EAPT: Efficient Attention Pyramid Transformer for Image Processing

Xiao Lin, Shuzhou Sun, Wei Huang, Bin Sheng, *Member, IEEE*, Ping Li, *Member, IEEE*, and David Dagan Feng, *Life Fellow, IEEE*

***Abstract*—Recent transformer-based models, especially patch-based methods, have shown huge potentiality in vision tasks. However, the split fixed-size patches divide the input features into the same size patches, which ignores the fact that vision elements are often various and thus may destroy the semantic information. Also, the vanilla patch-based transformer cannot guarantee the information communication between patches, which will prevent the extraction of attention information with a global view. To circumvent those problems, we propose an Efficient Attention Pyramid Transformer (EAPT). Specifically, we first propose the Deformable Attention, which learns an offset for each position in patches. Thus, even with split fixed-size patches, our method can still obtain non-fixed attention information that can cover various vision elements. Then, we design the Encode-Decode Communication module (En-DeC module), which can obtain communication information among all patches to get more complete global attention information. Finally, we propose a position encoding specifically for vision transformers, which can be used for patches of any dimension and any length. Extensive experiments on the vision tasks of image classification, object detection, and semantic segmentation demonstrate the effectiveness of our proposed model. Furthermore, we also conduct rigorous ablation studies to evaluate the key components of the proposed structure.**

***Index Terms*—Transformer, attention mechanism, pyramid, classification, object detection, semantic segmentation.**

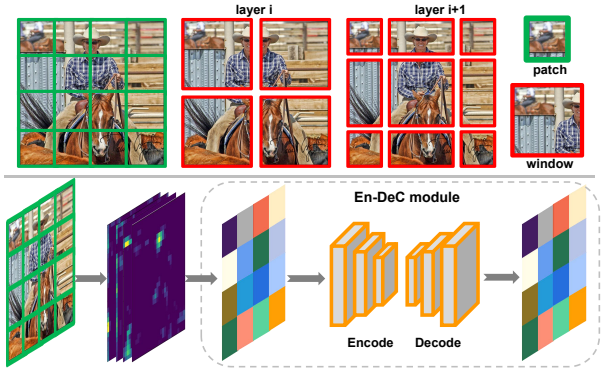


Fig. 1: The shifted window approach in Swin Transformer (this illustration refers to the original paper) and our proposed En-DeC module. Swin Transformer (top) shifts the windows of the i-th layer to obtain the windows of the $(i+1)$-th layer. Obviously, different windows cover different patches, so the communication among different patches can be guaranteed to a certain extent. To consider all the patches at once, our proposed En-DeC module (bottom) uses an encode-decode architecture to obtain communication information among all patches to get more complete global attention information.

I. Introduction

TRANSFORMER-BASED models have become de facto approaches of Natural Language Processing (NLP) because of their advantages in processing sequences [1]–[5]. And in the most recent, transformer has also achieved competitive performance in many vision tasks, such as image classification [6], [7], object detection [8], [9], segmentation [10], image generation [11], person re-identification [12], etc. Compared with language material, the resolution of visual data is higher, and thus the global-pixel-level attention calculations will yield an unbearable cost. To this problem, DEtection TRansformer (DETR) [13], which is the milestone of the vision transformer, uses a Convolutional Neural Network (CNN) to reduce the resolution of the input. Furthermore, ViT [6] abandons the CNN feature extractor and directly splits the input into fixed-size patches to build a pure vision transformer architecture. And many recent works also follow this method of processing of splitting high-resolution input. Albeit its prosperity, the above vanilla splitting-based methods still suffer from many thorny problems.

The one is split fixed-size patches may destroy semantic information. Unlike language elements, vision elements differ in size and shape. Thus, it is difficult for the fixed-size patches to cover various vision elements. To this issue, Deformable Patch-based Transformer (DPT) [14] splits the patches in a data-specific way to obtain non-fixed-size patches, i.e., it gets the learnable positions and scales in an adaptive way

Manuscript received June 01, 2021; revised September 09, 2021; accepted October 07, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61872241 and Grant 61572316, and in part by The Hong Kong Polytechnic University under Grant P0030419, Grant P0030929, and Grant P0035358. *(The first two authors contributed equally to this work.)*

Xiao Lin is with the Department of Computer Science, Shanghai Normal University, Shanghai 200234, China; and also with the Shanghai Engineering Research Center of Intelligent Education and Bigdata, Shanghai 200240, China (Email: lin6008@shnu.edu.cn).

Shuzhou Sun is with the Department of Computer Science, Shanghai Normal University, Shanghai 200234, China (Email: 1000479143@s-mail.shnu.edu.cn).

Wei Huang is with the Department of Computer Science and Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China (Email: 191380039@usst.edu.cn).

Bin Sheng is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (Email: shengbin@sjtu.edu.cn).

Ping Li is with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong (Email: p.li@polyu.edu.hk).

David Dagan Feng is with the Biomedical and Multimedia Information Technology Research Group, School of Information Technologies, The University of Sydney, Sydney, NSW 2006, Australia (Email: dagan.feng@sydney.edu.au).

TABLE I: Properties comparison of different position encoding methods.

Methods	Inductive	Data-driven	Symmetrical	Parallel	Parameter efficient
Sinusoidal [4]	✓	✗	✗	✓	✓
Embedding [15]	✗	✓	✗	✓	✗
Relative [16]	✗	✓	✗	✓	✓
FLOATER [17]	✓	✓	✗	✗	✓
MCMD	✓	✓	✓	✓	✓

for each patch. However, considering that the patch-based vision transformer usually calculates attention information in each patch, DPT may introduce a lot of extra calculations, especially when facing data with larger vision elements. In this paper, we propose Deformable Attention, which can combat the problem of fixed-size patches destroying semantic information without introducing additional attention. To be specific, the Deformable Attention provides a learnable offset for each position in the patch, and thus the calculation of attention information is no longer restricted by patches of fixed size and shape. Therefore, even with fixed-size patches, the Deformable Attention can still capture semantic information of various vision elements. In addition, the Deformable Attention does not change the size of the original patches, so it does not introduce additional attention calculation costs.

And the another thorny problem is that although the vanilla splitting-based transformer can avoid expensive global-pixel-level attention calculations, it also limits the communication of attention information among different patches. Obviously, the compromise of local-pixel-level attention calculations on computational cost affects the performance of the model. To this problem, Swin Transformer [8] calculates the attention information in the non-overlapping local windows (each window covers multiple patches) instead of in a single patch. Meanwhile, Swin Transformer uses the Shifted Window mechanism to make the window cover different patches in different attention calculation stages. Although Swin Transformer can promote communication among different patches, it is still local communication. In this work, we propose a global patch communication module based on the ecode-decode architecture, dubbed as En-DeC module (Encode-Decode Communication module). The En-DeC module first uses a encode model to compress the information of all patches. Then, the En-DeC module recovers the compressed information by a decode model and adds the recover results into the attention calculation results. Both our proposed En-DeC module and Shifted Window mechanism aim to communicate the information among different patches, but the former is global-communication and the latter is local-communication, and we show the difference in Fig. 1.

In addition to the above two thorny problems, current position encoding approaches in vision transformer either directly adopt low-dimensional encoding technology in NLP that cannot capture multi-dimensional location information or use learn-based encoding method that may increase learning cost [4], [15]–[17]. These facts motivate our novel position encoding design tailored for vision transformer. To achieve this goal, we propose to use Multi-dimensional Continuous Mixture Descriptor (MCMD) to describe the position information. Specifically, we use Gaussian to describe the position information of each dimension and then mix them. Benefits from the symmetrical and continuous properties of Gaussian, our descriptors can be of any length and the changes of the descriptors are smooth as well as symmetrical. Additionally, our position encoding method can establish the correlation between descriptors and parameters to be learned only by adjusting very few hyperparameters in Gaussian. Based on the above description, our proposed position encoding method obviously has the following properties: 1) **Inductive**. The ability to handle sequences of any dimension and any length; 2) **Data-Driven**. The encoding is affected by the data; 3) **Symmetrical**. The encoding of symmetrical position sequences is also symmetrical; 4) **Parallel**. Encoding does not affect the parallelization of the transformer; and 5) **Parameter efficient**. Encoding does not introduce too many additional parameters. We summarized the properties comparison of typical position encoding methods in Table I. Our work makes the following three main contributions:

- **Deformable Attention:** We propose a Deformable Attention, which starts with a learnable offset to remodel the rules for obtaining attention information. Compared with the vanilla fixed-size-patch-based transformer (e.g., ViT), Deformable Attention can better cover the various vision elements. And our work introduces less computational cost than similar deformable-based methods (e.g., DPT).
- **Encode-Decode Communication module (En-DeC module):** Although the vanilla splitting-based transformer can significantly reduce the computational cost, only calculating the attention information in a single patch also limits the communication among different patches. Some designs (e.g., Shifted Window in Swin Transformer) can alleviate this problem, but they are still local communication. Different from the previous methods, our proposed En-DeC module uses an encode-decode architecture to implement all patches communication.
- **Multi-dimensional Continuous Mixture Descriptor (MCMD):** We propose the MCMD, a position encoding tailored for vision transformer, which encodes different dimensions independently and then mixes them. MCMD satisfies the properties that an excellent position encoding should have, i.e., inductive, data-driven, symmetrical, parallel, parameter efficient.

