Container: Context Aggregation Network

Anonymous Author(s)
Affiliation
Address
email

Abstract

Convolutional neural networks (CNNs) are ubiquitous in computer vision, with a myriad of effective and efficient variations. Recently, Transformers – originally introduced in natural language processing – have been increasingly adopted in computer vision. While early adopters continue to employ CNN backbones, the latest networks are end-to-end CNN-free Transformer solutions. A recent surprising finding shows that a simple MLP based solution without any traditional convolutional or Transformer components can produce effective visual representations. While CNNs, Transformers and MLP-Mixers may be considered as completely disparate architectures, we provide a unified view showing that they are in fact special cases of a more general method to aggregate spatial context in a neural network stack. We present the CONTAINER (CONText AggregatIon NEtwoRk), a general-purpose building block for multi-head context aggregation that can exploit long-range interactions *a la* Transformers while still exploiting the inductive bias of the local convolution operation leading to faster convergence speeds, often seen in CNNs. Our CONTAINER architecture achieves 82.7 % Top-1 accuracy on ImageNet using 22M parameters, +2.8 improvement compared with DeiT-Small, and can converge to 79.9 % Top-1 accuracy in just 200 epochs. In contrast to Transformer-based methods that do not scale well to downstream tasks that rely on larger input image resolutions, our efficient network, named CONTAINER-LIGHT, can be employed in object detection and instance segmentation networks such as DETR, RetinaNet and Mask-RCNN to obtain an impressive detection mAP of 38.9, 43.8, 45.1 and mask mAP of 41.3, providing large improvements of 6.6, 7.3, 6.9 and 6.6 pts respectively, compared to a ResNet-50 backbone with a comparable compute and parameter size. Our method also achieves promising results on self-supervised learning compared to DeiT on the DINO framework.

1 Introduction

Convolutional neural networks (CNNs) have become the *de facto* standard for extracting visual representations, and have proven remarkably effective at numerous downstream tasks such as object detection [33], instance segmentation [19] and image captioning [1]. Similarly, in natural language processing, Transformers rule the roost [13, 39, 38, 4]. Their effectiveness at capturing short and long range information have led to state-of-the-art results across tasks such as question answering [41] and language understanding [53].

In computer vision, Transformers were initially employed as long range information aggregators across space (e.g., in object detection [5]) and time (e.g., in video understanding [56]), but these methods continued to use CNNs [31] to obtain raw visual representations. More recently however, CNN-free visual backbones employing Transformer modules [49, 14] have shown impressive performance on image classification benchmarks such as ImageNet [30]. The race to dethrone CNNs has now begun to expand beyond Transformers – a recent unexpected result shows that a multi-layer perceptron (MLP) exclusive network [47] can be just as effective at image classification.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

On the surface, CNNs [31, 8, 58, 20], Vision Transformers (ViTs) [14, 49] and MLP-mixers [47] are typically presented as disparate architectures. However, taking a step back and analyzing these methods reveals that their core designs are quite similar. Many of these methods adopt a cascade of neural network blocks. Each block typically consists of aggregation modules and fusion modules. Aggregation modules share and accumulate information across a predefined context window over the module inputs (e.g., the self attention operation in a Transformer encoder), while fusion modules combine position-wise features and produce module outputs (e.g., feed forward layers in ResNet).

In this paper, we show that the primary differences in many popular architectures result from variations in their aggregation modules. These differences can in fact be characterized as variants of an affinity matrix within the aggregator that is used to determine information propagation between a query vector and its context. For instance, in ViTs [14, 49], this affinity matrix is dynamically generated using key and query computations; but in the Xception architecture [8] (that employs depthwise convolutions), the affinity matrix is static – the affinity weights are the same regardless of position, and they remain the same across all input images regardless of size. And finally the MLP-Mixer [47] also uses a static affinity matrix which changes across the landscape of the input.

Along this unified view, we present CONTAINER (CONText AggregatIon NEtwoRk), a general purpose building block for multi-head context aggregation. A CONTAINER block contains both static affinity as well as dynamic affinity based aggregation, which are combined using learnable mixing coefficients. This enables the CONTAINER block to process long range information while still exploiting the inductive bias of the local convolution operation. CONTAINER blocks are easy to implement, can easily be substituted into many present day neural architectures and lead to highly performant networks whilst also converging faster and being data efficient.

