Vicinity Vision Transformer

Weixuan Sun, Zhen Qin, Hui Deng, Jianyuan Wang, Yi Zhang, Kaihao Zhang, Nick Barnes, Senior Member, IEEE, Stan Birchfield, Lingpeng Kong, and Yiran Zhong

Abstract—Vision transformers have shown great success on numerous computer vision tasks. However, their central component, softmax attention, prohibits vision transformers from scaling up to high-resolution images, due to both the computational complexity and memory footprint being quadratic. Linear attention was introduced in natural language processing (NLP) which reorders the self-attention mechanism to mitigate a similar issue, but directly applying existing linear attention to vision may not lead to satisfactory results. We investigate this problem and point out that existing linear attention methods ignore an inductive bias in vision tasks, i.e., 2D locality. In this article, we propose Vicinity Attention, which is a type of linear attention that integrates 2D locality. Specifically, for each image patch, we adjust its attention weight based on its 2D Manhattan distance from its neighbouring patches. In this case, we achieve 2D locality in a linear complexity where the neighbouring image patches receive stronger attention than far away patches. In addition, we propose a novel Vicinity Attention Block that is comprised of Feature Reduction Attention (FRA) and Feature Preserving Connection (FPC) in order to address the computational bottleneck of linear attention approaches, including our Vicinity Attention, whose complexity grows quadratically with respect to the feature dimension. The Vicinity Attention Block computes attention in a compressed feature space with an extra skip connection to retrieve the original feature distribution. We experimentally validate that the block further reduces computation without degrading the accuracy. Finally, to validate the proposed methods, we build a linear vision transformer backbone named Vicinity Vision Transformer (VVT). Targeting general vision tasks, we build VVT in a pyramid structure with progressively reduced sequence length. We perform extensive experiments on CIFAR-100, ImageNet-1k, and ADE20 K datasets to validate the effectiveness of our method. Our method has a slower growth rate in terms of computational overhead than previous transformer-based and convolution-based networks when the input resolution increases. In particular, our approach achieves state-of-the-art image classification accuracy with 50% fewer parameters than previous approaches.

Index Terms—2D vicinity, image classification, linear transformer, semantic segmentation, vision transformer.

I. INTRODUCTION

RECENT years have witnessed the success of the transformer structure in natural language processing [1], [2], [3] and computer vision [4], [5], [6]. However, transformers inherently suffer from quadratic computational complexity and a quadratic memory footprint. As a result, vision transformer networks have to adopt patch-wise image tokenization to reduce the sequence length. Despite the temporary relief provided by such a tokenization method, the quadratic complexity problem still exists. This limitation prohibits vision transformers from handling high-resolution images or fine-grained image patches.

Linear attention is a promising direction to solve this issue. This group of methods reorders the self-attention mechanism of transformers with Kernelization methods to reduce the quadratic complexity to linear [7]. Numerous methods have been proposed for attention decomposition, such as approximating the softmax [8], [9], [10], and finding a new similarity metric [11], [12]. However, most of these methods are only verified on NLP tasks and suffer from a crucial performance drop in computer vision [13], when compared with conventional softmax attention.

We investigate this issue and point out that existing linear attention methods ignore an inductive bias in vision, i.e., 2D locality. Our intuition is based on the fact that convolutional neural networks [14], [15], [16] have dominated computer vision tasks since the rise of deep networks. Most of them have such locality bias, that is, 2D neighbouring regions are more likely to be highly related. Recent transformer-based methods also adopt such an assumption and obtain improved results by attaching convolution-based 2D locality bias [17], [18], [19], [20], [21]. Additionally, some efficient transformer backbones use window attention [22], [23], neighbourhood attention [24], or deformable attention [25], [26] to enable lower complexity and 2D locality, but they still suffer limited receptive field and quadratic complexity within the sampled tokens. Therefore we hypothesize that the 2D locality bias is an important property and should be incorporated into linear transformers.

In this article, we present Vicinity Attention, a new linear attention method that effectively enforces 2D locality. Our locality re-weighting mechanism is inspired by the recently proposed cosFormer [12], which assumes locality bias in language and uses a cosine re-weighting mechanism for 1D NLP tasks. However, directly applying the cosFormer to vision tasks will lead to unsatisfactory results since the 2D locality enforces stronger connections only on the 1D tokenized neighbouring...
image patches, so the cosFormer is not compatible with 2D distance. In that case, it will assign less weight to vertically connected patches as they are far away when we tokenized the patches to 1D tokens as shown in Fig. 3(a). To solve this issue, we propose a 2D variant of cosFormer in this work. Specifically, we propose a 2D Manhattan distance decomposition to encode relative positions, and integrate it with the cosine function to encourage visual tokens to have a stronger connection to their neighbors in 2D Fig. 3(b), (c), and (d). Since this distancing mechanism is decomposable in two directions, it can be seamlessly applied to linear attention.

Compared with vanilla transformer-based methods, linear attention complexity grows linearly with respect to sequence length but quadratically with feature dimension, which becomes a new computational bottleneck in our Vicinity Attention. To address this concern so as to further reduce computation, we propose a novel Vicinity Attention Block. First, we propose Feature Reduction Attention (FRA) to reduce the input feature dimension by half, so that the overall theoretical complexity can be reduced by a factor of four. Then a Feature Preserving Connection (FPC) is added to retrieve the original feature distribution and strengthen the representational ability. Such a block structure is seamlessly integrated with our linear Vicinity Attention and we experimentally validate that our Vicinity Vision Block further reduces computational complexity without degenerating the accuracy.

Finally, we build a general-purpose linear vision backbone, termed Vicinity Vision Transformer (VVT). Since our linear Vicinity Attention has a better efficiency advantage over vanilla self-attention [4], which enables feature maps with higher resolutions. We build VVT in a pyramid structure, which starts from high-resolution image patches and progressively shrinks to adapt to different vision tasks with multi-scale outputs.

Fig. 1(a) compares our method with current state-of-the-art vision backbones on the ImageNet-1k benchmark. Our method outperforms all the competitors with only half the parameters. We also provide the growth rate of GFLOPs with different input resolutions for these methods in Fig. 1(b). Given its linear complexity in token numbers, VVT can efficiently process images with much larger resolutions. Further, our experiments validate that VVT does not compromise accuracy and achieves superior results over transformer-based methods as well as convolution-based methods of comparable model sizes.

Our main contributions are as follows: (1) We introduce a linear self-attention mechanism for vision called Vicinity Attention, which introduces a Manhattan distance-based 2D locality to linear vision transformers. (2) To further reduce computational complexity by targeting linear attention, we propose a novel attention block called the Vicinity Attention Block. It contains a feature reduction attention (FRA) to improve the efficiency and a feature preserving connection (FPC) to retain the feature extraction ability. (3) We correspondingly build the Vicinity Vision Transformer (VVT), which serves as a general-purpose vision backbone and can be easily applied to various vision tasks. Extensive experiments validate the effectiveness of VVT on various computer vision benchmarks.

II. RELATED WORK

A. Vision Transformer Backbones

In computer vision tasks, CNN networks [14], [15], [16] have achieved great successes, while recently transformers have gained strong emerging interest. In this section, we mainly discuss vision networks using self-attention.

Some works [21], [27], [28], [29] adopt self-attention mechanisms to replace some or all convolution layers in the CNN networks for image recognition. To further leverage the power of self-attention, [30] proposes a non-local operation and adds it within ResNet [31], the non-local block can capture long-range dependencies and lead to improvements on vision tasks such as video classification, object detection, segmentation, and pose estimation. Recently, pure transformer-based vision networks were introduced. ViT [4] splits images into local patches. Then
it projects and flattens patches into embedding sequences, subsequently, it employs a pure transformer structure for image classification.

