This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

PaCa-ViT: Learning Patch-to-Cluster Attention in Vision Transformers

Ryan Grainger[1], Thomas Paniagua[1], Xi Song[2], Naresh Cuntoor[3], Mun Wai Lee[3] and Tianfu Wu[1]*
[1]Department of ECE, NC State, [2]An Independent Researcher, [3]BlueHalo

Abstract

Vision Transformers (ViTs) are built on the assumption of treating image patches as "visual tokens" and learn patch-to-patch attention. The patch embedding based tokenizer has a semantic gap with respect to its counterpart, the textual tokenizer. The patch-to-patch attention suffers from the quadratic complexity issue, and also makes it non-trivial to explain learned ViTs. To address these issues in ViT, this paper proposes to learn Patch-to-Cluster attention (PaCa) in ViT. Queries in our PaCa-ViT starts with patches, while keys and values are directly based on clustering (with a predefined small number of clusters). The clusters are learned end-to-end, leading to better tokenizers and inducing joint clustering-for-attention and attention-for-clustering for better and interpretable models. The quadratic complexity is relaxed to linear complexity. The proposed PaCa module is used in designing efficient and interpretable ViT backbones and semantic segmentation head networks. In experiments, the proposed methods are tested on ImageNet-1k image classification, MS-COCO object detection and instance segmentation and MIT-ADE20k semantic segmentation. Compared with the prior art, it obtains better performance in all the three benchmarks than the SWin [32] and the PVTs [47, 48] by significant margins in ImageNet-1k and MIT-ADE20k. It is also significantly more efficient than PVT models in MS-COCO and MIT-ADE20k due to the linear complexity. The learned clusters are semantically meaningful. Code and model checkpoints are available at https://github.com/iVMCL/PaCaViT.

1. Introduction

A picture is worth a thousand words. Seeking solutions that can bridge the semantic gap between those words and raw image data has long been, and remains, a grand challenge in computer vision, machine learning and AI. Deep learning has revolutionized the field of computer vision in the past decade. More recently, Vision Transformers (ViTs) [13, 45] have witnessed remarkable progress in computer vision. ViTs are built on the basis of treating image patches as "visual tokens" using patch embedding and learning patch-to-patch attention throughout. Unlike the textual tokens that are provided as inputs in natural language processing, visual tokens need to be learned first and continuously refined for more effective learning of ViTs. The patch embedding based tokenizer is a workaround in practice and has a semantic gap with respect to its counterpart, the textual tokenizer. On one hand, the well-known issue of the quadratic complexity of vanilla Transformer models and the 2D spatial nature of images create a non-trivial task of developing ViTs that are applicable for many vision problems including image classification, object detection and semantic segmentation. On the other hand, explaining trained ViTs requires non-trivial and sophisticated methods [4] following the trend of eXplainable AI (XAI) [18] that has been extensively studied with convolutional neural networks.

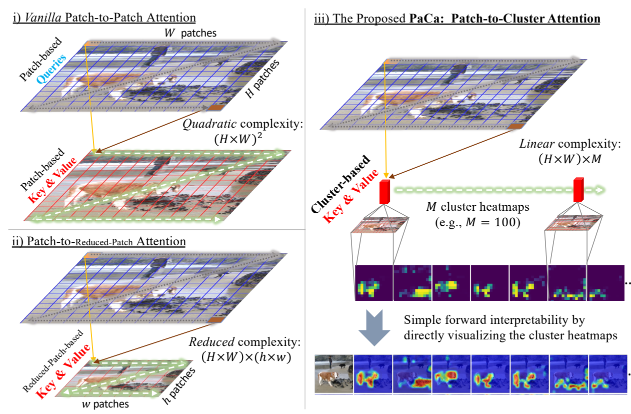

Figure 1. i) The vanilla patch-to-patch self-attention [13, 45] directly leverages image patch embeddings as visual tokens and suffers from its quadratic complexity. Every Query (e.g., the patches in the blue grid) needs to interact with every Key. ii) To address the quadratic complexity, one popular method is to leverage spatial reduction (e.g., implemented via a convolution with a stride $r > 1$) in computing the Key and the Value [47, 48]. It still performs patch-to-patch attention, but enjoys a reduced complexity. iii) We propose **Patch-to-Cluster attention (PaCa)** in this paper. A predefined number of M cluster assignments is first learned and then used in computing the Key and Value, resulting in not only linear complexity, but also more meaningful visual tokens.

To address the quadratic complexity, there have been

*T. Wu is the corresponding author.

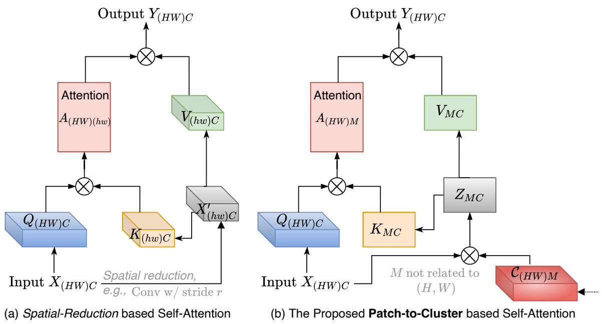

Figure 2. Illustration of (a) the spatial reduction based self-attention and (b) the proposed PaCa module in vision applications, where (HW) represents the number of patches in the input with H and W the height and width respectively, and M a predefined small number of clusters (e.g., $M = 100$). See text for details.

two main variants developed with great success: One is to exploit the vanilla Transformer model locally using a predefined window size (e.g., 7×7) such as the SWin-Transformer [32] and the nested variant of ViT [62]. The other is to exploit another patch embedding at a coarser level (i.e., nested patch embedding) to reduce the sequence length (i.e., spatial reduction) before computing the keys and values (while keeping the query length unchanged) [47, 48, 52], as illustrated in Fig. 1 (left-bottom) and Fig. 2 (a). Most of these variants follow the patch-to-patch attention setup used in the vanilla isotropic ViT models [13]. Although existing ViT variants have shown great results, patch embedding based approaches may not be the best way of learning visual tokens due to the underlying predefined subsampling of the image lattice. Additionally, patch-to-patch attention does not account for the spatial redundancy found in images due to their compositional nature and reusable parts [15]. Thus, it is worth exploring alternative methods towards learning more semantically meaningful visual tokens. A question arises naturally: *Can we rethink the patch-to-patch attention mechanism in vision tasks to hit three "birds" (reducing complexity, facilitating better visual tokenizer and enabling simple forward explainability) with one stone?*