Our proposed CONTAINER architecture obtains 82.7 % Top-1 accuracy on ImageNet using 22M parameters, improving +2.8 points over DeiT-S [49] with a comparable number of parameters. It also converges faster, hitting DeiT-S's accuracy of 79.9 % in just 200 epochs compared to 300.

We also propose a more efficient model, named CONTAINER-LIGHT that employs only static affinity matrices early on but uses the learnable mixture of static and dynamic affinity matrices in the latter stages of computation. In contrast to ViTs that are inefficient at processing large inputs, CONTAINER-LIGHT can scale to downstream tasks such as detection and instance segmentation that require high resolution input images. Using a CONTAINER-LIGHT backbone and 12 epochs of training, RetinaNet [33] is able to achieve 43.8 mAP, while Mask-RCNN [19] is able to achieve 45.1 mAP on box and 41.3 mAP on instance mask prediction, improvements of +7.3, +6.9 and +6.6 respectively, compared to a ResNet-50 backbone. The more recent DETR and its variants SMCA-DETR and Deformable DETR [5, 18, 68] also benefit from CONTAINER-LIGHT and achieve 38.9, 43.0 and 44.2 mAP, improving significantly over their ResNet-50 backbone baselines.

CONTAINER-LIGHT is data efficient. Our experiments show that it can obtain an ImageNet Top-1 accuracy of 61.8 using just 10% of training data, significantly better than the 39.3 accuracy obtained by DeiT. CONTAINER-LIGHT also convergences faster and achieves better kNN accuracy (71.5) compared to DeiT (69.6) under DINO self-supervised training framework [6].

The CONTAINER unification and framework enable us to easily reproduce several past models and even extend them with just a few code and parameter changes. We extend multiple past models and show improved performance – for instance, we produce a Hierarchical DeiT model, a multi-head MLP-Mixer and add a static affinity matrix to the DeiT architecture. Our code base and models will be released publicly. Finally, we analyse a CONTAINER model containing both static and dynamic affinities and show the emergence of convolution-like local affinities in the early layers of the network.

In summary, our contributions include: (1) A unified view of popular architectures for visual inputs – CNN, Transformer and MLP-mixer, (2) A novel network block – CONTAINER, which uses a mix of static and dynamic affinity matrices via learnable parameters and the corresponding architecture with strong results in image classification and (3) An efficient and effective extension – CONTAINER-LIGHT with strong results in detection and segmentation. Importantly, we see that a number of concurrent works are aiming to fuse the CNN and Transformer architectures [32, 59, 36, 21, 50, 64, 59, 43], validating our approach. We hope that our unified view helps place these different concurrent proposals in context and leads to a better understanding of the landscape of these methods.

2 Related Work

Visual Backbones. Since AlexNet [30] revolutionized computer vision, a host of CNN based architectures have provided further improvements in terms of accuracy including VGG [42], ResNet [20], Inception Net [44], SENet [25], ResNeXt [58] and Xception [8] and efficiency including Mobile-net v1 [23], Mobile-net v2 [23] and Efficient-net v2 [46]. With the success of Transformers [51] in NLP such as BERT [13] and GPT [39], researchers have begun to apply them towards solving the long range information aggregation problem in computer vision. ViT [14]/DeiT [49] are transformers that achieve better performance on ImageNet than CNN counterparts. Recently, several concurrent works explore integrating convolutions with transformers and achieve promising results. ConViT [11] explores soft convolutional inductive bias for enhancing DeiT. CeiT [61] directly incorporates CNNs into the Feedforward module of transformers to enhance the learned features. PVT [55] proposes a pyramid vision transformer for efficient transfer to downstream tasks. Pure Transformer models such as ViT/DeiT however, require huge GPU memory and computation for detection [55] and segmentation [66] tasks, which need high resolution input. MLP-Mixer [47] shows that simply performing transposed MLP followed by MLP can achieve near state-of-the-art performance. We propose CONTAINER, a new visual backbone that provides a unified view of these different architectures and performs well across several vision tasks including ones that require a high resolution input.