Variations of ViT like CVT [17], Swin [22], PVT [32], Twins [33] and T2T [19] were proposed. CVT [17] introduces Convolutional Token Embedding and Convolutional Projection for Attention to include desirable properties of CNNs into ViT [4] to improve both performance and efficiency. The Swin Transformer [22] introduces non-overlapping windows and applies self-attention in each window, then the window partitions are shifted between adjacent layers. Swin transformer reduces computational complexity and is suitable for different vision tasks. PVT [32] introduces a pure transformer network with a pyramid structure, and it further proposes spatial reduction attention which computes self-attention in reduced sequence length to save computation. Twins [33] combines locally-grouped self-attention from Swin [22] and sub-sampled attention from [32], which decreases the computational cost and enhances communications between sub-windows. DAT [26] builds a general vision transformer backbone with deformable attention. More recently, [34] adopts the quad-tree algorithm which progressively ignores less related image patches to improve efficiency. However, none of the above methods are able to directly process full-length self-attention on high-resolution inputs, requiring locally grouping or sub-sampling. Contrarily, our method can directly calculate self-attention on high-resolution input.

B. Efficient Transformers

Various works have been proposed to address the computational complexity problems of transformers in both computer vision and natural language processing. In addition to vision backbones such as CVT [17], Swin [22], PVT [32], and Twins [33] introduced in the previous section, in this section we introduce other more efficient transformer algorithms.

Existing efficient transformer methods can be generally grouped into two categories: pattern-based and kernel-based. Pattern-based methods sparsify the attention matrix with handcrafted or learnable patterns. [35] reduces the complexity by applying a combination of a strided pattern and a local pattern to the standard attention matrix. Beyond fixed diagonal windows and global windows, Longformer [36] also extends sliding windows with dilation to enlarge the receptive field. Instead of fixed patterns, Reformer [37] and [38] adopt locally sensitive hashing to group tokens into different buckets. On the other hand, kernel-based methods [8], [9], [10], [39] aim to replace softmax self-attention with approximations or decomposable functions, which change the order of scale dot product calculation and reduce the complexity of self-attention from quadratic to linear. [12] proposes a cosine re-weighting function to enforce locality in a 1D sequence and achieves linear complexity. Nevertheless, the aforementioned kernel-based methods only consider the linearization in a 1D sequence for natural language processing tasks, their performances on 2D vision tasks are not satisfactory.

In contrast, we aim at enforcing 2D locality in linear complexity.

Two works that are similar to ours are [13] and [40]. [40] applies softmax on $Q$ and $K$ respectively, then changes the dot product order to access a linear complexity. However, it is not validated on a large-scale classification benchmark as a backbone network. SOFT [13] uses a Gaussian kernel function to replace the dot-product similarity in self-attention. It assumes that the similarity is symmetrical, which may not hold true in practice. Further, it does not consider the 2D locality mechanism. In this article, we propose a linear self-attention that facilitates a 2D locality mechanism, and we validate the proposed method on various vision tasks.

III. PRELIMINARY

Self-attention was originally proposed for 1D NLP tasks [1]. To make it suitable for 2D vision tasks, we often convert an image to several patches and then embed them to a 1D sequence $x \in \mathbb{R}^{N \times d}$, where $d$ is the embedding size and $N = \frac{H}{p} \times \frac{W}{p}$ is the patch number, with $p$, $H$, $W$ denoting patch size, image height, and image width respectively. Mathematically, a transformer block $T: \mathbb{R}^{N \times d} \rightarrow \mathbb{R}^{N \times d}$ with input $x$ is defined as:

$$T(x) = F(Att(x) + x)$$

where $Att(\cdot)$ is the self-attention module and $F(\cdot)$ is a feed-forward module.

The self-attention module $Att(\cdot)$ adopts three learnable weights $W_q, W_k, W_v$ to project $x$ into query ($Q$), key ($K$), and value ($V$). It usually computes an attention matrix $A$ by a similarity function $S(\cdot)$ over queries and keys. In standard self-attention, $S(\cdot)$ is softmax normalization. The output of the self-attention module is $O = Att(x) = AV$, where $A \in \mathbb{R}^{N \times N}$ suffers from a quadratic space and time complexity with respect to the patch number $N$. Therefore, the computational complexity here is $O(N^2d) = O(\frac{H^2W^2}{p^2}d)$. Consequently the self-attention module becomes sensitive to image size $H, W$ and patch size $p$, which are the computation bottlenecks in vision transformer networks.

Linearization of self-attention aims to reduce the quadratic theoretical computation complexity to linear. It can be achieved by picking a decomposable similarity function $S(\cdot)$ to satisfy

$$S(Q_i, K_j) = \phi(Q_i)\phi(K_j)^\top$$

where $\phi(\cdot)$ is a kernel function. Given such a kernel, we can write the outputs of the self-attention module as follows:

$$O_i = \sum_j S(Q_i, K_j) V_j = \frac{\sum_{j=1}^{N} (\phi(Q_i)\phi(K_j)^\top) V_j}{\sum_{j=1}^{N} (\phi(Q_i)\phi(K_j)^\top)}$$

Fig. 2 illustrates the linearization process. Instead of calculating the attention matrix $A \in \mathbb{R}^{N \times N}$, we compute $\phi(K)^\top V \in \mathbb{R}^{d \times N}$ first:

$$[\phi(Q)\phi(K)^\top] V = \phi(Q) [\phi(K)^\top V]$$

so that the $O(N^2d)$ operation is converted to an $O(Nd^2)$ one. Its computational complexity grows linearly with respect to the sequence length $N$. A number of linear attention approaches have been proposed for NLP tasks, which use different kernel functions $\phi(x)$ to replace the quadratic softmax, as discussed in the Related Work. However, directly applying existing linear
attention [8], [10], [12] to vision suffers from a performance drop [13]. Notably, all of the aforementioned linear attention approaches and existing linear vision transformers [13], [34], [41] do not consider 2D locality in vision. To address this problem, we propose a linear self-attention that is aware of 2D position.

**Locality** is a widely used assumption in computer vision [14], [15], [16], [17], [18], [19], [20], [42], [43], i.e., neighbouring pixels should have a higher possibility to belong to the same object than distant pixels. In convolution-based networks, this assumption is inherently coupled into each layer throughout the whole model via convolutional kernels [14], [15], [16]. However, it does not hold for standard transformer-based networks due to the self-attention mechanism [4]. Several works show that with the same number of parameters, transformer-based networks cannot match the performance of the CNN counterparts, possibly due to the lack of locality bias [17], [44]. Recent state-of-the-art methods partially introduce locality bias to vision transformers at the architectural level, [17], [18], [19], [20] directly combine convolution with transformers. Some efficient transformer backbones use window attention [22], [23], neighbourhood attention [24], or deformable attention [25], [26] to achieve lower complexity and better performance. These methods enable 2D locality, however, they still suffer limited receptive field and quadratic complexity within the sampled tokens.

In this article, for the first time, we introduce locality bias into the linear self-attention. It can be smoothly integrated into existing vision transformer architectures for better performance. In fact, our method achieves better performance than previous state-of-the-art vision transformers in various computer vision benchmarks.

