J. Vis. Commun. Image R. 89 (2022) 103664

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Full Length Article

The encoding method of position embeddings in vision transformer[☆]

Kai Jiang, Peng Peng, Youzao Lian, Weisheng Xu [*]

Department of Control Science and Engineering, College of Electronic and Information Engineering, Tongji University, Caoan Highway No. 4800, Shanghai 201804, China

ARTICLE INFO

Keywords:
Vision transformer
Position embeddings
Gabor filters

ABSTRACT

In contrast to Convolutional Neural Networks (CNNs), Vision Transformers (ViT) cannot capture sequence ordering of input tokens and require position embeddings. As a learnable fixed-dimension vector, the position embedding improves accuracy while limiting the migration of the model between different input sizes. Hence, this paper conducts an empirical study on position embeddings of pre-trained models, which mainly focuses on two questions: (1) What do the position embeddings learn from training? (2) How do the position embeddings affect the self-attention modules?

This paper analyzes the pattern of position embedding in pre-trained models and finds that the linear combination of Gabor filters and edge markers can fit the learned position embeddings well. The Gabor filters and edge markers can occupy some channels to append the position information, and the edge markers have flowed to values in self-attention modules. The experimental results can guide future work to choose suitable position embeddings.

1. Introduction

Transformers have recently drawn significant attention in natural language processing (NLP) because of their competitive performance and superior capability in capturing long-range dependencies [1–3]. Inspired by the Transformers' success in NLP, the Vision Transformer (ViT) [4] splits an image into patches and provides images' linear embedding sequences as an input to a standard Transformer [4]. Networks similar to the Transformer structure have also been proposed [5–7].

The transformers utilize the Multi-Head Self-Attention module (MHSA) and Position-wise Feed-forward module (FFN) to extract features from the input sequences [1], while the self-attention module is permutation-invariant, unable to leverage the tokens' order in the input sequences [8]. Moreover, the FFN is a position-independent Multi-layer Perception (MLP) [1]. Thus, a learnable 1D matrix, named position embedding (PE), is added to the input sequences [9], where the PE has equal length and the same dimension as the input sequence [4], limiting the length extension of the input sequence.

Previous works have investigated PE as a representation of each position in the input sequences [4,10], with PE exposing the position information to the model [11]. The position information learned by PE can be visualized through similar maps of different locations [4,12]. As illustrated in Fig. 1, the position has the maximum cosine similarity with itself, visualized as the brightest point in the similarity maps, with the latter point having high similarity to the points around it.

According to the position information, closer locations should have more similar position embeddings [4]. Although the similarity of position embeddings should be monotonically decreasing as distance increases, it periodically changes as the distance increases in the similarity maps, and the period decay size also varies while training the hyper-parameters [4,13,14]. Hence, the regular feature opposes the position information assumption stating that closer patches tend to have more similarities. Thus, PE learns not only position information.

The periodic patterns in the PE similarity maps are also found in some NLP models [12]. Nevertheless, research has not been conducted on the reasons for the periodic patterns and what information is learned by the PE. Therefore, it is necessary to study the encoding method of the learned PE to explore the reasons for the appearance of periodic features. Encoding methods can also obtain the information contained in the PE.

Studying the position encoding method involves choosing the essential channels for the classification results. In other words, some channels may be redundant in the network and can be pruned with a slight accuracy loss [15]. Additionally, noise imposes a perception loss of each channel's value. Given that we cannot obtain the encoding method directly from each channel's value, research on position

[☆] This paper has been recommended for acceptance by Zicheng Liu.

[*] Corresponding author.

E-mail addresses: jkjiang@tongji.edu.cn (K. Jiang), peng.peng@tongji.edu.cn (P. Peng), youzaolian@tongji.edu.cn (Y. Lian), xuweisheng@tongji.edu.cn (W. Xu).

https://doi.org/10.1016/j.jvcir.2022.103664
Received 25 February 2022; Received in revised form 22 September 2022; Accepted 9 October 2022
Available online 14 October 2022

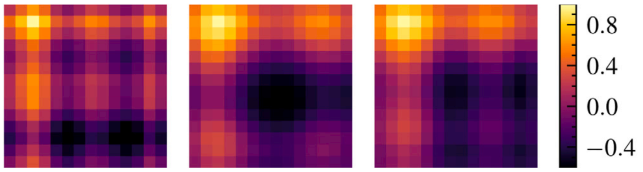

Fig. 1. The cosine similarity of the position embeddings of different models. From left to right: the similarity at index 16 of the position sequence with other position embedding from ViT-B [4], Deit-B [13], Cait-S [14].

embedding has been hampered, leading to a search for alternative PE approaches, e.g., the relative position [10,11,16] and the convolution position embeddings [17,18].

Spurred by the observations mentioned above, this work focuses on the geometry of each channel in the position embeddings rather than the actual number of each location. Specifically, we use a function to model the relationship between shapes and positions and employ this function to approximate the learned position embeddings in the pre-trained model. The accuracy results of the test are set to highlight that our model can fit the PE well in the pre-trained models.

This paper's contributions can be summarized as follows:

- Analyzing each PE channel in the pre-trained models and summarizing the types of shapes. Moreover, we propose an encoding method that fits the learned PE from the coordinates of each position.
- Analyzing the patch and position embeddings and finding that the position embeddings occupy some channels of patch embeddings.
- Analyzing the PE flow in the self-attention modules and finding that the query, key, and value matrices need position embeddings.

2. Related works

2.1. Vision Transformers

As illustrated in Fig. 2, the images are cut into non-overlapping N patches, each projected to a d-dimension vector via a linear layer. To accomplish image classification, a learnable d-dimension vector is concatenated to the patch embeddings, and a learnable matrix called position embeddings is added to the patch embeddings to capture the order information between patches. These patch embeddings are then applied to M standard Transformer encoders including MHSA and FFN. Finally, the learned class token is extracted as a vector for classification.

After ViT, various position embedding methods have been proposed to capture sequence information. According to the different positions and the way of joining, position embeddings can be classified into three types: Absolute Position Embedding (APE), Relative Position Embedding (RPE), and Convolution Position Embedding (CPE).

APE: In ViT, the APE adds the learnable position embeddings into patch embeddings before the encoders:

$$x = p + pos \qquad p, pos, x \in \mathbb{R}^{(N+1)\times d} \tag{1}$$

where x is the Transformer's input, p denotes the embedding sequence of all patches, pos is the position embedding, N is the length of the patch sequences, and d is the embedding dimension. The pos parameter can be a set of learnable position embeddings corresponding to the patch embedding [4] or two sets of learnable position embeddings [19], one per axis.