Transformer Variants. Vanilla Transformers are unable to scale to long sequences or high-resolution images due to the quadratic computation in self-attention. Several methods have been proposed to make Transformer computations more efficient for high resolution input. Reformer [29], Clusterform [52], Adaptive Clustering Transformer [66] and Asymmetric Clustering [10] propose to use Locality Sensitivity Hashing to cluster keys or queries and reduce quadratic computation into linear computation. Lightweight convolution [57] explore convolution architectures for replacing Transformers but only explore applications in NLP. RNN Transformer [28] builds a connection between RNN and Transformer and results in attention with linear computation. Linformer [54] changes the multiplication order of key,query,value into query,value,key by deleting the softmax normalization layer and achieve linear complexity. Performer [9] uses Orthogonal Random Feature to approximate full rank softmax attention. MLIN [17] performs interaction between latent encoded nodes, and its complexity is linear with respect to input length. Bigbird [3] breaks the full rank attention into local, randomly selected and global attention. Thus the computation complexity becomes linear. Longformer [63] uses local Transformers to tackle the problem of massive GPU memory requirements for long sequences. MLP-Mixer [47] is a pure MLP architecture for image recognition. In the unified formulation we provide, MLP-Mixer can be considered as a single-head Transformer with static affinity matrix weight. MLP-Mixer can provide more efficient computation than vanilla transformer due to no need to calculate affinity matrix using key query multiplication. Efficient Transformers mostly use approximate message passing which results in performance deterioration across tasks. Our CONTAINER unification performs global and local information exchange simultaneously using a mixture affinity matrix, while CONTAINER-LIGHT switches off the dynamic affinity matrix for high resolution feature maps to reduce computation. Although switching off the dynamic affinity matrix slightly hinders classification performance, CONTAINER-LIGHT still provides effective and efficient generalization to downstream tasks compared with popular backbones such as ViT and ResNet.

Transformers for Vision. Transformers enable high degrees of parallelism and are able to capture long-range dependencies in the input. Thus Transformers have gradually surpassed other architectures such as CNN [31] and RNN [22] on image [14, 5], audio [2] and language understanding [13]. In computer vision, Non-local Neural Network [56] has been proposed to capture long range interactions to compensate for the local information captured by CNNs and used for object detection [24] and semantic segmentation [16, 26, 69, 62]. However, these methods use Transformers as a refinement module instead of treating the transformer as a first-class citizen. ViT [14] introduces the first pure Transformer model into computer vision and surpasses CNNs with large scale pretraining on the non publicly available JFT dataset. DeiT [49] trains ViT from scratch on ImageNet-1k and achieve better performance than CNN counterparts. DETR [5] uses Transformer as an encoder and decoder architecture for designing the first end-to-end object detection system. Taming Transformer [15] use Vector Quantization [37] GAN and GPT [39] for high quality high-resolution image generation. Motivated by the success of DETR on object detection, Transformers have been applied widely on tasks such as semantic segmentation [67], pose estimation [60], trajectory estimation [35], 3D representation learning and self-supervised learning with MOCO v3 [7] and DINO [6].

3 Methods

In this section we first provide a generalized view of neighborhood/context aggregation modules commonly employed in present neural networks. Then we revisit three major architectures – Transformer [51], Depthwise Convolution [8] and the recently proposed MLP-Mixer [47], and show that they are special cases of our generalized view. We then present our CONTAINER module in Sec 3.3 and its efficient version – CONTAINER-LIGHT in Sec 3.5.

3.1 Contextual Aggregation for Vision

Consider an input image $X \in \mathbb{R}^{C\times H\times W}$, where C and $H \times W$ denote the channel and spatial dimensions of the input image, respectively. The input image is first flattened to a sequence of tokens $\{X_i \in \mathbb{R}^C | i = 1, \ldots, N\}$, where $N = HW$ and input to the network. Vision networks typically stack multiple building blocks with residual connections [20], defined as

$$\mathbf{Y} = \mathcal{F}(\mathbf{X}, \{\mathbf{W}_i\}) + \mathbf{X}. \tag{1}$$

Here, $\mathbf{X}$ and $\mathbf{Y}$ are the input and output vectors of the layers considered, and $\mathbf{W}_i$ represents the learnable parameters. $\mathcal{F}$ determines how information across $\mathbf{X}$ is aggregated to compute the feature at a specific location. We first define an affinity matrix $\mathcal{A} \in \mathbb{R}^{N\times N}$ that represents the neighborhood for contextual aggregation. Equation 1 can be re-written as:

$$\mathbf{Y} = (\mathcal{A}\mathbf{V})\mathbf{W}_1 + \mathbf{X}, \tag{2}$$

where $\mathbf{V} \in \mathbb{R}^{N\times C}$ is a transformation of $\mathbf{X}$ obtained by a linear projection $\mathbf{V} = \mathbf{X}\mathbf{W}_2$. $\mathbf{W}_1$ and $\mathbf{W}_2$ are the learnable parameters. $\mathcal{A}_{ij}$ is the affinity value between X_i and X_j. Multiplying the affinity matrix with $\mathbf{V}$ propagates information across features in accordance with the affinity values.

The modeling capacity of such a context aggregation module can be increased by introducing multiple affinity matrices, allowing the network to have several pathways to contextual information across $\mathbf{X}$. Let $\{\mathbf{V}^i \in \mathbb{R}^{N\times \frac{C}{M}} | i = 1, \ldots, M\}$ be slices of $\mathbf{V}$, where M is the number of affinity matrices, also referred to as the number of heads. The multi-head version of Equation 2 is

$$\mathbf{Y} = \mathrm{Concat}(\mathcal{A}_1\mathbf{V}_1, \ldots, \mathcal{A}_M\mathbf{V}_M)\mathbf{W}_2 + \mathbf{X}, \tag{3}$$

where $\mathcal{A}_m$ denotes the affinity matrix in each head. Different A_m can potentially capture different relationships within the feature space and thus increase the representation power of contextual aggregation compared with a single-head version. Note that only spatial information is propagated during contextual aggregation using the affinity matrices; cross-channel information exchange does not occur within the affinity matrix multiplication, and that there is no non-linear activation function.

3.2 The Transformer, Depthwise Convolution and MLP-Mixer

Transformer [51], depthwise convolution [27] and the recently proposed MLP-Mixer [47] are three distinct building blocks used in computer vision. Here, we show that they can be represented within the above context aggregation framework, by defining different types if affinity matrices.

Transformer. In the self-attention mechanism in Transformers, the affinity matrix is modelled by the similarity between the projected query-key pairs. With M heads, the affinity matrix in head m, $\mathcal{A}^{sa}_m$ can be written as

$$\mathcal{A}^{sa}_m = \mathrm{Softmax}(\mathbf{Q}_m\mathbf{K}^T_m/\sqrt{C/M}), \tag{4}$$

where $\mathbf{K}_m, \mathbf{Q}_m$ are the corresponding key, query in head m, respectively. The affinity matrix in self-attention is dynamically generated and can capture instance level information. However, this introduces quadratic computational, which requires heavy computation for high resolution feature.

Depthwise Convolution. The convolution operator fuses both spatial and channel information in parallel. This is different from the contextual aggregation block defined above. However, depthwise convolution [27] which is an extreme case of group convolution performs disentangled convolution. Considering the number of the heads from the contextual aggregation block to be equal to the channel size C, we can define the convolutional affinity matrix given the 1-d kernel $Ker \in \mathbb{R}^{C\times 1\times k}$:

$$\mathcal{A}^{conv}_{mij} = \begin{cases} Ker[m, 0, |i-j|] & |i-j| \leq k \\ 0 & |i-j| > k \end{cases}, \tag{5}$$

where $\mathcal{A}_{mij}$ is the affinity value between X_i and X_j on head m. In contrast with the affinity matrix obtained from self-attention whose value is conditioned on the input feature, the affinity values for convolution are static – they do not depend on the input features, sparse – only involves local connections and shared across the affinity matrix.