The rest of this paper is organized as follows: Section II discusses the related work for transformer, especially Vision transformer, and the position encoding in it. The design details of the Efficient Attention Pyramid Transformer (EAPT) are given in Section III. In Section IV we cover the experimental results. Finally, Section V concludes this paper.

II. Related Work

A. Vision Transformer

Due to the strong representation ability of transformer, transformer-based models have been applied to multiple vision tasks in most recent. From high-level vision tasks object detection [8], [9] and segmentation [10] to low-level tasks

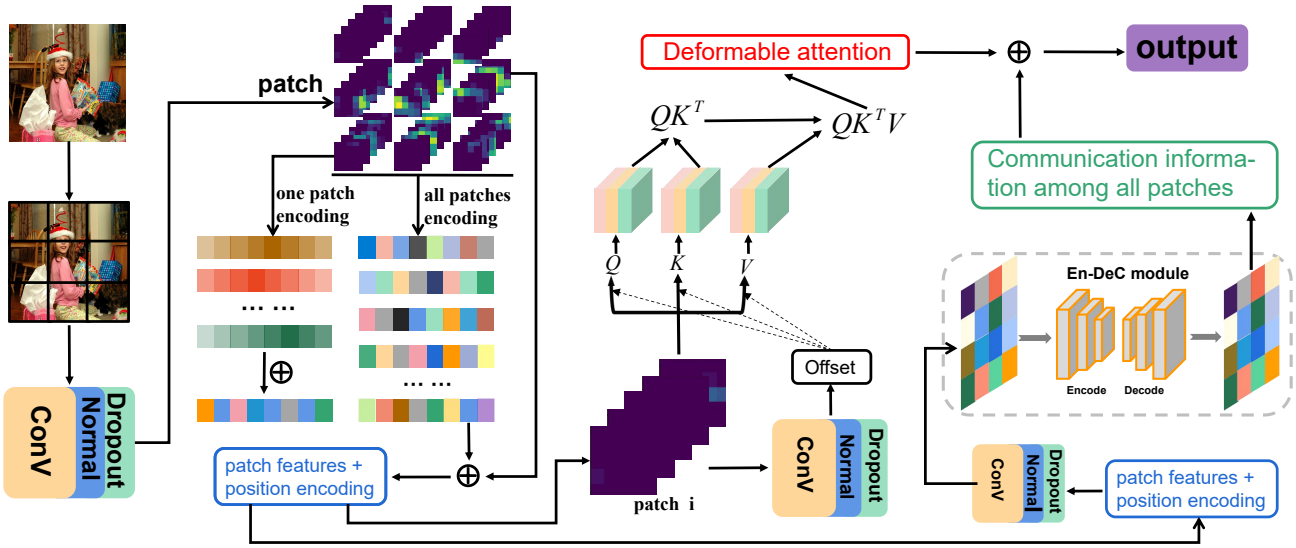


Fig. 2: The overall structure of our proposed Efficient Attention Pyramid Transformer (EAPT). EAPT first encodes the split fixed-size patches with our proposed Multi-dimensional Continuous Mixture Descriptor (MCMD), which is an efficient multi-dimensional feature position encoding method tailored for vision transformer (the result is the blue box). Then, EAPT uses the Deformable Attention to learn an offset for each position in the patches. Therefore, split fixed-size patches can still cover the various vision elements well (the result see red box). EAPT finally uses the En-DeC module to get communication information among all patches (see green box) and adds it to Deformable Attention above to get the final output.

such as image enhancement [18] and image generation [11] etc., vision transformer models have achieved competitive performance. Groundbreaking work DETR [13] introduces transformer to vision tasks for the first time. It starts with a convolutional neural network to extract features coupled with inputting those features to the transformer blocks to obtain the attention information. However, DETR [13] requires immense amounts of computational and gains poor performance on small objects due to its coarse-grained attention information extraction. Then, ViT [6] splits the input into fixed-size patches to avoid the unbearable cost of the global-pixel-level attention calculation. And much current work followed this patch-based processing mechanism. Although widely used, the patch-based methods still have many thorny troubles. The one is split fixed-size patches may destroy semantic information of vision elements. Deformable Patch-based Transformer (DPT) [14] splits the patches in a data-specific way to obtain non-fixed-size patches to combat this trouble. However, non-fixed-size patches may introduce a lot of additional attention calculation costs. The another is the patch-based transformers limit the communication of attention information among different patches. Although the non-overlapping local windows in Swin Transformer can promote communication among different patches, this is still local communication. For these thorny troubles, we propose Efficient Attention Pyramid Transformer (EAPT), which consists of Deformable Attention and Encode-Decode Communication module (En-DeC module). The former learns an offset for each position in the patch, so it makes the fixed size patches can better cover the various vision elements. The latter uses an encode-decode architecture to make the communications among all patches. Overall, EAPT is a patch-based transformer, but it can alleviate the thorny troubles in the vanilla fixed-size-patch-based transformer due to our proposed Deformable Attention and En-DeC module.

B. Position Encoding in Transformer

Unlike the Recurrent Neural Network (RNN) [19]–[21] the transformer cannot distinguish the position information of the tokens and thus requires to perform position encoding for inputs [22]. The vanilla transformer [4] adopts Sinusoidal-based position encoding methods, and its properties of continuous and unbounded allow it to encode tokens of any length. However, this method is non-data-driven, i.e., it cannot adaptively change the position encoding according to the token itself, so it cannot guarantee the same position guidance for tokens with different weights. Given the central role of position encoding, many attempts in the field of language processing improve the sinusoidal-based encoding method in the original transformer paper from multiple perspectives, and among these, absolute encoding [4], relative encoding [16], recursive encoding [17], learn-based encoding [15], etc. are typical approaches. Relative encoding [16] hypothesizes that precise relative position information is not useful beyond a certain distance and thus can use clipping to encode any amount of sequences. Learn-based encoding [15] learns position encoding through training, while recursive encoding [17] adopts Neural ODE. Although these methods have been effective for transformer model in NLP, we argue that it is obviously inappropriate to use them directly in vision transformer because the data in vision tasks are high-dimensional and these methods are not capable of capturing multi-dimensional position information, and we will analyze the limits of current position encoding methods in detail in Section III. To this issue, we propose a novel position encoding method specifically for high-dimensional data in this

paper. We encode different dimensions independently and then mix them to ensure that the encoding has the same guidance for the position information of different dimensions.

III. Proposed Approach

This section presents implementation methods and details of our proposed. First, we will introduce the deformable attention mechanism, which can obtain richer attention information without increasing the computation cost. Then, we propose the Encode-Decode Communication module (En-DeC module), which uses an encode-decode architecture to obtain communication information among all patches. We finally design a novel position encoding specifically for vision transformer named Multi-dimensional Continuous Mixture Descriptor (MCMD), which can efficiently model any amount of patches of multi-dimensional position information. We show the above three designs respectively in Fig. 2.

A. Deformable Attention

Following recent excellent work Swin Transformer [8], we also compute self-attention in non-overlapped windows. Suppose there is a multi-dimensional feature $\mathcal{V} \in \mathbb{R}^{\mathrm{h}\times w\times c}$, which is divided into $h_p \times w_p$ patches and denoted as $\mathcal{V}_p \in \mathbb{R}^{h_p\times w_p\times \frac{\mathrm{h}\times \mathrm{w}}{h_p\times w_p}\times c}$. We then further divide it into $h_w \times w_w$ windows, and the results can be denoted as $\mathcal{V}_w \in \mathbb{R}^{h_w\times w_w\times \frac{\mathrm{h_p}\times \mathrm{w_p}}{h_w\times w_w}\times c}$. Without loss of generality, we only discuss the attention calculation of a single window here. For a single window w_m in all windows $\mathrm{w} = \{w_1, w_2, \ldots, w_n\}$, assume that its all position on the original feature is (l_m^x, l_m^y). In order to allow the current window to have a larger view of attention, we add a learnable offset (o^x, o^y) to each position of this window, and the overall offset of (l_m^x, l_m^y) is (o_m^x, o_m^y). (o_m^x, o_m^y) is learned from Wf, specifically, $(o_m^x, o_m^y) = Dense(Conv(Wf))$, where $Dense(\cdot)$ and $Conv(\cdot)$ are fully connected layer and convolution layer, respectively. In order to reduce the learning difficulty of offset, we perform a value constraint on it, that is, $|\mathrm{o_m^x}| \leq \mathrm{h_w}$, $|\mathrm{o_m^y}| \leq \mathrm{w_w}$. We denote the position of window (l_m^x, l_m^y) added with the learned offset is (L_m^x, L_m^y), and it can be calculated as:

$$(L_m^x, L_m^y) = (l_m^x, l_m^y) + (o_m^x, o_m^y) \tag{1}$$

Considering that (L_m^x, L_m^y) should not exceed the boundary of features, we need to further restrict its value. Here we define the multi-dimensional boundary constraint kernel $C(\cdot,\cdot)$, and

$$\begin{aligned} C((\mathrm{x},\mathrm{y}),(\mathrm{h},\mathrm{w})) = (&\max(\min(x+h,h),1),\\ &\max(\min(y+w,w),1)) \end{aligned} \tag{2}$$

where (x, y) and (h, w) are the positions to be bounded and the boundary of features, respectively. We use $C(\cdot,\cdot)$ to restrict (L_m^x, L_m^y), and denote its result as $(\mathcal{L}_m^x, \mathcal{L}_m^y)$. Note that the offset is learnable and is typically fractional and thus $(\mathcal{L}_m^x, \mathcal{L}_m^y)$ corresponds to the virtual position in the feature. We use Bilinear Interpolation here to calculate the value of the virtual position. Suppose there is a virtual point (x, y) on the feature, and its value $V_{(x,y)}$ can be calculated as:

$$V_{(x,y)} = \sum_{q_{(x,y)}} |q_{(x,y)}^x - x||q_{(x,y)}^y - y| q_{(x,y)} \tag{3}$$

