Arbitrary Dimensions of Rotation Matrix in Rotary Position Embedding

Allen Chen
Department of Computer Science
Columbia University
New York, USA
atc2160@columbia.edu

***Abstract*—Rotary Positional Encodings (RoPE) have emerged as a powerful mechanism for encoding relative positional information in transformer architectures. It notably uses 2x2 rotation matrices to achieve efficient and effective positional encodings. This paper extends the RoPE framework by generalizing the rotation mechanism to arbitrary block sizes, enabling the use of higher-dimensional rotation matrices while preserving the critical relative positional encoding property. To test the effects on performance, we leverage vision transformer (ViT) on CIFAR10 benchmark dataset. The experimental results reveal that the performance of the original 2x2 rotation matrix remains to be the best option with the highest accuracy and relatively fast training and inference time comparing to those of other rotation matrix of arbitrary sizes. These findings highlight the intrinsic efficiency and adequacy of the 2x2 rotation mechanism, suggesting that the added complexity of higher-dimensional rotations does not necessarily translate to empirical gains. Nonetheless, this work provides a flexible and robust framework for exploring and applying RoPE variants in ViT, offering valuable insights and a foundation for advancing positional encoding mechanisms in the computer vision domain. The code is available at** *https://github.com/AllenChenCU/rotary_embedding*.

***Index Terms*—Rotary Postion Embedding, RoPE, Rotation Matrix, Vision Transformer**

I. Introduction

Transformers have revolutionized deep learning by excelling in a wide range of applications, from natural language processing to computer vision. A critical component of their success is the ability to incorporate positional information, compensating for the permutation invariance of self-attention mechanisms. Various techniques have been developed for position encoding in the domain of computer vision, with Absolute Positional Embedding (APE) and Relative Position Bias (RPB) being the two main approaches.

Absolute positional embeddings explicitly associate each input token with a unique positional vector through sinusoidal or learnable representation, typically added to the input embeddings before applying the attention mechanism. While straightforward and effective, this approach often struggles to generalize well across variable sequence lengths or unseen positional ranges. Relative position bias, on the other hand, represents positional relationships as pairwise offsets between tokens, directly encoding these relationships into the attention scores. Even though this is an improvement over APE approach, RPB is added after query and key are multiplied, thus limiting the interactions with attention weights, which causes limited utilization of relative position.

Among the recent advancements in positional encoding, Rotary Positional Encodings (RoPE) have proven to be an efficient and effective method for incorporating relative positional information, as demonstrated in Rotary Position Embedding for Vision Transformer [1]. The standard RoPE mechanism employs 2x2 rotation matrices to subspaces of query and key vectors, essentially rotating multiple 2D subspaces simultaneously. This mechanism inherits the benefits of both the absolute position and relative position information by encoding fixed positions and relationships as transformations in rotational subspaces. However, its reliance on fixed 2x2 blocks can limit its ability to capture positional relationships in high-dimensional spaces.

This paper addresses this limitation by introducing an extension of RoPE that utilizes rotation matrices of arbitrary block sizes, enabling the representation of richer positional dependencies. By increasing the number of subspaces in one rotational group, it aims to reduce the total number of groups, allowing more elements to be rotated in the same subspace. The experiments use the original ViT without RoPE, ViT with 1D axial 2x2 RoPE, and ViT with 2D axial 2x2 RoPE as the baseline, and compare the effects of ViT with variants of 2D axial NxN RoPE in CIFAR10 task. The results show that the proposed extensions do not consistently outperform the original ViT with 2D axial 2x2 RoPE in accuracy. However, this work presents a foundation to continue the experiments on advancing the RoPE mechanism in the computer vision domain.

II. Methods

A. RoPE with 2x2 rotation matrix

The standard RoPE mechanism relies on the application of 2x2 rotation matrix to encode positional relationships between tokens in a two-dimensional subspace. It is defined as:

$$R(m,\theta) = \begin{pmatrix} \cos m\theta & -\sin m\theta \\ \sin m\theta & \cos m\theta \end{pmatrix},$$

where m is the position in the sequence and θ is the angle of rotation, which varies across different consecutive 2 dimensional subspaces within the embedding dimension.