MLP-Mixer The recently proposed MLP-Mixer [47] does not rely on any convolution or self-attention operator. The core of MLP-Mixer is the transposed MLP operation, which can be denoted as $\mathbf{X} = \mathbf{X} + (\mathbf{V}^T \mathbf{W}_{MLP})^T$. We can define the affinity matrix as

$$\mathcal{A}^{mlp} = (\mathbf{W}_{MLP})^T, \tag{6}$$

where $\mathbf{W}_{MLP}$ represents the learnable parameters. This simple equation shows that the transposed-MLP operator is a contextual aggregation operator on a single feature group with a dense affinity matrix. Comparing with self-attention and depthwise convolution, the transpose-MLP affinity matrix is static, dense and with no parameter sharing.

The above simple unification reveals the similarities and differences between Transformer, depthwise convolution and MLP-Mixer. Each of these building blocks can be obtained by different formulating different affinity matrices. This finding leads us to create a powerful and efficient building block for vision tasks – the CONTAINER.

3.3 The CONTAINER Block

As detailed in Sec 3.2, previous architectures have employed either static or dynamically generated affinity matrices – each of which provides its unique set of advantages and features. Our proposed building block named CONTAINER, combines both types of affinity matrices via a learnable parameter. The single head CONTAINER is defined as:

$$\mathbf{Y} = ((\alpha \overbrace{\mathcal{A}(\mathbf{X})}^{Dynamic} + \beta \overbrace{\mathcal{A}}^{Static})V)W_2 + \mathbf{X} \tag{7}$$

$\mathcal{A}(\mathbf{X})$ is dynamically generated from $\mathbf{X}$ while $\mathcal{A}$ is a static affinity matrix. We now present a few special cases of the CONTAINER block. In the following, $\mathcal{L}$ denotes a learnable parameter.

- $\alpha = 1$, $\beta = 0$, $\mathcal{A}(x) = \mathcal{A}^{sa}$: A vanilla Transformer block with self-attention (denoted sa).
- $\alpha = 0$, $\beta = 1$, $M = C$, $\mathcal{A} = \mathcal{A}^{conv}$: A depthwise convolution block. In depthwise convolution, each channel has a different static affinity matrix. When $M \neq C$, the resultant block can be considered a Multi-head Depthwise Convolution block (MH-DW). MH-DW shares kernel weights.
- $\alpha = 0$, $\beta = 1$, $M = 1$, $\mathcal{A} = \mathcal{A}^{mlp}$: An MLP-Mixer block. When $M \neq 1$, we name the module Multi-head MLP (MH-MLP). MH-MLP splits channels into M groups and performs independent transposed MLP to capture diverse static token relationships.
- $\alpha = \mathcal{L}$, $\beta = \mathcal{L}$, $\mathcal{A}(x) = \mathcal{A}^{sa}$, $\mathcal{A} = \mathcal{A}^{mlp}$: This CONTAINER block fuses dynamic and static information, but the static affinity resembles the MLP-Mixer matrix. We call this block CONTAINER-PAM (Pay Attention to MLP).
- $\alpha = \mathcal{L}$, $\beta = \mathcal{L}$, $\mathcal{A}(x) = \mathcal{A}^{sa}$, $\mathcal{A} = \mathcal{A}^{conv}$: This CONTAINER block fuses dynamic and static information, but the static affinity resembles the depthwise convolution matrix. This static affinity matrix contains a locality constraint which is shift invariant, making it more suitable for vision tasks. This is the default configuration used in our experiments.

The CONTAINER block is easy to implement and can be readily swapped into an existing neural network. The above versions of CONTAINER provide variations on the resulting architecture and its performance and exhibit different advantages and limitations. The computation cost of a CONTAINER block is the same as a vanilla Transformer since the static and dynamic matrices are linearly combined.

3.4 The CONTAINER network architecture

We now present a base architecture used in our experiments. The unification of past works explained above allows us to easily compare self-attention, depthwise convolution, MLP and multiple variations of the CONTAINER block, and we perform these comparison using a consistent base architecture.