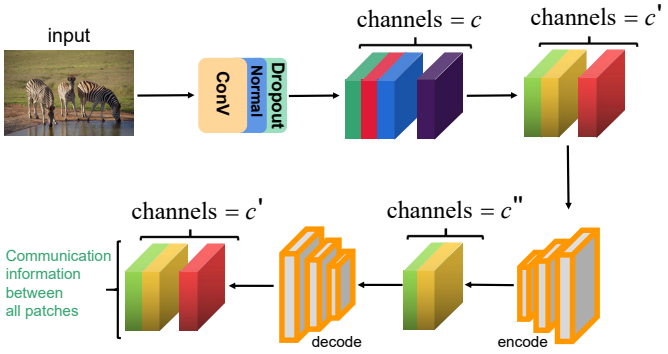


Fig. 3: The illustration of the En-DeC module. The En-DeC module first compresses the channels of features including all patches from c to c'. Then, the En-DeC module compresses and recovers the above results to obtain communication information among all patches. Note that the final feature channel is c' instead of c. The purpose of this design is to reduce the calculation cost of the En-DeC module.

where $q_(x, y)$ enumerates the four actual positions closest to this virtual position.

B. En-DeC module

The vanilla splitting-based transformer calculates attention information within a single patch, which will lead to weak performance. To this problem, we propose the En-DeC module, which is based on an encode-decode architecture and can allow information exchange among different patches. Without loss of generality, for a multi-dimensional inputs $\mathcal{V} \in \mathbb{R}^{\mathrm{h}\times w\times c}$, it is processed by the En-DeC module as follows: The En-DeC module first uses a encode architecture to extract features from $\mathcal{V}$, the rules are:

$$\begin{cases} \mathcal{V}' = \mathrm{Con}\,V\,(\mathcal{V}, c') \\ \mathcal{V}'' = \mathrm{Con}\,V\,(\mathcal{V}', c'') \\ \mathcal{V}''_{i,j} = \mathcal{Q}\left(\widetilde{W_{en}}\mathcal{V}'_{(i'\sim i'',j'\sim j'')}\right), \forall i \in h, \forall j \in w \end{cases} \tag{4}$$

where $\mathrm{Con}\,V(X, c)$ is the channel compression kernel that uses a convolution layer, which means compressing the channel of X into c dimensions. The channels with larger variances usually include more feature information, and this has been observed in many previous works [23]–[25]. $\mathcal{V}_{(i'\sim i'',j'\sim j'')}$ represents a matrix with the same shape as the convolution kernel weight $\widetilde{W_{en}}$. $\mathcal{Q}$ enumerates all convolution kernels and input matrix product operations. c' and c'' are the number of compressed channels and the number of convolution kernels, respectively. $\mathcal{V}'' \in \mathbb{R}^{\mathrm{h}'\times w'\times c''}$ is the output of the encode architecture, and $\mathcal{V}''_{i,j}$ is the value of a single position of the output. Note that, we only show here the convolution processing on the input once. We can perform convolution processing on it as many times as needed, and the same rules apply to the decode architecture. Meanwhile, the encode architecture first compresses the input channel to c' dimension, and the restore result of decode architecture is also c' dimension instead of c dimension. The purpose of this is to reduce the computational cost.

Then, the En-DeC module uses a decode architecture to restore the shape of $\mathcal{V}'' \in \mathbb{R}^{\mathrm{h}' \times w' \times c''}$ to $\mathbb{R}^{\mathrm{h} \times w \times c'}$, the rules are:

$$\begin{cases} \mathcal{V}''' = \mathrm{DeCon}\, V\left(\mathcal{V}'', c'\right) \\ \mathcal{V}'''_{i,j} = \mathcal{Q}\left(\widetilde{W_{de}} V''_{(i' \sim i'', j' \sim j'')}\right), \forall i \in h', \forall j \in w' \end{cases} \tag{5}$$

where $\mathrm{DeCon}\, V(X, c)$ is the Transposed Convolution kernel, which can expand the input to the target shape. $\mathcal{V}'''$ is the output of the decode architecture, and $\mathcal{V}'''_{i,j}$ is the value of a single position of the output. We argue that $\mathcal{V}'''$ includes patches communication information, and we directly add it to the attention information obtained by the vanilla splitting-based transformer. We show the details of the En-DeC module in Fig. 3.

C. Our Proposed Multi-dimensional Position Encoding

1) Limits of Current Position Encoding Methods: Multi-Head Attention is an important component of the transformer, For inputs $f \in \mathbb{R}^{\mathrm{L} \times d}$, its result is:

$$A(f) = \mathrm{soft\,max}[\frac{W^Q f (W^K f)^{\mathrm{T}}}{\sqrt{d}}] W^{\mathrm{V}} f \tag{6}$$

where $W^Q f$, $W^K f$, and $W^V f$ are query, keys, and values respectively. $\sqrt{d}$ is the dimension of $W^Q f (W^K f)^{\mathrm{T}}$. However, such an attention mechanism is not capable of distinguishing the position information of the input sequence and thus requires position encoding. Current position encoding approaches in vision transformer either directly adopt low-dimensional encoding technology in NLP or use learn-based encoding method that may increase the difficulty of model learning. We argue that those methods are obviously inappropriate to use directly in vision transformer because the data in vision tasks are high-dimensional and these methods are not capable of capturing multi-dimensional position information. Suppose there is a multi-dimensional feature $f \in \mathbb{R}^{\mathrm{h} \times w \times c}$, which is divided into $p \times p$ patches. If a one-dimensional position encoding is used, we denote the position encoding of the sequence $xp{+}y{+}t$ as t, where x and y are position information in different dimensions respectively. Then the output of this sequence through the model reasoning is:

$$F(xp{+}y{+}t) = F(xp) + \frac{\partial f}{\partial p}(y{+}t) + \frac{\partial^2 f}{\partial p^2}(y{+}t) + \cdots + R_n(p) \tag{7}$$

where $R_n(x)$ is the Taylor remainder, and $F(x)$ represents the model output of inputs x. We can observe that although the position information of different dimensions should be the same, the degree of influence of t on x and y is different. Therefore, we argue that low-dimensional encoding is not enough to represent the position information of high-dimensional data.

And if the encoding of position (x, y) is multi-dimensional, its output by model reasoning is:

$$\begin{aligned} F(x + t_x, y + t_y) = F(x, y) + \frac{\partial F}{\partial x} t_x + \frac{\partial F}{\partial y} t_y + \frac{1}{2} \frac{\partial^2 F}{\partial x^2} t_x^2 + \\ \frac{1}{2} \frac{\partial^2 F}{\partial y^2} t_y^2 + \frac{1}{2} \frac{\partial^2 F}{\partial x \partial y} t_x t_y + \cdots + R_n(x, y) \end{aligned} \tag{8}$$

where t_x and t_y are position information of different dimensions. $F(x, y)$ represents the model output of inputs (x, y), and $R_n(x, y)$ is the Taylor remainder. We can find that t_x and t_y have the same impact on x and y, which is in line with our expectations of position encoding for high-dimensional data. Inspired by this, we propose a novel position encoding method tailored for high-dimensional data named MCMD. It encodes different dimensions independently and then mixes them to ensure that the encoding has the same guidance for the position information of different dimensions.

2) Multi-dimensional Continuous Mixture Descriptor (M-CMD): Suppose there is a multi-dimensional feature $\mathcal{V} \in \mathbb{R}^{\mathrm{h} \times w \times c}$, which is divided into $p \times p$ patches and denoted as $\mathcal{V}_p \in \mathbb{R}^{\mathrm{p}^2 \times \frac{\mathrm{hwc}}{\mathrm{p}^2}}$. Obviously, the amount and length position encoding required for $\mathcal{V}_p$ are p^2 and $\frac{\mathrm{hwc}^2}{\mathrm{p}^2}$ respectively.