### IV. OUR METHOD

In this section, we start from presenting the details of Vicinity Attention mechanism in Section IV-A which enforces 2D locality in linear attention. Then in Section IV-B, we introduce a new attention structure named Vicinity Attention Block to address the computational bottleneck for linear attention targeting linear attention. In Section IV-C we introduce the overall architecture of VVT and show the structure illustration in Fig. 5. Targeting general vision tasks, we integrate our Vicinity Attention into a four-stage pyramid structure. Each stage consists of a patch embedding module followed by several Vicinity Vision Blocks and feed forward residual blocks. Image inputs are progressively down-sampled to generate multi-scale outputs. We provide a family of VVT variants and detail their configurations in Table I.

#### A. Vicinity Attention

It has been verified that the softmax normalization is the root of the quadratic complexity of self-attention [13]. In this section, we provide the details of our linear self-attention technique, i.e., vicinity attention for vision tasks. The key insight is to replace the standard softmax operation by another similarity function, which (1) can be decomposed to simple kernel functions and (2) introduces a locality constraint that is beneficial in visual understanding. Inspired by [12], we adopt ReLU(·) as the kernel function, and conduct a row-wise normalization to replace softmax. Our self-attention then can be reformulated as:

\[
O_i = \frac{\sum_{j=1}^{N} [\text{ReLU}(Q_i) \text{ReLU}(K_j)^\top] V_j}{\sum_{j=1}^{N} [\text{ReLU}(Q_i) \text{ReLU}(K_j)^\top]}
\]

where we can reorder the calculation so the complexity is reduced to linear with respect to the sequence length \( N \). This ReLU-based normalization method retains two important properties of softmax self-attention: (1) it is always positive, avoiding aggregating negatively-correlated information. (2) all the elements lie between [0,1].

**Enforcing 2D Locality Bias:** Locality bias has been discussed in language tasks [12], [45], [46] with 1D sequence distance encoding, but not considered by existing linear attention in vision. In vision transformers, the embedding sequence is flattened from a 2D mask, hence it is essential to consider token positions in 2D before being flattened. To enforce the locality bias in linear transformers, we need a kernel function that can 1) put more emphasis on 2D neighboring patches and 2) can be decomposed with (2).

Given two tokens \( q_i, k_j \) from \( Q \) and \( K \) respectively, the positions of these two tokens on the 2D feature maps before
flattening are:

\[ i = u_im + r_i, \quad j = u_jm + r_j, \quad 0 \leq r_i, r_j < m \]

(6)

where \( m \) is width of the feature map, \( u \) denotes the row index, and \( r \) denotes the column index. Following (2), we can define a re-weighted attention with a distance function \( D \) to enforce the locality bias between two tokens as:

\[
S(Q_i, K_j) = \phi(Q_i)G(D(i, j))\phi(K_j)\top
\]

(7)

where \( G \) produces the weight according to the distance so nearby patches can be emphasized. A naive choice might be directly using the euclidean distance \( \sqrt{(r_j - r_i)^2 + (u_j - u_i)^2} \). However, since this term cannot be decomposed into two terms relating to \( i \) and \( j \) separately, it cannot be applied to the linear transformers.

Instead, since 2D Manhattan distance [47] decouples relative position in two directions, we propose to use it as the distance function:

\[
S(Q_i, K_j) = \phi(Q_i)G(|r_j - r_i| + |u_j - u_i|)\phi(K_j)\top
\]

(8)

However, direct Manhattan distance decoupling is still hindered as the absolute operation cannot be decomposed.

Inspired by [12], the cosine function has two desirable features: 1) It cancels the absolute operation in (8) and applies non-linear emphasis on the nearby tokens; and 2) It can be decomposed into two terms relating to \( i \) and \( j \) separately, which fulfills the linear complexity. Thus, we propose to bind Manhattan distance and cosine function to achieve linear attention with 2D locality. First, given a feature map with size \( m \) by \( n \), following (6) we redefine the 2D positions as:

\[
a_i = \frac{\pi u_i}{2n}, \quad a_j = \frac{\pi u_j}{2n}, \quad b_i = \frac{\pi r_i}{2m}, \quad b_j = \frac{\pi r_j}{2m}
\]

(9)

Then, using the Manhattan distance, the self-attention calculation can be decomposed into four terms:

\[
S(Q_i, K_j) = \phi(Q_i)(\cos(a_i - a_j) + \cos(b_i - b_j))\phi(K_j)\top
\]

\[
= \phi(Q_i)(\cos(a_i)\cos(a_j) + \cos(b_i)\cos(b_j))\phi(K_j)\top
\]

\[
+ \phi(Q_i)(\sin(a_i)\sin(a_j))\phi(K_j)\top
\]

\[
+ \phi(Q_i)(\cos(b_i)\sin(b_j))\phi(K_j)\top
\]

\[
+ \phi(Q_i)(\sin(b_i)\sin(b_j))\phi(K_j)\top
\]

\[
= \phi^{\cos a_i}(Q_i)\phi^{\cos a_j}(K_j)\top
\]

\[
+ \phi^{\sin a_i}(Q_i)\phi^{\sin a_j}(K_j)\top
\]

\[
+ \phi^{\cos b_i}(Q_i)\phi^{\cos b_j}(K_j)\top
\]

\[
+ \phi^{\sin b_i}(Q_i)\phi^{\sin b_j}(K_j)\top
\]

(10)

where \( \phi^{\cos y}(x) \triangleq \cos(y)\phi(x), \phi^{\sin y}(x) \triangleq \sin(y)\phi(x) \).

Here the queries and keys are related to their own positions \( i \) and \( j \), and hence we can fulfill (4) to reorder the dot product in self-attention. Following our locality decomposition, we get:

\[
K'_i = [\phi^{\cos a_i}(Q_i), \phi^{\sin a_i}(Q_i), \phi^{\cos b_i}(Q_i), \phi^{\sin b_i}(Q_i)]
\]

(11)

where

\[
\phi^{\cos y}(x) \triangleq \cos(y)\phi(x), \quad \phi^{\sin y}(x) \triangleq \sin(y)\phi(x)
\]

(12)

Finally, we have the output as:

\[
O_i = \frac{\sum_{j=1}^{N}[Q'_iK'_j\top]V_j}{\sum_{j=1}^{N}[Q'_iK'_j\top]} = \frac{Q'_i\sum_{j=1}^{N}[K'_j\top]V_j}{Q'_i\sum_{j=1}^{N}K'_j}\top
\]

(13)

Our method provides three appealing properties: (1) Linear complexity: cosine re-weighting and Manhattan distance ensure decomposability, thus we avoid calculating \( QK \). The overall computational complexity is \( O(N(4d)^2) \) which grows linearly with sequence length. (2) Locality bias: if two tokens are close to each other in 2D feature maps, they are encouraged to have a stronger relationship, as shown in Fig. 3. (3) Global context: it retains a global receptive field as standard self-attention.

### B. Vicinity Attention Block

Compared to vanilla attention, the complexity of linear attention grows linearly with sequence length but quadratically with respect to feature embedding size. Thus, the computational bottleneck of linear attention is shifted from the input resolution to feature dimension, whilst it is not considered by existing linear vision transformers [13], [34], [40]. This issue is amplified in Vicinity Attention as our theoretical computational complexity is \( O(N(4d)^2) \). To achieve a better computational efficiency in linear attention, we redesign the structure of the multi-head self-attention (MSA) module to reduce the feature dimension without hindering the performance.

