How to encode location in the Vision Transformer? A study on position embeddings

Xiangxie Zhang

Delft University of Technology

TU Delft

How to encode location in the Vision Transformer? A study on position embeddings

by

Xiangxie Zhang

to obtain the degree of Master of Science in Computer Science
at the Delft University of Technology,
Faculty of Electrical Engineering, Mathematics and Computer Science

Student Number: 5343208
Project Duration: November, 2021 - June, 2022
Thesis committee:

Dr. Jan van Gemert,	TU Delft, Supervisor, Committee chair
Robert-Jan Bruintjes,	TU Delft, Daily supervisor
Dr. Jie Yang,	TU Delft, External committee member

TU Delft

Preface

This thesis report presents my Master's Project work about investigating the position embeddings of ViT models in the Computer Vision Lab at the Delft University of Technology. My research was conducted under the supervision of Dr. Jan van Gemert and Robert-Jan Bruintjes. This report consists of two parts. The first part is the scientific paper of the main work, while the second part provides some background knowledge related to the main research.

During the research, I learned a lot from this amazing journey. The most important thing I gained from this experience is how to conduct proper academic research. At the beginning of this project, I had a struggle time when I did not have a clear idea of what exactly I would like to research. After reading many previous papers related to the Vision Transformer models, I finally found this interesting topic about location information in ViT models. In the middle of the research, I learned many lessons about how to arrange my time and conduct experiments efficiently. From an academic point of view, of course, my knowledge of computer vision, especially the ViT models, has been expanded throughout the project. I hope this unforgettable experience could lay a solid foundation for my future career.

Now the Master's project has come to an end. I would like to express my great appreciation to my supervisors, Jan van Gemert and Robert-Jan Bruintjes, for their guidance during the research. In the first year of my Master's study, I took the course deep learning and seminar computer vision by deep learning, which was taught by Jan. He is always kind to students and patient in answering their questions. The nice experience in these two courses made me finally decide to do my Master's thesis in the Computer Vision lab. As my daily supervisor, Robert-Jan always gives me appropriate instructions on conducting academic research. I learned a lot from the weekly meetings with him. Additionally, I would like to express my sincere gratitude to Jie Yang, my external committee member. I took the information retrieval course last year taught by Jie Yang, and that experience was amazing. I also had the pleasure of contributing to a project under the supervision of Jie Yang and learning from him. Finally, I am deeply grateful to my family members and friends for their support in my life.

Xiangxie Zhang
Delft, June 2022

Contents

1

Scientific paper

How to encode location in the Vision Transformer? A study on position embeddings

Xiangxie Zhang
Delft University of Technology
X.Zhang-60@student.tudelft.nl

Robert-Jan Bruintjes (supervisor)
Delft University of Technology
R.Bruintjes@tudelft.nl

Jan van Gemert (supervisor)
Delft University of Technology
j.c.vanGemert@tudelft.nl

Abstract

Location information is essential for the ViT model. Image data has three types of location information: absolute location, relative direction, and relative distance. Various position embeddings methods have been used to introduce location information to the ViT model. Some existing methods are absolute position embeddings, relative position embeddings, fixed sinusoidal position embeddings, and learnable Fourier position embeddings. However, it is unclear what type of location information can be encoded by different position embeddings methods. This paper investigates this question by conducting fully-controlled experiments and feature-level analysis on synthetic datasets. The results suggest that the relative position embeddings cannot encode absolute location information, which leads to inferior performance. All the position embeddings approaches that we test can encode relative location information. However, they have different levels of relative location bias. The learnable absolute position embeddings do not contain any relative location bias and therefore need more data to learn. The fixed sinusoidal and learnable Fourier position embeddings are relatively better, but they also have minor drawbacks. The fixed sinusoidal position embeddings are not trainable, while the Fourier method does not have much bias on relative location information. We propose to make the fixed sinusoidal position embeddings learnable and use pretraining tasks to improve the Fourier method. Our two new approaches show promising results on the testing datasets, and they are competitive compared with a similar approach.

1. Introduction

Vision Transformer (ViT) [5] is a newly emerging model competing with Convolutional Neural Network (CNN) in computer vision. ViT uses a pure Transformer model [28] without the help of convolutions and achieves state-of-the-art performance on image classification tasks. The innovation of ViT is splitting the input image into grid-like non-overlapping patches. The global features across different image patches can be captured through the self-attention layers. However, the self-attention mechanism itself is permutation-invariant to the input sequence [14]. Changing the order of the image patches does not make any difference to the outputs of the self-attention layer. Consider the self-driving scenario shown in Figure 1, where the computer vision model is applied to determine the relative location between the red and blue vehicles. A pure self-

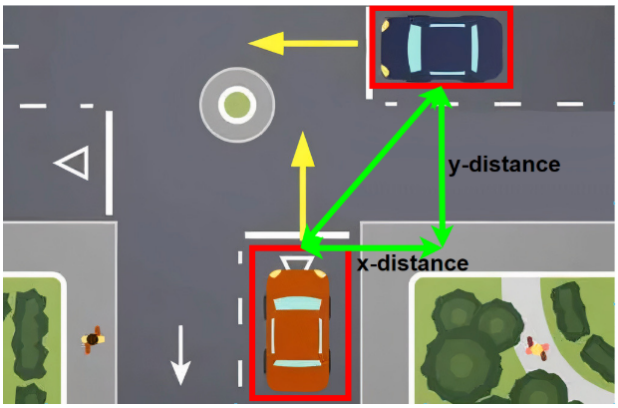


Figure 1. An overview of the three types of location information in images. This figure displays a scenario of self-driving techniques. The driving directions of the red and blue vehicles are indicated by the yellow arrows, which show that they will meet at the intersection. To avoid a car crash with the blue car, the computer vision model of the self-driving system in the red car should know the relative direction between them, that the blue car is at the front-right of the red car. In addition, the model should be aware of the relative distance between the two vehicles, which can be decomposed into horizontal and vertical distances, as shown by the green arrows. The model should also be sensitive to the absolute location to avoid problems like getting into the wrong lane. In this paper, we are interested in investigating whether the position embeddings approach in the ViT model could encode the three types of location information. This image is adapted from https://www.photophoto.cn.

attention layer is insensitive to the displacements of the two vehicles, which is not a desired property. Therefore, location information should be introduced to the ViT models to make them be sensitive to the order of the input image patches [28].

Previous work identifies three types of location information in image data: absolute location, relative direction, and relative distance [34], as shown in Figure 1. The absolute location information specifies where the target object is in an image. The relative direction and relative distance together describe the relative location between different objects. Therefore, both the absolute and relative location are important information in computer vision tasks [8].

Position embeddings are used to feed location information into the ViT models. The original ViT uses the learnable absolute position embeddings [4] as the default setting, while there are other methods like fixed sinusoidal position embeddings [28, 31], relative position embeddings [21, 23], and the recently proposed learnable Fourier position embeddings [13]. These methods have different mechanisms to encode locations. The relative position embeddings are applied in the self-attention layers, and they are designed to encode relative location information. Other methods give each image patch a unique absolute position embedding, and they are introduced to the ViT model before the self-attention layers. The levels of predefined location information injected into these models are also different. The absolute and relative position embeddings are initialized with random numbers, while the sinusoidal and Fourier methods introduce additional location information by using predefined features. Another difference lies in the ability to learn. The fixed sinusoidal position embeddings are not learnable, while other approaches can be trained to adapt to the datasets. Given all the differences, it is still unclear what type of location information can be encoded by these methods. In addition, it is not conclusive yet what the predefined features bring to the model and how the inability to learn could influence the model.

In this paper, we investigate what location information can be encoded by different position embeddings and explore their advantages and drawbacks by conducting fully-controlled experiments and feature-level analysis on synthetic datasets. Based on the results of these experiments, we propose solutions to improve the fixed sinusoidal and learnable Fourier position embeddings as they have the most desired properties compared with other approaches. We design pretraining tasks on synthetic datasets to help the learnable Fourier method acquire the desired relative location bias. In addition, using a similar structure as the Fourier approach, we construct learnable sinusoidal position embeddings. Our methods show promising results on realistic datasets, and they are competitive compared with a similar state-of-the-art approach.

2. Related work

The fixed sinusoidal position embeddings [28, 31], absolute position embeddings [4], and relative position embeddings [21, 23] are seminal position embeddings methods. Researchers have been trying to find better encoding schemes for location information. There are works that focus on modifying the structure of the relative position embeddings to better encode relative location information [3,7,20,34]. Some other works apply complex numbers and rotation matrices to enhance the encoding ability of the absolute [29] and relative position embeddings [25]. Introducing convolutions to the ViT models to enhance the spatial inductive bias is also a frequently proposed method. For example, Convolution Vision Transformer (CvT) [33] and Convolution enhanced image Transformer [35] use convolution blocks to capture low-level features and spatial information. Conditional Position encoding Vision Transformer (CPVT) [1] replaces the traditional position embeddings with convolution layers. Besides directly incorporating convolutions to ViT models, researchers also try to enhance ViT's locality inductive bias by injecting Shifted Patch Tokenization (SPT) and Locality Self-Attention (LSA) modules [12]. These works concentrate on modifying the structure of the position embeddings or the ViT model to improve the ability to encode location. Unlike them, we instead try to understand existing position embeddings more deeply and improve them by injecting more location information or making them more flexible. Our methods do not require complex modifications to the structures.

A method related to ours is proposed in [15]. This approach attaches an additional localization MLP head to the ViT model, which is used to predict the relative distance between two selected image patches. A corresponding loss function is designed for this additional relative localization auxiliary task [15]. Similar to our method, this work also concentrates on introducing more location information to the ViT models, and its solution is applying self-supervised learning. However, this work does not explicitly focus on improving position embeddings, which is the main research interest of our paper.

Encoding location information is also crucial for non-ViT models in other fields, such as neural representations for signals. The NeRF [18] model uses an MLP model being fed with coordinates as inputs to represent signals for synthesizing views of scenes. A mapping function is designed to project the input location coordinates to higher dimensions. Possible improvements to the coordinate-MLP models are learning instance-specific position embeddings [22] or using other mapping functions such as the Fourier feature [26]. The SIREN [24] model adds periodic activation functions to coordinate-MLP, which achieves better results than the previously proposed location encoding technique. These methods improve the ability to encode

location information for better approximation of higher frequency functions. Different from them, the ViT model applies position embeddings techniques to be sensitive to the order of the input sequence.

3. Existing position embeddings methods

3.1. Absolute position embeddings

We discuss two frequently used methods to implement absolute position embeddings. We could either use predefined fixed sinusoidal position embeddings [28, 31] or trainable embeddings with random initialization [4]. In ViT models, the absolute position embeddings are added to the patch embeddings before being fed into the self-attention layer.