Simple but without loss generality, here we only discuss the position encoding of one patch. For the patch of position (i, j), we first describe it as two binary vectors $\mathcal{V}_{(ij,x)}$ $[0, 0, \cdots, 1, \cdots]$ and $\mathcal{V}_{(ij,y)}$ $[0, 0, \cdots, 1, \cdots]$, where the length of vector is p, the i-th of $\mathcal{V}_{(ij,x)}$ and j-th of $\mathcal{V}_{(ij,y)}$ are 1 and others are 0. We use Gaussian Function $ae^{-(x-b)^2/2c^2}$ to describe $\mathcal{V}_{(ij,x)}$ and $\mathcal{V}_{(ij,y)}$ respectively and denote them as $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})$ and $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)})$, where,

$$\begin{aligned} \mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)}) &= [G(\mathcal{V}^{x^1}_{(\mathrm{ij},x)}), G(\mathcal{V}^{x^2}_{(\mathrm{ij},x)}), \cdots, G(\mathcal{V}^{x^{(\mathrm{hwc}/p^2)}}_{(\mathrm{ij},x)})] \\ \mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)}) &= [G(\mathcal{V}^{y^1}_{(\mathrm{ij},y)}), G(\mathcal{V}^{y^2}_{(\mathrm{ij},y)}), \cdots, G(\mathcal{V}^{y^{(\mathrm{hwc}/p^2)}}_{(\mathrm{ij},y)})] \end{aligned} \tag{9}$$

Among this, $x^t = (b^x_{ij} + 0.5) \times hwc/p^3 + t$, $y^t = (b^y_{ij} + 0.5) \times hwc/p^3 + t$, $t \in [1, hwc/p^2]$. And $G(\mathcal{V}^x_{(\mathrm{ij},x)})$ and $G(\mathcal{V}^y_{(\mathrm{ij},y)})$ can be calculated as follow:

$$\begin{aligned} G(\mathcal{V}^x_{(\mathrm{ij},x)}) &= a_{ij} e^{-(x - b^x_{ij})/2c^{x^2}_{ij}} \\ G(\mathcal{V}^y_{(\mathrm{ij},y)}) &= a_{ij} e^{-(y - b^y_{ij})/2c^{y^2}_{ij}} \end{aligned} \tag{10}$$

where $b^x_{ij} = i$ and $b^y_{ij} = j$, c^x_{ij} and c^y_{ij} are adjustable hyperparameters and $c^x_{ij} \neq c^y_{ij}$, this can guarantee the uniqueness of position encoding. a_{ij} is related to the parameters of $\mathcal{V}$, which we will describe in detail in the following section. We next add the Gaussian descriptors corresponding to each dimension to get the final position encoding, that is, for $\mathcal{V}_{(i,j)}$, its position encoding $\mathcal{P}_{\mathcal{V}(i,j)}$ can be calculated as:

$$\mathcal{P}_{\mathcal{V}_{(i,j)}} = \mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)}) + \mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)}) \tag{11}$$

where $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})$ and $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)}$ are the position codes of $\mathcal{V}_{(i,j)}$ in the x and y dimensions, respectively. As a relevant illustrate case, the feature $\mathcal{V}$ we use here is three dimensional. In fact, our position encoding method can be extended to any dimension features, that is, each dimension is encoded like $\mathcal{P}_{\mathcal{V}_{(i,j)}}$, and finally the position encoding results of all dimensions are added.

3) Design for Data-driven: Feature-agnostic encoding, such as Sinusoidal in the original Transformer paper, cannot guarantee that the coding of each patch can perform the same position guiding. For predefined position encoding, if the preset value is large, it will affect the semantic information of the feature itself. While if the preset value is small, it is difficult to guarantee that effective position information can be provided. Additionally, the values of different patches are also different, so it is obviously unreasonable to use value-agnostic

TABLE II: Detailed architecture of Efficient Attention Pyramid Transformer (EAPT). c, c', and c'' are the number of feature channels, which have been described in detail in Section III. The input of the example shown here is 256×256, and the patch size is 4×4, so initially, we can get 4096 patches.

	output size	EAPT-S	EAPT-M	EAPT-L
stage 1	64 × 64 (4×)	[dim = 96, head = 3 position encoding : MCMD En-DeC module : $c = 96, c' = 1, c'' = 2$] × 3	[dim = 128, head = 4 position encoding : MCMD En-DeC module : $c = 128, c' = 2, c'' = 4$] × 3	[dim = 192, head = 6 position encoding : MCMD En-DeC module : $c = 192, c' = 4, c'' = 8$] × 3
stage 2	32 × 32 (8×)	[dim = 192, head = 6 position encoding : MCMD En-DeC module : $c = 192, c' = 2, c'' = 4$] × 3	[dim = 256, head = 8 position encoding : MCMD En-DeC module : $c = 256, c' = 4, c'' = 8$] × 3	[dim = 384, head = 12 position encoding : MCMD En-DeC module : $c = 384, c' = 8, c'' = 16$] × 3
stage 3	16 × 16 (16×)	[dim = 384, head = 12 position encoding : MCMD En-DeC module : $c = 384, c' = 4, c'' = 8$] × 10	[dim = 512, head = 16 position encoding : MCMD En-DeC module : $c = 512, c' = 8, c'' = 16$] × 16	[dim = 768, head = 24 position encoding : MCMD En-DeC module : $c = 768, c' = 16, c'' = 32$] × 16
stage 4	8 × 8 (32×)	[dim = 768, head = 24 position encoding : MCMD En-DeC module : $c = 768, c' = 8, c'' = 16$] × 3	[dim = 1024, head = 32 position encoding : MCMD En-DeC module : $c = 1024, c' = 16, c'' = 32$] × 3	[dim = 1536, head = 48 position encoding : MCMD En-DeC module : $c = 1536, c' = 32, c'' = 64$] × 3

encoding for them. While the data driven property means that the encoding and the features are related, so it can guarantee the same position guidance for features with different values.

Because of $\int_{-\infty}^{+\infty} ae^{-(x-b)^2/2c^2} d_x = \sqrt{2}a|\mathrm{c}|\sqrt{\Pi}$, we only need to change the value of a or c to change the distribution of values. But we have used c as a hyperparameter to ensure the uniqueness of position encoding. Therefore, here we proposed to change a based on the value of the feature to be encoded to achieve data-driven. Following the definition in section 3.1, for $\mathcal{V}_{(i,j)}$, its position encoding is $\mathcal{P}_{\mathcal{V}_{(i,j)}} = [G(\mathcal{V}^{x^1}_{(\mathrm{ij},x)}), G(\mathcal{V}^{x^2}_{(\mathrm{ij},x)}), \cdots, G(\mathcal{V}^{x^{(\mathrm{hwc}/p^2)}}_{(\mathrm{ij},x)})] + [G(\mathcal{V}^{y^1}_{(\mathrm{ij},y)}), G(\mathcal{V}^{y^2}_{(\mathrm{ij},y)}), \cdots, G(\mathcal{V}^{y^{(\mathrm{hwc}/p^2)}}_{(\mathrm{ij},y)})]$, where $G(\mathcal{V}^{x}_{(\mathrm{ij},x)}) = a_{ij}e^{-(x-b^x_{ij})/2c^{x^2}_{ij}}$ and $G(\mathcal{V}^{y}_{(\mathrm{ij},y)}) = a_{ij}e^{-(y-b^y_{ij})/2c^{y^2}_{ij}}$. And we calculated a_{ij} as:

$$a_{ij} = \frac{p^2}{hwc}\,\mathrm{Sum}(\mathcal{V}_{(i,j)}) \tag{12}$$

where $\mathrm{Sum}(p)$ is the sum of value in patch p. Therefore, the $\mathcal{P}_{\mathcal{V}_{(i,j)}}$ and the values of $\mathcal{V}_{(i,j)}$ are related, so they can provide equal position guidance for different sequences.

4) Properties of Multi-dimensional Continuous Mixture Descriptor: **Inductive.** Inductive refers to the ability to handle sequences of any length, position encoding technology with this property can process inputs with any shapes. Obviously, our proposed MCMD satisfies this property. Following the illustrated case in Section 3.1, here we change the number of patches, $(p, p) \to (p', p')$. Therefore, the number and length of position encoding become $\mathrm{p'}^2$ and $\frac{\mathrm{hwc}^2}{\mathrm{p'}^2}$, respectively. We only need to replace p in Equation 1 with p' to get a Gaussian descriptor of length $\frac{\mathrm{hwc}^2}{\mathrm{p'}^2}$. Our method is works for p ($p \in \mathbb{N}^+$) patches. For the $\mathcal{V} \in \mathbb{R}^{\mathrm{h}\times w\times c}$, in extreme cases, the number of patches can be $\mathrm{h} \times w$. Similarly, we only need to replace $\frac{\mathrm{hwc}^2}{\mathrm{p}^2}$ in Equation 1 with $\mathrm{h} \times w$ to get the relative position encoding.