We present a schematic illustration of our Vicinity Attention Block in Fig. 4. Our block is generally composed of two steps: compressing and preserving. In the first compressing step, it receives an input \( X \in \mathbb{R}^{N \times d} \), which is projected into a compressed feature space \( Q \in \mathbb{R}^{N \times \frac{d}{2}}, K \in \mathbb{R}^{N \times \frac{d}{2}}, V \in \mathbb{R}^{N \times \frac{d}{2}} \). We call it Feature Reduction Attention (FRA). Then, following our locality decomposition in (10), we obtain \( Q' \in \mathbb{R}^{N \times \frac{d}{2}}, K' \in \mathbb{R}^{N \times \frac{d}{2}} \), which is used to calculate linear self-attention. Due to the essence of linear attention, FRA reduces the computational complexity by a factor of four.

In the second preserving step, we propose to preserve the original feature distribution to compensate the feature compression with negligible computational overhead. Inspired by [48], [49], we add a skip connection called Feature Preserving Connection (FPC) to capture the global context features of input \( X \). The FPC consists of an average pooling operation and two linear layers which have a complexity of \( 2d^2 \). FPC retains the original feature distribution and can strengthen the representational ability. Finally, the outputs of FRA and FPC are fused in a residual manner to get final attention output.

In summary, our Vicinity Attention Block proposes to calculate self-attention in a compressed feature dimension while still preserving the original feature space. It reduces the theoretical complexity to \( O(N(2d)^2 + 2d^2) \). Our experiments validate that it notably reduces computation without degenerating accuracy.
Fig. 3. Visualization of the locality re-weighting patterns. The red cross symbol denotes the query position and the mask denotes the locality weights to it. We use a lighter color to indicate a larger weight. (a) If we directly adopt 1D distance re-weighting [12] in self-attention, the query token will be assigned lower weights on vertical tokens as they are far away when processed in 1D. (b), (c), (d) In our method, although self-attention is calculated in 1D, our re-weighting mechanism ensures that a token is encouraged to have higher relation weights with its neighbours in two dimensions.

Fig. 4. Illustration of the Vicinity Attention Block. The input feature dimension is down-sampled to reduce the computational cost, while the original feature distribution is retained by the FPC (the skip connection at the top).

In the following, we discuss the relationships between our VVT and several existing methods.

Relationship to CosFormer: The cosFormer [12] was originally developed for natural language processing and achieves linear complexity in 1D. In this article we develop a 2D variant of the cosFormer. The cosFormer [12] attention adopts cosine re-weighting and considers 1D locality bias in NLP, but it shows poor results on vision tasks (Table VII). Our Vicinity Attention can be seen as a non-trivial extension of the cosFormer to vision. The combination of Manhattan distance and the cosine function composes a novel 2D positional encoding that enables the capture of long-range visual dependency with 2D locality and linear complexity. In addition, we propose a new vision attention block named the Vicinity Attention Block, which contains FRA and FPC to further reduce the complexity without diminishing performance. The Vicinity Attention Block is then used to build vision backbones with a pyramid structure and can process high-resolution images.

Relationship to Existing Window Attention and Neighborhood Attention: 2D locality has been introduced into vision transformers by several methods such as window attention [22], [23], deformable attention [25], [26] and neighborhood attention [50]. A key difference between these and our Vicinity Attention lies in the receptive field. These existing methods apply hard locality, i.e., they consider self-attention only within the sampled tokens (from nearby windows or deformable operations) unsampled tokens are disregarded. Further, complexity within the sampled tokens is still quadratic. Differently, Vicinity Attention has a constant global receptive field via soft locality, i.e., all spatial locations are visible, but spatially nearby ones are emphasized via our linear attention.

Relationship to GCNet: The main difference between the Vicinity Attention Block and GCNet block [48] is the attention generation method, which reflects the different motivations of the two blocks. The GCNet block adds a uniform context vector to every spatial location. This uniform context vector does not vary by query location and is motivated by the observation that different attention maps are mostly similar. Our Vicinity Attention Block performs a transformer operation that is modified to reduce complexity. That is, our attention maps vary by query location using the 2D locality constraint. We also have a channel attention-like connection called FPC which is an instantiation of the squeeze-excitation (SE) connection [49], but aims to reduce complexity while preserving representation ability. Results in Table VII show the superiority of the Vicinity Attention Block compared to the GCNet block.

C. The Vicinity Vision Transformer

In this section, we introduce the overall architecture of Vicinity Vision Transformer (VVT) as shown in Fig. 5.

The linear complexity of Vicinity Attention enables higher-resolution inputs, so we integrate the Vicinity Attention Block into a progressively shrinking pyramid structure that has four
Fig. 5. Overall architecture of our VVT. VVT adopts a pyramid structure and is divided into four stages. The illustrated sample is VVT-Small. The output resolution progressively shrinks to generate multi-scale feature maps.

Each stage contains a patch embedding layer and multiple stacked Vicinity Transformer Blocks and Feed-forward blocks. In detail, we use a patch size of $4 \times 4$ in the first stage. Given the input image of size $H \times W \times 3$, we first divide it into $\frac{H}{4} \times \frac{W}{4}$ patches. Then we feed the patches into a patch embedding module to obtain a flattened embedding sequence with a size of $\frac{HW}{16} \times C_1$. Here we adopt the overlapping patch embedding module and Convolutional Feed-Forward proposed by Wang et al. [51]. The embedding sequence is subsequently fed into several successive transformer blocks with $L_1$ layers.

In the second stage, the feature sequence from the first stage is reshaped back to $\frac{H}{4} \times \frac{W}{4} \times C_1$ and down-sampled to an embedding sequence of size $\frac{HW}{16} \times C_2$, and then processed by the transformer blocks of the second stage. We follow the same approach to obtain multi-scale output feature maps of the third and fourth stage with output sizes of $\frac{HW}{32} \times C_3$ and $\frac{HW}{32} \times C_4$, respectively. Hierarchical feature maps can be easily leveraged to many downstream vision tasks. In Fig. 6, we show qualitative examples of Grad-CAM [52] obtained from different stages of VVT and ViT [4] respectively. As shown, our approach is able to produce fine-grained features whereas the ViT [4] can only capture low-resolution features.

Table I details different architecture variants of VVT from VVT_Tiny to VVT_Large. We empirically choose our network settings following the common principles of vision backbones [15], [53]: (1) spatial resolution is decreased progressively with feature dimension increased. (2) stage 3 has the most of the computational cost. The architecture hyper-parameters of VVT are:
- $C$: the input channel dimension of the attention block.
- $P$: the patch size of the patch embedding.
- $R$: the feature reduction ratio of the Vicinity Attention Block.
- $H$: the number of heads in the Vicinity Attention Block.
- $E$: the feature expansion ratio of the feed forward layer.

V. EXPERIMENTS

To verify the effectiveness of our method, we conduct extensive experiments on the CIFAR-100 [59] and ImageNet-1k [54] datasets for image classification, and on the ADE20K [60] dataset for semantic segmentation. Specifically, we first make a comparison with existing state-of-the-art methods, and then give an ablation study over the design of VVT.