Learnable absolute position embeddings are initialized with random numbers drawn from the normal distribution with a mean of 0 and a standard deviation of 0.02 [4]. Although this simple approach is frequently used in ViT models, previous work has pointed out that the absolute position embeddings method is not translation equivariant because it gives each image patch a unique position embedding [1, 2]. Translation equivariance ensures that an object's features do not rely on its absolute position in an image. Translation equivariance is an important property in image tasks related to data efficiency [9]. We will show by experiments that the learnable absolute position embeddings are data-inefficient compared with other methods.

Instead of using random values, the fixed sinusoidal position embeddings method [28] applies sine and cosine functions to initialize the position embeddings. Since we are dealing with image data, in this paper, we use the 2D sinusoidal position embeddings [31]:

$$\begin{aligned} \mathrm{PE}(x, y, 2i) &= \sin(x/10000^{\frac{4i}{D}}) \\ \mathrm{PE}(x, y, 2i+1) &= \cos(x/10000^{\frac{4i}{D}}) \\ \mathrm{PE}(x, y, 2j+D/2) &= \sin(y/10000^{\frac{4j}{D}}) \\ \mathrm{PE}(x, y, 2j+1+D/2) &= \cos(y/10000^{\frac{4j}{D}}), \end{aligned} \tag{1}$$

where x and y represent the horizontal and vertical coordinates of the image patch, and D denotes the dimension of position embeddings. i and j specify the location of the value within each individual position embedding. The fixed sinusoidal position embeddings use the same strategy as the learnable absolute position embeddings, giving each image patch a unique position embedding. We will show by experiments that they are defined to contain relative location bias, leading to higher data efficiency.

3.2. Relative position embeddings

Instead of directly adding to the patch embeddings, relative position embeddings [21, 23, 34] are used in the self-attention layer. A self-attention operation with 2D relative position embeddings [21] can be calculated as:

$$\mathbf{y_{ij}} = \sum_{a=0}^{i}\sum_{b=0}^{j} \mathrm{softmax}_{ab}(\mathbf{q}_{ij}^{T}\mathbf{k}_{ab} + \mathbf{q_{ij}^{T}r_{a-i,b-j}})\mathbf{v_{ab}}, \tag{2}$$

in which $\mathbf{r_{a-i,b-j}}$ is the learned relative position embedding. This term is only related to the relative horizontal and vertical shift $\mathbf{a-i}$ and $\mathbf{b-j}$ between the query and key patch, which implies that the relative position embeddings are translation equivariant. However, since relative position embeddings ignore the absolute position of the image patches, which is proven to be crucial to image classification tasks [8], relative position embeddings perform worse than absolute position embeddings [1].

3.3. Learnable Fourier position embeddings

Recent research raises the idea of using learnable Fourier features to encode locations in ViT models [13]. This method uses raw coordinates of the image patches as the inputs and trains an encoding function to extract Fourier features. The Fourier features are calculated as:

$$\mathbf{r_x} = \frac{1}{\sqrt{D}}[\cos \mathbf{x}W_r^T || \sin \mathbf{x}W_r^T], \tag{3}$$

where x is the input coordinates, D is the dimension of the feature embeddings, and W_r is the encoding function. The $||$ represents concatenation. The extracted Fourier features are then fed to an MLP layer to get the final position embeddings. The encoding function W_r is initialized by drawing random values from a Gaussian distribution $N \sim (0, \gamma^{-2})$, where γ is a hyperparameter. In such a way, the encoding function is initialized to make the Fourier features contain relative location information. According to the original paper [13], given two positions x and y, the Gaussian kernel over the two positions can now be approximated by the inner product of the two Fourier features $\mathbf{r_y}$ and $\mathbf{r_y}$ as in:

$$\mathbf{r_x} \cdot \mathbf{r_y} \approx \exp(-\frac{||x-y||^2}{\gamma^2}). \tag{4}$$

As a result, a useful inductive bias of L_2 distance is introduced to the model. We will show by experiments that the Fourier method actually introduces bias on both relative direction and relative distance to the ViT model.

4. Fully-controlled experiments

4.1. What method can encode relative location?

For image tasks, both relative direction and relative distance are important relative location information [34], therefore we investigate them separately. We construct red-green datasets and conduct fully-controlled experiments on them. We create a binary classification task to explore what methods can encode relative direction information, and the sample images are shown in Figure 2. Images in the first class

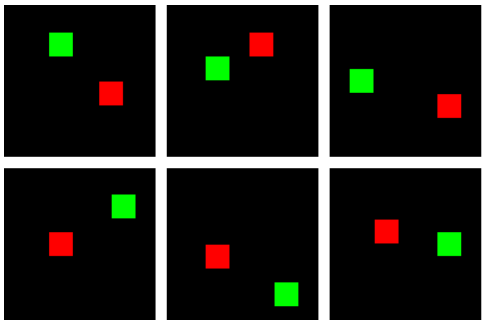


Figure 2. Sample images in the synthetic red-green dataset which is used to test relative direction information. Class 1 has the green square to the left of the red square, while the green square is to the right of the red square in class 2.

consist of a green square to the left of a red square. On the contrary, the green square is to the right of the red square for images in the second class. We design a regression task to investigate what position embeddings method can encode the relative distance information. We still use images with red and green squares, while the target now is the relative horizontal and vertical distance from the green square to the red square. We use accuracy and the R^2 score to evaluate the model's performance on the classification and regression tasks respectively. The R^2 score is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \quad (5)$$

A R^2 score of 1 indicates the best model, while it can be arbitrarily low. If the model outputs the average value of the true results as the prediction of all samples, the R^2 score will be 0.

We use a simplified version of the ViT model, which has only one encoder block. The rest components are the same as the original ViT model. The benefit of using a smaller model is that it saves time and resources, given that the tasks are relatively more straightforward than those on realistic datasets. We train the ViT model with different position embeddings methods, and each experiment is repeated 10 times using various random seeds to obtain an accurate measure of the performance. We use 5000 training images, 1000 validation images, and 1000 testing images. The results are shown in Table 1, which illustrates that all the position embeddings methods we test can solve the two tasks, although they have tiny differences in performance.

To verify whether these position embeddings methods indeed learn the left-and-right relative direction information and the relative distance information, or they solve the two tasks by simply hard-remembering all possible positions of the red and green squares, we need to analyze what location features these methods capture from the synthetic red-green dataset. In [30], a feature-level empirical analysis is proposed to investigate what position embeddings learn in different language models. In our research, we conduct similar experiments.

Pos. emb.	Direction (Acc)	Distance (R^2)
None	52.72 $\pm$ 1.08	-0.01 $\pm$ 0.01
Relative [21]	99.92 $\pm$ 0.06	0.84 $\pm$ 0.04
Absolute [4]	99.43 $\pm$ 0.36	0.92 $\pm$ 0.06
Fixed sinusoidal [28]	99.81 $\pm$ 0.16	0.96 $\pm$ 0.01
Learnable Fourier [13]	99.64 $\pm$ 0.32	0.94 $\pm$ 0.03

Table 1. Accuracy of the binary classification experiment testing the relative direction and R^2 score of the regression experiment testing the relative distance. The results suggest that all the position embeddings can encode relative location information.

We build a binary classification dataset for the relative direction task by obtaining pairs of learned position embeddings. For each pair of position embedding (x_i, x_j) that represent position (p_i, p_j), we construct the feature embedding $X(i, j)$ as $x_i - x_j$, while the target is 0 if position p_i is to the left of p_j otherwise 1. We then train a logistic regression model on this dataset. If the learned position embeddings capture the horizontal left-and-right relative direction information, then the logistic regression model should be able to solve the binary classification problem. For comparison, we make another binary classification dataset whose target is the vertical up-and-down relative direction information, which is not present in the red-green dataset and therefore cannot be learned by the position embeddings. We also conduct experiments on the initial position embeddings before training on the red-green dataset. Applying the same feature-level analysis for the relative distance task, we construct a dataset by collecting pairs of position embeddings, and the target is the relative shift between the two positions. We train a linear regression model on this dataset to see whether the learned embeddings indeed contain relative distance information. We conduct all experiments using 10-fold cross-validation for each of the 10 trained models on the red-green datasets.

The average accuracy and R^2 score of the 10 models in the above experiments can be found in Table 2, while the full results are shown in Appendix A. The absolute position embeddings method uses random initialization, therefore it does not contain any relative location information at the initial state. This is reflected in both the logistic regression and the linear regression experiments. Before training, the logistic regression classifier only gives random outputs with an average accuracy of around 50%, while the linear regression model also has random outputs with an average R^2 score of around 0. However, after training, the logistic regression model reaches above 99% average accuracy on the left-and-right relations and remains around 50% average accuracy on the up-and-down relations. The linear

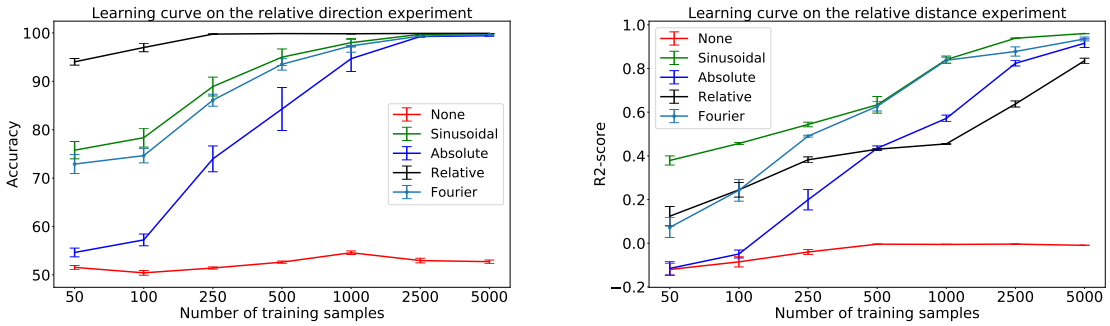


Figure 3. The learning curve of the relative direction experiment (left) and the relative distance experiment (right), targeting on testing the data efficiency in learning relative location information. The figures show that the fixed sinusoidal position embeddings are more data-efficient than the Fourier and the absolute position embeddings.