As shown in Fig. 1 (right) and Fig. 2 (b), this paper proposes to learn **Patch-to-Cluster attention (PaCa)**, which provides a straightforward way to address the aforementioned question: Given an input sequence $X_{N,C}$ (e.g., $N = H \cdot W$), a light-weight clustering module finds meaningful clusters by first computing the cluster assignment, $\mathcal{C}_{N,M}$ (Eqn. 4 and Eqn. 5) with a predefined small number of clusters M (e.g., $M = 100$). Then, M latent "visual tokens", $Z_{M,C}$ are formed via simple matrix multiplication between $\mathcal{C}^T_{N,M}$ (transposed) and $X_{N,C}$. In inference, we can directly visualize the clusters $\mathcal{C}_{N,M}$ as heatmaps to reveal what has been captured by the trained models (Fig. 1, right-bottom). The proposed PaCa module induces jointly learning clustering-for-attention and attention-for-clustering in ViT models. We study four aspects of the PaCa module:

- *Where to compute the cluster assignments?* Consider the stage-wise pyramidical architecture (Fig. 3) of assembling ViT blocks [47, 48], a stage consists of a number of blocks. We test two settings: *block-wise* by computing the cluster assignment for each block, or *stage-wise* by computing it only in the first block in a stage and then sharing it with the remaining blocks. Both give comparable performance. The latter is more efficient when the model becomes deeper.
- *How to compute the cluster assignment?* We also test two settings: using 2D convolution or Multi-Layer Perceptron (MLP) based implementation. Both have similar performance. The latter is more generic and sheds light on exploiting PaCa for more general Token-to-Cluster attention (ToCa) in a domain agnostic way.
- *How to leverage an external clustering teacher?* We investigate a method of exploiting a lightweight convolution neural network (Fig. 4) in learning the cluster assignments that are shared by all blocks in a stage. It gives some interesting observations, and potentially pave a way for distilling large foundation models [3].
- *What if the number of clusters is known?* We further extend the PaCa module in designing an effective head sub-network for dense prediction tasks such as image semantic segmentation (Fig. 5) where the number of clusters M is available based on the ground-truth number of classes and the learned cluster assignment $\mathcal{C}_{N,M}$ has direct supervision. The PaCa segmentation head significantly improves the performance with reduced model complexity.

In experiments, the proposed PaCa-ViT model is tested on the ImageNet-1k [12] image classification, the MS-COCO object detection and instance segmentation [31] and the MIT-ADE20k semantic segmentation [64]. It obtains consistently better performance across the three tasks than some strong baseline models including the Swin-Transformers [32] and the PVTv2 models [47].

2. Related Work and Our Contributions

Since the pioneering work of ViT [13], there has been a vast body of work leveraging and developing variants of the Transformer model [45] that dominates the NLP field in computer vision (see a recent survey [19]). We briefly review some of the related efforts on addressing the quadratic complexity of ViT models. There has been rapid progress in developing efficient Transformer models. [41] provides an excellent survey of different efforts in the literature.

One family of approaches is to leverage inductive bias back in the Transformer, including the local window partition based methods [1, 2, 6, 32, 36], random sparse patterns [59] and the locality-sensitive hashing (LSH) based Reformer [28]. Although both computational and model performance can be improved, these models achieve them at the expense of limiting the capacity of a self-attention layer due to the locality constraints. Also, sophisticated designs might be needed such as the shifted window and the masked attention method in the SWin-Transformer [32]. The quad-tree based aggregating method proposed in the NesT [62] shows another promising direction. In a similar spirit, the Evo-ViT [55] presents a method of selecting top-k informative tokens for applying the self-attention to reduce the cost. The A-ViT [57] presents a method of halting tokens via reformulating the adaptive computation time (ACT) method. Both Evo-ViT and A-ViT can achieve linear complexity, but they mainly focus on image classification tasks, and it is not clear how to extend them for downstream tasks such as object detection and semantic segmentation. Like the PVT models [47, 48], our goal is to develop a versatile variant of ViT that can tackle not only image classification, but also many downstream vision tasks which use high-resolution images as inputs and need to retain sufficient high-resolution information throughout.

Another family of approaches exploits low-rank projections to form a coarser-grained representation of the input sequence, which have shown successful applications for certain NLP tasks such as the LinFormer [46], Nyströmformer [54] and Synthesizer [40]. Even though these methods retain the capability of enabling each token to attend to the entire input sequence, they suffer from the loss of high-fidelity token-wise information, and on tasks that require fine-grained local information, their performance can fall short of full attention or the aforementioned sparse attention. Similarly, the Performer [7] presents a method for Softmax attention kernel approximation via positive Orthogonal Random features for the Query and the Key. The proposed PaCa-ViT is motivated by addressing the redundancy of information within patches in patch-to-patch attention in computer vision applications, and shares the spirit of low-rank projection based efficient Transformer models. Our PaCa is most similar to Linformers [46], but with two main differences: Our PaCa applies clustering (i.e. $\mathcal{C}_{N,M}$) before computing the Key and the Value (Fig. 2 (b)), unlike Linformers which apply direct projection after the Key and the Value are computed (i.e., $E \cdot K$ and $F \cdot V$, see Eqn.7 in the Linformer paper). Our PaCa reduces the sequence length via a learnable and data-adaptive cluster assignment $\mathcal{C}_{N,M}$, rather than treating the projection(s), E and F, as sequence length specific model parameters.

In addition to achieve the efficiency, driven by XAI [18] and the natural curiosity of humanity, it is always desirable to understand what is going on inside different ViTs. Most XAI efforts have been focused on convolutional neural networks. More recently, some attention has been attracted to explaining the vanilla isotropic ViT models based on the attention scores themselves. As pointed out in the Improved LRP [4], reducing the explanation to only the attentions scores may be myopic since many other components are ignored. The proposed PaCa provides a direct forward explainer by visualizing the learned cluster assignments as heatmaps, which is complementary to existing approaches.