RPE: The RPE adds the relative position information into the attention module. The self-attention computes a weighted sum over all values v in the sequence. The attention weights A rely on the pairwise similarity between two elements of the sequence and their respective query q_i and key k_j representations. The parameters q, k, and v are computed from input x by linearly combining every channel.

$$[q, k, v] = xU_{qkv} \qquad U_{qkv} \in \mathbb{R}^{(N+1)\times 3d} \tag{2}$$

$$\begin{aligned} A &= qk^T \qquad A \in \mathbb{R}^{(N+1)\times(N+1)} \\ &= xU_qU_k^Tx^T \\ &= \underbrace{pU_qU_k^Tp^T}_{1} + \underbrace{pU_qU_k^Tpos_k^T}_{2} \\ &\quad + \underbrace{posU_qU_k^Tp^T}_{3} + \underbrace{posU_qU_k^Tpos_k^T}_{4} \end{aligned} \tag{3}$$

$$z = Softmax(\frac{A}{\sqrt{d}})v \qquad z \in \mathbb{R}^{(N+1)\times d} \tag{4}$$

where U_{qkv} is the linear layer to compute q, k, and v, while z is the output of the self-attention modules and FFN's input, when the absolute position embeddings are considered, the A parameter can be divided into four parts.

The RPE value is commonly calculated via a lookup table involving learnable parameters, with a lookup process relying on the relative distance between patches [10]. Hence this method can be extended to sequences of different lengths, but the RPE can slow down the training and testing processing [8].

The location to add the RPE depends on the four APE parts. Shaw et al. [16] removed parts 3 and 4 in Eq. (3), while Huang et al. [11] discarded part 4 of the same equation. Finally, Wu et al. [10] removed parts 2, 3, and 4, and added the relative position embeddings into A in bias mode.

CPE: Unlike NLP, the input sequences of the Transformers are flattened from 2D feature maps. The 2D convolution can capture position information using zero-padding [20]. Hence, some networks use convolution layers to add the position information, with these layers added at the self-attention modules [18], or FFN [7,21], or between two encoder layers [17].

2.2. Gabor filter

Gabor wavelets were introduced by Gabor [22] using complex functions as a basis for Fourier Transform in information theory applications. The original Gabor functions are generated based on a fixed Gaussian distribution while the modulating wave frequency varies. Examples of filters generated from 2D Gabor functions are illustrated in Fig. 3. The Gabor filters are widely adopted in traditional filter design due to their enhanced capability of decomposing the signal's scale and orientation [23,24]. For example, the Gabor filters have excellent spatial locality and directional selectivity [25] and capture spatial information mainly in the horizontal and vertical directions [26].

Combining Gabor and CNN originates from the visualization of convolution kernels [26], where the Gabor filter's output is the CNN's input [27]. It is found that several convolution layers from AlexNet's first layer [28] are similar to Gabor filters and are primarily present in shallow layers, i.e., low semantic levels. Then the Gabor filters are integrated into the convolution filters to reduce training time [29] and improve the network's robustness [30].

3. What is the position embedding?

This section analyzes the geometry of the absolute position embeddings in pre-trained models and finds a model to describe the relationship between coordinates and shapes.

3.1. Shapes of learned position embeddings

Among APE, RPE, and CPE, the APE has the least a priori information and is independent of image inputs, posing an appealing method for further study. Hence, we obtain the learned position embedding from ViT's base model (ViT-B) [31] and DeiT's base model (DeiT-B) [31]. The DeiT-B model employed without a distillation module has the same network structure as the ViT-B model [4]. Their main

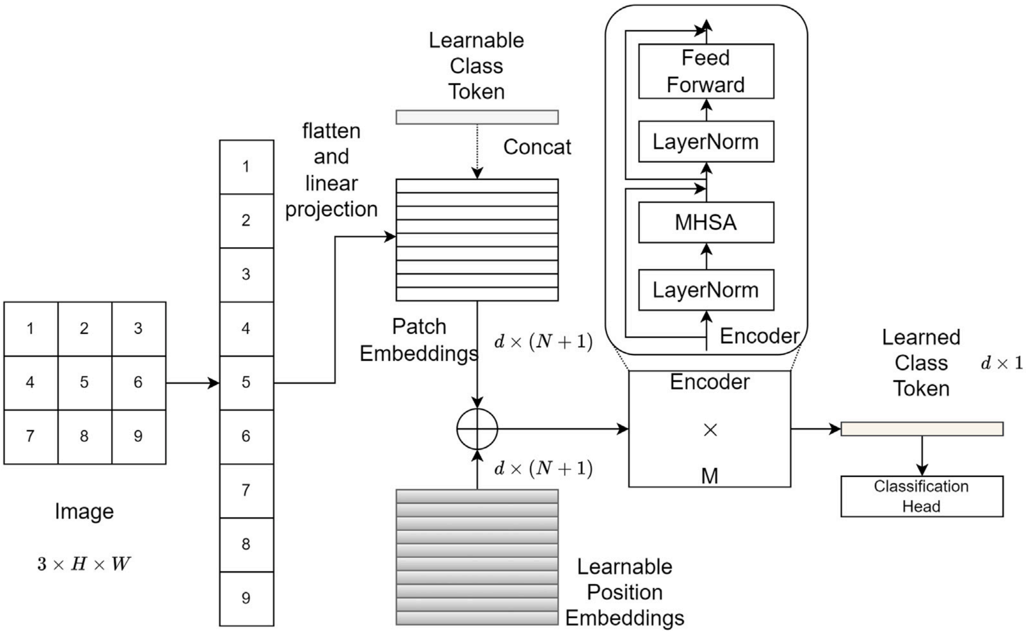

Fig. 2. The network architecture of ViT and DeiT. The DeiT without distillation modules shares the same architecture as the ViT but uses more complex training strategies.

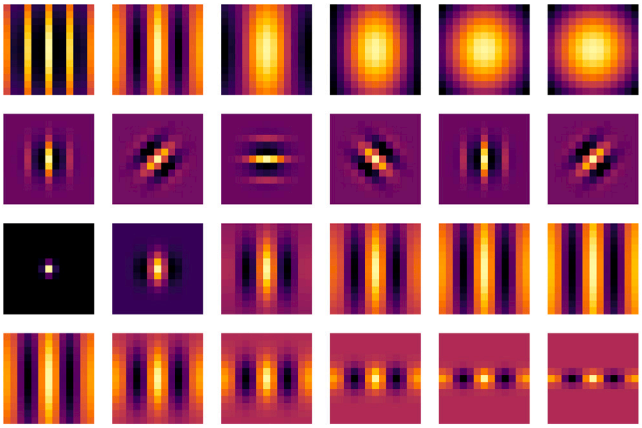

Fig. 3. Examples of traditional 2D Gabor filters with different parameters.

difference is that the training strategies of DeiT are more complex, and specifically, DeiT-B applies more data augmentation methods, such as CutMix [32], Mixup [33], and Label Smoothing [34]. DeiT-B achieves higher accuracy on the ImageNet datasets based on these data augmentation schemes enhancing training, although DeiT-B and ViT-B have the same network architecture and settings.

Furthermore, for visualization purposes, we remove the position of the class token and restore the position sequences to the 2D position maps. The model's patch size is set to 16 × 16, and when the input image size is 224 × 224, the feature map size is 14 × 14. Examples of feature maps from different ViT-B channels are illustrated in Fig. 4.

As depicted in Fig. 4 and ignoring specific values and noise, the geometric shape of the position embedding within a channel can be divided into four types: Chaos, Raster, Striped, and Box.