In this paper, we focus on the Axial frequency implementation of the RoPE mechanism. We start with the 1D axial frequency method, discussed in the RoFormer paper [3]. This implementation indexes patches in the image as consecutive numbers. In other words, an image of size 32x32 with a patch size value of 1 would have the values of m ranging from 0 to 1023. We also have another implementation with 2D axial frequency, discussed in the RoPE in ViT paper [1]. Instead of assigning each patch a consecutive number, it uses both the column and row number, rotating the first half of the query or key embedding with the column index multiplied by the angle of rotation at the corresponding 2D subspace and the second half of the embedding with the row index multiplied by the corresponding angle of rotation. These two methods with 2x2 rotation matrix serve as the baseline for the subsequent experiments.

B. RoPE with 3x3 rotation matrix

Next we look at 3x3 rotation matrices. In three-dimensional space, a rotation matrix is a 3x3 orthogonal matrix with determinant 1, meaning the rotation happens around a predefined axis. In this study, we focus on rotations around the primary axis (x, y, and z), with the corresponding matrices defined as follows:

1. Rotation around the x-axis:

$$R_x(m,\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(m\theta) & -\sin(m\theta) \\ 0 & \sin(m\theta) & \cos(m\theta) \end{pmatrix}$$

2. Rotation around the y-axis:

$$R_y(m,\theta) = \begin{pmatrix} \cos(m\theta) & 0 & \sin(m\theta) \\ 0 & 1 & 0 \\ -\sin(m\theta) & 0 & \cos(m\theta) \end{pmatrix}$$

3. Rotation around the z-axis:

$$R_z(m,\theta) = \begin{pmatrix} \cos(m\theta) & -\sin(m\theta) & 0 \\ \sin(m\theta) & \cos(m\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Now we want to show that the above matrices preserve the relative positional encoding property - the resulting rotation matrix $R_1 R_2^T$ should depend only on the delta-vectors between the coordinate vectors associated with the tokens. We take the 3x3 rotation matrix around the z-axis as an example. Suppose we have:

$$R_1(m,\theta) = \begin{bmatrix} \cos(m\theta) & -\sin(m\theta) & 0 \\ \sin(m\theta) & \cos(m\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

$$R_2(n,\theta) = \begin{bmatrix} \cos(n\theta) & -\sin(n\theta) & 0 \\ \sin(n\theta) & \cos(n\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The transpose of R_2 is given by:

$$R_2^\top(n\theta) = \begin{bmatrix} \cos(n\theta) & \sin(n\theta) & 0 \\ -\sin(n\theta) & \cos(n\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Now, compute $R_1(m\theta)R_2^\top(n\theta)$:

$$R_1(m\theta)R_2^\top(n\theta) =$$

$$\begin{bmatrix} \cos(m\theta) & -\sin(m\theta) & 0 \\ \sin(m\theta) & \cos(m\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(n\theta) & \sin(n\theta) & 0 \\ -\sin(n\theta) & \cos(n\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Performing the matrix multiplication:

$$R_1 R_2^\top =$$

$$\begin{bmatrix} \cos(m\theta)\cos(n\theta)+\sin(m\theta)\sin(n\theta) & \cos(m\theta)\sin(n\theta)-\sin(m\theta)\cos(n\theta) & 0 \\ \sin(m\theta)\cos(n\theta)-\cos(m\theta)\sin(n\theta) & \sin(m\theta)\sin(n\theta)+\cos(m\theta)\cos(n\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Using the trigonometric identities:

$$\cos(m\theta)\cos(n\theta)+\sin(m\theta)\sin(n\theta) = \cos((m-n)\theta),$$

$$\cos(m\theta)\sin(n\theta)-\sin(m\theta)\cos(n\theta) = \sin((m-n)\theta),$$

we simplify:

$$R_1 R_2^\top = \begin{bmatrix} \cos((m-n)\theta) & \sin((m-n)\theta) & 0 \\ -\sin((m-n)\theta) & \cos((m-n)\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The result $R_1 R_2^T$ is a rotation matrix around the z-axis with angle $(m-n)\theta$. This means that the relative positional encoding depends only on the difference $m-n$, which corresponds to the delta-vectors between the two tokens.