Our Contributions. This paper makes two main contributions for developing efficient and interpretable variants of Transformers in computer vision applications: (i) It proposes a Patch-to-Cluster Attention (PaCa) module that facilitates learning more expressive and meaningful "visual tokens" beyond patches in ViTs. It addresses the quadratic complexity of vanilla patch-to-patch attention, while accounting for the spatial redundancy of patches in the patch-to-patch attention. It also enables a forward explainer for interpreting the trained models based on the learned semantically meaningful clusters. (ii) It proposes a PaCa semantic segmentation head network that is lightweight and more expressive than the widely used UperNet [53] and the semantic FPN [27]. With the two main contributions, the proposed PaCa ViTs show superior performance consistently in image classification, object detection and instance segmentation and image semantic segmentation than the prior art including SWin-Transformers [32] and PVTs [47, 48].

3. Approach

In this section, we present details of the proposed PaCa module and the resulting PaCa-ViT models.

3.1. From Patch-to-Patch Attention to Patch-to-Cluster Attention

Denote by $X_{N,C}$ an input sequence consisting of N "tokens" which are embedded in a C-dimensional space. In computer vision, the N tokens are formed via patch embedding. We have $N = H \times W$ where H and W are the height and width of the patch grid respectively [1]. Positional encoding can also be added to counter the permutation insensitivity of the self-attention computation [13, 45].

The core of the Transformer model is to compute the scaled dot-product attention in transforming the input $X_{N,C}$ to the output $Y_{N,C}$,

$$A_{N,M} = \text{Softmax}(\frac{Q_{N,C} \cdot K^T_{M,C}}{\sqrt{C}})_{dim=1},$$
$$Y_{N,C} = A_{N,M} \cdot V_{M,C}, \tag{1}$$

where $Q_{N,C}, K_{M,C}$ and $V_{M,C}$ are the Query/Key/Value computed from the input $X_{N,C}$, e.g., via linear transformations in the patch-to-patch attention where $M = N$, which leads to the quadratic complexity of computing $A_{N,N}$. The Softmax is applied for each row as indicated by the subscript $dim = 1$. In practice, the multi-head self-attention (MHSA) is used to capture the attention in different subspaces and fused by a linear projection [45]. To address the

[1] We will use the three notations $X_{N,C}$, $X_{H,W,C}$ and $X_{(HW)C}$ interchangeably in this paper.

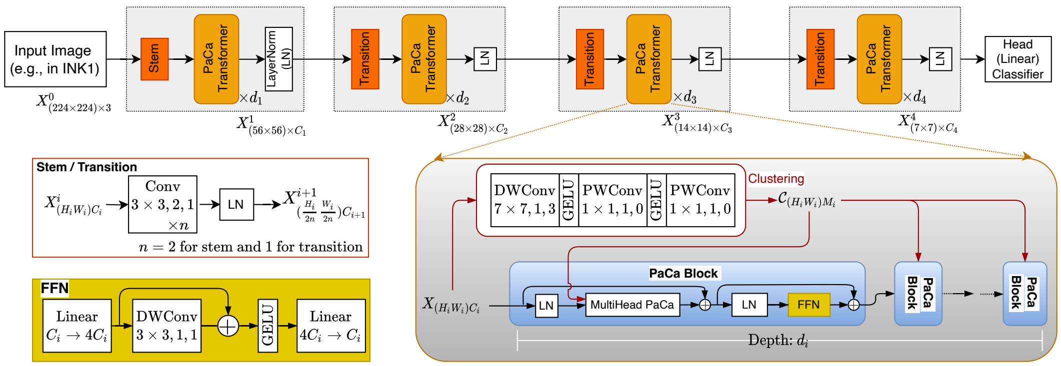

Figure 3. Illustration of the proposed PaCa-ViT using the stage-wise convolution-based clustering setting. It consists of four stages each of which has a number, d_i of the proposed PaCa Transformer block. The FFN refers to a feed-forward network. See text for details.

quadratic complexity, **the key** is to ensure $M \ll N$, preferably a predefined constant (e.g., $M = 100$) to induce the linear complexity.

To that end, one popular method is to exploit spatial reduction via strided convolution (nested patch embedding) or adaptive average pooling as done in the PVT models [47,48] (Fig. 2 (a)). Note that i) the typically used strided convolution method for spatial reduction does not truly prevent quadratic complexity, but rather reduces it by a ratio corresponding to the patch size, and ii) the adaptive average pooling may suffer from treating each element in a pooling window with equal importance, thus lacking the necessary adaptability and data-driven reweighing capability. Meanwhile, the vanilla MLP in the Transformer block has been substituted by the inverted bottleneck block proposed in the MobileNets-v2 [37], termed *MBlock* (the left-bottom of Fig. 3), which adds a depth-wise convolution in the hidden layer. And, the non-overlapping patch embedding has been replaced by overlapping ones. With these modifications, positional encoding is not used, partially due to the implicit positional encoding capability of the zero-padding convolutions [25] used in the Stem, the Transition modules, and the MBlock.

On top of the best practices used in PVTv2 [47], we propose the Patch-to-Cluster attention (PaCa), as illustrated in Fig. 2 (b), which not only achieves the linear complexity (with the overhead of the lightweight clustering module), but also provides a simple cluster assignment visualization method for explaining the attention module.

3.2. The Proposed Patch-to-Cluster Attention

As shown in Fig. 2 (b), given an input sequence $X_{N,C}$ and a predefined number M of clusters (e.g., $M = 100$), we first compute the cluster assignment $\mathcal{C}_{N,M}$, whose goal is to cluster the input sequence into M latent "visual tokens",

$$Z_{M,C} = \text{LayerNorm}(\mathcal{C}_{N,M}^T \cdot X_{N,C}), \tag{2}$$

which then is used to compute the Key, $K_{M,C}$ and the Value, $V_{M,C}$ via linear transformations in computing the attention (Eqn. 1). We present two methods of computing the cluster assignment $\mathcal{C}_{N,M}$: One is the **onsite clustering** as illustrated in Fig. 3, and the other is the **external clustering** via a lightweight teacher network as illustrated in Fig. 4.