- **Chaos** means no obvious pattern exists in the value distribution at different feature map locations. This channel type is more frequent in ViT-B and almost absent in DeiT-B. Considering the advantages of DeiT-B over ViT considering test accuracy, a reasonable guess is that these channels are untrained or not fully trained.

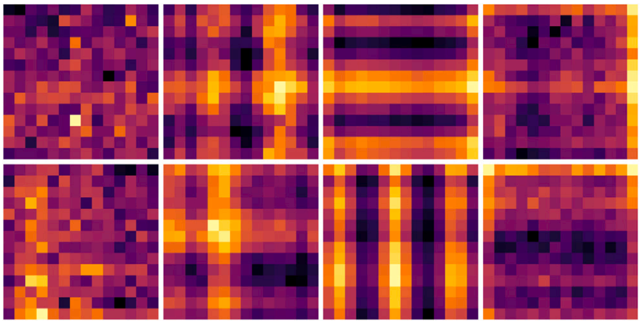

Fig. 4. Four types of position embeddings from ViT-B. From left to right: Chaos, Raster, Striped, and Box.

- **Raster** is similar to the sum of horizontal and vertical stripes, where the position embeddings have different values in the horizontal and vertical directions. This position channel usually has a maximum or minimum value, which decreases periodically as the distance from the maximum point increases.
- **Striped** denotes that the differences in the values within the position map are mainly in the x-axis or the y-axis direction. In all position embedding channels, the oblique stripes are almost absent, i.e., these position embedding channels mainly focus on the horizontal or vertical directions. Usually, the stripes do not have the same brightness in one position channel, implying an attenuation between the stripes.
- **Box** means that the position maps of the boundaries and interior have a significant division. The channel maps have no apparent pattern in this case, but the edges are highlighted. These channels focus on the information of the edges, with the position embeddings' shape corresponding to the convolution layer's zero-padding [20], making the boundary stand out.

In general, the APE shape has the following characteristics:

- The encoding method contains both periodic and decay components.
- The encoding is oriented mainly in the horizontal and vertical directions.

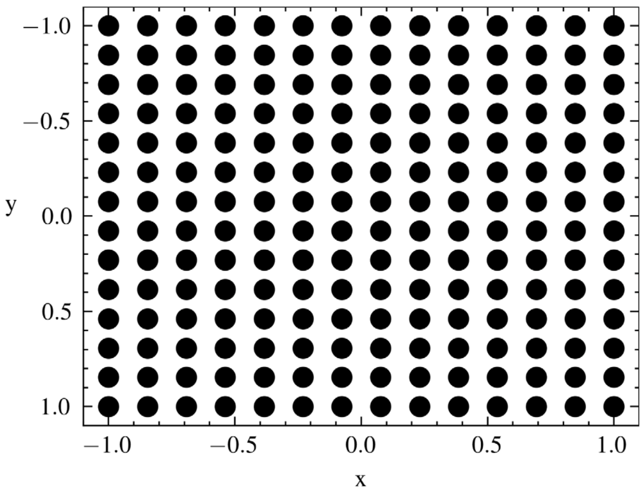

Fig. 5. Coordinates of each position. The boundaries of the feature map are set to 1 or −1, and the position intervals inside the feature map are equally distributed according to the number of patches.

- The values at the edges and the internal of the feature maps have gaps.

These characteristics can be the assumptions to model location information.

3.2. Models of learned position embeddings

Based on the above visualization and analysis, we attempt to model the last three types of position embedding except for the Chaos type.

Initially, we use the 2D coordinates (x_i, y_i) to represent feature maps in the patch at the i_{th} columns and j_{th} rows and deflate them within the interval $[-1, 1]$, as shown in Fig. 5. In other words, the patch's coordinates in the upper left corner are $(-1, -1)$, which is first in the position embedding sequence except for the class-token. The coordinates of the 14_{th} patch in the sequence at the upper right corner of the feature map are set to $(1, -1)$, and the subsequent models are based on these coordinates.

We model the Striped position embedding type utilizing the Gabor function defined by a Gaussian function multiplied by a sinusoidal wave [35]:

$$g(x) = exp(-\frac{(x-x_a)^2}{2\sigma^2}) \cdot exp(j(2\pi\frac{x}{\lambda} + \psi)) \tag{5}$$

where x_a is the center of the receptive field in the spatial domain, λ is the optimal spatial frequency of the filter in the frequency domain, and σ is the standard deviation of the elliptical Gaussian function. The real and the imaginary parts of $g(x)$ are presented in Eqs. (6) and (7):

$$g(x) = exp(-\frac{(x-x_a)^2}{2\sigma^2}) \cdot cos(2\pi\frac{x}{\lambda} + \psi) \tag{6}$$

$$g(x) = exp(-\frac{(x-x_a)^2}{2\sigma^2}) \cdot sin(2\pi\frac{x}{\lambda} + \psi) \tag{7}$$

To avoid operating complex numbers, we model the real part of the Gabor function (Eq. (6)). The Gabor function for the x coordinates is employed to fit the longitudinal stripes, while the Gabor function for the y-coordinates can be used to fit transverse stripes.

We choose the horizontal and vertical Gabor functions to model the Centroid type of the position embedding. To preserve the position embeddings of the Stripe type and oppose the traditional 2D Gabor filter, we add the horizontal and vertical Gabor filters rather than multiply them [36].

Moreover, we use the following forms to model the Box type of position embedding. As presented in Eq. (8), we set only one edge of the feature map to one and the other positions to zero. Each matrix in Eq. (8), from left to right, represents the left, right, upper, and lower bounds of feature maps, respectively. The number of rows and columns of these four matrices is equal to the length and width of the feature map. These four channels mark the different edges of the position map.

Table 1
Performance of the generated and original learnable position embeddings on ImageNet Test Datasets.

Model	#Param.	Method	Top 1 Acc(%)	Top 5 Acc(%)
ViT-B	151K	LE	75.4	92.8
ViT-B	10.7K	G+E	77.0(+1.6)	94.3
ViT-B	7.6K	G	76.4(+1.0)	93.1
ViT-B	4.6K	E	60.5(−14.9)	82.8
DeiT-B	151K	LE	81.8	95.6
DeiT-B	10.7K	G+E	81.6(−0.2)	95.4
DeiT-B	7.6K	G	80.7(−1.1)	94.9
DeiT-B	4.6K	E	72.0(−9.8)	90.2

#Param. refers to the number of parameters related to position embeddings,
LE is the learnable position embedding,
G refers to the linear combination of Gabor filters channels,
E denotes the linear combination of edge marker channels,
G+E refers to the linear combination of Gabor filter channels and edge marker channels.

$$\left[\begin{array}{c|c} 1 & \\ \vdots & 0 \\ 1 & \end{array}\right], \left[\begin{array}{c|c} & 1 \\ 0 & \vdots \\ & 1 \end{array}\right], \left[\begin{array}{c} 1 \quad \cdots \quad 1 \\ \hline \\ 0 \end{array}\right], \left[\begin{array}{c} 0 \\ \\ \hline 1 \quad \cdots \quad 1 \end{array}\right] \tag{8}$$

Then, we combine these six channels (horizontal Gabor channel, vertical Gabor channel, top edge channel, bottom edge channel, left channel, and right edge channel) with a linear layer.

$$\begin{aligned} PE(x,y) &= g(x) * W_x + g(y) * W_y + E * W_E \\ &= exp(-\frac{(x)^2}{2\sigma_x^2}) \cdot cos(2\pi\frac{x}{\lambda_x} + \psi_x) * W_x \\ &+ exp(-\frac{(y)^2}{2\sigma_y^2}) \cdot cos(2\pi\frac{y}{\lambda_y} + \psi_y) * W_y \\ &+ E * W_E + B \end{aligned} \tag{9}$$

where W_x and W_y are the weights of the Gabor function in the x and y directions, E is the images' four edge markers, W_E are the weight of each edge marker, and B is the bias of this channel. Ultimately, modeling the Centroid, Stripe, and Box types of position embedding is achieved by controlling the weights W_x, W_y, and W_E.