Motivated by networks in past works [20, 55], our base architecture contains 4 stages. In contrast to ViT/DeiT which down-sample the image to a low resolution and keep this resolution constant, each

stage in our architecture down-samples the image resolution gradually. Gradually down-sampling can retain image details, which is important for downstream tasks such as segmentation and detection. Each of the 4 stages contains a cascade of blocks. Each block contains two sub-modules, the first to aggregate spatial information (named spatial aggregation module) and the second to fuse channel information (named feed-forward module). In this paper, the channel fusion module is fixed to a 2-layer MLP as proposed in [51]. Designing a better spatial aggregation module is the main focus of this paper. The 4 stages contain 2, 3, 8 and 3 blocks respectively. Each stage uses patch embeddings which fuse spatial patches of size $p \times p$ into a single vector. For the 4 stages, the values of p are 4,4,2,2 respectively. The feature dimension within a stage remains constant – and is set to 128, 256, 320, and 512 for the four stages. This base architecture augmented with the CONTAINER block results in a similar parameter size as DeiT-S [49].

3.5 The CONTAINER-LIGHT network

We also present an efficient version known as CONTAINER-LIGHT which uses the same basic architecture as CONTAINER, but switches off the dynamic affinity matrix in the first 3 stages. The absence of the computation heavy dynamic attention at the early stages of computation help efficiently scale the model to process large image resolutions and achieve superior performance on downstream tasks such as detection and instance segmentation.

$$\mathcal{A}_m^{\text{CONTAINER-LIGHT}} = \begin{cases} \mathcal{A}_m^{conv} & Stage = 1, 2, 3 \\ \alpha \mathcal{A}_m^{sa} + \beta \mathcal{A}_m^{conv} & Stage = 4 \end{cases}, \tag{8}$$

α and β are learnable parameters. In network stage 1, 2, 3, CONTAINER-LIGHT will switch off $\mathcal{A}_m^{sa}$.

4 Experiments

We now present experiments with CONTAINER for ImageNet and with CONTAINER-LIGHT for the tasks of object detection, instance segmentation and self-supervised learning. We also present appropriate baselines. Please see the appendix for details of the models, training and setup.

4.1 ImageNet Classification

Top-1 Accuracy. Table 1 compares several highly performant models within the CNN, Transformer, MLP, Hybrid and our proposed CONTAINER families. CONTAINER and CONTAINER-LIGHT outperform the pure Transformer models ViT [14] and DeiT [49] despite far fewer parameters. They outperform PVT [55] which employ a hierarchical representation similar to our base architecture. They also outperform the recently published state-of-the-art SWIN [36] (they outperform Swin-T which has more parameters). The best performing models continue to be from the EfficientNet [45] family, but we note that EfficientNet [45] and RegNet [40] apply an extensive neural architecture search, which we do not. Finally note that CONTAINER-LIGHT not only achieves a high accuracy but does so at lower FLOPs and much faster throughput than models with comparable capacities.

The CONTAINER framework allows us to easily reproduce past architectures but also to create effective extensions over past work (outlined in Sec 3.3), several of which are compared in Table 2. H-DeiT-S is a hierarchical version of DeiT-S obtained by simply using $\mathcal{A}^{sa}$ within our hierarchical architecture and provides 1.2 gain. Conv-3 (naive convolution (conv) with 3×3 kernel) aggregates spatial and channel information, where as Group Conv-3 splits input features and performs convs using different kernels – it is cheaper and more effective. When group size = channel dim., we get depth-wise conv. DW-3 is a depthwise convs with 3 by 3 kernel that only aggregates spatial information. Channel information is fused using 1×1 convs. MH-DW-3 is a multi-head version of DW-3. MH-DW-3 shares kernel parameters within the same group. With fewer kernels, MH-DW-3 achieves comparable performance with DW-3. MLP is an implementation of transposed MLP for spatial propagation. MLP-LR stands for MLP with low-rank decompostion. MLP-LR provides better performance with fewer parameters. MH-MLP-LR adds a multi-head mechanism over MLP-LR and provides further improvements. In contrast to the original MLP-Mixer [47], we do not add any non-linearity like GELU into CONTAINER as is specified in the contextual aggregation equation.

Data Efficiency. CONTAINER-LIGHT has a built-in shift-invariance and parameter sharing mechanism. As a result it is more data efficient in comparison to DeiT [49]. Table 3 shows that at the low data regime of 10%, CONTAINER-LIGHT outperforms DeiT by a massive 22.5 points.