A. Image Classification

ImageNet-1k: The image classification experiments are conducted on the ImageNet-1k [54] dataset, which contains 1.28 million training images and 50 thousand validation images from 1,000 categories. We follow the training hyper-parameters of [32]. In detail, the model is trained using an AdamW [61] optimizer with a weight decay of 0.05 and a momentum of 0.9. We use an initial learning rate of $5 \times 10^{-4}$ and decrease it by a cosine schedule [62], with 5 epochs for warming up. We also adopt the same data augmentation strategy as in [32], including random cropping, random horizontal flipping, etc. All the models
TABLE I
DETAILED SPECIFICATIONS OF OUR VVT VARIANTS. WE SHOW FOUR VARIANTS OF OUR VVT STRUCTURE. C: FEATURE DIMENSION, P: PATCH SIZE, R: FEATURE REDUCTION, H: HEAD NUMBER, E: EXPANSION RATIO OF THE FEED-FORWARD LAYER. THE NUMBER OF LAYERS OF THE FOUR STAGES Follows THE RATIOS OF 1 : 1 : 3 : 1 or 1 : 1 : 9 : 1.

| Stage | Output Size | Layer Name | Tiny | Small | Medium | Large |
|-------|-------------|------------|------|-------|--------|-------|
|       |             | Patch Name | C = 95, P = 4 |       |        |       |
| Stage 1 | $\frac{H}{4} \times \frac{W}{4}$ | Transformer Encoder | R = 2, H = 1, E = 8 | R = 2, H = 1, E = 8 | R = 2, H = 1, E = 8 | R = 2, H = 1, E = 8 |
| Stage 2 | $\frac{H}{8} \times \frac{W}{8}$ | Patch Embedding | C = 160, P = 2 |       |        |       |
| Stage 3 | $\frac{H}{16} \times \frac{W}{16}$ | Transformer Encoder | R = 2, H = 2, E = 8 | R = 2, H = 2, E = 8 | R = 2, H = 2, E = 8 | R = 2, H = 2, E = 8 |
| Stage 4 | $\frac{H}{32} \times \frac{W}{32}$ | Patch Embedding | C = 512, P = 2 |       |        |       |

Fig. 7. Qualitative results of semantic segmentation on ADE20K [60]. The results are generated by VVT-L based Semantic FPN [64].

are trained on the training set for 300 epochs with a crop size of 224 × 224. We use the top-1 accuracy as the evaluation metric.

The quantitative results are shown in Table II, which cover the popular transformer-based and CNN-based classification networks. For fair comparison, we compare the VVT variants respectively with the networks using similar parameters, split by solid lines. For example, VVT-L performs better than PVTv2-B5 [32] but only uses around 70% parameters of the latter. In particular, our medium-size variant VVT-M surpasses the large models Twins-SVT-L [33] and Swin-B [22] with substantially fewer parameters (51.7% and 45.5% respectively). Moreover, compared to the state-of-the-art CNN-based networks such as RegNet [63] and ResNeXt [58], VVT shows stronger performance while using a similar number of parameters.

It is worth noting that VVT directly computes self-attention on the high-resolution feature maps, while the competitors such as PVT [32], Swin Transformer [22], Twins [33] and SOFT [13] rely on sub-sampling or window self-attention to reduce computational cost. The GFLOPs of VVT could be further reduced if similar subsampling or windowing was used, we validate this in Fig. 8.

CIFAR-10 and CIFAR-100: It is known that vision transformers may suffer critical performance drops on small-scale datasets. To validate the performance of our method on small-scale datasets, we use well-known CIFAR-10 and CIFAR-100 [59] as target datasets. Tables III and IV show our VVT-S and competitors’ results. All models are trained from scratch using an AdamW optimizer with a weight decay of 0.05 and a momentum of 0.9. We use an initial learning rate of $5 \times 10^{-4}$ and train for 300 epochs. All models are trained and tested on 224 × 224 resolution for fair comparison. We see that our VVT outperforms all competing methods on both CIFAR-10 and CIFAR-100. We hypothesize that for small datasets, stronger inductive bias may make the model more data efficient. We can...
### TABLE II

COMPARISON OF DIFFERENT BACKBONES ON IMAGENET1K [54]. FOR A FAIR COMPARISON, WE COMPARE VVT WITH COMPETING MODELS UNDER SIMILAR PARAMETER SIZES. ALL MODELS ARE TRAINED AND TESTED ON 224 × 224 RESOLUTION. GFLOPS ARE ALSO CALCULATED UNDER THE INPUT SCALE OF 224 × 224. "MS OUT" REPRESENTS MULTI-SCALE OUTPUTS.

| Method         | Style       | MS out | Param | GFLOP | Top-1 Acc (%) |
|----------------|-------------|--------|-------|-------|---------------|
| ResNet18 [31]  | ConvNets    | ✓      | 11.7  | 1.8   | 69.8          |
| DeiT-Tiny/16 [55] | Transformers | x     | 5.7   | 1.3   | 72.2          |
| PVTv2-B1 [51]  | Transformers| ✓      | 13.1  | 2.1   | 78.7          |
| SOFT [13]      | Transformer | ✓      | 13.0  | 1.9   | 79.3          |
| VVT (ours)     | Transformers| ✓      | 12.9  | 3.0   | 79.4          |
| ResNet30 [31]  | ConvNets    | ✓      | 23.6  | 4.1   | 76.1          |
| PVT-Small [32] | Transformers| ✓      | 24.5  | 3.8   | 79.8          |
| RegNetY-4G [56] | ConvNets    | ✓      | 21.0  | 4.0   | 80.0          |
| TNT-S [18]     | Transformers| x      | 23.8  | 5.2   | 81.3          |
| Swin-T [22]    | Transformers| ✓      | 29.0  | 4.5   | 81.6          |
| CVT-13 [17]    | Hybrid      | ✓      | 20.0  | 4.5   | 81.6          |
| Twins-SVT-S [33] | Hybrid     | ✓      | 24.0  | 2.8   | 81.7          |
| XCLF-S12 [41]  | Linear Transformers | x | 26.0  | 4.8   | 82.0          |
| DAT-T [26]     | Transformer | ✓      | 29.0  | 4.6   | 82.0          |
| PVTv2-B2-Li [51] | Transformers | ✓ | 22.6  | 9.1   | 82.1          |
| SOFT [13]      | Linear Transformers | ✓ | 24.0  | 3.3   | 82.5          |
| Focal-Tiny [57] | Transformers | ✓ | 29.1  | 4.9   | 82.2          |
| Quadruple-B-b3 [34] | Linear Transformers | ✓ | 24.2  | 4.5   | 82.7          |
| VVT-S (ours)   | Linear Transformers | ✓ | 23.5  | 5.6   | 82.7          |

| Method         | Style       | MS out | Param | GFLOP | Top-1 Acc (%) |
|----------------|-------------|--------|-------|-------|---------------|
| ResNet101 [31] | ConvNets    | ✓      | 44.7  | 7.9   | 77.4          |
| ViT-Small [16] | Transformers| x      | 48.8  | 9.9   | 80.8          |
| PVT-Medium [32] | Transformers | ✓ | 44.2  | 6.7   | 81.2          |
| RegNetY-8G [56] | ConvNets    | ✓      | 39.0  | 8.0   | 81.7          |
| CVT-2v [17]    | Hybrid      | ✓      | 32.0  | 7.1   | 82.5          |
| XCLF-S24 [41]  | Linear Transformers | x | 48.0  | 9.1   | 82.6          |
| SOFT [13]      | Linear Transformers | ✓ | 45.0  | 7.2   | 82.9          |
| Swin-S [22]    | Transformers| ✓      | 50.0  | 8.7   | 83.0          |
| Twins-SVT-B [33] | Hybrid     | ✓      | 56.0  | 8.3   | 83.2          |
| PVTv2-B3 [51]  | Transformers| ✓      | 45.2  | 6.9   | 83.2          |
| Focal-Small [57] | Transformers | ✓ | 51.1  | 9.1   | 83.5          |
| PVTv2-B4 [51]  | Transformers| ✓      | 62.6  | 10.1  | 83.6          |
| DAT-S [26]     | Transformer | ✓      | 50.0  | 9.0   | 83.7          |
| Quadruple-B-b3 [34] | Linear Transformers | ✓ | 46.3  | 7.8   | 83.7          |
| VVT-M (ours)   | Linear Transformers | ✓ | 47.9  | 9.4   | 83.8          |