		Pos. emb.		
		Abs	Sinusoidal	Fourier
DIR.	left-right before	51.47	100.00	92.41
	left-right after	99.99	100.00	99.71
	up-down before	50.69	100.00	92.47
	up-down after	49.61	100.00	92.55
DIS.	before	0.00	1.0	0.91
	after	0.93	1.0	0.96

Table 2. Average accuracy of the logistic regression experiments for testing relative direction (DIR.) and average R^2 score of the linear regression experiments for testing relative distance (DIS.) before and after training on the synthetic red-green dataset. The feature-level analysis verifies that the position embeddings methods can learn relative location information since the accuracy and R^2 score increase after training. We also conclude that these methods have different levels of relative location bias as the accuracy and R^2 score are different before training.

regression model has an average R^2 score of approximately 0.93 after training. These results imply that the absolute position embeddings truly learn the relative location information existing in the training dataset. For fixed sinusoidal position embeddings, the logistic regression model could achieve 100% accuracy on both horizontal and vertical directions, while the linear regression model can reach a R^2 score of 1. These results indicate that the sinusoidal position embeddings are pre-defined to have a robust relative location bias. We also conduct experiments for the learnable Fourier position embeddings method, where the logistic regression model could reach 92% average accuracy on both horizontal and vertical directions before training. After training, the average accuracy on the left-and-right direction is around 99% while on the up-and-down direction is still 92%. The linear regression model can reach an average R^2 score of 0.91 before training, while after training, this figure increases to 0.96. We conclude that both sinusoidal position embeddings and Fourier position embeddings have a bias on relative location before training, while the sinusoidal position embeddings have a stronger one. The absolute position embeddings do not have any bias on relative location, but they can acquire this knowledge through training.

Based on the above results, we expect the sinusoidal position embeddings to be the most data-efficient method in learning relative location information as they have bias on it. In contrast, the absolute position embeddings are not data-efficient and need more data to learn. To verify this point, we conduct experiments to test the data efficiency of different position embeddings approaches. We still train simplified ViT models with different position embeddings techniques on classification and regression tasks using the synthetic red-green dataset. However, we vary the number of training samples this time and plot the learning curve. The result is displayed in Figure 3. The figure demonstrates that the sinusoidal position embeddings method is more data-efficient than the Fourier position embeddings and learnable absolute position embeddings, as its learning curve is higher than the other two.

4.2. What method can encode absolute location?

Previous work has stated the importance of absolute location to image tasks [8]. For ViT models, each image patch should be given a unique position embedding to encode the absolute location. The absolute position embeddings, sinusoidal position embeddings, and Fourier position embeddings can handle this task. However, when the relative position embeddings take place in the self-attention layer, the model only takes the relative location between the query and key image patch into account and neglects the absolute location of each image patch. Therefore, the relative position embeddings cannot encode absolute location. We construct another binary classification dataset to verify this statement, in which the images in class 1 have two squares in the upper part of the image while images in class 2 have

	Accuracy
None	49.79 ± 1.86
Fixed sinusoidal [28]	99.94 ± 0.10
Absolute [4]	99.85 ± 0.13
Relative [21]	54.02 ± 7.12
Learnable Fourier [13]	99.99 ± 0.03

Table 3. Accuracy of the binary classification experiment testing the absolute location. The results suggest that the relative position embeddings cannot encode the absolute location information.

both squares in the bottom part of the image. The relative location between the two squares does not matter anymore in this dataset, while their absolute locations play an important role. The results are shown in Table 3, which demonstrates that the relative position embeddings cannot encode the absolute location information, while the other three approaches can handle it.

4.3. Weakness of fixed sinusoidal position embeddings

Previous sections suggest that the fixed sinusoidal position embeddings contain useful relative location bias, and therefore they are more data-efficient than other methods. However, do fixed sinusoidal position embeddings have drawbacks? Previous research state that the fixed sinusoidal position embeddings method is not flexible and cannot adapt to learn the location features of the datasets [13, 15]. We design another synthetic red-green dataset to determine how its inflexibility may influence the model. The dataset is similar to the one we use in 4.2. However, in this experiment, we use different colors for training images and testing images. Figure 4 shows some sample images.

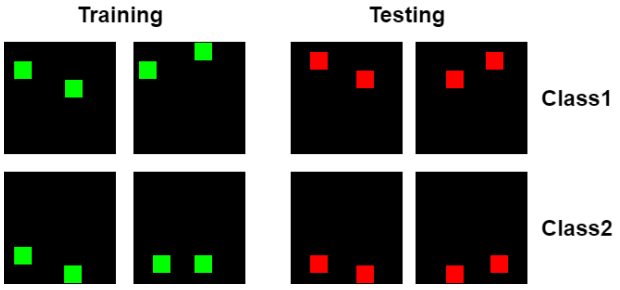
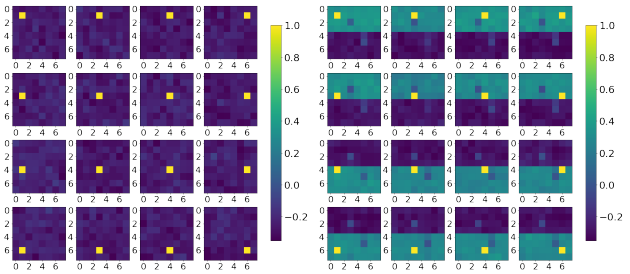


Figure 4. Sample images in the synthetic red-green dataset which is used to test the influence of the inflexibility of the fixed sinusoidal position embeddings. The color of the squares are different between the training and testing set.

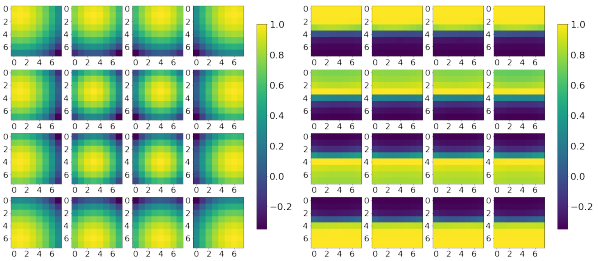
The result of using different position embeddings on this dataset is shown in Table 4. This task still requires the position embeddings method to encode absolute location. However, the fixed sinusoidal position embeddings do not

	Accuracy
None	50.06 ± 0.16
Fixed sinusoidal [28]	57.03 ± 14.90
Absolute [4]	97.46 ± 3.26
Relative [21]	49.70 ± 0.60
Learnable Fourier [13]	98.49 ± 3.97

Table 4. Accuracy of the binary classification experiment testing the influence of inflexibility of the fixed sinusoidal position embeddings. The results suggest that the fixed sinusoidal position embeddings do not work anymore when the color is changed.



(a) Learnable absolute position embedding



(b) Learnable Fourier position embeddings

Figure 5. The cosine similarity heatmap of using learnable position embeddings methods. The heatmap is able to display the similarity of position embeddings between a selected image patch and all other image patches. The left images show the heatmaps before training, while the right images show the heatmaps after training. The heatmaps illustrate that the learnable position embeddings can adapt themselves to capture the location information in the datasets.

work anymore when the color is changed. On the other hand, the learnable absolute position embeddings and learnable Fourier position embeddings could still solve the problem. To find out why this happens, we analyze what the position embeddings learn by plotting the cosine similarity heatmaps, which are displayed in Figure 5.

Since the learnable absolute position embeddings are initialized with random values, there is no specific pattern before training. For a certain image patch, the cosine similarities between its position embedding and all other image patches are the same. However, after training on the dataset, a clear boundary that separates the upper and bottom parts

	Relative [21]	Absolute [4]	Learnable Fourier [13]	Fixed sinusoidal [28]
Encode absolute location	✗	✓	✓	✓
Encode relative direction	✓	✓	✓	✓
Encode relative distance	✓	✓	✓	✓
Have relative direction bias	✗	✗	$\sim$	✓
Have relative distance bias	✗	✗	$\sim$	✓
Learnable	✓	✓	✓	✗

Table 5. A summary on properties of different position embeddings. ✓means the method holds this property while ✗ indicates that the method does not have the corresponding property. $\sim$ implies that it is questionable that whether the method completely hold the the property. It depends on other factors like the setting of hyperparameter.

of the image can be observed. This implies that the absolute position embeddings can know that all image patches in the same half are more similar to each other than those image patches in the other half. For the learnable Fourier position embeddings, we could observe a clear pattern showing relative location bias before training. Each image patch is more similar to the image patches close to it than those further away. However, after training, a completely different pattern could be observed, that images in the same half are similar to each other. These heatmaps clearly illustrate that the learnable position embeddings can adapt to capture the location information in the dataset through training.

The fixed sinusoidal position embeddings are not learnable, and their relative location bias does not fit the dataset. Therefore, the ViT model using fixed sinusoidal position embeddings cannot solve this problem simply depending on position embeddings. The solution relies on the fused appearance and position features used in the self-attention layer. Therefore, when the color is changed in the testing set, the fixed sinusoidal position embeddings no longer work. However, the learnable position embeddings can adapt to capture crucial location information in the dataset. Therefore, the ViT models using learnable position embeddings can decouple the appearance features and the location features.

In a nutshell, the fixed sinusoidal position embeddings are not trainable, leading the ViT model to be trained in a different manner from those models using learnable position embeddings, which may result in inferior performances under certain situations.

5. Methods

Based on the analysis in the previous section, we make a summary of properties that different position embeddings methods hold in Table 5. For realistic datasets, both absolute and relative location information is important, and therefore we need position embeddings methods that could encode both. From the data-efficiency perspective, an ideal position embeddings method should have a bias on relative location. In addition, it should be sufficiently flexible to adapt to the dataset. From Table 5, we believe that the Learnable Fourier position embeddings and the fixed sinusoidal position embeddings have the most required properties.

5.1. Improving the learnable Fourier method

The learnable Fourier position embeddings almost meet all the requirements of an ideal position embeddings method. However, from the fully-controlled experiments in 4.1 we show that it does not have sufficient relative location bias compared with the sinusoidal position embeddings. In fact, when conducting the fully-controlled experiments, we find that the Fourier method's relative location bias heavily depends on the hyperparameter γ. It needs careful tuning to obtain a satisfying relative location bias, which may require a long time.

Tuning the hyperparameter does not ensure an optimal result. Instead, we propose to pretrain the Fourier position embeddings to acquire desired relative location bias without the need to tune the hyperparameter. Pretraining is a helpful technique in computer vision that enable a model to learn from one dataset and generalize to other datasets [36]. In our case, we propose to pretrain the Fourier position embeddings on two designed red-green datasets. The first task is a classification problem where the model needs to determine the relative direction between the red and green squares in both horizontal and vertical directions. Therefore there are four classes in total. The second task is the same as what we used to test relative distance in 4.1. We pretrain the model on these two tasks linearly. After pretraining, we apply the logistic regression and linear regression experiments again to evaluate the level of relative location bias that the Fourier position embeddings obtain. The accuracy of the logistic regression classifier on both directions increases from 92% to 99%, while the R^2 score of the linear regression model also increases from 0.91 to 0.98. These results demonstrate that the Fourier method acquires more relative location bias by the pretraining tasks compared with the initial state.