Inductive in MCMD not only allows position encoding for sequences of any length but also includes relative position between sequences. For $\mathcal{V}_{(i,j)}$ and $\mathcal{V}_{(i+t',j)}$, without considering $a_{ij} \neq a_{ji}$, it is obviously $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)}) = \mathcal{G}(\mathcal{V}_{((\mathrm{i+t'})\mathrm{j},y)})$. Meanwhile, $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)}) = [G(\mathcal{V}^{x^1}_{(\mathrm{ij},x)}), \cdots, G(\mathcal{V}^{x^{(\mathrm{hwc}/p^2)}}_{(\mathrm{ij},x)})]$, $\mathcal{G}(\mathcal{V}_{((\mathrm{i+t'})\mathrm{j},x)}) = [G(\mathcal{V}^{x^{1+t'}}_{((\mathrm{i+t'})\mathrm{j},x)}), \cdots, G(\mathcal{V}^{x^{(\mathrm{hwc}/p^{2}+t')}}_{((\mathrm{i+t'})\mathrm{j},x)})]$, the latter can be regarded as the result of translation of the former on the Gaussian function. Therefore, we can get $\mathcal{G}(\mathcal{V}_{((\mathrm{i+t'})\mathrm{j},x)})$ based on $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})$, that is, our encoding method including relative position information, which is very useful for sequence processing but some other methods (e.g., learning-based methods) cannot benefit from it.

Symmetrical. Symmetrical refers to the position encoding of the symmetrical patch is also symmetrical. Symmetrical allows encoding to benefit from the relative positions information of the patches. Our proposed position encoding is approximately symmetrical, that is, for $\mathcal{P}_{\mathcal{V}_{(i,j)}} \approx \overleftarrow{\mathcal{P}_{\mathcal{V}_{(j,i)}}}$, where $\overleftarrow{\mathcal{P}_{\mathcal{V}_{(\mathrm{j,i})}}}$ is symmetric descriptor of $\mathcal{P}_{\mathcal{V}_{(j,i)}}$. For simplicity, we only prove that $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})^t \approx \overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^t}$, where $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})^t$ is t-th value of of $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})$.

Because $G(\mathcal{V}^{x}_{(\mathrm{ij},x)})$ is symmetrical, $\overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^t} = \overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^{p-t-1}} = \mathrm{a_{ji}e}^{-\frac{[(\mathrm{p-t-1})-\mathrm{b^x_{ji}}]^2}{2\mathrm{c^x_{ij}}^2}}$. Further translate $\overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^{p-t-1}}$ to get $\overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^{p+t+2j+1}}$, and $\overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^{p+t+2j+1}} = \mathrm{a_{ji}e}^{-\frac{[(-\mathrm{p+t+2j+1})-\mathrm{b^x_{ji}}]^2}{2\mathrm{c^x_{ij}}^2}} = \mathrm{a_{ji}e}^{-\frac{(\mathrm{t-b^x_{ij}})^2}{2\mathrm{c^x_{ji}}^2}}$. c^x_{ji} and c^x_{ij} are just hyperparameters used to ensure the uniqueness of position encoding, so we can set $c^x_{ji} = c^x_{ij}$. But note that a_{ij} and a_{ji} are related to the value of patch, so $a_{ij} \neq a_{ji}$ and thus the only difference between $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})^t$ and $\overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^t}$ is a_{ij} and a_{ji}. therefore, under our proposed position encoding method, $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},x)})^t \approx \overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},x)})^t}$. Similarly, $\mathcal{G}(\mathcal{V}_{(\mathrm{ij},y)})^t \approx \overleftarrow{\mathcal{G}(\mathcal{V}_{(\mathrm{ji},y)})^t}$. Because our proposed position encoding method is data-driven, it is approximately symmetrical rather than absolutely symmetrical. However, we argue that this is sufficient to provide relative position information. Additionally, due to different values of patches, absolutely symmetrical encoding is unreasonable.

Parallel. The main difference between Transformer and RNNs is that the former is parallel while the latter is not. Our position encoding method can be integrated into the parallel of Transformer at a very low cost, while some of the other encoding methods require very high time and memory costs, such as recursive-based approaches. Following formula 1, our method can be directly integrated into Transformer block, that

is:

$$A(f) = \text{soft}\max\{\frac{W^Q(f+P_f)[W^K(f+P_f)]^{\mathrm{T}}}{\sqrt{d}}\}W^{\mathrm{V}}(f+P_f) \tag{13}$$

where $\mathcal{P}_f$ is the position encoding of the input f. $W^Q(f+P_f)$, $W^K(f+P_f)$, and $W^{\mathrm{V}}(f+P_f)$ are respectively query, keys, and values with position codes $\mathcal{P}_f$. $\sqrt{d}$ is the dimension of $W^Q(f+P_f)[W^K(f+P_f)]^{\mathrm{T}}$.

IV. Experiments

We evaluate our proposed model on three vision tasks, i.e., classification, object detection, and semantic segmentation. First, we introduce the details of the experiments, including network architecture, compared tasks and dataset, and training settings. Then, we compare our architecture with traditional methods including convolution-based models and transformer-based models. We lastly conduct rigorous ablation studies to evaluate the key components of the proposed structure including Deformable Attention, En-DeC module, Multi-dimensional Continuous Mixture Descriptor (MCMD).

A. Experiments details

Network architecture. Follow Swin Transformer [8], our network structure also consists of multiple stages, and each includes multiple transformer blocks. The transformer blocks within a stage will not change the shape of the feature map, and we perform downsampling between stages, which makes our network easy to apply to the existing vision framework. In order to fit the different computational requirements, we design three network architectures with different complexity, dubbed as EAPT-S, EAPT-M, and EAPT-L respectively. The main difference between these three architectures is the amount of feature map channels and transformer blocks. Generally speaking, the more transformer blocks and channels, the higher the complexity of the network. Meanwhile, each transformer block includes En-DeC modules with different settings, which can ensure the communication of information among all patches. We report the detailed network architectures in Table II. As a relevant use case, the input size we show in Table II is 256×256, and we have also verified multiple input sizes in the following experiment.

Compared tasks and dataset. We use ImageNet [34], COCO 2017 [35], and ADE20K [36] to verify our work on the vision tasks classification, object detection, and semantic segmentation, respectively. ImageNet is a very large dataset, and its commonly used subset ILSVRC2012 (ImageNet2012) still has more than one million training images. We use ImageNet-1K here, it has 1.28M training images and 50K validation images. COCO 2017 is a challenging dataset with 80 categories, and including 118k training images, 5k validation images, and 20k test images. ADE20K is a semantic segmentation dataset, which contains 20k training images, 2k validation images and, 3k test images.

Training settings. We follow most of the training techniques and tricks in Swin Transformer [8] and [7], the detailed settings are as follows. Training settings for classification: the optimizer is AdamW [37], initial learning rate=0.001, the warmup learning rate is 0.05 and, warmup steps is 20 epochs, batchsize = 1024. Most of the augmentation and regularization strategies in Swin Transformer [8] and [7] are used in experiments. Training settings for object detection: We follow the four object detectors used in Swin Transformer, i.e., Cascade Mask R-CNN [28], [29], ATSS [30], RepPoints v2 [31], and Sparse RCNN [32]. The optimizer is AdamW, initial learning rate=0.0001, the warmup learning rate is 0.05, batchsize = 16. Training settings for semantic segmentation: we adopt UperNet in mmsegmentation as the evaluation framework. The optimizer is AdamW, initial learning rate=0.00006, the warmup learning rate is 0.01, and warmup step is 1500 iterations. We also adopt the training techniques and tricks used in Swin Transformer, such as data augmentation, stochastic

TABLE III: Classification results on ImageNet-1K. The baselines we compared here include convolutional neural network based architecture (RegNetY [26] and EfficientNet [27]) and transformer-based architecture (ViT [6], Swin Transformer [8], and DeiT [7]). Follow Swin Transformer, throughput is measured on a V100 GPU.

method	inputs	#param	FLOPs	throughput	top-1 acc.
RegNetY-4G [26]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [26]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [26]	224^2	84M	16.0G	334.7	82.9
EffNet-B3 [27]	300^2	12M	1.8G	732.1	81.6
EffNet-B4 [27]	380^2	19M	4.2G	349.4	82.9
EffNet-B5 [27]	456^2	30M	9.9G	169.1	83.6
EffNet-B6 [27]	528^2	43M	19.0G	96.9	84.0
EffNet-B7 [27]	600^2	66M	377.0G	55.11	84.3
ViT-B/16 [6]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [6]	384^2	307M	170.7G	27.3	76.5
DeiT-S [7]	224^2	22M	4.6G	940.4	79.8
DeiT-B [7]	224^2	86M	17.5G	292.3	81.8
DeiT-B [7]	384^2	86M	55.4G	85.9	83.1
Swin-T [8]	224^2	29M	4.5G	755.2	81.3
Swin-S [8]	224^2	50M	8.7G	436.9	83.0
Swin-B [8]	224^2	88M	15.4G	278.1	83.3
Swin-B [8]	384^2	88M	40.7G	84.7	84.2
EAPT-S	224^2	39M	6.5G	695.2	82.9
EAPT-M	224^2	107M	18.3G	242.6	84.6
EAPT-M	384^2	116M	53.6G	75.3	86.1

TABLE IV: Frameworks-level comparison results on the task of object detection on COCO 2017. The baselines of detection frameworks we compared here include Cascade Mask R-CNN [28], [29], ATSS [30], RepPoints v2 [31], and Sparse R-CNN [32]. And the compared network architectures include ResNet [33] and Swin Transformer [8].

Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param	FLOPs	FPS
Cascade Mask R-CNN	R-50	46.3	64.3	50.5	82M	739G	18.0
	Swin-T	50.5	69.3	54.9	86M	745G	15.3
	EAPT-S	51.8	70.4	56.2	93M	771G	13.7
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	47.2	66.5	51.3	36M	215G	22.3
	EAPT-S	48.3	67.9	52.6	42M	236G	19.7
RepPoints-V2	R-50	46.5	64.6	50.3	42M	274G	13.6
	Swin-T	50.0	68.5	54.2	45M	283G	12.0
	EAPT-S	51.3	59.9	55.7	53M	301G	10.1
Sparse R-CNN	R-50	44.5	63.4	48.2	106M	166G	21.0
	Swin-T	47.9	67.3	52.3	110M	172G	18.4
	EAPT-S	48.9	68.6	54.1	129M	201G	15.9

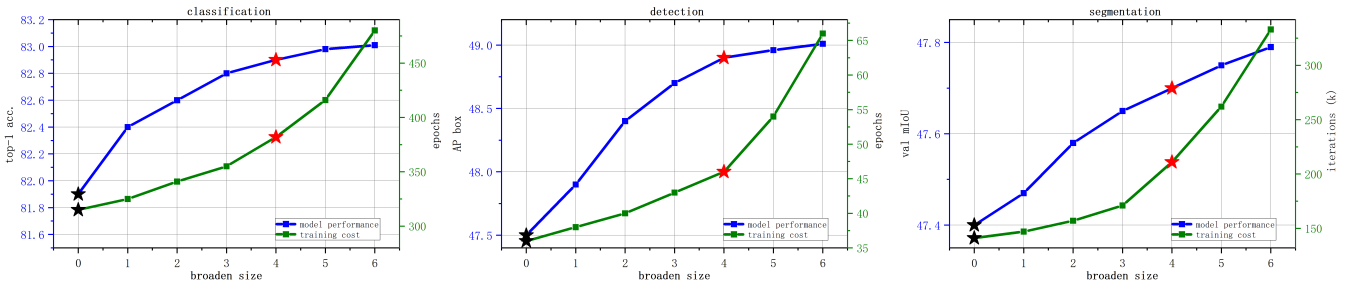


Fig. 4: The performance of EAPT-S in different broaden sizes. The experiment follows the settings in Table III, Table IV, and Table VI. ★($|o_m^x| = |o_m^y| = 0$) means that the Deformable Attention is not used, and ★ ($|o_m^x| = |o_m^y| = 4$) is the used broaden size in this paper.

TABLE V: System-level comparison results on the task of object detection on COCO 2017. The baselines we compared here include convolutional neural network based architecture and transformer-based architecture. HTC++ is the ingenious design improved from HTC in Swin Transformer, and we follow and improve this design and denote it HTC+++. * indicates multi-scale testing.

Method	mini-val		test-dev		#param	FLOPs
	AP^{box}	AP^{mask}	AP^{box}	AP^{mask}		
RepPointsV2* [31]	-	-	52.1	-	-	-
GCNet* [38]	51.8	44.7	52.3	45.4	-	1041G
RelationNet++* [39]	-	-	52.7	-	-	-
SpineNet-190 [40]	52.6	-	52.8	-	164M	1885G
ResNeSt-200* [41]	52.5	-	53.3	47.1	-	-
EfficientDet-D7 [42]	54.4	-	55.1	-	77M	410G
DetectoRS* [43]	-	-	55.7	48.5	-	-
YOLOv4 P7* [44]	-	-	55.8	-	-	-
Copy-paste [45]	55.9	47.2	56.0	47.4	185M	1440G
X101-64 (HTC++)	52.3	46.0	-	-	155M	1033G
Swin-B (HTC++) [8]	56.4	49.1	-	-	160M	1043G
Swin-L (HTC++) [8]	57.1	49.5	57.7	50.2	284M	1470G
Swin-L (HTC++)* [8]	58.0	50.4	58.7	51.1	284M	-
EAPT-M (HTC+++)	57.8	50.3	58.0	50.6	182M	1183G
EAPT-L (HTC+++)	58.2	50.6	59.0	51.6	319M	1630G
EAPT-L (HTC+++)*	59.1	51.8	59.7	52.2	309M	-

TABLE VI: Semantic segmentation results on ADE20K. The baselines we compared here include convolutional neural network based architecture and transformer-based architecture. The used backbone include ResNet [33], HRNet [46], ResNeSt [41], Swin Transformer [8].

ADE20K		val	test	#param	FLOPs	FPS
Method	Backbone	mIoU	score			
DANet [47]	ResNet-101	45.2	-	69M	1119G	15.2
DLab.v3+ [48]	ResNet-101	44.1	-	63M	1021G	16.0
ACNet [49]	ResNet-101	45.9	38.5	-	-	-
DNL [50]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [51]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [52]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [51]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [48]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [48]	ResNeSt-200	48.4	-	88M	1381G	8.1
UperNet [52]	DeiT-S	44.0	-	52M	1099G	16.2
UperNet [52]	Swin-T	46.1	-	60M	945G	18.5
UperNet [52]	Swin-S	49.3	-	81M	1038G	15.2
UperNet [52]	EAPT-S	47.7	-	76M	1123G	16.8
UperNet [52]	EAPT-M	51.5	-	119M	1362G	13.6

depth, etc.

B. Comparison to state-of-the-art

Evaluate on classification. For the task of classification, we compare our model with both convolutional neural network based architectures and transformer-based architectures, and we report comparison results in Table III. Compared with recent similar works, i.e., ViT [6], DeiT [7], and Swin Transformer [8], our results are significantly more optimized. With the same level of structural complexity, our model has higher accuracy (e.g., +1.6% for EAPT-S (82.9%) over Swin-T (81.3%)). At the same level of accuracy, the structure of our model is simpler (e.g., $\times 3$ throughput for EPAT-M (242.6) over DeiT-B (85.9)). Compared with convolutional neural networks based models, our proposed and recent transformer-based works have no other obvious advantages except for a slightly better speed-accuracy trade-off. We argue that the main reasons are as follows: 1) The model is mainly used for Natural Language Processing (NLP). Although we and recent vision transformers have made adaptive improvements to make it more suitable for vision tasks, considering the differences between vision elements and language elements, We believe that the existing improvements are not enough to establish the attention relationship between vision elements. 2) The compared CNN models in Table III have been optimized (e.g., RegNet [26] and EfficientNet [27] are based on Neural Architecture Search (NAS)), and our proposed and compared recent vision transformers follow the vanilla transformer design. Therefore, we have reason to believe that the use of advanced technology to optimize the vision transformer can further improve its performance.

Evaluate on object detection. Under typical object detection frameworks, we compare our proposed network architectures with convolutional-based networks and recent advanced transformer models. We report the framework-level comparison results in Table IV. From Table IV, we have several observations: 1) EAPT demonstrates the clear advantages over the convolutional-based networks. Under different frameworks, EAPT is increased by +5.5% AP^{box} at most (EAPT-S over R-50 under the method of Cascade Mask R-CNN), and at least by +1.0% (EAPT-S over Swin-T under the method of Sparse Mask R-CNN). 2) EAPT we proposed outperforms existing advanced transformer models Swin Transformer with accept-

TABLE VII: Experiment results under different component combinations on the task of classification, object detection, and semantic segmentation. ✓ and - stand for using and discarding this component, respectively. Comp.1, Comp.2, and Comp.3 stand for the Deformable Attention, the En-DeC module, and MCMD. The experiment follows the settings in Table III, Table IV, and Table VI.

Method / task	backbone	Comp.1	Comp.2	Comp.3	top-1 acc. / AP^{box} / mIoU
EAPT-S classification		✓	✓	✓	82.9
		✓	✓	-	81.9
		✓	-	✓	81.7
		-	✓	✓	81.8
		✓	-	-	81.2
		-	✓	-	81.6
		-	-	✓	81.0
		-	-	-	80.6
Sparse R-CNN detection	EAPT-S	✓	✓	✓	48.9
		✓	✓	-	48.5
		✓	-	✓	47.6
		-	✓	✓	48.1
		✓	-	-	47.4
		-	✓	-	47.9
		-	-	✓	47.3
		-	-	-	46.8
UperNet segmentation	EAPT-S	✓	✓	✓	47.7
		✓	✓	-	47.3
		✓	-	✓	46.6
		-	✓	✓	47.9
		✓	-	-	45.9
		-	✓	-	47.5
		-	-	✓	46.0
		-	-	-	45.7

able increased costs of parameters and FLOPs. 3) Compared with loose evaluation metric (AP^{box}_{50}), we have a more obvious advantage in strict one (AP^{box}_{75}). We argue that this benefits from non-fixed size attention information that our network can obtain and thus can better establish the attention relationship between vision elements of different sizes. We will further discuss these benefits in more detail in the ablation study.

Additionally, we also report the results of comparison with previous state-of-the-art methods in Table V. Our detection framework refers to the ingenious design of the Swin Transformer, i.e., HTC++ [8] improved from HTC [53], and pretrained on ImageNet-22K. We follow the most of improvement tricks in HTC++, and named it HTC+++. Although the complexity of our model is slightly higher than these methods, the performance of our model is promising and competitive (e.g., +1.3% AP^{box} and 1.4% AP^{mask} for EAPT-L (HTC+++) (59.0% AP^{box} and 51.6% AP^{mask}) over Swin-L (HTC++) (57.7% AP^{box} and 50.2% AP^{mask})).