| Method         | Style       | MS out | Param | GFLOP | Top-1 Acc (%) |
|----------------|-------------|--------|-------|-------|---------------|
| ResNet152 [31] | ConvNets    | ✓      | 60.2  | 11.6  | 78.3          |
| PVT-Large [32] | Transformers| ✓      | 61.4  | 9.8   | 81.7          |
| ViT-Base [44]  | Transformers| x      | 86.6  | 16.0  | 81.8          |
| T2T-VT [19]    | Transformers| ✓      | 63.9  | 13.2  | 82.2          |
| XCLF-M24 [41]  | Linear Transformers | x | 84.0  | 16.2  | 82.7          |
| ResNeXt101-64x4d [58] | ConvNets | ✓ | 84.0  | 16.0  | 82.9          |
| TNT-B [18]     | Transformers| ✓      | 65.6  | 14.1  | 82.9          |
| RegNetY-16G [56] | ConvNets  | ✓      | 84.0  | 16.0  | 82.9          |
| SOFT [13]      | Linear Transformers | ✓ | 64.0  | 11.0  | 83.1          |
| Swin-B [22]    | Transformers| ✓      | 88.0  | 15.4  | 83.3          |
| Twins-SVT-L [33] | Hybrid     | ✓      | 99.2  | 14.8  | 83.7          |
| PVTv2-B5 [51]  | Transformers| ✓      | 82.0  | 11.8  | 83.8          |
| Focal-Base [57] | Transformers | ✓ | 89.8  | 16.0  | 83.8          |
| DAT-B [26]     | Transformer | ✓      | 88.0  | 15.8  | 84.0          |
| Quadruple-B-b4 [34] | Linear Transformers | ✓ | 64.2  | 11.5  | 84.0          |
| VVT-L (ours)   | Linear Transformers | ✓ | 61.8  | 10.8  | 84.1          |

### TABLE III

CIFAR-10 [59] CLASSIFICATION RESULTS

| Method         | Style       | MS out | Top-1 Acc (%) |
|----------------|-------------|--------|---------------|
| ViT-Small [4]  | Transformers| x      | 81.9          |
| TNT-S [18]     | Transformers| x      | 85.8          |
| Swin-T [22]    | Transformers| ✓      | 89.8          |
| PVT-Small [32] | Transformers| ✓      | 91.1          |
| ResNet50 [31]  | ConvNets    | ✓      | 92.5          |
| Twins-SVT-S [33] | Hybrid     | ✓      | 93.8          |
| PVTv2-B2 [51]  | Transformers| ✓      | 95.7          |
| VVT-S (ours)   | Linear Transformers | ✓ | 96.1          |

### TABLE IV

CIFAR-100 [59] CLASSIFICATION RESULTS

| Method         | Style       | MS out | Top-1 Acc (%) |
|----------------|-------------|--------|---------------|
| ViT-Small [4]  | Transformers| x      | 47.4          |
| TNT-S [18]     | Transformers| x      | 71.5          |
| Swin-T [22]    | Transformers| ✓      | 74.8          |
| PVT-Small [32] | Transformers| ✓      | 76.3          |
| ResNet50 [31]  | ConvNets    | ✓      | 79.2          |
| Twins-SVT-S [33] | Hybrid     | ✓      | 81.6          |
| PVTv2-B2 [51]  | Transformers| ✓      | 83.9          |
| VVT-S (ours)   | Linear Transformers | ✓ | 84.6          |
see ViT does not have inductive bias and performs poorly. Other models that have different types of locality bias achieve better results than ViT. Our VVT introduces self-attention with 2D locality and achieves the best results.

### B. Semantic Segmentation

We utilize the challenging ADE20K [60] dataset to evaluate our idea in semantic segmentation. It has 150 semantic classes, with 20,210, 2,000 and 3,352 images for training, validation and testing. With the VVT models (pre-trained on ImageNet-1k) as the backbone, we provide the results using two segmentation methods in Table V and show qualitative samples in Fig. 7. Specifically, we pick Semantic FPN [64] and UPerNet [65] as the semantic segmentation architecture, respectively following the settings of PVT [32] and Swin Transformer [22]. The quantitative results show that our method is consistently superior to the competing CNN-based methods. For example, using Semantic FPN, VVT-S outperforms ResNet50 [31] by 8.9%, and VVT-L is 7.7% better than ResNeXt101-32x4d [58], using mIoU as the metric. VVT also shows a more favourable segmentation result than the transformer-based competitors such as the Swin Transformer. A similar phenomenon is observed if taking UPerNet [65] as the segmentation method. These results validate that the extracted features of VVT are also valuable for semantic segmentation, which benefits from the locality mechanism together with the global attention.

### C. Ablation Study

#### Computational Overhead: We plot the GFLOPs growth rates with respect to the input image size on the right of Fig. 1. The growth rate of VVT (purple) is substantially lower than other transformer-based methods, and even slightly lower than ResNet [31]. However, theoretical GFLOPs may arguably not reflect real computational overhead. To further demonstrate the computational efficiency of Vicinity Attention, we compare GPU memory footprints of Vicinity Attention against recent efficient vision transformer methods [10], [13], [34], [39], [51] in Fig. 8. All results are obtained under the same settings including feature dimension, number of heads etc. Note that although VVT, SOFT [13], Performer [10], Quad [34] and Linformer [39] all have linear complexity, their actual memory consumption may vary according to their specific algorithms.

From Fig. 8, we can make the following observations: (i) Our method consumes relatively more memory than Performer [10] and Linformer [39], as both methods are designed for 1D sequences and show inferior performance on vision tasks. (ii) Compared to efficient vision transformers, i.e., PVTv2 [51], SOFT [13] and Quadtree [34], the memory consumption of Vicinity Attention grows notably slower, allowing inputs with much higher resolutions. (iii) The spatial reduction (sr) is a computation reduction technique proposed by PVT [53]. Specifically, it reduces the spatial dimensions of the key and value vectors in the self-attention module with convolution or pooling operations, thus the complexity of the self-attention computation is reduced. PVTv2 [51], SOFT [13], and Quadtree [34] all adopt the sr strategy to reduce computation, while the standard VVT directly calculates attention on the original sequence without sr. If we also integrate sr, the VVT_sr in Fig. 8 validates that the efficiency can be further improved with a spatial reduction ratio of 4. Finally, in Table VIII, we show memory footprints of the models with various attention modules that correspond to Table VII. In summary, this ablation study validates that VVT successfully mitigates the computational issue in the vision transformers.

#### Effect of Locality Constraint: The locality mechanism is introduced in the self-attention module of VVT. We show its effectiveness on the CIFAR-100 [59] and the ImageNet-1k [54] datasets in Table VI. Experiments are performed on the VVT-Tiny structure. We compare VVT with a variation without locality (denoted as VVT w/o locality) and the one with 1D locality based on cosFormer [12] (denoted as VVT 1D locality).
Fig. 9. We compare the class-wise attention maps of VVT against VVT w/o locality (VVT-wol in the figure), PVT [32] and ViT [4]. The attention maps are generated using Grad_CAM [52]. We observe that the activation of PVT and VVT w/o locality is more concentrated than ViT due to the fact that overlapping patch embedding or pyramid structure may also introduce locality. In VVT, we enforce 2D locality bias in terms of the attention module, so that the activation areas are more concentrated and appropriately located on the object.