Our method has two advantages compared with pretraining on a large-scale dataset like ImageNet or JFT-300M. First, our pretraining tasks are simple and do not require a

large amount of training data. Therefore they are not time nor resource consuming. We use 5000 training data in both pretraining datasets, and the training completes within 15 minutes using a single V-100 GPU for each task. As the pretraining datasets are constructed ourselves, we could use as much data as possible. Moreover, the Fourier position embeddings only learn the relative location bias from the red-green dataset. However, if we pretrain it on a realistic dataset, it may learn other knowledge specific to that dataset, which could hurt the performance on the target task. For example, previous research points out that vision tasks related to medical images may not benefit from pretraining on ImageNet because of the modality difference between medical images and natural images [32].

5.2. Improving the fixed Sinusoidal method

The fixed sinusoidal position embeddings have sufficient bias on relative location information, while its main weakness is its inability to be trained. Therefore, we propose to make the sinusoidal position embeddings learnable. Observing Eq 1 and 3, we find that the Fourier features are similar to the sinusoidal features, while the only difference lies in the ordering of the sine and cosine functions. For a single position embedding, the first half of the Fourier features is computed by using the cosine function, while the second half comes from the sine function. On the other hand, the sinusoidal features alternately apply cosine and sine functions. Based on this observation, we propose to construct learnable sinusoidal position embeddings using a similar structure as the learnable Fourier method. We use raw 2D-coordinate $\mathbf{x} = [x, y]$ as the input, and a linear layer $W \in \mathbb{R}^{\frac{D}{2} \times 2}$ as the encoding function, where D specifies the dimension of the position embeddings. The encoding function is initialized in such a way that the elements in the resulting vector $\mathbf{y} = \mathbf{x}W^T$ satisfies

$$\mathbf{y_i} = \begin{cases} x/10000^{\frac{4\times i}{D}} & \text{if } i < \frac{D}{4} \\ y/10000^{\frac{4\times(i-D/4)}{D}} & \text{if } i \geq \frac{D}{4}, \end{cases} \tag{6}$$

which are the same as those inside the cosine and sine bracket in Eq 1. We then apply cosine and sine function on $\mathbf{y}$ and get two new vectors $\mathbf{y_{cos}}$ and $\mathbf{y_{sin}}$. Afterwards, instead of directly concatenating $\mathbf{y_{cos}}$ and $\mathbf{y_{sin}}$ as in Eq 3, we concatenate each of their values alternately such that the results are the same as the values of the sinusoidal position embeddings. The sinusoidal position embeddings can be trained now and adapt to capture location information in the datasets, as the encoding function W is trainable.

6. Experiment

6.1. Experimental setup

To investigate how the properties in Table 5 influence the model and whether our proposed methods could improve on realistic datasets, we test these methods based on the image classification task using some common benchmark datasets. We use Oxford Flower-102 [19], ImageNet-tiny [11], CIFAR-10 and CIFAR-100 [10]. The statistics of these datasets can be found in Table 6. We train the ViT-base model using different position embeddings methods from scratch. To test the data efficiency, we change the number of training samples per class and analyze the learning curve. We repeat each experiment five times to get a reliable measure of the model's accuracy and record the average accuracy and standard deviation. In addition, we also compare our methods with a similar method which uses a self-supervised learning task to introduce relative location information [16], using full images of these four datasets.

Dataset name	Train size	Test size	Num of classes
CIFAR-10	50000	10000	10
CIFAR-100	50000	10000	10
Flower-102	6552	818	102
ImageNet-tiny	100000	10000	200

Table 6. The statistics of the realistic datasets

6.2. Results

Figure 6 demonstrates the modified learning curves of different position embeddings approaches on the four different testing datasets. The curves illustrate the amount of improvements in accuracy compared with the baseline model. The baseline model uses the default learnable absolute position embeddings.

When the number of training samples is small, the relative position embeddings can perform better than the baseline model, which uses learnable absolute position embeddings. However, as the number of training samples increases, the relative position embeddings become worse. This could be explained by the fact that the relative position embeddings are translation equivariant. However, a small amount of training samples is insufficient to make the learnable absolute position embeddings learn relative location information, and the absolute position embeddings are not translation equivariant. Therefore, the relative position embeddings are better. When there is more data, the absolute position embeddings can encode relative location information, enhancing translation equivariance. The drawback of relative position embeddings becomes more prominent because they cannot encode absolute location information, resulting in lower accuracy. The fixed sinusoidal and Fourier position embeddings have bias on relative location information, and therefore they are always better and more data-efficient than the learnable absolute position embeddings. In a nutshell, the properties of different position embeddings we explore on synthetic datasets could influence the performance on realistic datasets.

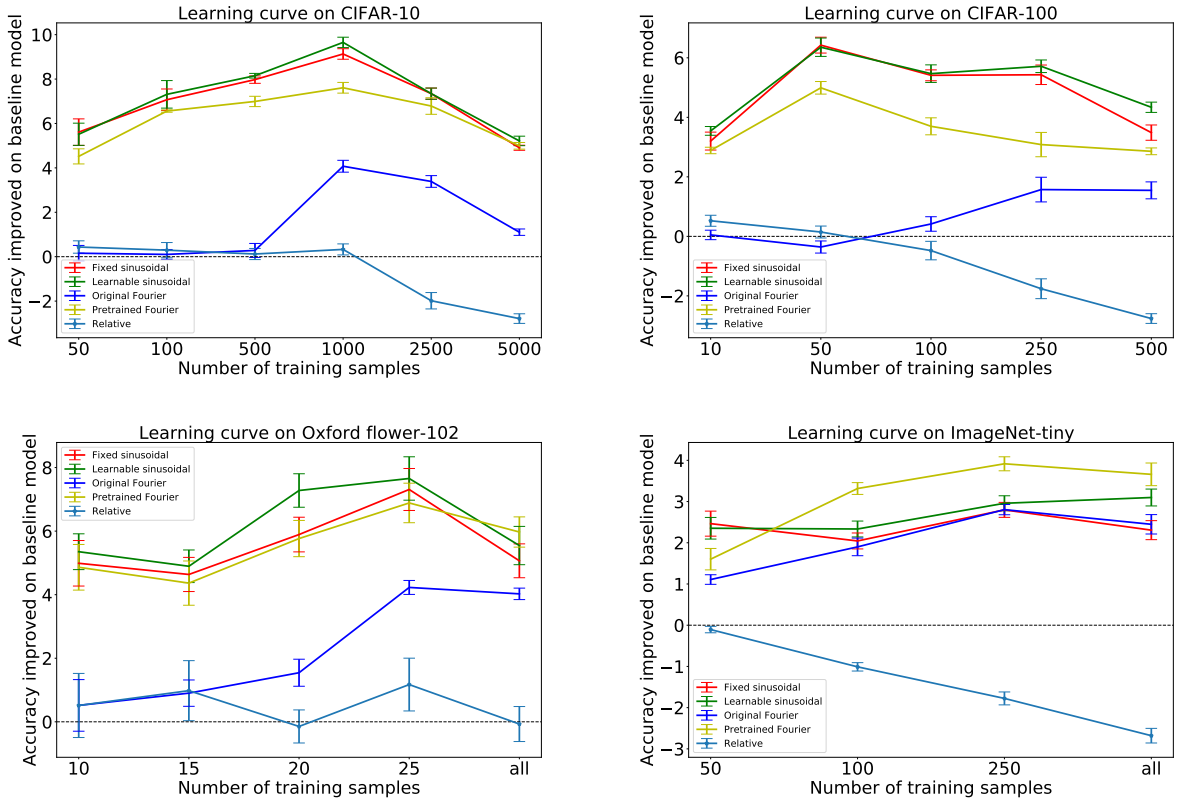


Figure 6. Modified learning curves. The plots record the amount of accuracy that different methods improve on the baseline model, which uses learnable absolute position embeddings. The baseline model is represented by the black dashed line, while other methods are represented by colored lines. The plots illustrate that our proposed methods have improvements on the existing position embeddings.

	CIFAR10	CIFAR100	Oxford Flower-102	ImageNet-Tiny
Absolute	80.61 ± 0.10	55.03 ± 0.43	70.80 ± 0.80	42.36 ± 0.32
Regularization by self-supervision [16]	84.83 ± 0.11	57.93 ± 0.27	75.76 ± 0.49	44.80 ± 0.18
Learnable sinusoidal	**85.82 ± 0.40**	**59.37 ± 0.23**	76.35 ± 0.83	45.46 ± 0.28
Pretrained learnable Fourier	85.60 ± 0.35	57.89 ± 0.25	**76.78 ± 0.65**	**46.02 ± 0.36**

Table 7. The accuracy on full datasets using learnable absolute position embeddings, our proposed methods and a similar method which applies self-supervised learning for regularization. The results indicate that our methods are competitive.

For our proposed pretraining method, Figure 6 illustrates that the learning curve of the pretraining method is always higher than the one of the original Fourier method on all four datasets. These results imply that our pretraining method helps the learnable Fourier position embeddings enhance the data efficiency by introducing more relative location bias. The learnable sinusoidal method also improves the original fixed sinusoidal position embeddings, although with a relatively smaller increase compared to what pretraining brings to the Fourier method. On CIFAR-10 and CIFAR-100, when there are a small amount of training data per class, there is no difference between the learnable and fixed sinusoidal method. The learnable sinusoidal method gradually improves as the number of training samples increases. This could be explained by the fact that the learnable method still needs data to learn to show its effectiveness. On the Oxford flower and ImageNet-tiny dataset, the improvements brought by the learnable sinusoidal method are more prominent. Across different methods, on CIFAR10/100 and Oxford flower dataset, the learnable sinusoidal position embeddings are more data-efficient than the Fourier position embeddings, even with the pretraining technique applied. However, on the ImageNet-tiny dataset, the pretrained Fourier method is better. In conclu-

sion, our proposed methods show promising results on realistic datasets.

Table 7 display the comparison in accuracy between using our proposed methods and the regularization method by self-supervision [16], which also focuses on introducing more relative location information to the ViT model. The results indicate that our proposed methods are competitive compared to this similar approach, as our accuracy is slightly better on all testing datasets.

7. Conclusion

In this paper, we investigate how the location information is encoded in the ViT model. We compare different position embeddings methods and analyze their advantages and drawbacks based on fully-controlled experiments and feature-level analysis on synthetic datasets. The default learnable absolute position embeddings lack the relative location bias, and they are not translation equivariant. The relative position embeddings are not sensitive to absolute location information, which is crucial in visual tasks. The fixed sinusoidal position embeddings are not flexible enough, while the learnable Fourier position embeddings do not contain much relative location bias. We propose pre-training tasks to enhance the relative location bias of the Fourier method, and we also make the fixed sinusoidal position embeddings learnable. The results on four realistic datasets show the effectiveness of our new methods and suggest that they are competitive compared with a similar state-of-the-art approach.