3.3. Fitting of learned position embeddings

We set each parameter in Eq. (9) as a learnable parameter added to the patch embedding to evaluate our models' effectiveness in replacing the original position embedding. The corresponding architecture is illustrated in Fig. 6.

Then, the coordinates and the learnable Gabor parameters are employed to generate the Gabor channels separately in the x and y directions. The Gabor and edge channels are linearly combined to generate a position embedding channel. After flattening to 1D and providing an appending learnable position for the class token, the generated position embedding is added to the patch embedding, similar to the original position embedding.

In this experiment, the whole vision transformer becomes a black box as a tool to verify the model fitting ability. As a result, all parameters from the pre-trained model ViT-base [4] and DeiT-base [13] are frozen, except for the position-related parameters. In other words, only the parameters $\lambda, \sigma, \psi, W_x, W_y, W_E$, and B in Eq. (9) are trained.

After 10 epochs of training, the results are reported in Table 1, highlighting that accuracy can be improved by 1.6% by simply replacing the learnable embeddings with the linear combination of Gabor

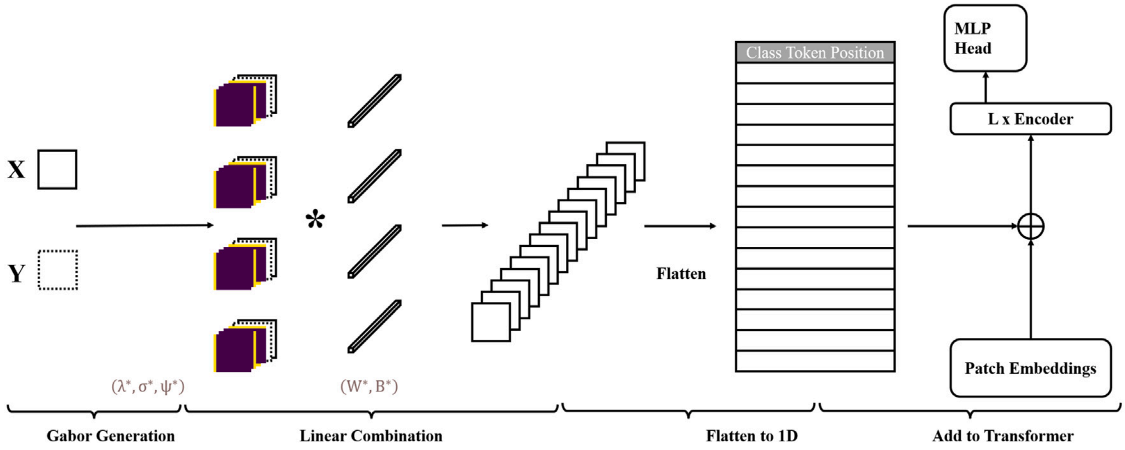

Fig. 6. Illustration of generating position embedding process.

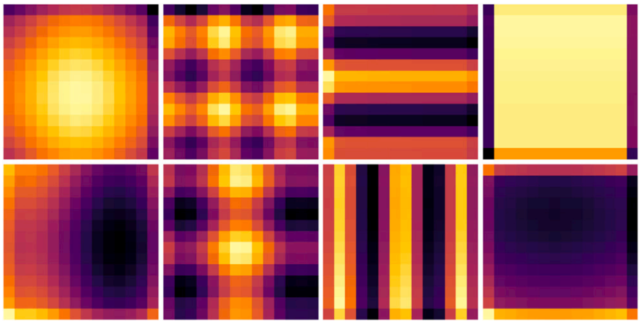

Fig. 7. Some channels of generated position embedding.

filters and edge markers without changing other parameters in the ViT-B models. Additionally, the generated positional embeddings result in a 0.2% accuracy loss compared to the learnable positional embeddings in the DeiT-B model. Fig. 7 depicts some channels of the generated position embeddings. By visualizing the different channels in the fitting position embeddings, we find that the Chaos channels in ViT-B have disappeared. Moreover, the learned position embeddings in DeiT-B have fewer Chaos channels. Therefore. positional embedding replacement can improve the test accuracy of ViT-B, but not DeiT-B, imposing the Chaos channels to disappear, i.e., the Chaos channels are undertrained positional embeddings, and the training strategy used by ViT-B prevents the model from converging.

Meanwhile, the results in Table 1 reveal that neither the Gabor filters nor edge markers can be fully adapted to the position embeddings. Compared with edge markers, the Gabor filters play a more critical role in position embeddings.

Regarding visualization and accuracy, the generated position embedding employing a linear combination of Gabor functions and edge markers can effectively replace the original position embedding. In other words, the position embedding mainly contains the Gabor function in the x and y directions along with the four boundaries' identification. In this case, the position embedding does not represent location information for every patch, but a group of global filters, covering all positions in each channel.

The periodic patterns in the similarity maps can also be explained, as these maps do not indicate the similarity of each patch position but the similarity of each position in the global filters. These global filters involve an encoding method that contains the periodic component. Although position embedding is more like a global filter than embedding each location, we still name it *Position Embedding* to avoid inconsistency.

Table 2
Types of images for models with or without position embeddings.

	MWPE: True	MWPE: False
MW/OPE: True	Ignored	Mislead
MW/OPE: False	Correct	Useless

4. How the position embedding works?

This section explores the position embeddings flow in the self-attention modules from a case perspective.

4.1. The process of examples selection

The position embeddings are a plug-in to enhance the Transformers' ability to capture location information. Transformers without position embeddings are still effective in vision tasks but have lower testing accuracy than those with position embeddings. Hence, the MHSA and FFN are powerful enough to capture critical features without adding position information for most images, as these images are meaningless for analyzing the flow of position embeddings in self-attention modules.

In this work, we removed the position embeddings, fine-tuned the remaining parameters based on the pre-trained models, and obtained the models without PE. However, the channels from the models neglecting PE do not correspond to the channels with PE, as the linear projection for the patches has changed. Hence, we freeze all model parameters that do not involve PE and re-add the learnable APE into models. After retraining the position embedding parameters, we find that adding PE improves accuracy without changing other model parameters. Hence, we have two model types: models with position embeddings (MWPE) and models without position embeddings (MW/OPE) that differ only in adding the position embeddings.

