text{dim} = 2)$  with shape  $(n, h, \frac{d}{h}, \frac{d}{h})$   
 $\hat{\mathbf{Q}} \leftarrow \mathbf{R}\mathbf{Q}, \hat{\mathbf{K}} \leftarrow \mathbf{R}\mathbf{K}$   
**return**  $\hat{\mathbf{Q}}, \hat{\mathbf{K}}$

---

Angle base matrix  $\mathbf{A}$  is defined by the RoPE method, and the extra parameters are brought by the definition of  $\mathbf{A}$ . Time complexity of 1) calculating the element-wise product over each axis is  $O(n \times h \times \frac{d}{hb} \times b^2) = O(ndb)$ ; 2) calculating sum of  $\mathbf{M}$  is  $O(N \times n \times h \times \frac{d}{hb} \times b^2) = O(ndbN)$ ; 3) calculating matrix exponential is  $O(n \times h \times \frac{d}{hb} \times b^3) = O(ndb^2)$  based on [1]; 4) applying rotation is  $O(n \times h \times (\frac{d}{h})^2) = O(\frac{nd^2}{h})$ . Thus, the overall time complexity of rotation is  $O(ndbN + ndb^2 + \frac{nd^2}{h})$ .

#### E.2.1 Vanilla RoPE

No extra parameters are presented in vanilla RoPE, and the angle base matrix  $\mathbf{A}$  can be calculated during pre-processing. Besides, in vanilla RoPE, block size  $b = 2$ ,

Positional Encoding Method	Commutativity	Extra Parameters	Extra Time Complexity
APE	–	$nd$	$O(nd)$
Vanilla RoPE	Yes	0	$O(Lnd(bN + b^2 + \frac{d}{h})) \approx O(\frac{Lnd^2}{h})$
LieRE	Commonly Not	$LNdb$	$O(Lnd(bN + b^2 + \frac{d}{h}))$
ComRoPE-AP (ours)	Yes	$Ldb$	$O(Lnd(bN + b^2 + \frac{d}{h}))$
ComRoPE-LD (ours)	Yes	$Ld(b + \frac{N}{b})$	$O(Lnd(bN + b^2 + \frac{d}{h}))$

Table 7. Comparison of different types of positional encoding methods.  $n$  represents for count of patches (tokens),  $d$  represents for dimension of hidden states,  $L$  represents for count of layers,  $b$  represents for block size,  $N$  represents for count of axes, and  $h$  represents the count of attention heads.

so  $\frac{d}{h} \gg bN + b^2 = 2N + 4$  in most cases. Thus, count of extra parameters are 0 and extra time complexity is  $O(ndbN + ndb^2 + \frac{nd^2}{h}) \approx O(\frac{nd^2}{h})$  where  $b = 2$ .

### E.2.2 LieRE

For LieRE, the angle base matrix can be formulated as  $\mathbf{A} = \mathbf{P} - \mathbf{P}^\top$  where the parameters in  $\mathbf{P}$  are all independent. The only extra step to get  $\mathbf{A}$  from  $\mathbf{P}$  is the subtraction whose time complexity is  $O(Ndb)$ . Thus, count of extra parameters are  $N \times h \times \frac{d}{hb} \times b^2 = Ndb$  and extra time complexity is  $O(ndbN + ndb^2 + \frac{nd^2}{h} + ndb) = O(ndbN + ndb^2 + \frac{nd^2}{h})$ .

### E.2.3 ComRoPE-AP

For ComRoPE-AP, we compose the angle base matrix  $\mathbf{A}$  whose shape is  $(N, h, \frac{d}{hb}, b, b)$  with matrices with shape  $(N, h, \frac{d}{hbN}, b, b)$  by filling the blocks that are irrelevant to the corresponding coordinate axes with zeros. Thus, similarly, count of extra parameters are  $N \times h \times \frac{d}{hbN} \times b^2 = db$  and extra time complexity is  $O(ndbN + ndb^2 + \frac{nd^2}{h})$ .

### E.2.4 ComRoPE-LD

For ComRoPE-LD, the angle base matrices in  $\mathbf{A}$  are pairwise linearly dependent on the first dimension (i.e., axis dimension). Therefore, it can be presented by a matrix with shape  $(h, \frac{d}{hb}, b, b)$  and a multiplication factor with shape  $(N, h, \frac{d}{hb})$  by a multiplication step with time complexity  $O(N \times h \times \frac{d}{hb} \times b^2) = O(Ndb)$ . Thus, count of extra parameters are  $h \times \frac{d}{hb} \times b^2 + N \times h \times \frac{d}{hb} = d(b + \frac{N}{b})$ , and extra time complexity is  $O(ndbN + ndb^2 + \frac{nd^2}{h})$ .