3.2.1 Onsite Clustering

For the *onsite clustering* method (Fig. 3), we have,

$$\mathcal{C}_{N,M} = \text{Clustering}(X_{N,C}; \theta), \tag{3}$$

where θ collects the parameters of the clustering module. We investigate two simple designs in this paper:

i) Clustering via Convolution: It uses depth-wise convolution (DWConv) and point-wise convolution (PWConv),

$$X_{N,C} \xrightarrow[k=7,s=1]{\text{DWConv+GELU}} \cdot \xrightarrow[k=1,s=1]{\text{PWConv+GELU}} U_{N,C}, \tag{4}$$

$$U_{N,C} \xrightarrow[k=1,s=1]{\text{PWConv}} \cdot \xrightarrow[\text{along } N]{\text{Softmax}} \mathcal{C}_{N,M}, \tag{5}$$

where the first DWConv module uses a relatively large $k = 7$ kernel with stride $s = 1$ and zero padding 3, which is to integrate local information with a larger receptive field.

ii) Clustering via MLP: To be more generic by eliminating the dependence on 2D convolution in Eqn. 4, it utilizes a MLP implementation,

$$X_{N,C} \xrightarrow[C\rightarrow 4C]{\text{Linear+GELU}} \cdot \xrightarrow[4C\rightarrow M]{\text{Linear}} \cdot \xrightarrow[\text{along } N]{\text{Softmax}} \mathcal{C}_{N,M}, \tag{6}$$

where the expansion ratio of the hidden layer of the MLP is set to 4 by default.

To encourage forming meaningful clusters (i.e., visual tokens) that can capture underlying visual patterns that are often spatially sparse, we apply Softmax along the spatial dimension in Eqns. 5 and 6, which also enables directly visualizing $\mathcal{C}_{N,M}$ as M heatmaps *for diagnosing the interpretability* of a trained model at the instance level in a forward computation.

Where to compute $\mathcal{C}_{N,M}$ in the onsite clustering setting? As aforementioned, $\mathcal{C}_{N,M}$ can be computed in either a block-wise or a stage-wise setting (the right-bottom of Fig. 3). We observed that the latter is not only more computationally efficient, but also more effective in terms of accuracy in our ablation study. Intuitively, sharing the cluster assignment in a stage facilitates consistency between differ-

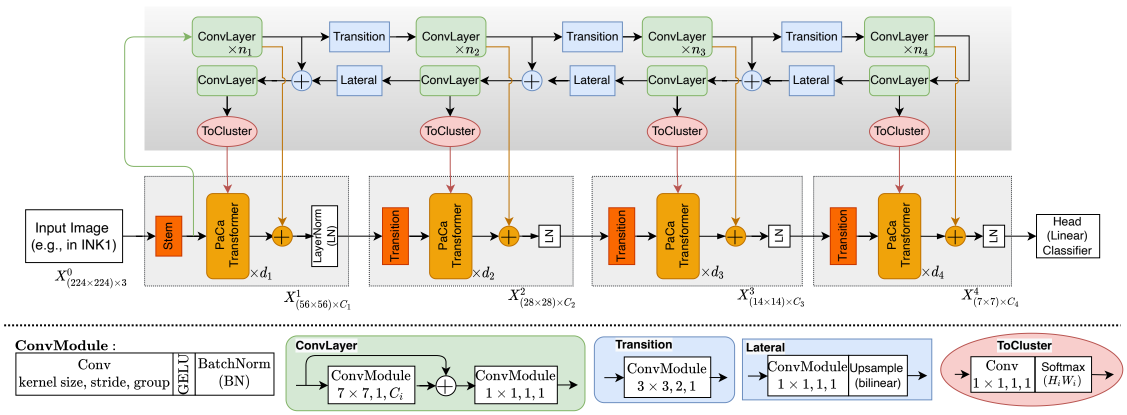

Figure 4. Illustration of computing the cluster assignment via an external clustering teacher network. See text for details.

ent Transformer blocks, and may induce meaningful latent features at the front-end of a stage (Eqn. 4) based on the collective feedback from all the blocks in a stage during training.

3.2.2 Understanding the PaCa Module

Intuitively, computing $Z_{M,C}$ via the matrix multiplication (Fig. 1 (b) and Eqn. 2) can be understood as a depth-wise global weighted pooling of the input $X_{N,C}$ with learned weights, $\mathcal{C}_{N,M}$. It can also be seen as a dynamic MLP-Mixer [42] with data-driven weight parameters $\mathcal{C}_{N,M}$ for the spatial transformation and integration component, rather than using top-down model parameters, making it more flexible by not restricting the trained models to a specific input size. The learned clusters (latent visual tokens) $Z_{M,C}$ share similar spirit to the class-token(s) or task prompts used in the vanilla ViT models (single class token) and its variants with multiple class-tokens. The former are data driven, while the latter are treated as model parameters.

Furthermore, the learned clustering assignment $\mathcal{C}_{N,M}$ has the same form of the attention matrix $A_{N,M}$ (Eqn. 1). Computing $Z_{M,C}$ (Eqn. 2) can thus be understood as performing cross-covariance attention (XCA) [14].

Learning $\mathcal{C}_{N,M}$ itself can be understood as a way of learning better visual tokens to bridge the gap between patches and the textual tokens used in natural language processing Transformer models. This type of visual tokenizer has also been observed to be useful in integrating Transformer models on top of convolution neural networks (CNNs) such as ResNets in Visual Transformer [50].

3.2.3 External Clustering

With the onsite clustering setting, the cluster assignments $\mathcal{C}_{N,M}$'s at the early stages are based on the low-to-middle level information. To address this issue, we are inspired by three lines of work: the feature pyramid network (FPN) [30] that is widely used for integrating visual information at different levels, the slow-fast thinking paradigm [26] (i.e., System 1 v.s. System 2) recently prompted for inducing reasoning capabilities in neural networks [17], and the empirical observations of Transformers focusing more on low-frequency information and CNNs focusing more on high-frequency information [35]. We introduce an *external clustering* teacher CNN (Fig. 4) that is concurrently trained with the PaCa ViT end-to-end. To be lightweight, we use the ConvMixer layer [44] in the FPN-style CNN clustering teacher network [2].

With the clustering teacher network, we first compute all the stage-wise clustering assignments, and then learn the PaCa ViT. We also integrate the stage outputs from the teacher network into the PaCa ViT. The clustering teacher network can be interpreted as a fast learner to provide informative guidance (the cluster assignment) to the relatively slower PaCa learner. It can also be intuitively interpreted as a type of working memory [8] that "manipulate" the input data to facilitate the "post-processing" via the PaCa. In addition, this integration may facilitate harnessing the joint expressive power of learning high-frequency information by the CNN teacher and of learning low-frequency information by the Transformer based models [35].