Based on the test results of MEPE and MW/OPE, the images of the test set can be divided into four types: Ignored, Misled, Correct, and Useless. Table 2 highlights that the Ignore type represents images classified correctly with or without position embedding, and the Correct class refers to images classified correctly and incorrectly without position embedding. The Misdirected type means that position embedding wrongly directs the classification result of the test image, while the Useless type means that adding position embedding does not affect the results. Misdirected and Corrected types of images can better represent the role of position embedding and therefore are objects gaining our attention.

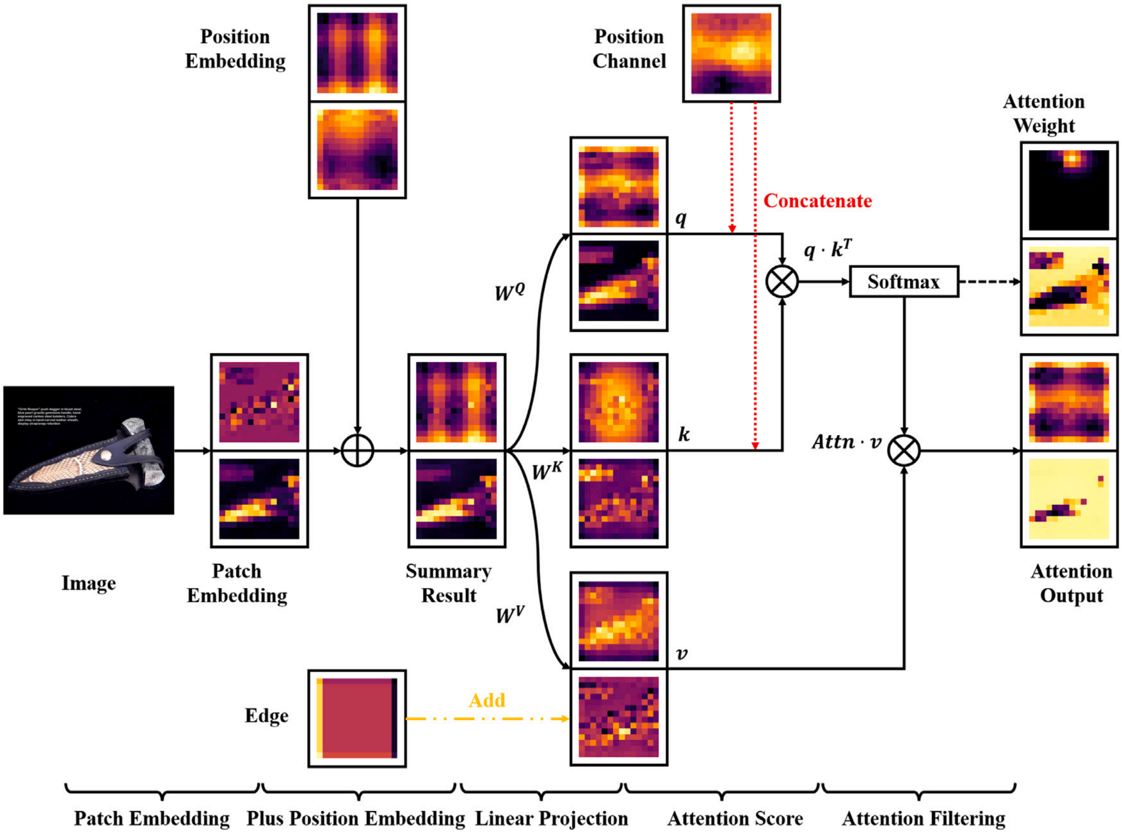

Fig. 8. Position embeddings flow in the self-attention module. Removing the class token and recovery from 1D sequences into 2D feature maps for visualization. Each rectangular box represents the channel in which the location information dominates, and below the box is the channel in which the picture information dominates.

4.2. Position embeddings in patch embedding

We choose the Correct type of image to visualize its intermediate feature maps in the network operations, as illustrated in Fig. 8. Initially, the processing flow splits the image into patches, which are linearly projected into Patch Embeddings. From the latter, we have selected some channels for visualization in Fig. 8, vaguely observing the original object's shape from the position embeddings' feature maps.

Then the position embeddings are added to the patch embeddings to obtain the Summary Result, highlighting that some channels become the position embeddings' shapes, and others remain in their original shape. This position embedding phenomenon occupying a channel was found in past studies but was considered to deteriorate by the visual transformers [37]. At this time, the input x of the vision transformer becomes the concatenated patch embedding p and position embedding pos.

$$\begin{aligned} & x = p + pos = [p_m, pos_m] \\ & p, pos, x \in \mathbb{R}^{N\times d}, p_m \in \mathbb{R}^{N\times(d-m)}, pos_m \in \mathbb{R}^{N\times m} \end{aligned} \tag{10}$$

where d is the dimension of the patch and position embeddings and m is the number of channels that position embedding dominates. In other words, even though pos is added to the patch embedding p, p and pos are still separate and apart in different channels.

According to the above analysis, the position embedding can be decomposed into Gabor filters and edge markers, i.e., the transformer input is a mixture of feature maps from images and global filters.

4.3. Position embeddings in attention weight

In Eq. (10), the transformer input is the concatenation of the patch and position embeddings. Then, the linear combination of all channels generates q, k, and v in the attention module. Including position embeddings by summation or concatenation is equivalent to employing a linear layer in self-attention modules.

However, the linear combination U_{qkv} only changes the order of the channels without mixing the position embedding and patch embedding, i.e., the position and patch embedding channels are still separate, as illustrated in the upper box of q and k in Fig. 8.

Then, the attention weight A is expressed as Eq. (11), with the position channels separating from patch channels.

$$\begin{aligned} A &= [q^1, pos_q^2, q^3, pos_q^4]\cdot[k^1, k^2, pos_k^3, pos_k^4]^T \\ &= \underbrace{q^1\cdot(k^1)^T}_{1} + \underbrace{pos_q^2\cdot(k^2)^T}_{2} \\ &+ \underbrace{q^3\cdot(pos_k^3)^T}_{3} + \underbrace{pos_q^4\cdot(pos_k^4)^T}_{4} \end{aligned} \tag{11}$$

where q^1, q^3, k^1, and k^2 are the block channels of q or k, containing patch information, and pos_q^3, pos_q^4, pos_k^3, and pos_k^4 are the block channels of q or k, mainly containing the position information. The numbers in the superscript represent only the order and not the specific number of channels. In Eq. (11), the results of qk^T can be divided into four parts depending on the channel indexes of the position embedding within q and k. The first part, i.e., $q^1(k^1)^T$ represents that q and k channels originate from the patch embedding. Moreover, the $pos_q^2\cdot(k^2)^T$ and $q^3\cdot(pos_k^3)^T$ represent that one channel originates from the position embedding while the other is from the patch embedding. The parts 2 and 3 of Eq. (11) computes the patch and position embeddings similarity. The $pos_q^4\cdot(pos_k^4)^T$ refers to the similarity of the two channels solely involving position channels, while the last three parts of qk^T introduce the position filter within these channels of A by calculating similarity.