To replace a 2x2 rotation matrix with a 3x3 one, the input size of the image would have to be resized so that it is divisible by 6 (2 times 3), which is derived from 2D Axial frequency and the 3x3 rotation matrix. This means the original input size of images is resized from 32x32 to 36x36. The slight increased input size would have an effect on the training and inference time, and this is taken into consideration when comparing to the baselines.

C. RoPE with 4x4 rotation matrix and above

In four-dimensional space, rotations occur within hyperplanes defined by two coordinates axes. There are six possible planes for rotation in 4D subspaces, corresponding to all unique pairs of coordinate axes. This results in a total of six independent rotation matrices. An example 4x4 rotation matrix in the x-y plane is defined as follows:

$$R_{xy}(m,\theta) = \begin{pmatrix} \cos(m\theta) & -\sin(m\theta) & 0 & 0 \\ \sin(m\theta) & \cos(m\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Similarly in higher dimensional spaces, rotations are defined within hyperplanes spanned by two basis vectors. For an N-dimensional space, the number of possible rotations corresponds to the number of unique axis pairs, given by N choose 2. This implies there are 10 rotation planes in 5D space and 15 in 6D space. The 5x5 rotation matrix in the x-y plane and the 6x6 rotation matrix in the x-y plane are arbitrarily selected to be used in this study. Similar to the 3x3 case, the input sizes of the image are resized from 32x32 to 40x40 and 36x36 respectively to ensure the key and query embedding can be evenly divided into groups.

TABLE I
MAIN EXPERIMENTS RESULTS

Model	*dim*	*Acc@1*	*Loss*	*InferenceTime*	*TrainingTime*
vit_no_rope_cifar10	2x2	0.570	1.293	0.698	4.267
vit_rope_1D_axial_2x2_cifar10	2x2	0.642	1.178	26.514	33.719
vit_rope_2D_axial_2x2_cifar10	2x2	0.704	1.045	27.591	36.243
vit_rope_2D_axial_3x3_2_cifar10	3x3	0.691	1.020	30.304	40.397
vit_rope_2D_axial_4x4_0_cifar10	4x4	0.679	1.077	27.195	36.091
vit_rope_2D_axial_5x5_0_cifar10	5x5	0.649	1.155	32.151	42.616
vit_rope_2D_axial_6x6_0_cifar10	6x6	0.663	1.174	30.788	40.572
vit_weighted_rope_2D_axial_3x3_cifar10	3x3	0.610	1.186	43.837	58.147
vit_weighted_rope_2D_axial_4x4_cifar10	4x4	0.609	1.201	51.666	124.240

TABLE II
3X3 ROTATION MATRICES RESULTS

Model	*Dim*	*Acc@1*	*Loss*	*InferenceTime*	*TrainingTime*
vit_no_rope_cifar10	2x2	0.570	1.293	0.698	4.267
vit_rope_1D_axial_2x2_cifar10	2x2	0.642	1.178	26.514	33.719
vit_rope_2D_axial_2x2_cifar10	2x2	0.704	1.045	27.591	36.243
vit_rope_2D_axial_3x3_0_cifar10	3x3	0.690	1.070	31.168	41.111
vit_rope_2D_axial_3x3_1_cifar10	3x3	0.677	1.124	30.140	40.132
vit_rope_2D_axial_3x3_2_cifar10	3x3	0.691	1.020	30.304	40.397

D. RoPE with weighted rotation matrix

In the three-dimensional subspaces, we experiment with three different rotation matrices around its corresponding axis. One additional experiment was to use all 3 rotation matrices to output 3 query and key embeddings, and equally weight them to produce a final weighted embedding. In the 4x4 case, we rotate the query and key embeddings with the 6 different rotation matrices with the corresponding hyperplanes, and compute the final weighted embeddings by taking the average of those 6 embeddings. The preliminary result for this approach has not shown improvement in performance, so no further experiments are conducted.