3.2.4 Task-Specific Tuning of M

The number of clusters M can be changed accounting for the task specific information. For example, when using an ImageNet-1k pretrained PaCa model that uses a relative small M (e.g., 100) as the backbone in a downstream task (e.g., the 150-class MIT-ADE20k dataset), we observe that we can change M to a large number (e.g., 200), which only results in minor changes of the network (e.g., the PWConv in Eqn. 5) and has no training issue observed in our experiments (see Sec. 4.3).

3.2.5 Complexity Analyses

Compared with PVTv2 [47], our PaCa (Eqn. 2) leads to *linear* complexity in computing the self-attention matrix (Eqn. 1) since the number of clusters M is predefined and

[2]Many other lightweight CNNs, such as the MobileNets [37], can be straightforwardly applied.

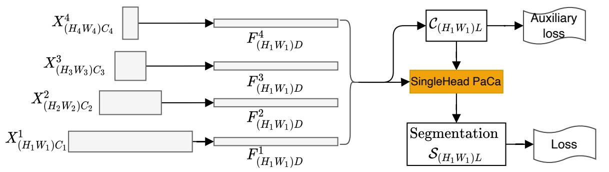

Figure 5. The proposed PaCa head network for semantic segmentation. The feature pyramid from the backbone (Fig. 3) is projected to a D-dim feature space and resized to the resolution of the first feature layer (via bilinear interpolation). L is the number of classes (e.g., 150 in the MIT-ADE20k dataset). Both the losses are cross-entropy, but with a smaller weight for the auxiliary loss.

fixed in our PaCa-ViT models. This advantage is achieved at the expense of the overhead cost in Eqn. 4 or Eqn. 5 and the matrix multiplication in Eqn. 2. For relatively small images (e.g., in image classification), the overhead cost slightly outweighs the reduction in computing the self-attention matrix (see Sec. 4.1). For large images (e.g., in object detection and instance semantic segmentation), the overhead cost is well paid off, leading to significant reduction of computing cost and memory footprint (see Sec. 4.2 and Sec. 4.3).

3.3. Network Interpretability via PaCa

To select the most important clusters in $\mathcal{C}_{N,M}$ for an input image I in a vision task (e.g., image classification), we adopt a straightforward approach. We use the clustering assignment maps $\mathcal{C}_{N,M}$ before the Softmax and then apply the Sigmoid transformation. For each cluster m, we reshape the slice $\mathcal{C}_{N,m}$ back to a 2D spatial heatmap , denoted by $\mathcal{H}^m_{h,w}$. We first compute a binary mask by keeping locations whose clustering scores are greater than the mean score,

$$\mathbb{M}^m_{h,w} = \mathcal{H}^m_{h,w} > mean(\mathcal{H}^m_{h,w}), \tag{7}$$

which is then upsampled to the resolution of input images (e.g., 224× 224) using the nearest interpolation, denoted by $\mathbb{M}^m$. The upsampled mask is used to mask the input image I that can be correctly classified by the model, and we have,

$$I^m = I \odot \mathbb{M}^m, \tag{8}$$

where $\odot$ represents element-wise product. I^m is then used as the input to the model.

We divide the learned clusters into two groups: the positive group in which a masked image I^m can still predict the ground-truth label, and the negative group in which a masked image $I^{m'}$ has the wrong predicted label. The positive group means that the masked portion in an image based on the clustering assignment retains sufficient information.

3.4. The PaCa Segmentation Head

The image semantic segmentation task can provide direct supervision signals to the clustering assignment $\mathcal{C}_{N,M}$ (with $M = L$ the number of the ground-truth classes). We elaborate the design shown in Fig. 5 in this section and present results in Sec. 4.3. With the feature pyramid $X^i_{(H_iW_i)C_i}$'s (e.g., $i = 1, 2, 3, 4$) from the backbone, we first transform each pyramid layer into a D-dim feature space, $F^i_{H_1 W_1} D$ via a vanilla convolution block (ConvBlock) consisting of 1×1 convolution, BatchNorm [24] and ReLU [29], followed by a bilinear upsampling (for layers other than the first one). Denote by $F_{(H_1W_1)4D}$ as the concatenated feature map as the multi-scale fused input, from which the clustering assignment $\mathcal{C}_{N_1,L}$ ($N_1 = H_1 \times W_1$) is computed,

$$F_{N_1,4D} \xrightarrow[4D\to D]{\text{ConvBlock}} \cdot \xrightarrow[k=1,s=1]{\text{PWConv}} \cdot \xrightarrow[\text{along } N_1]{\text{Softmax}} \mathcal{C}_{N_1,L}. \tag{9}$$

Then, a single-head PaCa module is used with the Query, Key and Value computed as follows,

$$\text{Query: } F_{N_1,4D} \xrightarrow{\text{ConvBlock}} Q_{N_1,D}, \tag{10}$$

$$\text{Clusters: } Z_{L,4D} = \mathcal{C}^T_{N_1,L} \cdot F_{N_1,4D}, \tag{11}$$

$$\text{Key \& Value: } Z_{L,4D} \xrightarrow{\text{LinearBlock}} K_{L,D}, V_{L,D}, \tag{12}$$

where a LinearBlock consisting of a linear projection layer, a BatchNorm1D and the ReLU. The output of the PaCa module is computed by,

$$\text{Softmax}_L(Q_{N_1,D} \cdot K^T_{L,D}) \cdot V_{L,D} \xrightarrow{\text{ConvBlock}} \mathbf{F}_{N_1,D}. \tag{13}$$

The final segmentation result is regressed via,

$$\mathcal{S}_{N_1,L} = \text{PWConv}(\mathbf{F}_{N_1,D}). \tag{14}$$

4. Experiment

In this section, we present experimental results of the proposed method in ImageNet-1k (IN1K) [12] classification, MS-COCO 2017 object detection and instance segmentation [31] and MIT-ADE20k semantic segmentation [64]. In implementation, we use the popular `timm` PyTorch toolkit [49] for image classification, the `mmdetection` and `mmsegmentation` toolkits [5] for object detection and semantic segmentation respectively.