Evaluate on semantic segmentation. We also adopt UperNet as the base framework for a fair comparison, and we report the comparison results in Table VI. From Table VI, our network architectures outperform both convolutional neural network based architectures and transformer-based architectures (e.g., +1.6% for EAPT-S (47.7%) over Swin-T (46.1%), +3.1% for EAPT-M (51.5%) over ResNeSt-200 (48.4%)).

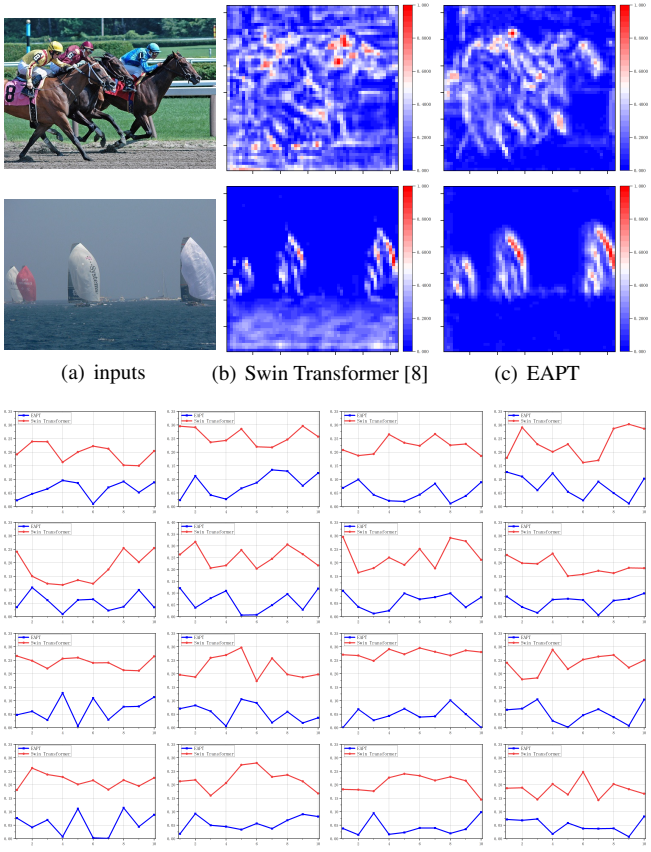
(a) inputs (b) Swin Transformer [8] (c) EAPT

Fig. 5: The feature visualization of various vision elements. The object in the first-row image is a bird, and the showed vision elements include head (orange boxes), beak (yellow boxes), and paw (red boxes). The object in the second-row image are two sheep, and the showed vision elements include head (yellow boxes) and feet (red boxes). The object in the third-row image is a person, and the showed vision elements include head (yellow boxes) and hand (red boxes).

C. Ablation Study

As described in section III, our work is mainly composed of three components, including Deformable Attention, En-DeC module, and MCMD. To further prove the effectiveness of those components, we do ablation study here. Three components can produce eight combinations, we reported the detect results of those combinations in Table VII, and we can conclude from the results that each component can help improve the performance of the EPAT. In the following, we will further prove the effectiveness of those components and explore their functionality.

Effectiveness of Deformable Attention. Different from the vanilla transformer, we introduce the Deformable Attention in this paper, which learns an offset for each position in patches and thus can better cover the various vision elements. And from Table VII, Deformable Attention can indeed improve the performance of our proposed. In order to make better use of the Deformable Attention, we will continue to explore its parameters here, in particular, the broaden size of the attention field in Deformable Attention. We have set the broaden size in the third section, that is $|\mathrm{o}^{\mathrm{x}}_{\mathrm{m}}| \leq \mathrm{h_w}$, $|\mathrm{o}^{\mathrm{y}}_{\mathrm{m}}| \leq \mathrm{w_w}$, where $|\mathrm{o}^{\mathrm{x}}_{\mathrm{m}}|$ and $|\mathrm{o}^{\mathrm{y}}_{\mathrm{m}}|$ represent the broaden size, and $\mathrm{h_w} \times \mathrm{w_w}$ is the shape of the non-overlapping local windows. We argue that the broaden size will affect the performance of Deformable Attention. To prove this, we report the performance of EAPT in different broaden sizes in Fig. 4. Meanwhile, we also show training costs in Fig. 4. From Fig. 4, we can find that the performance of EAPT is improving as the broaden size increases. Additionally, when the broaden size increases to a certain value, the improvement of model performance begins to slow down. The parameter used in this paper is $|\mathrm{o}^{\mathrm{x}}_{\mathrm{m}}| = |\mathrm{o}^{\mathrm{y}}_{\mathrm{m}}| = 4$, which is obviously not the optimal performance. This is mainly



(a) inputs (b) Swin Transformer [8] (c) EAPT

Fig. 6: The heat map of the features (lines 1 and 2) and the difference and coherence of edge information among different patches (lines 3-6). The edge information comparisons in lines 3-6 correspond to the patch position in the input image. For example, the first column of the third row corresponds to the patch in the upper left corner of the image.

because we have considered the tradeoff of model performance and training cost, i.e., the training cost of the model learning increases when the broaden size is large. And this also shows that if the training cost is increased, the performance of EAPT will further increase.

In addition, we also demonstrate the advantages of our proposed framework in terms of covering the various vision elements by feature visualization in Fig. 5. From Fig. 5, we can see that our proposed method obviously can better cover the vision elements in objects to be classified/detect/segment. We argue that this mainly benefits from the Deformable Attention, which can obtain non-fixed attention information through the learnable offsets.

Effectiveness of the En-DeC module. Although the vanilla splitting-based transformer can avoid expensive global-pixel-level attention calculations, it also limits the communication of attention information among different patches. To this problem, we propose Encode-Decode Communication module (En-DeC module), which uses an encode to compress features and uses a decode to restore, and thus its results contain communication information among all patches. To further observe the feature extraction ability of the different network architectures, we show the heat map of the features in Fig. 6 (lines 1 and 2). Meanwhile, we also show the difference and coherence of edge information among different patches. We argue that the smaller the difference and the stronger the coherence, the better the ability of the network to extract attention information across patches. To be specific, we directly calculate the difference between the edge feature values of adjacent patches (each edge gets ten values based on Bilinear Interpolation) and finally report the average of the four edges (two edges for peripheral patches) of each patch (see lines 3-6 in Fig. 6). From heat maps in Fig. 6, we can clearly see that our method can better cover the features of the object to be classified/detect/segment, while the features of the baseline method obviously include a lot of useless information. In addition, from lines 3-6 in Fig. 6, we can also see that our method can obtain more continuous attention information. We argue that this is mainly beneficial from the En-DeC module we proposed, which guarantees the information communication among all patches through an encode-decode architecture.

TABLE VIII: Performance under different position encoding methods on the task of classification, object detection, and segmentation. The used backbone is EAPT-S, and the experiment follows the settings in Table III, Table IV, and Table VI.

Method	ImageNet top-1 acc.	COCO AP^{box}	ADE20k val mIoU
Sinusoidal [4]	81.6	47.8	46.4
Embedding [15]	81.6	48.0	46.5
Relative [16]	81.8	48.2	46.7
FLOATER [17]	81.9	48.2	46.8
MCMD	82.9	48.9	47.7

Effectiveness of Multi-dimensional Continuous Mixture Descriptor (MCMD). Transformer does not have the ability to distinguish the position information of the input sequence like Recurrent Neural Networks (RNNs) and thus requires to perform position encoding for inputs. However, current position encoding approaches in vision transformer either directly adopt low-dimensional encoding technology in NLP or use learn-based encoding method that may increase the difficulty of model training. To this problem, we propose MCMD in this paper, which can encode the patches with any dimension and any length. We can see from Table VII that MCMD indeed contributes to our proposed method. Here we will further compare it with other position encoding methods to show its advantages, and we report the comparison results in Table VIII. From Table VIII we can see that our method outperforms the compared position encoding methods, including Sinusoidal [4], Embedding [15], Relative [16], and FLOATER [17]. Additionally, Sinusoidal performs the worst. We argue that this is mainly because it has no data-driven mechanism, that is, it cannot adaptively adjust the encoding according to the patches weights and thus cannot guarantee the same position guidance for all patches.

V. Conclusion

This paper presented a new vision transformer architecture called Efficient Attention Pyramid Transformer (EAPT). Our rigorous evaluation demonstrates that EAPT outperforms the

state-of-the-art methods on the task of classification, object detection, and semantic segmentation. We argue that the superior performance of EAPT is attributed to our novel designs, including Deformable Attention, En-DeC module, and Multi-dimensional Continuous Mixture Descriptor (MCMD). Deformable Attention can broaden the attention field without increasing too much computational cost, and En-DeC module uses an encode-decode architecture to obtain communication information among all patches. MCMD can encode patches with any dimensions and lengths and thus more fit the vision features than other methods. We argue that combining some existing technologies can further reduce the complexity of the transformer, i.e., Neural Architecture Search, Knowledge Distillation, and Network Pruning, and this is what we plan to do in immediate future.