In this research, we only focus on the original ViT model. For future research, we could investigate whether our proposed methods generalize to other ViT-based models like Deit [27], Levit [6] or Swin-transformer [17]. We could also research whether our methods also contribute to convolution-enhanced ViT models.

References

[1] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 2, 3

[2] Zihang Dai, Hanxiao Liu, Quoc Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[3] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019. 2

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2, 3, 4, 6, 7

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[6] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12259–12269, 2021. 10

[7] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020. 2

[8] Md Amirul Islam, Sen Jia, and Neil DB Bruce. How much position information do convolutional neural networks encode? *arXiv preprint arXiv:2001.08248*, 2020. 2, 3, 5

[9] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14274–14285, 2020. 3

[10] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 8

[11] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 8

[12] Seung Hoon Lee, Seunghyun Lee, and Byung Cheol Song. Vision transformer for small-size datasets. *arXiv preprint arXiv:2112.13492*, 2021. 2

[13] Yang Li, Si Si, Gang Li, Cho-Jui Hsieh, and Samy Bengio. Learnable fourier features for multi-dimensional spatial positional encoding. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 3, 4, 6, 7

[14] Tatiana Likhomanenko, Qiantong Xu, Gabriel Synnaeve, Ronan Collobert, and Alex Rogozhnikov. Cape: Encoding relative positions with continuous augmented positional embeddings. *Advances in Neural Information Processing Systems*, 34, 2021. 1

[15] Xuanqing Liu, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. Learning to encode position for transformer with continuous dynamical model. In *International Conference on Machine Learning*, pages 6327–6335. PMLR, 2020. 2, 6

[16] Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, 34, 2021. 8, 9, 10

[17] 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. 10

[18] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 2

[19] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 8

[20] 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. *arXiv preprint arXiv:1910.10683*, 2019. 2

[21] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. *Advances in Neural Information Processing Systems*, 32, 2019. 2, 3, 4, 6, 7

[22] Sameera Ramasinghe and Simon Lucey. Learning positional embeddings for coordinate-mlps. *arXiv preprint arXiv:2112.11577*, 2021. 2

[23] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018. 2, 3

[24] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 2

[25] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. 2

[26] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 2

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 10

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 2, 3, 4, 6, 7

[29] Benyou Wang, Donghao Zhao, Christina Lioma, Qiuchi Li, Peng Zhang, and Jakob Grue Simonsen. Encoding word order in complex embeddings. *arXiv preprint arXiv:1912.12333*, 2019. 2

[30] Yu-An Wang and Yun-Nung Chen. What do position embeddings learn? an empirical study of pre-trained language model positional encoding. *arXiv preprint arXiv:2010.04903*, 2020. 4

[31] Zelun Wang and Jyh-Charn Liu. Translating math formula images to latex sequences using deep neural networks with sequence-level training. *International Journal on Document Analysis and Recognition (IJDAR)*, 24(1):63–75, 2021. 2, 3

[32] Yang Wen, Leiting Chen, Yu Deng, and Chuan Zhou. Rethinking pre-training on medical imaging. *Journal of Visual Communication and Image Representation*, 78:103145, 2021. 8

[33] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22–31, 2021. 2

[34] 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, 3

[35] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 579–588, 2021. 2

[36] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. Rethinking pre-training and self-training. *Advances in neural information processing systems*, 33:3833–3845, 2020. 7

A. Results of the logistic regression and linear regression experiments

		left-right acc before	left-right acc after	up-down acc before	up-down acc after
Absolute pos. emb.	model 0	51.50 ± 18.76	99.97 ± 0.08	53.69 ± 27.51	52.38 ± 26.94
	model 1	50.64 ± 17.51	100.00 ± 0.00	46.10 ± 24.70	45.44 ± 26.45
	model 2	50.53 ± 16.62	100.00 ± 0.00	49.26 ± 29.14	47.36 ± 29.60
	model 3	51.45 ± 15.64	100.00 ± 0.00	47.81 ± 28.45	45.99 ± 28.51
	model 4	52.76 ± 18.34	100.00 ± 0.00	52.60 ± 23.04	50.57 ± 23.93
	model 5	51.28 ± 18.89	100.00 ± 0.00	52.30 ± 29.19	51.88 ± 28.11
	model 6	49.91 ± 16.19	100.00 ± 0.00	51.29 ± 28.45	51.88 ± 27.76
	model 7	52.14 ± 17.74	100.00 ± 0.00	54.33 ± 25.36	53.27 ± 25.30
	model 8	52.12 ± 18.83	100.00 ± 0.00	47.78 ± 25.13	46.52 ± 25.92
	model 9	52.34 ± 18.13	99.94 ± 0.17	51.71 ± 25.78	50.85 ± 26.20
Fourier pos. emb.	model 0	92.34 ± 4.32	99.98 ± 0.01	92.21 ± 3.78	92.43 ± 3.97
	model 1	92.99 ± 3.12	99.78 ± 0.01	92.09 ± 3.62	91.92 ± 3.09
	model 2	93.29 ± 4.31	99.99 ± 0.01	92.64 ± 3.19	92.93 ± 2.78
	model 3	91.53 ± 4.65	99.45 ± 0.04	92.43 ± 3.51	92.98 ± 3.31
	model 4	93.13 ± 2.89	100.00 ± 0.00	92.49 ± 3.77	92.23 ± 3.53
	model 5	92.11 ± 3.45	99.77 ± 0.03	92.94 ± 3.32	91.99 ± 3.13
	model 6	92.23 ± 3.04	100.00 ± 0.00	92.18 ± 3.74	92.44 ± 3.89
	model 7	91.19 ± 3.52	99.16 ± 0.04	92.47 ± 3.64	92.68 ± 3.59
	model 8	93.49 ± 3.01	100.00 ± 0.00	93.83 ± 3.14	93.57 ± 3.28
	model 9	91.79 ± 3.03	98.97 ± 0.02	91.46 ± 3.15	92.29 ± 3.45
Sinusoidal pos. emb.	model 0	100.00 ± 0.00		100.00 ± 0.00	
	model 1	100.00 ± 0.00		100.00 ± 0.00	
	model 2	100.00 ± 0.00		100.00 ± 0.00	
	model 3	100.00 ± 0.00		100.00 ± 0.00	
	model 4	100.00 ± 0.00		100.00 ± 0.00	
	model 5	100.00 ± 0.00		100.00 ± 0.00	
	model 6	100.00 ± 0.00		100.00 ± 0.00	
	model 7	100.00 ± 0.00		100.00 ± 0.00	
	model 8	100.00 ± 0.00		100.00 ± 0.00	
	model 9	100.00 ± 0.00		100.00 ± 0.00	

Table 8. Results of the logistic regression experiments on the position embeddings before and after the ViT model being trained on the left-and-right red-green dataset, using different position embeddings approaches.

	Absolute pos. emb. R^2 score		Fourier pos. emb. R^2 score		Fourier pos. emb. R^2 score	
	before training	after training	before training	after training	before training	after training
model 0	0.0153 ± 0.47	0.9176 ± 0.06	0.9123 ± 0.04	0.9621 ± 0.03	1.00 ± 0.00	
model 1	-0.1051 ± 0.50	0.9428 ± 0.02	0.9192 ± 0.03	0.9679 ± 0.04	1.00 ± 0.00	
model 2	-0.0377 ± 0.50	0.9332 ± 0.04	0.9155 ± 0.02	0.9597 ± 0.05	1.00 ± 0.00	
model 3	-0.0527 ± 0.51	0.9408 ± 0.03	0.9237 ± 0.02	0.9699 ± 0.01	1.00 ± 0.00	
model 4	0.0807 ± 0.41	0.9326 ± 0.02	0.9152 ± 0.05	0.9636 ± 0.04	1.00 ± 0.00	
model 5	-0.0419 ± 0.45	0.9319 ± 0.03	0.9194 ± 0.02	0.9633 ± 0.02	1.00 ± 0.00	
model 6	-0.0271 ± 0.49	0.9463 ± 0.01	0.9258 ± 0.03	0.9741 ± 0.02	1.00 ± 0.00	
model 7	0.0297 ± 0.44	0.9337 ± 0.02	0.9169 ± 0.01	0.9648 ± 0.02	1.00 ± 0.00	
model 8	-0.0151 ± 0.44	0.9504 ± 0.03	0.9170 ± 0.03	0.9614 ± 0.01	1.00 ± 0.00	
model 9	0.0311 ± 0.43	0.9239 ± 0.02	0.9093 ± 0.05	0.9528 ± 0.03	1.00 ± 0.00	

Table 9. Results of the logistic regression experiments on the position embeddings before and after the ViT model being trained on the left-and-right red-green dataset, using different position embeddings approaches.

2

Introduction

In 2017, the transformers model was proposed for Natural Language Processing (NLP) task [34], in which the self-attention mechanism plays an essential role. In recent years, numerous work has focused on adapting the transformers model to the field of Computer Vision (CV). One pioneering work is the Vision Transformer (ViT) [7]. Before ViT came out, the Convolutional Neural Network (CNN) was dominant in CV. However, the ViT model uses the structure of a pure transformers model without the help of convolutions and achieves state-of-the-art performances on image classification tasks. Unlike CNN, the revolutionary innovation of ViT is splitting the input image into grid-like non-overlapping patches and treating each patch as a token, like a word in a sentence for NLP tasks. This allows the global features across different image patches to be captured through the self-attention layers. Building upon the ViT models, there are many variants such as DeiT [33], LeViT [11], and Swin-Transformer [26]. With more and more advantages of ViT have been revealed, the interest in research of ViT models has grown.

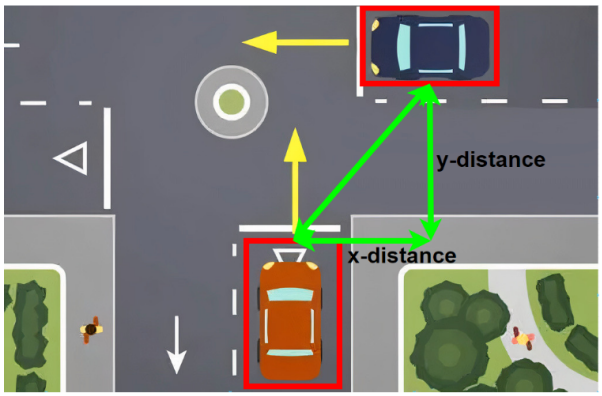


Figure 2.1: An overview of the three types of location information in images. This figure displays a scenario of self-driving techniques. The driving directions of the red and blue vehicles are indicated by the yellow arrows, which show that they will meet at the intersection. To avoid a car crash with the blue car, the computer vision model of the self-driving system in the red car should know the relative direction between them, that the blue car is at the front-right of the red car. In addition, the model should be aware of the relative distance between the two vehicles, which can be decomposed into horizontal and vertical distances, as shown by the green arrows. The model should also be sensitive to the absolute location to avoid problems like getting into the wrong lane. In this paper, we are interested in investigating whether the position embeddings approach in the ViT model could encode the three types of location information. This image is adapted from `https://www.photophoto.cn`.