We mainly test three stage-wise onsite clustering-via-convolution PaCa models (Fig. 3): PaCa-Tiny (12.2M), PaCa-Small (22.0M) and PaCa-Base (46.9M). For comparisons, we one stage-wise onsite clustering-via-MLP PaCa mode: PaCa^{mlp}-Small (22.6M). For onsite clustering, the PaCa is used in the first three stages with the number of clusters $M = 100$. We also test two stage-wise external clustering based PaCa models (Fig. 4): PaCa^{ec}-small (21.1M) and PaCa^{ec}-base (47.65M), both with $M = 100$ for all the four stages. Detailed architectural specifications are provided in the supplementary. The training recipes are the same with the prior art and provided in the supplementary too.

4.1. Image Classification

The IN1K classification dataset [12] consists of about 1.28 million images for training, and $50,000$ for validation, from $1,000$ classes. All models are trained on the training set for fair comparisons and report the Top-1 accuracy on the validation set. We follow the training recipe used by the PVTv2 which in turn is adopted from the DeiT [43].

Accuracy. Table 1 shows the comparisons. The pro-

Method	#Params (M)$\downarrow$	FLOPs (G) $\downarrow$	Top-1 Acc. (%)$\uparrow$
DeiT-T/16 [43]	5.7M	1.3	72.2
PVT-T [48]	13.2	1.9	75.1
PVTv2-B1 [47]	14.0	2.1	78.7
PaCa-Tiny (ours)	12.2	3.2	**80.9** $\uparrow$2.2
DeiT-S/16 [43]	22.1	4.6	79.9
T2T-ViT$_t$-14 [58]	22.0	6.1	80.7
PVT-S [48]	24.5	3.8	79.8
TNT-S [20]	23.8	5.2	81.3
SWin-T [32]	29.0	4.5	81.3
CvT-13 [51]	20.0	4.5	81.6
Twins-SVT-S [9]	24.0	2.8	81.3
FocalAtt-Tiny [56]	28.9	4.9	82.2
PVTv2-B2 [47]	25.4	3.9	82.0
PVTv2-B2-li [47]	22.6	4.0	82.1
PaCa-Small (ours)	22.0	5.5	83.08 $\uparrow$0.98
PaCa^{mlp}-Small (ours)	22.6	5.9	83.13 $\uparrow$1.03
PaCa^{ec}-Small (ours)	21.1	5.4	**83.17** $\uparrow$1.07
T2T-ViT$_t$-19 [58]	39.0	9.8	81.4
T2T-ViT$_t$-24 [58]	64.0	15.0	82.2
PVT-M [48]	44.2	6.7	81.2
PVT-L [48]	61.4	9.8	81.7
CvT-21 [51]	32.0	7.1	82.5
TNT-B [20]	66.0	14.1	82.8
SWin-S [32]	50.0	8.7	83.0
SWin-B [32]	88.0	15.4	83.3
Twins-SVT-B [9]	56.0	8.3	83.2
Twins-SVT-L [9]	99.2	14.8	83.7
FocalAtt-Small [56]	51.1	9.4	83.5
FocalAtt-Base [56]	89.8	16.4	83.8
PVTv2-B3 [47]	45.2	6.9	83.2
PVTv2-B4 [47]	62.6	10.1	83.6
PVTv2-B5 [47]	82.0	11.8	83.8
PaCa-Base (ours)	46.9	9.5	83.96 $\uparrow$0.76
PaCa^{ec}-Base (ours)	46.7	9.7	**84.22** $\uparrow$1.02

Table 1. Top-1 accuracy comparison in IN1K validation set using the single center crop (224×224) evaluation protocol. The relative improvement of our PaCa models are computed with respect to the PVTv2 models (underlined) with similar parameters.

posed PaCa ViT obtains consistently better performance than many variants of ViTs including the baseline PVTv2, which justifies the effectiveness of the proposed patch-to-cluster attention. With the onsite stage-wise clustering setting, clustering-via-MLP (Eqn. 6) is slightly better than clustering-via-convolution (Eqns. 4 and 5). The external clustering (Fig. 4) outperforms the onsite clustering (Fig. 3) slightly. Fig. 6 shows some examples of the learned clusters. **Efficiency.** In terms of efficiency based on FLOPs, the proposed PaCa models are slightly worse at the resolution of 224×224 in IN1K. As aforementioned, the efficiency will significantly improve and outperform other variants in downstream tasks with higher resolution images.

4.2. Object Detection and Instance Segmentation

The challenging MS-COCO 2017 benchmark [31] is used, which consists of a subset of train2017 (118k images) and a subset of val2017 (5k images). Following the common settings, we use the IN1K pretrained PaCa ViT models as the feature backbone, and test them using the Mask R-CNN framework [21] under the 1x schedule.

Accuracy. Table 2 shows the comparisons. The proposed PaCa models obtain consistently better performance than other ViT variants. The clustering-via-MLP obtains slightly better performance than both the clustering-via-convolution and the external clustering with the small model configuration. With the base model configuration, the external clustering is slightly better than the clustering-via-convolution. **Efficiency.** Overall, our PaCa models are significantly more efficient as shown by the FLOPs comparing with the baseline PVTv2.

Backbone	#Params (M)	FLOPs (G)	AP^b	AP$^b_{50}$	AP$^b_{75}$	AP^m	AP$^m_{50}$	AP$^m_{75}$
PVT-T [48]	32.9	-	39.8	62.2	43.0	37.4	59.3	39.9
PVTv2-B1 [47]	33.7	259*	41.8	64.3	45.9	38.8	61.2	41.6
PaCa-Tiny (ours)	32.0	252*	**43.3**	66.0	47.5	**39.6**	62.9	42.4
ResNet-50 [22]	44.2	260	41.0	61.7	44.9	37.1	58.4	40.1
SWin-T [32]	47.8	264	43.7	66.6	47.7	39.8	63.3	42.7
Twins-SVT-S [9]	44.0	228	43.4	66.0	47.3	40.3	63.2	43.4
FocalAtt-T [56]	48.8	291	44.8	67.7	49.2	41.0	64.7	44.2
PVT-S [48]	44.1	245	43.0	65.3	46.9	39.9	62.5	42.8
PVTv2-B2 [47]	45.0	325*	45.3	67.1	49.6	41.2	64.2	44.4
PaCa-Small (ours)	41.8	296*	46.4	68.7	50.9	41.8	65.5	45.0
PaCa^{mlp}-Small (ours)	42.4	303*	**46.6**	69.0	51.3	**41.9**	65.7	45.0
PaCa^{ec}-Small (ours)	40.9	292*	45.8	68.0	50.3	41.4	64.9	44.5
SWin-S [32]	69.1	354	46.5	68.7	51.3	42.1	65.8	45.2
SWin-B [32]	107.1	497	46.9	69.2	51.6	42.3	66.0	45.5
FocalAtt-S [56]	71.2	401	47.4	69.8	51.9	42.8	66.6	46.1
FocalAtt-B [56]	110.0	533	47.8	70.2	52.5	43.2	67.3	46.5
Twins-SVT-B [9]	76.3	340	45.2	67.6	49.3	41.5	64.5	44.8
PVT-M [48]	63.9	302	42.0	64.4	45.6	39.0	61.6	42.1
PVT-L [48]	81.0	364	42.9	65.0	46.6	39.5	61.9	42.5
PVTv2-B3 [47]	64.9	413*	47.0	68.1	51.7	42.5	65.7	45.7
PVTv2-B4 [47]	82.2	516*	47.5	68.7	52.0	42.7	66.1	46.1
PVTv2-B5 [47]	101.6	573*	47.4	68.6	51.9	42.5	65.7	46.0
PaCa-Base (ours)	66.6	373*	48.0	69.7	52.1	42.9	66.6	45.6
PaCa^{ec}-Base (ours)	61.4	372*	**48.3**	70.5	52.6	**43.3**	67.2	46.6