As shown in the table, our VVT achieves the best performance among all the others on the CIFAR-100 dataset. By comparing the VVT w/o locality and VVT 1D locality, we find that if we wrongly enforce the locality bias, the performance will drop critically. To further demonstrate its effectiveness on large-scale data, we also conduct the same ablation on ImageNet-1k. We observe that the model fails to converge with 1D locality which indicates that the 1D cosFormer cannot be trivially transplanted to 2D vision data. Finally, our VVT outperforms the VVT w/o locality by 1.2%, showing the efficiency of our linear locality attention.

Furthermore, we show the qualitative examples of Grad-CAM [52] from VVT and the competitors in Fig. 9. As shown, the locality mechanism encourages the tokens to assign higher attention to 2D neighbours, and hence the class-wise activations are more concentrated and accurately located on the target object regions of Grad-CAM. These results validate that the invention of the 2D locality mechanism helps the VVT models generate more reliable object features, which tend to be beneficial in image classification and downstream tasks.

Comparison With Existing Attention Modules: We compare with several existing self-attention methods: cosFormer [12], Performer [10], Linformer [39] PVTv2 [51] and vanilla self-attention [1]. For all methods, we adopt the VVT-Tiny network structure and only replace the attention block with competing attention modules for a fair comparison. All experiments are conducted using the same network configuration and training setting. We report results on the CIFAR-100 and the ImageNet1k in Table VII. As shown, on CIFAR-100 our Vicinity attention outperforms both the vanilla attention and all the alternative efficient methods. On ImageNet1k, we observe that cosFormer [12] fails to converge due to the false 1D locality. Compared to other efficient methods including GCNet [48], Performer [10], Linformer [39] and PVTv2 [51], VVT still achieves better classification result. Further, vanilla softmax attention [4] is 0.6% higher than our VVT but at a cost of substantially higher computational complexity (see Table VIII), which makes it implausible to extend to any larger models with pyramid structures. In Table VIII, we display the quantitative memory footprints of the aforementioned models in Table VII.
All models share the same network structure except the attention module with a batch size of 16, allowing for a fair comparison of different attention mechanisms. Our VVT has linear complexity and similar memory footprints to existing linear attention methods, i.e., cosFormer [12], Linformer [39] and Performer [10], whereas our performances on both datasets are superior. Finally, VVT has better memory efficiency than PVTv2 [51] and vanilla softmax attention [4].

**Analysis of Vicinity Attention Block:** In this section, we investigate the proposed Vicinity Attention Block from three perspectives: (1) we show that feature reduction attention (FRA) can significantly reduce computation without harming performance; (2) We validate that feature preserving connection (FPC) is effective to strengthen feature extraction ability; and (3) we report the effectivenss of FRA and FPC on an existing linear attention method. In summary, we demonstrate that FRA effectively reduces computational complexity and FPC preserves representation ability without causing degeneration of performance.

By FRA, we reduce the feature dimension to fulfill the assumption of $N \gg d$, which forces the computational complexity to approach linear. In Fig. 10, we ablate the computational cost under different feature reduction (FR) ratios. We can observe an obvious GFLOPs decrease when increasing the FR ratio from 1 to 2 (reducing the feature dimension by half). However, when the FR ratio further increases, the reduction tends to saturate. This is because the computational cost of the self-attention module is already significantly reduced and the remaining model parts dominate the computational overhead. Additionally, we ablate the performance under different FR ratios on the CIFAR-100 dataset. As shown in Table IX. When the FR ratio is small, the model retains a similar performance (i.e., 82.89% versus 82.92%), indicating that the result is not affected. When the FR ratio increases to a number like 8, a clear performance drop is observed. As a trade-off, we set the FR ratio as 2 for our formal model settings.

In the Vicinity Attention Block, FPC is added to compensate for reduced feature dimensions. Ablation of FPC is reported in Table X. The results indicate that FPC effectively retains the original feature distribution, and improves the feature extraction ability of the Vicinity attention module even when the feature dimension is reduced.

To further demonstrate the effectiveness of Vicinity Attention Block, we integrate FRA and FPC with an existing linear attention method, i.e., Performer [10]. As shown in Table XI, we attach the Performer attention block on VVT-T as a baseline. After integrating FRA and FPC, the computation is reduced while we can still observe an accuracy improvement (i.e., 77.2% versus 77.3%). It validates that our FRA and FPC yield performance gains in the existing linear attention approach.

**Image Classification With Different Input Size:** Table XII lists the performance of VVT with higher input resolution i.e., 384. Obviously, larger input resolution leads to better top-1 accuracy but requires larger computation. Compared to competing methods such as ViT [4], Swin [22] and CVT [17], VVT achieves better results with fewer parameters and a comparable computational overhead.

**Analysis of Overlapping Patch Embedding and Convolutional Feed-Forward:** In standard VVT, we adopt the overlapping patch embedding module and Convolutional Feed-Forward proposed by Wang et al. [51]. Both modules may also contribute to locality bias and improve results [17]. Table XIII ablates our Vicinity Attention without these two modules, compared to existing
VI. CONCLUSION

We have introduced Vicinity Vision Transformer (VVT), a general-purpose vision backbone that produces hierarchical feature representations through a pyramid structure. At its core, we propose Vicinity Attention, which introduces 2D locality into linear self-attention modules. We validate that the locality mechanism improves results and helps generate better class activation. Then, targeting the computational bottleneck of the linear attention, we propose a novel Vicinity Attention Block to further reduce computation. In addition, because VVT has linear complexity, it facilitates processing feature maps or input images with higher resolutions. The proposed VVT has been validated with strong performance on both image classification and semantic segmentation. Future work will be aimed at using neural architecture search to further improve upon architectures specifically designed for linear vision transformers.

REFERENCES

[1] A. Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., 2017.
[2] T. Brown et al., “Language models are few-shot learners,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 1877–1901.
[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.
[4] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Representations, 2021, pp. 14138–14148.
[5] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 16000–16009.
[6] R. Ranftl, A. Bochkovskiy, and V. Koltun, “Vision transformers for dense prediction,” in Proc. Int. Conf. Comput. Vis., 2021, pp. 12179–12188.
[7] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, “Efficient transformers: A survey,” ACM Comput. Surv., New York, NY, USA: ACM, vol. 55, no. 6, pp. 1–28, 2021.
[8] Y. Xiong et al., “Nyströmformer: A nyström-based algorithm for approximating self-attention,” in Proc. AAAI Conf. Artif. Intell., 2021, pp. 14138–14148.
[9] H. Peng, N. Pappas, D. Yogatama, R. Schwartz, N. A. Smith, and L. Kong, “Random feature attention,” in Proc. Int. Conf. Learn. Representations, 2021.
[10] K. Choromanski et al., “Rethinking attention with performers,” in Proc. Int. Conf. Learn. Representations, 2021.
[11] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, “Transformers are RNNs: Fast autoregressive transformers with linear attention,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 5156–5165.
[12] Q. Zhen et al., “coFormer: Rethinking softmax in attention,” in Proc. Int. Conf. Learn. Representations, 2022.
[13] J. Li et al., “SOFT: Softmax-free transformer with linear complexity,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 21297–21309.
[14] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556.
[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Proc. Conf. Learn. Representations, 2012.
[17] H. Wu et al., “CVT: Introducing convolutions to vision transformers,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 22–31.
[18] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, “Transformer in transformer,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 15908–15919.
[19] L. Yuan et al., “Tokens-to-token VT: Training vision transformers from scratch on ImageNet,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 558–567.
[20] Z. Dai, H. Liu, Q. Le, and M. Tan, “CoAtNet: Marrying convolution and attention for all data sizes,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 3965–3977.
[21] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, “Bottleneck transformers for visual recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 16519–16529.
[22] Z. Liu et al., “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proc. Int. Conf. Comput. Vis., 2021, pp. 10012–10022.
[23] A. Vaswani, P. Ramachandran, A. Srinivas, N. Parmar, B. Hechtman, and J. Shlens, “Scaling local self-attention for parameter efficient visual backbones,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 12894–12904.
[24] P. Zhang et al., “Multi-scale vision longformer: A new vision transformer for high-resolution image encoding,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 2998–3008.
[25] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable DETR: Deformable transformers for end-to-end object detection,” in Proc. Int. Conf. Learn. Representations, 2021.
A. Hassani, S. Walton, J. Li, S. Li, and H. Shi, “Neighborhood attention,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 3464–3473.