2.1. Motivation

Previous work identifies three types of location information in image data: absolute location, relative direction, and relative distance [40], as shown in Figure 2.1. The absolute location information specifies where the target object is in an image. The relative direction and relative distance together describe

the relative location between different objects. Therefore, both the absolute and relative location are important information in computer vision tasks [17].

Position embeddings are used to feed location information into the ViT models. The original ViT uses the learnable absolute position embeddings [6] as the default setting, while there are other methods like fixed sinusoidal position embeddings [34, 38], relative position embeddings [30, 31], and the recently proposed learnable Fourier position embeddings [21]. These methods have different mechanisms to encode locations. The relative position embeddings are applied in the self-attention layers, and they are designed to encode relative location information. Other methods give each image patch a unique absolute position embedding, and they are introduced to the ViT model before the self-attention layers. The levels of predefined location information injected into these models are also different. The absolute and relative position embeddings are initialized with random numbers, while the sinusoidal and Fourier methods introduce additional location information by using predefined features. Another difference lies in the ability to learn. The fixed sinusoidal position embeddings are not learnable, while other approaches can be trained to adapt to the datasets. Given all the differences, it is still unclear what type of location information can be encoded by these methods. In addition, it is not conclusive yet what the predefined features bring to the model and how the inability to learn could influence the model.

2.2. Research questions

Driven by the above motivations, in our research, we try to answer the following main research question:

- What location information can be encoded by different position embeddings methods?

We conduct several fully-controlled experiments to answer the above research question. The results indicate that the relative position embeddings cannot encode absolute location information, while other methods can encode it. All the position embeddings methods we test can encode relative location information, while they have different levels of bias on it. The absolute position embeddings do not have any bias, while the fixed sinusoidal position embeddings and learnable Fourier position embeddings have. The fixed sinusoidal position embeddings have a stronger one. Based on the results of the experiments, we believe that the fixed sinusoidal position embeddings are the best since they can encode both absolute and relative location information. In addition, the fixed sinusoidal position embeddings have relative location bias, leading to higher data efficiency in learning relative location information. However, the fixed sinusoidal position embeddings are not trainable, which is the main difference compared with those learnable position embeddings. Therefore, we have the following research question:

- How would the inability to learn influence the model?

We conduct another fully-controlled experiment, and the results suggest that the inability to learn leads the ViT model to be trained in a different manner compared with those learnable methods, which may cause inferior results under certain situations.

We summarize the properties of all position embeddings methods based on the above results. We believe that the learnable Fourier position embeddings and fixed sinusoidal position embeddings have the most required properties of an ideal position embeddings method. Therefore, we propose improvements to these two approaches. We design pretraining tasks to introduce more relative location bias to the learnable Fourier method. In addition, we make the fixed sinusoidal position embeddings trainable using a similar structure as the learnable Fourier method. We then conduct experiments on realistic datasets to investigate whether our proposed methods work. We also compare with a related state-of-the-art approach.

3

Vision Transformer

In this project, we are interested in investigating the Vision Transformer (ViT) model [7] as it has shown promising performances in vision tasks. In this section, we will give an introduction to how ViT works. We will discuss some critical components in the ViT structure. In addition, we will briefly introduce some variants that are built based on the original ViT model.

3.1. Overall structure

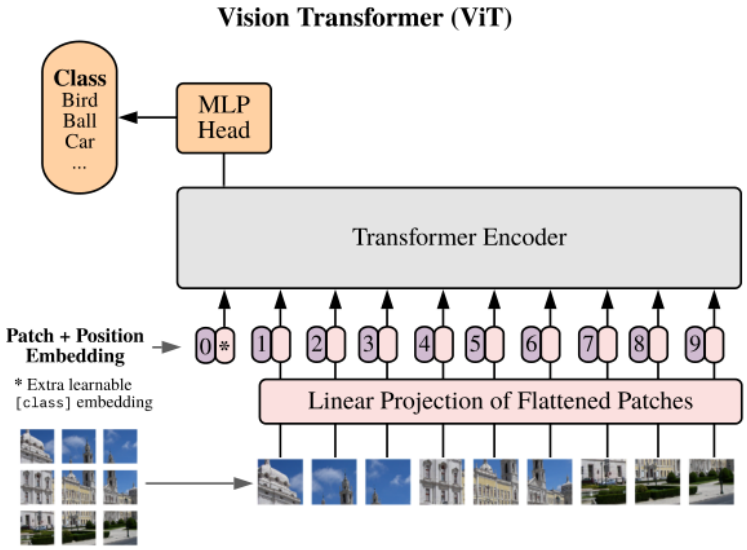


Figure 3.1: An overview of the Vision Transformer model. This figure is adapted from the original ViT paper [7].

Figure 3.1 illustrates the overall structure of the Vision Transformer model. Given an input image, the pre-processing step splits the image into grid-like non-overlapping image patches of size $P \times P$. Suppose the original image has a resolution of $H \times W$. In that case, we would have N image patches, where $N = HW/P^2$. Afterwards, the image patches are flattened and reshaped. The results are sequential 2D patches in the size of $N \times (P^2 \cdot C)$, where C represents the number of channels of the original image, typically 3 for color images. The flattened patches are then sent to a trainable linear projection layer and mapped to the dimension D. The resulting patch embeddings are then added with position embeddings, and we now get the feature embeddings. Similar to the design of the BERT model [6], for the image classification task, a class embedding is appended at the beginning of the feature embeddings. These embeddings are then fed into the Transformer encoder. The size of the

resulting encoded embeddings is not changed. The encoded embedding of the class token is then fed into an MLP head to predict the class of the image.

3.2. Encoder

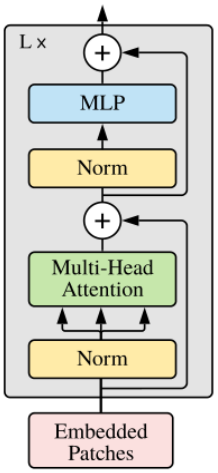


Figure 3.2: An overview of an individual encoder block in the ViT model. This figure is adapted from the original ViT paper [7].

The Transformer encoder consists of several encoder blocks. Figure 3.2 displays the structure of an individual encoder block. The multi-head attention layer and the MLP layer are two primary components of the encoder block. These two layers are connected sequentially. A layer normalization is applied before the attention layer and the MLP layer. Inspired by the structure of ResNet [13], residual connections are applied to both the attention and MLP layers. All the encoder blocks have the same structure. The number of encoder blocks is a hyperparameter and is determined before training. Using more encoder blocks enables the model to have a more powerful encoding function, while it also requires longer time and more resources to execute.

3.3. Self-attention

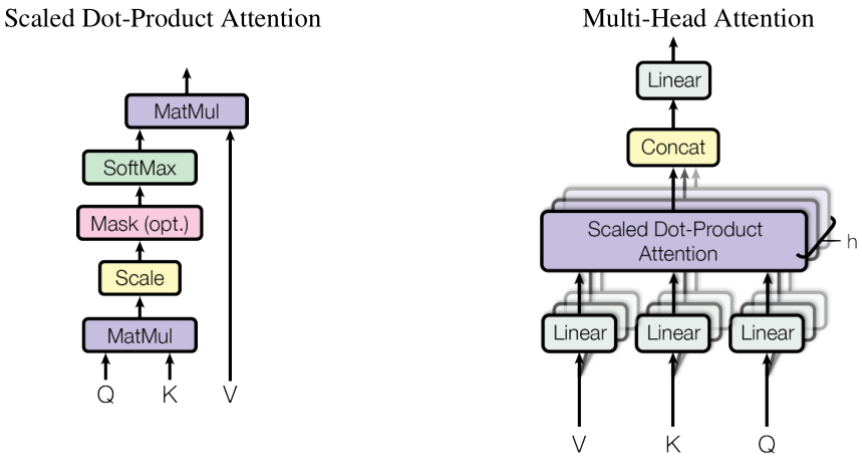


Figure 3.3: An overview of the self attention layer (left) and the multi-head attention layer (right). This figure is adapted from the original Transformer paper [34]

The self-attention mechanism, which is able to capture global features, plays a crucial role in the ViT model. It was originally proposed in the Transformer model for Natural Language Processing (NLP)

tasks [34]. The principle of the self-attention mechanism is similar to that of an information retrieval system or a recommendation system. The patch embeddings are first fed into three linear projection layers separately to get the query embeddings, key embeddings, and value embeddings. For each query embedding, a similarity score between it and all the key embeddings is calculated by the inner product. Afterwards, the weights between the query and all the keys could be calculated by mapping the similarity scores to the range from 0 to 1, using the softmax function. Finally, a weighted sum over the value embeddings is calculated to get the resulting encoded embedding. The whole process can be done in parallel by applying matrix multiplication:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{3.1}$$

where the term d_k is the vector dimension of the query and key embeddings. $\sqrt{d_k}$ is used as a scaling factor to prevent the gradient vanishing problem.

Instead of applying the single-head self-attention mechanism, we could also use a multi-head attention layer in which multiple groups of query, key, and value embeddings are involved. The self-attention mechanism is applied for each group individually. Eventually, the results of these groups are concatenated to get the final encoded embeddings. The advantage of a multi-head attention layer is that richer information or features can be captured by it.

3.4. Position embeddings

The self-attention mechanism itself is permutation-invariant to the input sequences [23]. Changing the order of the input image patches does not influence the final results. This is not a desired feature in vision tasks. Therefore, position embeddings are introduced such that the ViT model can be aware of the spatial relations between the input image patches [34]. This section introduces the widely used methods for position embeddings.

3.4.1. Absolute position embeddings

The absolute position embeddings approach is initially proposed in the BERT model [6] for NLP tasks, which is also used as the default method in the ViT model. This method initializes the learnable position embeddings with random numbers drawn from the normal distribution with a mean of 0 and a standard deviation of 0.02. The absolute position embeddings have the same vector dimension as the patch embeddings, and they are directly added together before being fed into the encoder. During the training, the absolute position embeddings and the patch embeddings are updated simultaneously.