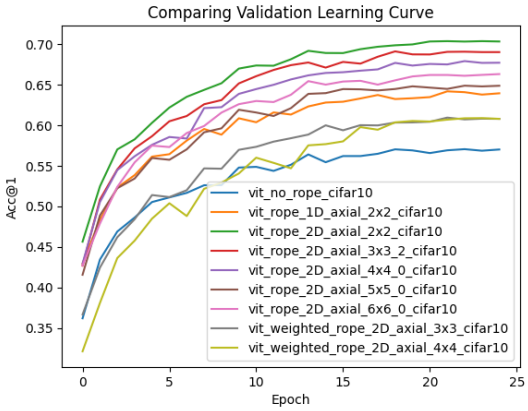

Fig. 1. Comparing validation learning curves from highlighted experiments (See Table I)

III. RESULTS

In this paper, we apply 2D RoPE mechanism with different rotation matrices to ViT on CIFAR10 benchmark dataset. For each model, we train for 25 epochs from scratch with 4 Nvidia T4 GPUs and report the best accuracy and loss from all the epochs and take the average training and inference time. Looking at Table I, we first obtain the baseline performance for models with and without the standard 1D and 2D RoPE with 2x2 rotation matrix. Next we configure and replace the standard 2x2 matrix with higher dimensional rotation matrices from 3x3 to 6x6, and record their model performance. Lastly, we show the performance for the two weighted 2D RoPE with 3x3 and 4x4 experiments. Comparing the model performance, the standard 2D RoPE mechanism with 2x2 rotation matrix achieves the highest accuracy and lowest loss across all models. Even though we see the model with a 4x4 rotation matrix has the lowest average inference and training time, it was not so much a gap from the baseline models. Looking further into Figure 1, we have no further doubt that the standard 2D RoPE machanism remain the best option as its validation learning curve leads in accuracy from all models throughout the epochs. We also see the performance and the number of dimension in rotation matrix has a negative relationship - as the number of dimension increases, the accuracy decreases. We can safely conclude that models with higher dimensional rotation matrix do not necessarily improve model performance.

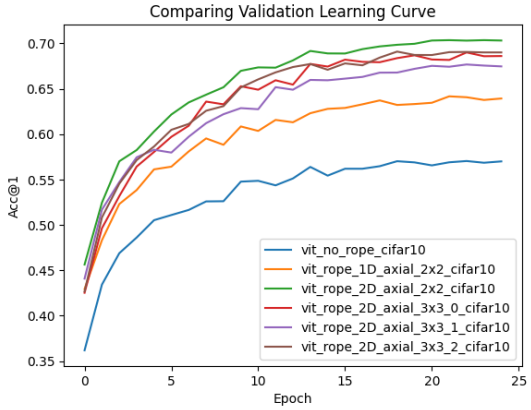

Fig. 2. Comparing validation learning curves from experiments with 3x3 rotation matrices (See Table II)

We also report the results for all 3x3 rotation matrices around the 3 primary axis (x, y, z-axis). From table II and Figure 2, it is observed that the results for different 3x3 rotation matrices are close

to each other. One interesting observation is that the model with 3x3 rotation matrix around the y-axis has the worst performance. Due to this noticeable discrepancy, the axis at which the higher dimensional rotation matrix rotates around can be the subject of future studies.

IV. Conclusion

In conclusion, this study set out to extend the Rotary Positional Encoding (RoPE) mechanism in Vision Transformer by exploring the application of rotation matrices with arbitrary block sizes, while preserving the critical relative positional encoding property. Despite the potential for richer positional representations through higher-dimensional rotations, our results demonstrate that the standard 2D RoPE with 2x2 rotation matrix remains the most effective, consistently outperforming its higher-dimensional counterparts. Furthermore, the analysis revealed a negative correlation between the dimensionality of the rotation matrix and model performance, with accuracy declining as the number of dimensions increased. This suggests that the added complexity introduced by higher-dimensional rotations does not translate to improved model outcomes. These findings reaffirm the robustness and efficiency of the standard 2x2 rotation matrix in RoPE while providing valuable insights and a solid foundation for future research to explore novel approaches and refinements in leveraging rotation matrices for positional encoding mechanisms.

V. Acknowledgements

We are thankful to Professor Choromanski for the invaluable guidance, insights, and ideas throughout the course of this study.

References

[1] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. *ECCV*, 2024.

[2] Github repository vit-pytorch. https://github.com/lucidrains/vit-pytorch/blob/main/vit_pytorch/vit.py. Accessed: 2024-11-26.

[3] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2023.