Table 2. Object detection and instance segmentation on MS-COCO val2017 [31] using the IN1K pretrained backbones and the Mask R-CNN [21] with the 1x (12-epoch) training schedule in training. FLOPs are computed at the input resolution of 1280×800. *computed using the torchprofile package.

Backbone	Head	#Params (M)	FLOPs (G)	mIOU
PVT-T [48]	Semantic FPN [27]	17.0	33.2	35.7
PVT-S		28.2	44.5	39.8
PVT-M		48.0	61.0	41.6
PVT-L		65.1	79.6	42.1
PVTv2-B1 [47]		17.8	34.2	42.5
PVTv2-B2		29.1	45.8	45.2
PVTv2-B3		49.0	62.4	47.3
PVTv2-B4		66.3	81.3	47.9
PVTv2-B5		85.7	91.1	48.7
SWin-T [32]	UperNet [53]	60	941	44.5
SWin-S		81	1038	47.6
SWin-B		121	1188	48.1
FocalAtt-T [56]	UperNet [53]	62	998	45.8
FocalAtt-S		85	1130	48.0
FocalAtt-B		126	1354	49.0
PaCa-Tiny (ours)	UperNet [53]	41.6	229.9*	44.49
PaCa-Small (ours)		51.4	242.7*	47.6
PaCa-Base (ours)		77.2	264.1*	49.67
PaCa-Tiny (ours)	PaCa (ours)	13.3	34.4*	45.65
PaCa-Small (ours)		23.2	47.2*	48.3
PaCa^{mlp}-Small (ours)		24.0	50.0*	48.2
PaCa^{ec}-Small (ours)		22.2	46.4*	46.2
PaCa-Base (ours)		48.0	68.5*	**50.39**
PaCa^{ec}-Base (ours)		48.8	68.7*	48.4

Table 3. Semantic segmentation on MIT-ADE20k [64] with the crop size 512×512 using the IN1K pretrained backbones. FLOPs are computed at the input resolution of 512×512. *computed using the torchprofile package.

4.3. Semantic Segmentation

The MIT-ADE20k [64] benchmark is used, which is a challenging dense prediction task consisting of $L = 150$

#Clusters	Where?	IN1K			MS-COCO w/ Mask RCNN 1x								MIT-ADE20K w/ PaCa Head		
		#Params	FLOPs	Top-1(%)	#Params	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	#Params	FLOPs	mIOU
(100, 100, 100, 100)	stage-wise	22.3	5.6	83.05	42.0	294	**46.4**	68.8	51.0	**41.8**	65.6	44.6	23.4	47.3	**48.3**
(49, 64, 81, 100)		22.2	5.3	82.98	42.0	289	46.1	68.7	50.4	41.5	65.3	44.3	23.4	47.3	47.8
(100, 81, 64, 49)		22.2	5.3	82.87	42.0	291	46.1	68.4	50.3	41.7	65.4	44.7	23.7	47.3	47.6
(49, 49, 49, 0)		22.0	5.0	82.95	41.8	289	46.2	68.6	50.6	41.6	65.4	44.3	23.2	47.2	48.1
(2, 2, 2, 0)		22.0	4.7	82.28	41.6	283	45.5	68.4	49.9	41.1	64.9	43.9	23.2	47.2	47.7
(100, 100, 100, 0)		22.0	5.5	**83.08**	41.8	296	**46.4**	68.7	50.9	**41.8**	65.5	45.0	23.2	47.2	**48.3**
(100, 100, 100, 100)	block-wise	24.2	6.1	82.93	44.0	304	46.5	68.7	51.0	41.8	65.6	45.0	25.8	50.9	48.0

Table 4. Ablation study on the number M of clusters using the onsite clustering-via-convolution `PaCa-Small` model on IN1K (Top-1), MS-COCO (with Mask-RCNN 1x) and MIT-ADE20k (with the proposed PaCa head). As mentioned in the submission (Sec. 4.3), on MIT-ADE20k, the number of clusters in a stage of the ImageNet-pretrained backbone is reset to 200 if it is not zero. See details in text.

ground-truth classes. We use the IN1K pretrained PaCa ViT models as the feature backbone, and the proposed PaCa segmentation head network (Fig. 5). Since the pretrained PaCa backbones are trained with $M = 100$ clusters that is smaller than L, we change $M = 200$ in this task, and observe no issues of training, and better performance than the counterpart during our development.

Table 3 shows the comparisons. With the same UperNet [53] head, the proposed PaCa-ViT backbones (stage-wise onsite clustering-via-convolution) consistently outperform other methods. With the proposed PaCa head, the proposed PaCa-ViT models further improve the performance, while significantly reducing the complexity compared with the UperNet head. This shows the effectiveness of the proposed PaCa head for semantic segmentation, which has a simple structure by design. It is also more effective than the semantic FPN [27] head. With the PaCa segmentation head, the stage-wise onsite clustering-via-convolution models obtain better performance than the counterparts.