H. Hu, Z. Zhang, Z. Xie, and S. Lin, “Local relation networks for image recognition,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 7794–7803.

J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 68–80.

X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 10076–10085.

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–777.

W. Wang et al., “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 568–578.

X. Chu et al., “Twins: Revisiting the design of spatial attention in vision transformers,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 9355–9366.

S. Tang, J. Zhang, S. Zhu, and P. Tan, “Quadtree attention for vision transformers,” 2022, arXiv:2201.02767.

R. Child, S. Gray, A. Radford, and I. Sutskever, “Generating long sequences with sparse transformers,” 2019, arXiv:1904.10509.

I. Beltagy, M. E. Peters, and A. Cohan, “Longformer: The long-document transformer,” 2020, arXiv:2004.05150.

N. Kotaev, L. Kaiser, and A. Levskaya, “Reformer: The efficient transformer,” in Proc. Int. Conf. Learn. Representations, 2020.

G. Daras, N. Kotaev, A. Odena, and A. G. Dimakis, “SMYRF: Efficient attention using asymmetric clustering,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 6476–6489.

S. Wang, B. Z. Li, M. Khabba, H. Fang, and H. Ma, “Linformer: Self-attention with linear complexity,” 2020, arXiv:2006.04768.

Z. Shen, M. Zhang, H. Zhao, S. Yi, and H. Li, “Efficient attention: Attention with linear complexities,” in Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis., 2021, pp. 3531–3539.

A. Ali et al., “XCiT: Cross-covariance image transformers,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 20014–20027.

J. Wang et al., “Deep two-view structure-from-motion revisited,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 8953–8962.

X. Cheng, Y. Zhong, Y. Dai, P. Ji, and H. Li, “Noise-aware unsupervised deep Lidar-Stereo fusion,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 6332–6341.

X. Chu, B. Zhang, Z. Tian, X. Wei, and H. Xia, “Do we really need explicit positional encodings for vision transformers?” 2021, arXiv:2102.10882v1.

K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, “What does BERT look at? An analysis of BERT’s attention,” 2019, arXiv:1906.04341.

O. Kovaleva, A. Romanov, A. Rogers, and A. Rumshisky, “Revealing the dark secrets of BERT,” 2019, arXiv:1908.08393.

H. Minkowski, Geometrie Der Zahlen, vol. 40. Rome, Germany: Teubner, 1910.

Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, “GCNet: Non-local networks meet squeeze-excitation networks and beyond,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops, 2019.

J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 712–7141.

A. Hussani, S. Walton, J. Li, S. Li, and H. Shi, “Neighborhood attention transformer,” 2022, arXiv:2204.07143.

W. Wang et al., “PVT V2: Improved baselines with pyramid vision transformer,” Comput. Vis. Media, vol. 8, no. 3, pp. 1–10, 2022.

R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 618–626.

W. Wang et al., “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 568–578.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 248–255.

H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers and distillation through attention,” 2021, arXiv:2012.12877.

Zhi-wei Wang received the double BE degrees in mechatronics engineering and mechanics engineering from ANU and Beijing Institute of Technology, Beijing, China. He is currently working toward the PhD degree with the Australian National University(ANU), Canberra, Australia. During his PhD he joined SenseTime research as a intern to work on Multi-Modal learning. His research interests include weakly-supervised segmentation and efficient transformer.

Weiuxan Sun received the BS degree in mathematics and applied mathematics from Fudan University, in 2017 and the MS degree in applied statistics from Tsinghua University, in 2020. He is a researcher from SenseTime. His current research areas are Efficient Network Design and Transformer.

Hui Deng received the BE and ME degrees from Northwestern Polytechnical University Xi’an, China, in 2019 and 2022 respectively. He is currently working toward the PhD degree with the Northwestern Polytechnical University(NWPU). During his PhD he joined sensetime research as a research intern to work on MultiModal Machine Learning. His current research interests including generative models, non-rigid structure from motion and depth estimation.

Jianyuan Wang received the BE degree with first-class honors from the Australian National University, in 2019. He is currently working toward the DPhil degree in the Visual Geometry Group, University of Oxford. He serves as the reviewer for IEEE Transactions on Pattern Analysis and Machine Intelligence, ICLR, CVPR, ICCV, and so on. His research interests include visual geometry and generative modelling.
Yi Zhang received the MS degree in advanced computer science from the University of Birmingham (UOB), Birmingham, U.K., in 2019. He is currently a computer vision researcher with SenseTime. His research interests include deep learning, computer vision and object detection, especially dense pedestrian detection, cross-modality object detection, and 3D reconstruction.

Kaihao Zhang is currently working toward the PhD degree with the College of Engineering and Computer Science, The Australian National University, Canberra, ACT, Australia. His research interests focus on computer vision and deep learning. He has more than 30 referred publications in international conferences and journals, including CVPR, ICCV, ECCV, NeurIPS, AAAI, ACMMM, IJCV, TIP, TMM, etc.

Nick Barnes (Senior Member, IEEE) received the BSc (Hons) and PhD degrees in robotic vision from the University of Melbourne, in 1994 and 1999, respectively. He is a professor with the School of Computing with the Australian National University. With NICTA, 2003-16, and CSIRO, he was a senior principal researcher and led the Computer Vision Research Group. He has best paper awards and nominations including from CVPR, Robotics and Systems Science, IROS, and DICTA.

Stan Birchfield received the PhD degree in electrical engineering, with a minor in computer science, from Stanford University. He is a principal research scientist and senior research manager from NVIDIA. He remains an adjunct faculty member with Clemson. He conducted research with Microsoft, was the Principal architect of a commercial product with a startup company in the Bay Area, co-founded a startup with collaborators with Clemson, and consulted for various companies. He has authored or co-authored more than 100 publications and a textbook on image processing and analysis. He regularly serves on the program committees and editorial boards of various leading conferences and journals in computer vision and robotics.

Lingpeng Kong received the PhD degree from School of Computer Science, Carnegie Mellon University. He is an assistant professor in the Department of Computer Science with the University of Hong Kong (HKU), and a co-director of the HKU NLP Lab. His work lies at the intersection of natural language processing (NLP) and machine learning (ML), with a focus on representation learning, structured prediction, and generative models. Before joining HKU, he was a research scientist with Google DeepMind.

Yiran Zhong received the PhD degree of engineering from the Australian National University, in 2021 and the MEng with the first class honor in information and electronics engineering from the Australian National University, in 2014. He is currently a principal investigator in Shanghai AI Laboratory. His research interests include self-supervised learning, visual geometry learning, multimodality learning, machine learning, and natural language processing. He won the ICIP Best Student Paper Award, in 2014.