REFERENCES

[1] H. Chen, D. Jiang, and H. Sahli, "Transformer encoder with multi-modal multi-head attention for continuous affect recognition," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[2] H. Zhong, J. Chen, C. Shen, H. Zhang, J. Huang, and X.-S. Hua, "Self-adaptive neural module transformer for visual question answering," *IEEE Transactions on Multimedia*, vol. 23, pp. 1264–1273, 2021.

[3] Y. Qian, M. Yang, X. Zhao, C. Wang, and B. Wang, "Oriented spatial transformer network for pedestrian detection using fish-eye camera," *IEEE Transactions on Multimedia*, vol. 22, no. 2, pp. 421–431, 2020.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, p. 60006010.

[5] X. Chen, X. Song, S. Cui, T. Gan, Z. Cheng, and L. Nie, "User identity linkage across social media via attentive time-aware user modeling," *IEEE Transactions on Multimedia*, pp. 1–12, 2020.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

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

[8] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *arXiv preprint arXiv:2103.14030*, 2021.

[9] X. Zhu, W. Su, L. Lu, B. Li, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[10] J. Liang, N. Homayounfar, W.-C. Ma, Y. Xiong, R. Hu, and R. Urtasun, "Polytransform: Deep polygon transformer for instance segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9128–9137.

[11] H. Chen, Y. Wang, T. Guo, C. Xu, and W. Gao, "Pre-trained image processing transformer," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[12] H. Luo, W. Jiang, X. Fan, and C. Zhang, "Stnreid: Deep convolutional networks with pairwise spatial transformer networks for partial person re-identification," *IEEE Transactions on Multimedia*, vol. 22, no. 11, pp. 2905–2913, 2020.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision (ECCV)*, 2020, p. 213229.

[14] Z. Chen, Y. Zhu, C. Zhao, G. Hu, W. Zeng, J. Wang, and M. Tang, "DPT: Deformable patch-based transformer for visual recognition," *arXiv preprint arXiv:2107.14467*, 2021.

[15] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2019.

[16] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

[17] L. Xuanqing, Y. Hsiang-Fu, D. Inderjit, and H. Cho-Jui, "Learning to encode position for transformer with continuous dynamical model," in *ICML*, 2020, pp. 6327–6335.

[18] Z. Pan, F. Yuan, J. Lei, W. Li, N. Ling, and S. Kwong, "Miegan: Mobile image enhancement via a multi-module cascade neural network," *IEEE Transactions on Multimedia*, pp. 1–1, 2021.

[19] F. Lyu, Q. Wu, F. Hu, Q. Wu, and M. Tan, "Attend and imagine: Multi-label image classification with visual attention and recurrent neural networks," *IEEE Transactions on Multimedia*, vol. 21, no. 8, pp. 1971–1981, 2019.

[20] Q. Wang, C. Yuan, J. Wang, and W. Zeng, "Learning attentional recurrent neural network for visual tracking," *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 930–942, 2019.

[21] C. Shi and C.-M. Pun, "Multiscale superpixel-based hyperspectral image classification using recurrent neural networks with stacked autoencoders," *IEEE Transactions on Multimedia*, vol. 22, no. 2, pp. 487–501, 2020.

[22] L. Nie, X. Wei, D. Zhang, X. Wang, Z. Gao, and Y. Yang, "Data-driven answer selection in community QA systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 6, pp. 1186–1198, 2017.

[23] J. Wagner, J. M. Khler, T. Gindele, L. Hetzel, J. T. Wiedemer, and S. Behnke, "Interpretable and fine-grained visual explanations for convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9089–9099.

[24] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3319–3327.

[25] X. Wang, X. He, F. Feng, L. Nie, and T.-S. Chua, "TEM: Tree-enhanced embedding model for explainable recommendation," in *World Wide Web Conference*, 2018, pp. 1543–1552.

[26] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollr, "Designing network design spaces," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 425–10 433.

[27] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, 2019, p. 61056114.

[28] K. He, G. Gkioxari, P. Dollr, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020.

[29] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6154–6162.

[30] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9756–9765.

[31] Y. Chen, Z. Zhang, Y. Cao, L. Wang, S. Lin, and H. Hu, "Reppoints v2: Verification meets regression for object detection," in *NeurIPS*, 2020.

[32] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, and C. Wang, "Sparse R-CNN: End-to-end object detection with learnable proposals," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[34] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "ImageNet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[35] T. Y. Lin, M. Maire, S. Belongie, J. Hays, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[36] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ADE20K dataset," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 302–321, 2016.

[37] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[38] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "GCNet: Non-local networks meet squeeze-excitation networks and beyond," in *IEEE Conference on Computer Vision (ICCV) Workshops*, 2019.

[39] C. Chi, F. Wei, and H. Hu, "RelationNet++: Bridging visual representations for object detection via transformer decoder," in *NeurIPS*, 2020.

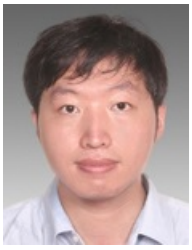
[40] X. Du, T.-Y. Lin, P. Jin, G. Ghiasi, M. Tan, Y. Cui, Q. V. Le, and X. Song, "SpineNet: Learning scale-permuted backbone for recognition and localization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 589–11 598.

[41] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, and R. Manmatha, "ResNeSt: Split-attention networks," *arXiv preprint arXiv:2004.08955*, 2020.
[42] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 778–10 787.
[43] S. Qiao, L. C. Chen, and A. Yuille, "DetectoRS: Detecting objects with recursive feature pyramid and switchable atrous convolution," *arXiv preprint arXiv:2006.02334*, 2020.
[44] A. Bochkovskiy, C. Y. Wang, and H. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.
[45] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, and B. Zoph, "Simple Copy-Paste is a strong data augmentation method for instance segmentation," *arXiv preprint arXiv:2012.07177*, 2020.
[46] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5686–5696.
[47] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3141–3149.
[48] L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *European Conference on Computer Vision (ECCV)*, 2018, p. 801818.
[49] J. Fu, J. Liu, Y. Wang, Y. Li, Y. Bao, J. Tang, and H. Lu, "Adaptive context network for scene parsing," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 6747–6756.
[50] M. Yin, Z. Yao, Y. Cao, X. Li, Z. Zhang, S. Lin, and H. Hu, "Disentangled non-local neural networks," in *European Conference on Computer Vision (ECCV)*, 2020.
[51] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *European Conference on Computer Vision (ECCV)*, 2020.
[52] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and S. Jian, "Unified perceptual parsing for scene understanding," in *European Conference on Computer Vision (ECCV)*, 2018, p. 418434.
[53] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Hybrid task cascade for instance segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4969–4978.

Xiao Lin received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China. She is currently a Full Professor with the Department of Computer Science, Shanghai Normal University (SHNU), Shanghai, China. She is also a Visiting Scholar with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. She has authored and co-authored a set of research papers in internal journals and conferences such as IEEE TMM, Elsevier CAD, IEEE ICME, etc. Her current research interests include image processing, computer vision, and machine learning.

Shuzhou Sun received the B.Eng. degree in computer science and technology from Henan Agricultural University, Zhengzhou, China, in 2018. He is currently pursuing the M.Eng. degree in computer science with the Department of Computer Science, Shanghai Normal University, Shanghai, China. His current research interests include computer vision, deep learning, and image processing.

Wei Huang received the B.Eng. degree in computer science from the Henan University of Science and Technology, Luoyang, China, in 2007. He is currently pursuing the Ph.D. degree in computer science with the Department of Computer Science and Engineering, University of Shanghai for Science and Technology, Shanghai, China. His current research interests include image processing, deep learning, and computer vision.



Bin Sheng (Member, IEEE) received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Shatin, Hong Kong, in 2011. He is currently a Full Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. He is an Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology. His current research interests include virtual reality and computer graphics.

Ping Li (Member, IEEE) received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Shatin, Hong Kong, in 2013. He is currently a Research Assistant Professor with The Hong Kong Polytechnic University, Kowloon, Hong Kong. He has one image/video processing national invention patent and has excellent research project reported worldwide by *ACM TechNews*. His current research interests include image/video stylization, artistic rendering and synthesis, and creative media.

David Dagan Feng (Life Fellow, IEEE) received the M.Eng. degree in electrical engineering and computer science (EECS) from Shanghai Jiao Tong University, Shanghai, China, in 1982, and the M.Sc. degree in biocybernetics and the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), Los Angeles, CA, in 1985 and 1988, respectively, where he received the Crump Prize for Excellence in Medical Engineering. He is currently the head in the School of Information Technologies, the director in the Biomedical & Multimedia Information Technology Research Group, and the research director in the Institute of Biomedical Engineering and Technology at the University of Sydney, Sydney, Australia. He has published over 700 scholarly research papers, pioneered several new research directions, and made a number of landmark contributions in his field. More importantly, however, is that many of his research results have been translated into solutions to real-life problems and have made tremendous improvements to the quality of life for those concerned. He has served as the chair in the International Federation of Automatic Control (IFAC) Technical Committee on Biological and Medical Systems, has organized/chaired over 100 major international conferences/symposia/workshops, and has been invited to give over 100 keynote presentations in 23 countries and regions. He is a fellow of the IEEE and Australian Academy of Technological Sciences and Engineering.