3.4.2. Sinusoidal position embeddings

The fixed sinusoidal position embeddings are used in the original transformer model [34], which are calculated as:

$$\begin{aligned} \text{PE}(pos, 2i) &= \text{sin}(pos/10000^{\frac{2i}{D}}) \\ \text{PE}(pos, 2i+1) &= \text{cos}(pos/10000^{\frac{2i}{D}}), \end{aligned} \tag{3.2}$$

where pos represents the position of the current token, i is the location of the value within an individual position embedding. As we mainly work on image data in this research, we use the 2D sinusoidal position embeddings [38], which are calculated as:

$$\begin{aligned} \text{PE}(x, y, 2i) &= \text{sin}(x/10000^{\frac{4i}{D}}) \\ \text{PE}(x, y, 2i+1) &= \text{cos}(x/10000^{\frac{4i}{D}}) \\ \text{PE}(x, y, 2j+D/2) &= \text{sin}(y/10000^{\frac{4j}{D}}) \\ \text{PE}(x, y, 2j+1+D/2) &= \text{cos}(y/10000^{\frac{4j}{D}}), \end{aligned} \tag{3.3}$$

where x and y represent the horizontal and vertical coordinates of the token, D denotes the total dimension of position embeddings, i and j specifies the location of the value within each individual position embedding.

Similar to the absolute position embeddings, the sinusoidal position embeddings have the same size as the patch embeddings, and they are directly added together before being fed into the encoder. However, the sinusoidal position embeddings are fixed. In other words, they do not get updated during the training.

3.4.3. Relative position embeddings

Another line of method is the relative position embeddings [30, 31, 40]. Instead of directly adding to the patch embeddings, relative position embeddings are used in the self-attention layer. A self-attention operation with 2D relative position embeddings [30] can be calculated as:

$$\mathbf{y_{ij}} = \sum_{a=0}^{i} \sum_{b=0}^{j} \text{softmax}_{ab}(\mathbf{q}_{ij}^{T} \mathbf{k}_{ab} + \mathbf{q_{ij}^{T}} \mathbf{r_{a-i,b-j}}) \mathbf{v_{ab}}, \tag{3.4}$$

in which $\mathbf{r_{a-i,b-j}}$ is the learned relative position embedding. The relative position embeddings take the relative shift between the query and the key patch into account, both vertically and horizontally.

3.4.4. Learnable Fourier position embeddings

The learnable Fourier method is a recently proposed position embeddings method which matches our research interest. This method uses raw coordinates of the image patches as the inputs, and train a encoding function to extract Fourier features. The Fourier features are calculated as:

$$\mathbf{r_x} = \frac{1}{\sqrt{D}} [\cos \mathbf{x} W_r^T || \sin \mathbf{x} W_r^T], \tag{3.5}$$

where x is the input coordinates, D is the dimension of the feature embeddings, and W_r is the encoding function. The $||$ represents concatenation. The extracted Fourier features are then sent to an MLP layer to get the final position embeddings. The encoding function W_r is initialized by drawing random values from a Gaussian distribution $N \sim (0, \gamma^{-2})$, where γ is a hyperparameter.

3.4.5. Other position embeddings methods

A lot of research focuses on building new methods for position embeddings in the Transformer model. The deep learning models like LSTM [15] can encode positions by nature as it takes the inputs sequentially. Inspired by this idea, the FLOATER [25] is proposed, which uses a continuous dynamic model to encode position information. There is also research that uses complex numbers to encode positions [35]. Using the similar idea of applying complex numbers to the Transformer model, the RoFormer model [32] uses a rotation matrix to encode absolute position information and incorporates relative position information. Another popular idea to improve position embeddings is building a hybrid model by introducing convolutions. For example, CPVT [4] uses convolutions to build positional encoding generators that can encode positions dynamically.

3.5. Variants of ViT

Building upon the original ViT model, there are many variants that enhance the performance. The DeiT model [33] adds a distillation token to the ViT model to implement knowledge distillation and achieve efficient training. The LeViT model [11] proposes a multi-stage architecture for Vision Transformer. The attention mechanism is used as a down-sampling technique in LeViT. Similarly, the Pyramid Vision Transformer (PVT) [37] proposes hierarchical structure for ViT. An improved version named PVTv2 [36] has also been built. The Swin-Transformer model [26] builds a multi-stage hierarchical structure based on the ViT model. This model uses local windows where the local self-attention mechanism is applied. This idea of computing self-attention locally is also used in the Transformer in Transformer (TNT) model [12].

The original ViT model is proposed for the image classification task. However, ViT has been applied to many other visual tasks. For object detection, the detection Transformer (DETR) [2] is a pioneering work. Researchers have also tried to replace the CNN module in traditional object detection models with Vision Transformer. For example, ViT-FRCNN [1] uses ViT as the backbone for object detector. Image segmentation is also a popular task in computer vision, and there are many works that try to apply ViT to this task. SETR [42] considers semantic segmentation tasks from a sequence-to-sequence perspective

and proposes a pure transformer model to encode images as sequential patches. TransUnet [3] applies Vision Transformer to UNet as a powerful encoder for medical image segmentation tasks. ViT can also be applied to many other vision tasks, such as pose estimation [16, 24, 41], action recognition [8, 9, 10], or re-identification [14, 22, 27].

4

Translation equivariance

4.1. Definition

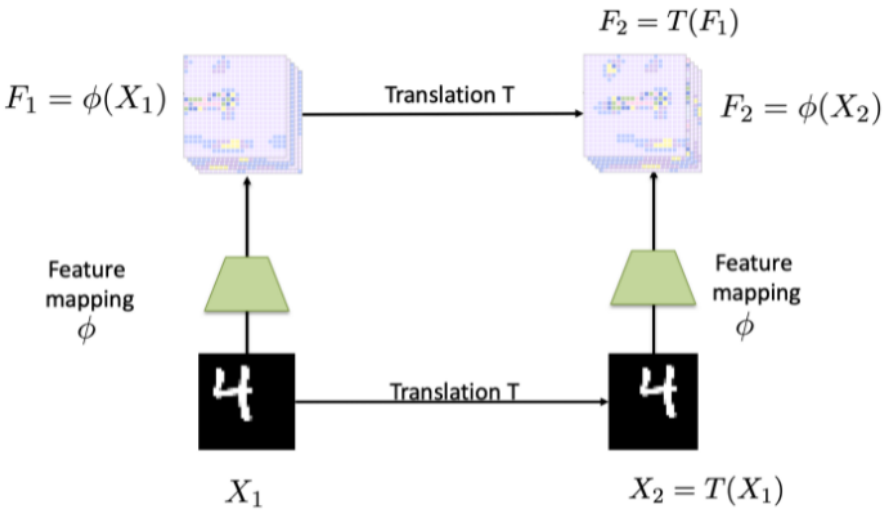


Figure 4.1: An illustration of translation equivariance adapted from an online blog [39]. An input image from the MNIST dataset contains a digit 4. A feature mapping function ϕ is used to get the feature map. If the digit is translated from the left to the right, then the corresponding feature map also has the same translation.

Translation equivariance is an essential property in visual tasks. It denotes the idea that the translation of an object in an image leads to the same translation on its corresponding feature map. Translation equivariance ensures that the visual features do not depend on the absolute position in an image.

4.2. Translation equivariance in CNN

Translation equivariance naturally holds in convolutions, and it is an essential inductive bias in CNN models. This is realized by the weight-sharing mechanism of the convolution kernels. Figure 4.2 illustrates how convolution kernels work. The kernel is used as a feature detector applied to any part of the image without being modified. In other words, the weight w_{i-j} in a convolution kernel, which corresponds to the position i and j, only relies on the relative shift $i-j$ but is not sensitive to the absolute locations of i and j [5]. This ensures that convolutions have translation equivariance. With the usage of the global pooling layer, the CNN models are translation invariant. Therefore, the translation of objects will not influence the final output of CNN models. This property enables CNN models to generalize well and be trained in a data-efficient manner. The translation equivariance and invariance can be utilized

by applying the data augmentation technique, in which we could apply different kinds of transformation like translation, rotation, or scaling to enrich the dataset [28].

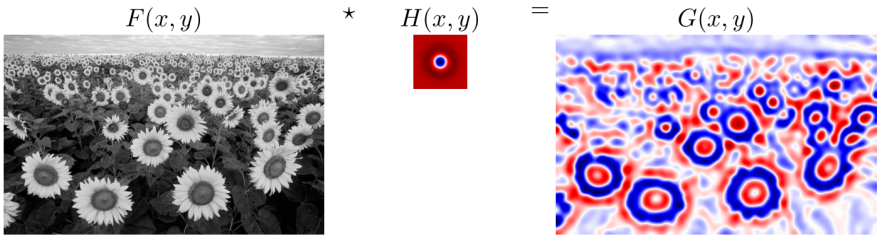


Figure 4.2: An illustration of how convolution kernels work. A kernel is used to detect the same visual features at every part of the image. This is done by sliding the small kernel H over the whole image F to get the feature map G. This figure is adapted from the slides of TU Delft CS4240 Deep learning course.

4.3. Translation equivariance in ViT

Previous work claims that the usage of absolute position embeddings destroys the translation equivariance in ViT models [4]. This is because the absolute position embeddings method gives each image patch a unique position embedding. Therefore the relative position information between two image patches relies on their absolute positions. Our research shows that absolute position embeddings can encode relative location information, enhancing its translation equivariance.

5 Datasets

In this section, we will give an introduction to the dataset that we use in our research. There are two types of datasets. The first one is the designed red-green dataset, which is used to conduct fully-controlled experiments to test some hypotheses. The second one is the realistic dataset, which is used to conduct experiments to compare our proposed methods with previous approaches.

5.1. Red-green dataset

Inspiring by this paper [18], we design red-green datasets to test some hypotheses. The red-green datasets consist of images that only have a red square and a green square, while the background is black. An example dataset can be found in 5.1. The advantage of using red-green datasets is that researchers can inject any features that they want to investigate into the datasets. The datasets are constructed in a target-driven way. In addition, the red-green datasets are light and do not require too much time and resources. In our research, we construct several red-green datasets containing different position features to test different aspects of location information in image data. Compared with realistic datasets, whose images come from the real world and have various visual features, the red-green datasets allow us to test what we want to investigate.

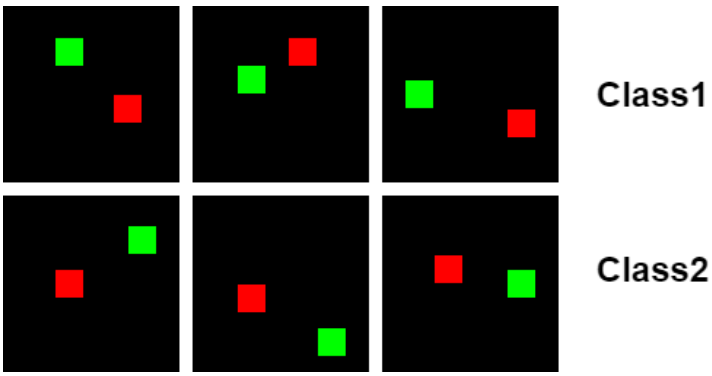


Figure 5.1: An example red-green dataset that is used in our research. This dataset presents a binary classification task. Images in class 1 have a green square to the left of a red square, while images in class 2 have a green square to the right of a red square. This dataset is used to test whether the position embeddings method could encode relative direction information.