4.4. Ablation Study

In this section, we present an ablation study on the number M of clusters in each of the four stages (Fig. 3). The results are shown in Table 4. Interestingly, in terms of image classification performance, the number of clusters does not have a significant impact based on the cases tested (even with the number of clusters pushed to 2), which shows the robustness of the proposed PaCa models, but also suggests a potential improvement that may be worth exploring: Similar in spirit to the auxiliary loss used in the PaCa segmentation head (Fig. 5), some self-supervised loss functions (e.g., the loss function proposed in the Barlow Twins [60]) could be leveraged to induce learning diverse and complementary clusters for capturing underlying meaningful patterns (reusable and composable parts) at scene-/object-/part-levels. Based on the diversity of clusters, instance-sensitive cluster masks can be learned to filter out redundant clusters on the fly. Based on the visualization of learned clusters (Fig. 6), we observe redundant clusters and cluttered clusters. We leave those for the future work.

5. Conclusion

This paper presents a patch-to-cluster attention (PaCa) module for learning efficient and interpretable Vision Transformers (ViTs). The proposed PaCa can address the quadratic complexity issue and account for the spatial redundancy of patches in the commonly used patch-to-patch attention. It also provides a forward explainer for diagnosing the explainability of ViTs. A simple learnable clustering module is introduced for easy integration in the ViT models. The proposed PaCa is also used in designing a lightweight yet effective semantic segmentation head network. In experiments, the proposed PaCa is tested in IN1K, MS-COCO and MIT-ADE20k benchmarks. It obtains consistently better performance than the prior art including the SWin-Transformers and PVTs. It also shows semantically meaningful qualitative results of the learned clusters.

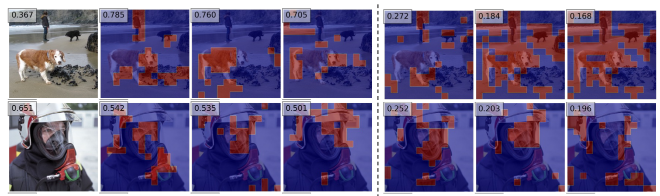

Figure 6. Examples of visualizing the clusters using the PaCa-Small model using the method presented in Sec. 3.3. Both images are correctly classified by the model. The left three clusters are the top-3 in the positive group, and the right three clusters are the top-3 in the negative group. The top-k in either group is defined based on the prediction probability of the ground-truth class. For the first image, it is interesting to see the positive group leads to higher prediction probabilities than the raw input image. The full visualization is provided in the supplementary.

Acknowledgements

This research is partly supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via Contract No. 2021-21040700003, ARO Grant W911NF1810295, NSF IIS-1909644, ARO Grant W911NF2210010, NSF IIS-1822477, NSF CMMI-2024688 and NSF IUSE-2013451. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, ARO, NSF, DHHS or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright annotation thereon. The authors are grateful for the constructive comments by anonymous reviewers and area chairs.

References

[1] Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. Etc: Encoding long and structured inputs in transformers. *arXiv preprint arXiv:2004.08483*, 2020. 2

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 2

[3] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2

[4] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 782–791, 2021. 1, 3

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6, 12

[6] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. 2

[7] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *International Conference on Learning Representations*, 2020. 3

[8] Thomas B Christophel, P Christiaan Klink, Bernhard Spitzer, Pieter R Roelfsema, and John-Dylan Haynes. The distributed nature of working memory. *Trends in cognitive sciences*, 21(2):111–124, 2017. 5

[9] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *arXiv preprint arXiv:2104.13840*, 1(2):3, 2021. 7

[10] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 13

[11] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020. 13

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 6, 12

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 3

[14] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681*, 2021. 5

[15] Stuart Geman, Daniel Potter, and Zhi Yi Chi. Composition systems. *Quarterly of Applied Mathematics*, 60(4):707–736, 2002. 2

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010. 12, 13

[17] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A*, 478(2266):20210068, 2022. 5

[18] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science Robotics*, 4(37), 2019. 1, 3

[19] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020. 2

[20] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *Advances in Neural Information Processing Systems*, 34:15908–15919, 2021. 7

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 7, 12

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[23] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016. 12

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 448–456. JMLR Workshop and Conference Proceedings, 2015. 6

[25] Md Amirul Islam, Sen Jia, and Neil DB Bruce. How much position information do convolutional neural networks encode? *arXiv preprint arXiv:2001.08248*, 2020. 4

[26] Daniel Kahneman. *Thinking, fast and slow*. Macmillan, 2011. 5

[27] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6399–6408, 2019. 3, 7, 8

[28] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*. 2
[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems (NIPS)*, pages 1106–1114, 2012. 6
[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 5
[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 6, 7, 12
[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 1, 2, 3, 7
[33] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 12
[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 12, 14
[35] Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022. 5
[36] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018. 2
[37] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 4, 5, 13
[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 12
[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 12
[40] Y Tay, D Bahri, D Metzler, D Juan, Z Zhao, and C Zheng. Synthesizer: Rethinking self-attention in transformer models. arxiv 2020. *arXiv preprint arXiv:2005.00743*, 2020. 3
[41] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020. 2
[42] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021. 5
[43] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 6, 7, 12
[44] Asher Trockman and J Zico Kolter. Patches are all you need? *arXiv preprint arXiv:2201.09792*, 2022. 5, 13
[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1, 2, 3
[46] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and H Linformer Ma. Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 3
[47] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer, 2021. 1, 2, 3, 4, 5, 7, 13
[48] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021. 1, 2, 3, 4, 7
[49] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 6, 12
[50] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020. 5
[51] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021. 7
[52] Yu-Huan Wu, Yun Liu, Xin Zhan, and Ming-Ming Cheng. P2t: Pyramid pooling transformer for scene understanding. *arXiv preprint arXiv:2106.12011*, 2021. 2
[53] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018. 3, 7, 8, 12
[54] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nystr\" omformer: A nystr\" om-based algorithm for approximating self-attention. *arXiv preprint arXiv:2102.03902*, 2021. 3
[55] Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2964–2972, 2022. 3

[56] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *arXiv preprint arXiv:2107.00641*, 2021. 7

[57] Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10809–10818, 2022. 3

[58] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021. 7

[59] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. In *NeurIPS*, 2020. 2

[60] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021. 8

[61] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 12

[62] Zizhao Zhang, Han Zhang, Long Zhao, Ting Chen, and Tomas Pfister. Aggregating nested transformers. *arXiv preprint arXiv:2105.12723*, 2021. 2, 3

[63] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 12

[64] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 2, 6, 7, 12