Unlike Eq. (3), the dimension of the four parts in Eq. (11) is less than the Transformer's dimension d, i.e., the values of q^2 and k^3 are set

Table 3
Performance of each part in attention scores.

Part	#Param.	Top 1 Acc (%)↑	Top 5 Acc (%)↑
–	–	68.10	88.5
APE	37.8k	70.02(+1.92)	89.79
4	78.3k	**70.83(+2.73)**	90.17
2,3	69.1k	70.07(+1.97)	89.84
2,3,4	96.8k	70.80(+2.7)	90.26

Table 4
The effect of Edge Markers in v.

Methods	p	q	k	v	#Param.	Top 1 Acc (%)↑	Top 5 Acc (%)↑
–	–	–	–	–	–	68.12	88.50
APE	LE	–	–	–	37.8k	70.02(+1.9)	89.79
Plus	–	–	–	E	13.8k	68.95(+0.83)	89.01
Plus	–	LE	LE	–	907.8k	70.19(+2.07)	89.81
Plus	–	LE	LE	E	921.6k	70.71(+2.59)	90.26
Plus	–	G+E	G+E	–	64.5k	70.29(+2.17)	89.87
Plus	–	G+E	G+E	G+E	96.8k	70.75(+2.63)	90.37
Plus	–	G+E	G+E	E	78.3k	**70.80(+2.68)**	90.26

p refers to the place we add the position embedding, which is the patch embedding before the first transformer layer,
Plus means that the position embeddings are added into the p, q, k, or v.

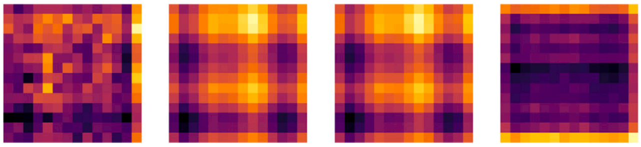

Fig. 9. The linear combination of position embedding in ViT at channel 14. From left to right: pos, $pos \cdot U_q$, $pos \cdot U_k$, and $pos \cdot U_v$.

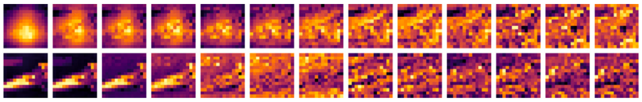

Fig. 10. The first column is the input of the first encoder, and columns 2–13 are the output of layers 1–12. The first row is the position channel, and the second row is the feature channel.

negligibly compared with the position embedding in order to separate the position and patch embedding. This strategy reduces the dimension of the patches' representation and decreases the network's expressiveness. In addition, the position channels must remain separated from the patch channels after linear layer U_{qkv}.

Compared with APE, the RPE methods add the position embeddings into the self-attention module to avoid the linear combination. Recent works always pay attention to parts 2 and 3 but neglect part 4 in Eq. (3) [11,16,38]. For example, the position embedding is only added into k to avoid the channel occupation and linear combination [10,16,39].

$$qk^T = [q^1, q^2, q^3, q^4] \cdot [k^1, k^2, k^3, k^4]^T + [q^1, q^2, q^3, q^4] \cdot [pos_k^1, pos_k^2, pos_k^3, pos_k^4]^T \tag{12}$$

To further explore how part 4 of Eq. (3) impacts accuracy, we develop a method to separate the patch and position channels that concatenate the position channels to the q and k from MW/OPE. We do not concatenate the position channels with patch embeddings before the first encoder because the linear combination U_{qkv} would combine them directly with the position embedding into a patch embedding. Hence the position and patch channels are separate, as presented in Eq. (13).

$$qk^T = [q^1, pos_q^2] \cdot [k^1, pos_k^2]^T = q^1 \cdot (k^1)^T + pos_q^2 \cdot (pos_k^2)^T \tag{13}$$

Moreover, we perform several experiments to validate further the role of parts 2, 3, and 4 of Eq. (11) in improving accuracy. First, we choose a pre-training model from DeiT-Tiny (DeiT-T) as the baseline, removing the model's positional embedding and fine-tuning it. The DeiT-T model without position embeddings achieves 68.1% accuracy on the ImageNet [40] validation set, as shown in Table 3. Then, we freeze all parameters in the baseline and add position embeddings back into the model.

Table 3 reveals that adding APE affords a 1.92% improvement in test accuracy compared to baseline but is lower than directly adding parts 2, 3, and 4 into attention modules. Based on the previous analysis, APE adds location information by invading specific channels, but this method does not work well. In other words, adding location information through the APE method is not as straightforward as the RPE method.

In RPE methods, adding part 4 to the attention mechanism can achieve similar results as adding parts 2, 3, and 4, while adding only part 4 achieves the best results. In addition, the model's accuracy with only parts 2 and 3 is 0.76% inferior to the model with part 4. This proves that the input independent of part 4 is the positional embedding part that improves accuracy.

4.4. Position embedding in value matrix

The absolute position embedding joins the location information by dominating a part of the channels, so the location information must flow into the value matrix in the attention mechanism. This section analyzes the flow of location information in the value matrix.

Observing the two feature maps of v in Fig. 8, the main difference is that the top picture has smaller values at the feature maps' edge. It shows that the edge markers flow into the v through linear layer U_{qkv}. In order to show the linear layer's effect, the linear combination of position embeddings without patch embeddings is illustrated in Fig. 9. Compared with $pos \cdot U_q$ and $pos \cdot U_k$, $pos \cdot U_v$ is similar to the Box type.

Based on the above shape analysis, we propose the hypothesis that the edge markers in the absolute position embeddings flow into the value matrix of the self-attention module. To test this hypothesis, we conduct a series of experiments on the previously mentioned baseline, with the corresponding results reported in Table 4. First, we only add the edge markers to v, achieving an accuracy improvement of 0.83% compared to the baseline without location information. We also add the complete relative position embedding in q and k. Indeed, adding both learnable and generated relative position embeddings in q and k improves accuracy by adding the edge marker in v. Moreover, the combination of Gabor filters and edge markers in v is less effective than only adding edge markers because the edge markers contained in the position embeddings flow to v.

In summary, adding relative position embedding to the attention scores is not sufficient; therefore, it is necessary to add edge markers to v. This explains why the accuracy of the RPE model tested in [10] is inferior to the model using RPE and APE.

4.5. Position embedding in every encoder

In CPVT [8], the authors suggest placing 2D convolution layers after each encoder, using convolution instead of APE. Moreover, ViT experiments in [4] claim no significant accuracy difference between placing a position embedding at the beginning of the first encoder and the beginning of each encoder. Hence, adding position embeddings operates across the encoder layers.

According to the above analysis, the position embedding occupies some channels from the patch embedding, which is visualized by presenting the output of each encoder layer (Fig. 10). In Fig. 10, the first and second columns are the first and second encoder layers,

Table 5
Performance of different encoders contains PE.