## F. Distribution of elements in angle matrices

In this section, we analyze the element distribution in angle matrices obtained from the 2D classification experiments. Specifically, we extract all elements from the upper triangular parts of the matrices. The standard deviations of these elements are summarized in Table 8, and their density plot is presented in Figure 8.

Method	Block Size	Standard Deviations
LieRE	2	0.326
ComRoPE-AP		0.271
ComRoPE-LD		0.384
LieRE	4	0.246
ComRoPE-AP		0.208
ComRoPE-LD		0.278
LieRE	8	0.195
ComRoPE-AP		0.171
ComRoPE-LD		0.238

Table 8. The standard deviations of elements in angle matrices obtained from the 2D classification experiments.

To provide a clearer view of the long-tail distribution, we present the density plot using both linear and logarithmic scales in Figure 8. From the linear scale plot, it can be observed that elements near zero exhibit the highest variance in the angle matrices of LieRE, while ComRoPE-AP demonstrates the most moderate variance. On the other hand, the logarithmic scale reveals notable differences in range. For instance, ComRoPE-LD retains a broader distribution at values farther from zero. Consequently, as indicated in Table 8, ComRoPE-LD exhibits the largest overall variance among the angle matrix elements. This phenomenon is likely due to the linear dependencies between angle matrices across different coordinate axes, which necessitate significant frequency differences to distinguish them effectively.

## G. More Analysis

**Computational complexity and time consumption.** Time consumption is shown in Table 9. While small block sizes should have minimal impact, parallel optimization issues in `torch.matrix_exp` lower GPU utilization, increasing time costs for LieRE and ComRoPE.