5.2. Realistic dataset

To determine whether our proposed methods generalize well and compare with other approaches, we need to test the models on realistic datasets. The four datasets we used in this research are relatively smaller compared with commonly used datasets like ImageNet-1k. The reason is that we want to test the data efficiency of various position embeddings methods. This section gives a brief introduction to the datasets that we use.

5.2.1. CIFAR10 and CIFAR100

CIFAR10 and CIFAR100 [19] datasets are commonly used small-scale image classification datasets in the computer vision research community. Both datasets contain 60000 32x32 color images. 50000 images are used as the training data, while the rest 10000 are used for testing. The CIFAR10 dataset has 10 labels, and each label has 6000 images. On the other hand, the CIFAR100 dataset has 100 labels, and each label has 600 images. Figure 5.2 displays some sample images in the CIFAR10 datasets.

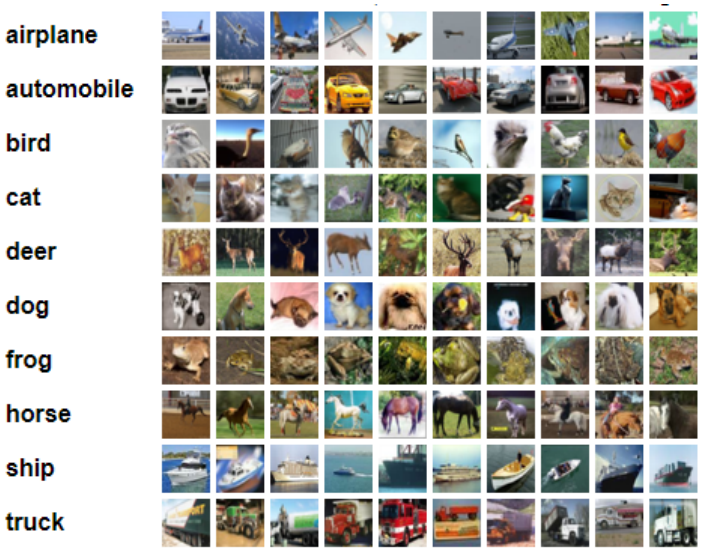


Figure 5.2: Sample images in the CIFAR10 dataset which is present in this paper [19]

5.2.2. Oxford flower 102

The Oxford flower 102 dataset [29] is a widely used image classification dataset. It contains 102 different types of flowers found in United Kingdom. It is an imbalanced dataset as different flower categories have various amounts of image samples. In addition, the images in this dataset have different sizes. In our research 6552 images are used for training, while 818 images are used for testing. Figure 5.3 displays some sample images.

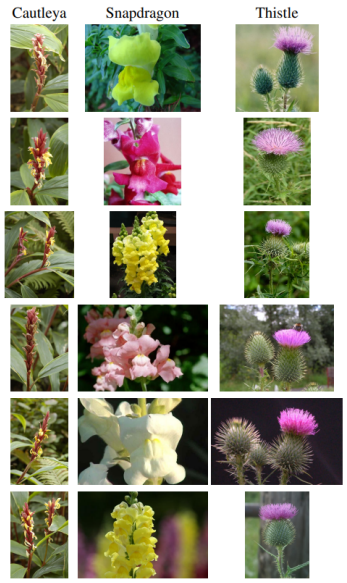


Figure 5.3: Sample images in the Oxford flower 102 dataset which is present in this paper [29]

5.2.3. ImageNet-tiny

The tiny ImageNet dataset [20] is a subset of the ImageNet dataset. This dataset contains 200 categories, and each category has 550 64x64 images. Among them, 500 images are used for training, while 50 images are used for testing. Therefore, there are in total 100000 images in the training dataset and 10000 images in the testing dataset. The ImageNet-tiny dataset was originally used for a course challenge. Figure 5.4 displays some sample images.

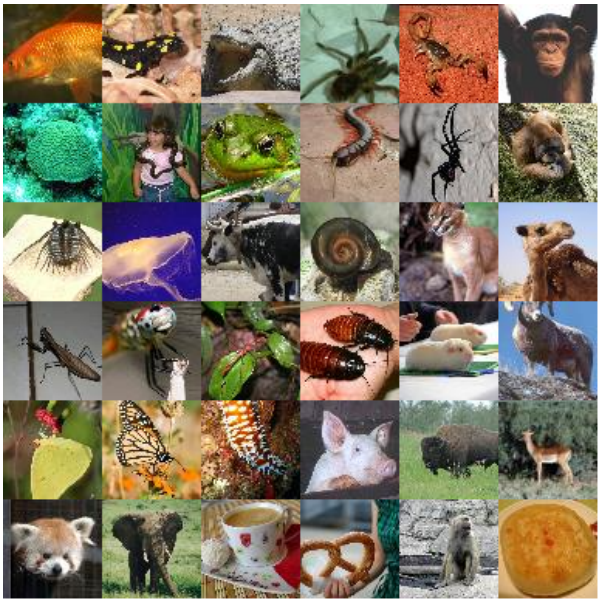


Figure 5.4: Some sample images in the ImageNet-tiny dataset which is present in this paper [20]

6

Conclusion

In this paper, we investigate how the location information is encoded in the ViT model. We compare different position embeddings methods and analyze their advantages and drawbacks based on fully-controlled experiments and feature-level analysis on synthetic datasets. The default learnable absolute position embeddings lack the relative location bias, and they are not translation equivariant. The relative position embeddings are not sensitive to absolute location information, which is crucial in visual tasks. The fixed sinusoidal position embeddings are not flexible enough, while the learnable Fourier position embeddings do not contain much relative location bias. We propose pretraining tasks to enhance the relative location bias of the Fourier method, and we also make the fixed sinusoidal position embeddings learnable. The results on four realistic datasets show the effectiveness of our new methods and suggest that they are competitive compared with a similar state-of-the-art approach.

In this research, we only focus on the original ViT model. For future research, we could investigate whether our proposed methods generalize to other ViT-based models like Deit [33], Levit [11] or Swin-transformer [26]. We could also research whether our methods also contribute to convolution-enhanced ViT models.

References

[1] Josh Beal et al. “Toward transformer-based object detection”. In: *arXiv preprint arXiv:2012.09958* (2020).

[2] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *European conference on computer vision*. Springer. 2020, pp. 213–229.

[3] Jieneng Chen et al. “Transunet: Transformers make strong encoders for medical image segmentation”. In: *arXiv preprint arXiv:2102.04306* (2021).

[4] Xiangxiang Chu et al. “Conditional positional encodings for vision transformers”. In: *arXiv preprint arXiv:2102.10882* (2021).

[5] Zihang Dai et al. “Coatnet: Marrying convolution and attention for all data sizes”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[6] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[7] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[8] Mohsen Fayyaz and Jurgen Gall. “Sct: Set constrained temporal transformer for set supervised action segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 501–510.

[9] Kirill Gavrilyuk et al. “Actor-transformers for group activity recognition”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 839–848.

[10] Rohit Girdhar et al. “Video action transformer network”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 244–253.

[11] Benjamin Graham et al. “LeViT: a Vision Transformer in ConvNet’s Clothing for Faster Inference”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 12259–12269.

[12] Kai Han et al. “Transformer in transformer”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[13] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[14] Shuting He et al. “Transreid: Transformer-based object re-identification”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15013–15022.

[15] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[16] Lin Huang et al. “Hand-transformer: non-autoregressive structured modeling for 3D hand pose estimation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 17–33.

[17] Md Amirul Islam, Sen Jia, and Neil DB Bruce. “How much position information do convolutional neural networks encode?” In: *arXiv preprint arXiv:2001.08248* (2020).

[18] Osman Semih Kayhan and Jan C van Gemert. “On translation invariance in cnns: Convolutional layers can exploit absolute spatial location”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 14274–14285.

[19] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009).

[20] Ya Le and Xuan Yang. “Tiny imagenet visual recognition challenge”. In: *CS 231N* 7.7 (2015), p. 3.

[21] Yang Li et al. “Learnable Fourier Features for Multi-Dimensional Spatial Positional Encoding”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[22] Yulin Li et al. “Diverse part discovery: Occluded person re-identification with part-aware transformer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2898–2907.

[23] Tatiana Likhomanenko et al. “Cape: Encoding relative positions with continuous augmented positional embeddings”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[24] Kevin Lin, Lijuan Wang, and Zicheng Liu. “End-to-end human pose and mesh reconstruction with transformers”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1954–1963.

[25] Xuanqing Liu et al. “Learning to encode position for transformer with continuous dynamical model”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6327–6335.

[26] Ze Liu et al. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022.

[27] Hao Luo et al. “An empirical study of vehicle re-identification on the AI City Challenge”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4095–4102.

[28] Divyanshu Mishra. *Translational invariance vs translational equivariance*. Feb. 2020. URL: `https://towardsdatascience.com/translational-invariance-vs-translational-equivariance-f9fbc8fca63a`.

[29] Maria-Elena Nilsback and Andrew Zisserman. “Automated flower classification over a large number of classes”. In: *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. IEEE. 2008, pp. 722–729.

[30] Prajit Ramachandran et al. “Stand-alone self-attention in vision models”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[31] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. “Self-attention with relative position representations”. In: *arXiv preprint arXiv:1803.02155* (2018).

[32] Jianlin Su et al. “Roformer: Enhanced transformer with rotary position embedding”. In: *arXiv preprint arXiv:2104.09864* (2021).

[33] Hugo Touvron et al. “Training data-efficient image transformers & distillation through attention”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 10347–10357.

[34] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[35] Benyou Wang et al. “Encoding word order in complex embeddings”. In: *arXiv preprint arXiv:1912.12333* (2019).

[36] Wenhai Wang et al. “PVT v2: Improved baselines with Pyramid Vision Transformer”. In: *Computational Visual Media* 8.3 (2022), pp. 415–424.

[37] Wenhai Wang et al. “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 568–578.

[38] Zelun Wang and Jyh-Charn Liu. “Translating math formula images to LaTeX sequences using deep neural networks with sequence-level training”. In: *International Journal on Document Analysis and Recognition (IJDAR)* 24.1 (2021), pp. 63–75.

[39] Christian Wolf. *What is translation equivariance, and why do we use convolutions to get it?* Oct. 2020. URL: `https://chriswolfvision.medium.com/what-is-translation-equivariance-and-why-do-we-use-convolutions-to-get-it-6f18139d4c59`.

[40] Kan Wu et al. “Rethinking and improving relative position encoding for vision transformer”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10033–10041.

[41] Sen Yang et al. “Transpose: Keypoint localization via transformer”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 11802–11812.

[42] Sixiao Zheng et al. “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 6881–6890.