Positions	#Param.	Top 1 Acc (%)↑	Top 5 Acc (%)↑
–	–	68.10	88.5
APE	37.8k	70.02 (+1.92)	89.79
1–3	19.6k	70.38 (+2.28)	89.87
1–5	32.6k	70.64 (+2.54)	90.12
1–6	39.2k	70.73 (+2.63)	90.16
1–9	58.8k	70.79 (+2.69)	90.32
1–12	78.3k	**70.80 (+2.70)**	90.26

respectively, indicating that the position and patch channels maintain the original shape in the shallow layer and obtain the semantic shape at a deeper level.

Table 5 presents the results after adding the generated position embeddings in the encoder layers of different depths. The model accuracy increases by including position embeddings in more encoder layers. However, the accuracy improvement when adding positional embedding in a deep encoder is relatively smaller than submitting positional embeddings in a shallow encoder.

This finding confirms that Gabor-shape filters appear only in the shallow layer [29], i.e., the position embedding as a global filter only works in shallow layers capturing shallow features, like shapes and edges.

5. Experiment

5.1. Datasets and experiment settings

All experiments utilize PyTorch [41] and the Timm library [31], and our method is evaluated on the ImageNet dataset [40]. Considering training, we adopt the schedule employed in the recent vision Transformers model DeiT [13], i.e., the AdamW optimizer with a weight decay of 0.05 [42], 512 batch size, 0.1 label smoothing, and stochastic depth with survival rate probability of 0.9. Additionally, we use same data augmentation methods as in Deit: CutMix [32], Mixup [33], and random augmentation [43]. Opposing to DeiT, we do not utilize a warmup epoch. All models are trained with 2 NVIDIA TITAN XP GPUs.

Next, we present the implementation details of several experiments conducted in this paper.

Fitting position embeddings in pre-training models: This experiment utilizes a linear combination of Gabor filters and edge markers to fit the learned position embedding in pre-trained models. We exploit the pre-trained base model parameters of ViT [4] and DeiT [13] and freeze the independent parameters of the position embedding. Moreover, we remove the original position embedding and add the generated position embedding at the original locations. After 10 training epochs with a 0.001 learning rate, we obtain the results presented in Table 1.

The flow of the position embeddings in attention module: The tiny model of DeiT [13] is used as a baseline. This experiment removes the position embedding and trains the remaining parameters utilizing a learning rate of 1×10^{-5} for 50 epochs. The top-1 accuracy of the ImageNet validation set is 68.1%, i.e., 4.1% less than the original model. Then we freeze all model parameters and train our newly added parameters related to the position embedding. The initial learning rate is set to 1×10^{-3} and the minimized learning rate is set to 1×10^{-5}. The results after 50 epochs of training are presented in Table 4.

Position Embeddings in every encoder: These experiments aim to verify the impact of the position embeddings' depth on a network. These experiments still use the fine-tuned DeiT-T model without position embeddings as the baseline. Then we add generative positional embedding in the encoder layers of different depths. The generative position embeddings include Gabor functions, edge markers in q and k, and edge markers in v. The training strategy remains the same as in the previous section, and the corresponding results are reported in Table 5.

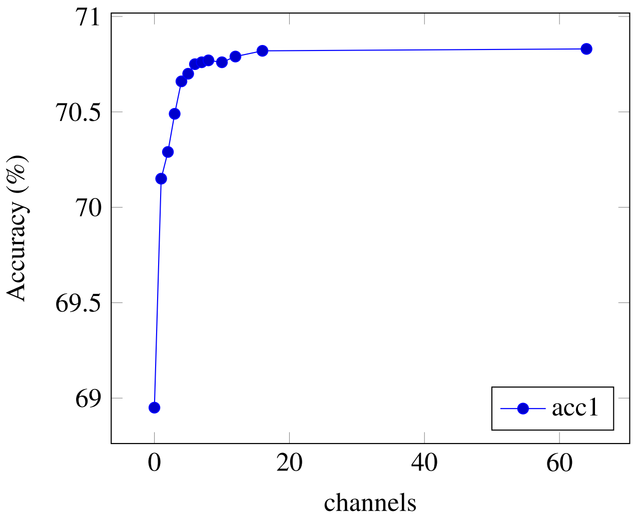

Fig. 11. Different channels of position embeddings concatenating with q and k.

5.2. Ablation study

This section verifies the effect of our parameter settings on the generative positional embedding. These experiments also employ the DeiT-T model without position embeddings as the baseline, similar to the trials investigating the position embedding flow. Furthermore, the training schedule is the same as in the previous experiments, and all experiments add the edge markers in v. In this experiment, we consider the following cases.

The channel number of concatenation: We examine the interplay between the different position channels and the performance attained. As illustrated in Fig. 11, the more the position channels concatenated to q and k, the higher the accuracy. However, the accuracy increase becomes insignificant when the number of position channels exceeds 16.

6. Conclusion

This paper concludes that the learnable position embedding in vision transformers is a linear combination of Gabor filters and edge markers, with Gabor filters mainly focusing on vertical and horizontal directions. Additionally, the position and patch embeddings are separate even after summation and involve linear layers in the self-attention modules, as the edge markers of the position embeddings flow into the v in the self-attention modules.

Future work will design other modeling methods for position information related to other vision tasks like object detection and video understanding while utilizing the position encoding embeddings proposed in this work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The authors would like to thank the Innovation Program of Science and Technology Commission of Shanghai Municipality under Grant No. 19DZ1209200. The authors would like to express their gratitude to EditSprings (https://www.editsprings.cn/) for the expert linguistic services provided.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5998–6008, URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[2] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for Computational Linguistics, 2019, pp. 4171–4186, http://dx.doi.org/10.18653/v1/n19-1423.

[3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, RoBERTa: A robustly optimized BERT pretraining approach, 2019, CoRR 1907.11692 arXiv:1907.11692.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021, URL https://openreview.net/forum?id=YicbFdNTTy.

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, 2021, CoRR abs/2103.14030 arXiv:2103.14030.

[6] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021, arXiv:2102.12122.

[7] Y. Li, K. Zhang, J. Cao, R. Timofte, L.V. Gool, LocalViT: Bringing locality to vision transformers, 2021, CoRR abs/2104.05707 arXiv:2104.05707.

[8] X. Chu, Z. Tian, B. Zhang, X. Wang, X. Wei, H. Xia, C. Shen, Conditional positional encodings for vision transformers, 2021, arxiv Preprint 2102.10882 URL https://arxiv.org/pdf/2102.10882.pdf.

[9] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y.N. Dauphin, Convolutional sequence to sequence learning, in: D. Precup, Y.W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, in: Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 1243–1252, URL http://proceedings.mlr.press/v70/gehring17a.html.

[10] K. Wu, H. Peng, M. Chen, J. Fu, H. Chao, Rethinking and improving relative position encoding for vision transformer, 2021, CoRR abs/2107.14222 arXiv:2107.14222.

[11] Z. Huang, D. Liang, P. Xu, B. Xiang, Improve transformer models with better relative position embeddings, in: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020, in: Findings of ACL, EMNLP 2020, Association for Computational Linguistics, 2020, pp. 3327–3335, http://dx.doi.org/10.18653/v1/2020.findings-emnlp.298.

[12] Y. Wang, Y. Chen, What do position embeddings learn? An empirical study of pre-trained language model positional encoding, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Association for Computational Linguistics, 2020, pp. 6840–6849, http://dx.doi.org/10.18653/v1/2020.emnlp-main.555.