**Application to LLMs.** ComRoPE can be incorporated into

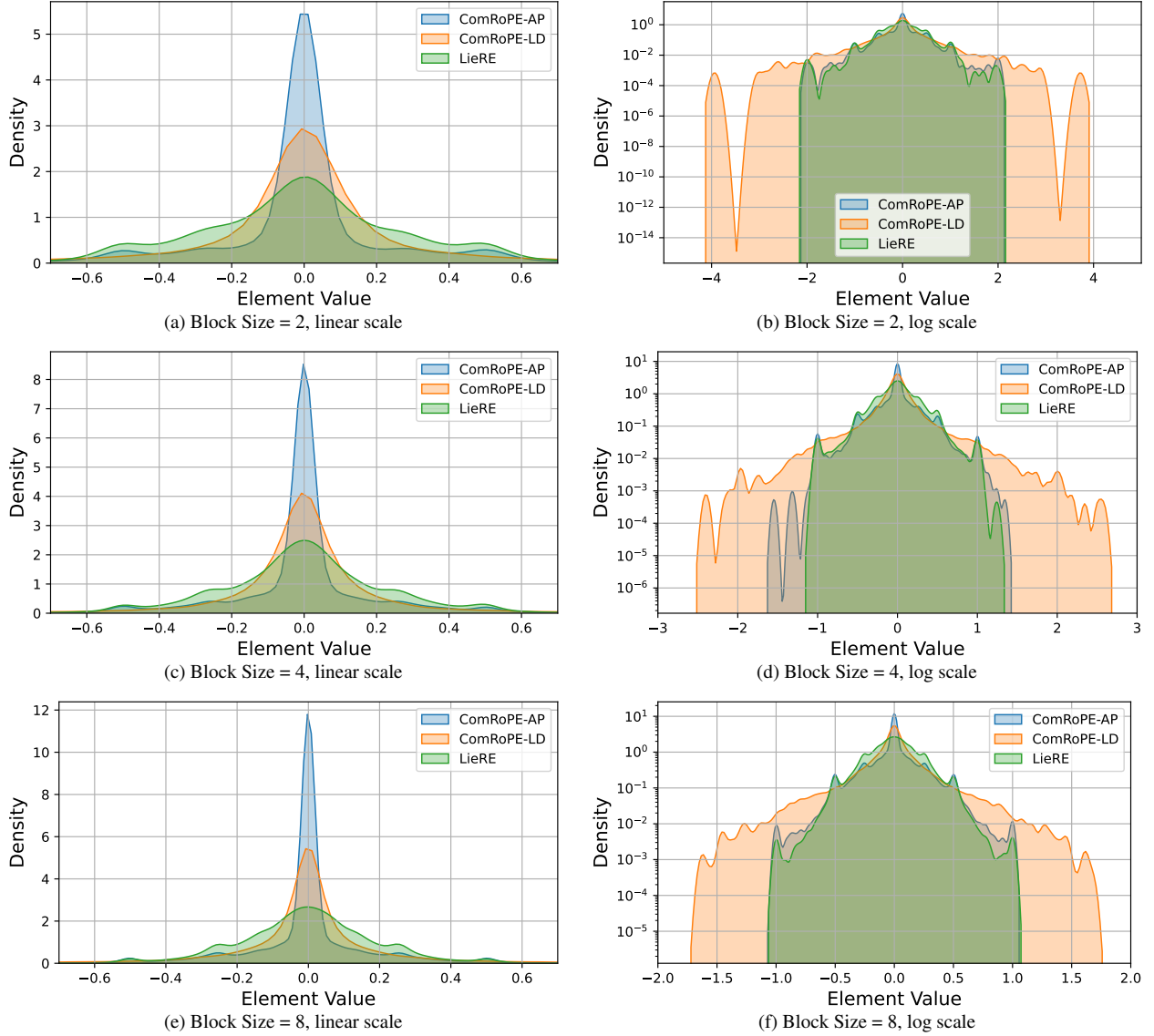


Figure 8. Density distribution of elements in the upper triangular sections of angle matrices from 2D classification experiments. Subfigures (a-b), (c-d), and (e-f) show the distributions for different block sizes: 2, 4, and 8, respectively.

Method	Vanilla RoPE	LieRE block = $8 \times 8$	ComRoPE-LD block = $4 \times 4$	ComRoPE-AP block = $4 \times 4$	ComRoPE-LD block = $8 \times 8$	ComRoPE-AP block = $8 \times 8$
Training Time per Epoch (min)	16	21	19	19	20	20
Inference Time on Valid Split (s)	32	36	35	34	35	35

Table 9. Computational costs compared on A800 $\times$ 4.

large language models as a *drop-in* substitute for the rotary position embeddings used in most pre-trained checkpoints, requiring no extra architectural changes during fine-tuning. Because language modeling operates along a single sequence dimension, the commuting property of our angle matrices holds automatically. At present, however, the `torch.matrix_exp` implementation incurs substantial memory overhead on large models, making end-to-end

training prohibitively expensive. Addressing this bottleneck, thereby unlocking full-scale LLM experiments, remains a key priority for future work.

**Implementation with sota codebase and settings.** Our work compares RoPE designs under consistent settings to highlight relative advantages, as demonstrated by our experiments. For a more thorough and convincing comparison, we conduct additional experiments in the RoPE-Mixed codebase with DeiT data augmentation (C.f. Table 10).



Position Encoding Method	Evaluation Resolution							
	128	192	224	256	320	384	448	512
RoPE-Mixed	68.99	79.75	81.42	82.31	82.75	82.11	80.61	78.39
ComRoPE-AP	68.48	<b>80.94</b>	<b>82.01</b>	82.59	82.43	81.65	80.58	<b>79.75</b>
ComRoPE-LD	<b>69.88</b>	79.91	81.78	<b>83.24</b>	<b>83.36</b>	<b>82.32</b>	<b>80.79</b>	<u>78.97</u>

Table 10. Results with DeiT recipe. RoPE-Mixed corresponds to ComRoPE-LD with  $2 \times 2$  blocks. ComRoPE-LD consistently outperforms RoPE-Mixed.

## H. Limitations

Despite the merits of our approach, it has two notable constraints that call for further investigation. The first one is computational overhead. Our implementation depends on `torch.matrix_exp`, which is slow and memory-intensive on large models. Cutting training time and GPU memory use is therefore an urgent engineering goal. And the other is strict commutativity restrictions. We currently require relatively strong conditions for the angle matrices to commute, which may restrict the expressiveness of the resulting embeddings. Identifying weaker—yet still sufficient—conditions could broaden the method’s capacity and applicability. Addressing these two issues will be the cornerstone of our future work, paving the way for more efficient training and richer modeling flexibility.