[13] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, 2020, arXiv preprint arXiv:2012.12877.

[14] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, H. Jégou, Going deeper with image transformers, 2021, arXiv preprint arXiv:2103.17239.

[15] M. Zhu, K. Han, Y. Tang, Y. Wang, Visual transformer pruning, 2021, CoRR abs/2104.08500 arXiv:2104.08500.

[16] P. Shaw, J. Uszkoreit, A. Vaswani, Self-attention with relative position representations, in: M.A. Walker, H. Ji, A. Stent (Eds.), Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers), Association for Computational Linguistics, 2018, pp. 464–468, http://dx.doi.org/10.18653/v1/n18-2074.

[17] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, C. Shen, Twins: Revisiting the design of spatial attention in vision transformers, 2021, arxiv Preprint 2104.13840, URL https://arxiv.org/pdf/2104.13840.pdf.

[18] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, L. Zhang, CvT: Introducing convolutions to vision transformers, 2021, CoRR abs/2103.15808 arXiv:2103.15808.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-end object detection with transformers, in: A. Vedaldi, H. Bischof, T. Brox, J. Frahm (Eds.), Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I, in: Lecture Notes in Computer Science, vol. 12346, Springer, 2020, pp. 213–229, http://dx.doi.org/10.1007/978-3-030-58452-8_13.

[20] M.A. Islam, M. Kowal, S. Jia, K.G. Derpanis, N.D.B. Bruce, Position, padding and predictions: A deeper look at position information in CNNs, 2021, CoRR abs/2101.12322 arXiv:2101.12322.

[21] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, PVTv2: Improved baselines with pyramid vision transformer, 2021, arXiv:2106.13797.

[22] D. Gabor, Theory of communication. Part 1: The analysis of information, J. Inst. Electr. Eng. III 93 (1946) 429–441.

[23] C. Li, W. Wei, J. Li, W. Song, A cloud-based monitoring system via face recognition using gabor and CS-LBP features, J. Supercomput. 73 (4) (2017) 1532–1546, http://dx.doi.org/10.1007/s11227-016-1840-6.

[24] L. Fei, S. Teng, J. Wu, I. Rida, Enhanced minutiae extraction for high-resolution palmprint recognition, Int. J. Image Graph. 17 (4) (2017) 1750020:1–1750020:15, http://dx.doi.org/10.1142/S0219467817500206.

[25] X.-d. Hu, X.-q. Wang, F.-j. Meng, X. Hua, Y.-j. Yan, Y.-y. Li, J. Huang, X.-l. Jiang, Gabor-CNN for object detection based on small samples, Defence Technol. 16 (6) (2020) 1116–1129.

[26] S. Luan, C. Chen, B. Zhang, J. Han, J. Liu, Gabor convolutional networks, IEEE Trans. Image Process. 27 (9) (2018) 4357–4366, http://dx.doi.org/10.1109/TIP.2018.2835143.

[27] H. Yao, L. Chuyi, H. Dan, Y. Weiyu, Gabor feature based convolutional neural network for object recognition in natural scene, in: 2016 3rd International Conference on Information Science and Control Engineering, ICISCE, 2016, pp. 386–390, http://dx.doi.org/10.1109/ICISCE.2016.91.

[28] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: P.L. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a Meeting Held December 3-6, 2012, Lake Tahoe, Nevada, United States, 2012, pp. 1106–1114, URL https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

[29] F. Meng, X. Wang, F. Shao, D. Wang, X. Hua, Energy-efficient gabor kernels in neural networks with genetic algorithm training method, Electronics (ISSN: 2079-9292) 8 (1) (2019) http://dx.doi.org/10.3390/electronics8010105, URL https://www.mdpi.com/2079-9292/8/1/105.

[30] J.C. Pérez, M. Alfarra, G. Jeanneret, A. Bibi, A.K. Thabet, B. Ghanem, P. Arbeláez, Gabor layers enhance network robustness, in: A. Vedaldi, H. Bischof, T. Brox, J. Frahm (Eds.), Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part IX, in: Lecture Notes in Computer Science, vol. 12354, Springer, 2020, pp. 450–466, http://dx.doi.org/10.1007/978-3-030-58545-7_26.

[31] R. Wightman, PyTorch image models, 2019, http://dx.doi.org/10.5281/zenodo.4414861, GitHub Repository https://github.com/rwightman/pytorch-image-models.

[32] S. Yun, D. Han, S. Chun, S.J. Oh, Y. Yoo, J. Choe, CutMix: Regularization strategy to train strong classifiers with localizable features, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE, 2019, pp. 6022–6031, http://dx.doi.org/10.1109/ICCV.2019.00612.

[33] H. Zhang, M. Cissé, Y.N. Dauphin, D. Lopez-Paz, Mixup: Beyond empirical risk minimization, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018, URL https://openreview.net/forum?id=r1Ddp1-Rb.

[34] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 2818–2826, http://dx.doi.org/10.1109/CVPR.2016.308.

[35] J. Bai, Y. Zeng, Y. Zhao, F. Zhao, Training a V1 like layer using gabor filters in convolutional neural networks, in: International Joint Conference on Neural Networks, IJCNN 2019 Budapest, Hungary, July 14-19, 2019, IEEE, 2019, pp. 1–8, http://dx.doi.org/10.1109/IJCNN.2019.8852439.

[36] T.S. Lee, Image representation using 2D gabor wavelets, IEEE Trans. Pattern Anal. Mach. Intell. 18 (10) (1996) 959–971, http://dx.doi.org/10.1109/34.541406.

[37] Z. Peng, W. Huang, S. Gu, L. Xie, Y. Wang, J. Jiao, Q. Ye, Conformer: Local features coupling global representations for visual recognition, 2021, CoRR abs/2105.03889 arXiv:2105.03889.

[38] P. Dufter, M. Schmitt, H. Schütze, Position information in transformers: An overview, 2021, CoRR abs/2102.11090 arXiv:2102.11090.

[39] A. Srinivas, T. Lin, N. Parmar, J. Shlens, P. Abbeel, A. Vaswani, Bottleneck transformers for visual recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, June 19-25, 2021, Computer Vision Foundation / IEEE, 2021, pp. 16519–16529, URL https://openaccess.thecvf.com/content/CVPR2021/html/Srinivas_Bottleneck_Transformers_for_Visual_Recognition_CVPR_2021_paper.html.

[40] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA, IEEE Computer Society, 2009, pp. 248–255, http://dx.doi.org/10.1109/CVPR.2009.5206848.

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: H.M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E.B. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, 2019, pp. 8024–8035, URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[42] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, la, USA, May 6-9, 2019, OpenReview.net, 2019, URL https://openreview.net/forum?id=Bkg6RiCqY7.

[43] E.D. Cubuk, B. Zoph, J. Shlens, Q. Le, RandAugment: Practical automated data augmentation with a reduced search space, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual, 2020, URL https://proceedings.neurips.cc/paper/2020/hash/d85b63ef0ccb114d0a3bb7b7d808028f-Abstract.html.