Family	Network	Top-1 Acc	Params	FLOPs	Throughput	Input dim	NAS
CNN	ResNet-50 [20]	78.5	25.6M	4.1G	1250.3	224^2	✗
	ResNet-101 [20]	79.8	44.7M	7.9G	753.7	224^2	✗
	Xception71 [8]	79.9	42.3M	N/A	423.5	299^2	✗
	RegNetY-4G [40]	80.0	21M	4.0G	1156.7	224^2	✓
	RegNetY-8G [40]	81.7	39M	8.0G	591.6	224^2	✓
	RegNetY-16G [40]	82.9	84M	16.0G	334.7	224^2	✓
	EfficientNet-B3 [45]	81.6	12M	1.8G	732.1	300^2	✓
	EfficientNet-B4 [45]	82.9	19M	4.2G	349.4	380^2	✓
	EfficientNet-B5 [45]	83.6	30M	9.9G	169.1	456^2	✓
	EfficientNet-B6 [45]	84.0	43M	19.0G	96.9	528^2	✓
	EfficientNet-B7 [45]	84.3	66M	37.0G	55.1	600^2	✓
Transformer	ViT-B/16 [14]	77.9	86M	55.4G	85.9	384^2	✗
	ViT-L/16 [14]	76.5	307M	190.7G	27.3	384^2	✗
	DeiT-S [49]	79.9	22.1M	4.6G	940.4	224^2	✗
	DeiT-B [49]	81.8	86M	17.5G	292.3	224^2	✗
	PVT-T [55]	75.1	13.2M	1.9G	N/A	224^2	✗
	PVT-S [55]	79.8	24.5M	3.8G	N/A	224^2	✗
	PVT-Medium [55]	81.2	44.2M	6.7G	N/A	224^2	✗
	PVT-L [55]	81.7	61.4M	9.8G	N/A	224^2	✗
	ViL-T [65]	76.3	6.7M	1.3G	N/A	224^2	✗
	ViL-S [65]	82.0	24.6M	4.9G	N/A	224^2	✗
	Swin-T [36]	81.3	29M	4.5G	755.2	224^2	✗
	Swin-S [36]	83.0	50M	8.7G	436.9	224^2	✗
	Swin-B [36]	83.3	88M	15.4G	278.1	224^2	✗
MLP	Mixer-B/16 [47]	76.4	79M	N/A	N/A	224^2	✗
	ResMLP-24 [48]	79.4	30M	6.0G	715.4	224^2	✗
Hybrid	ConvViT [11]	81.3	27M	5.4G	N/A	224^2	✗
	BoT-S1-50 [43]	79.1	20.8M	4.3G	N/A	224^2	✗
	BoT-S1-59 [43]	81.7	33.5M	7.3G	N/A	224^2	✗
Container (Ours)	CONTAINER	82.7	22.1M	8.1G	347.8	224^2	✗
	CONTAINER-LIGHT	82.0	20.0M	3.2G	1156.9	224^2	✗

Table 1: ImageNet [12] Top-1 accuracy comparison for CNN, Transformer, MLP, Hybrid and Container models. Throughput (images/s) is not reported in all papers (noted as N/A). Models that have fewer parameters than CONTAINER or upto 10% more parameters are highlighted.

Method	Top-1 Acc	Params	α	β	$\frac{C}{M}$	$\mathcal{A}^{dynamic}$	$\mathcal{A}^{static}$
H-DeiT-S	81.0	22.1M	1	0	32	$\mathcal{A}^{sa}$	N/A
Conv-3	79.6	33.8M	N/A	N/A	N/A	N/A	N/A
Group Conv-3	79.7	20.5M	N/A	N/A	N/A	N/A	N/A
DW-3	80.1	18.7M	0	1	1	N/A	$\mathcal{A}^{conv}$
MH-DW-3	79.9	18.6M	0	1	32	N/A	$\mathcal{A}^{conv}$
MLP	77.5	50.9M	0	1	C	N/A	$\mathcal{A}^{mlp}$
MLP-LR	78.9	36.5M	0	1	C	N/A	$\mathcal{A}^{mlp}$
MH-MLP-LR	79.6	41.6 M	0	1	32	N/A	$\mathcal{A}^{mlp}$
CONTAINER	82.7	22.1M	$\mathcal{L}$	$\mathcal{L}$	32	$\mathcal{A}^{sa}$	$\mathcal{A}^{conv}$
CONTAINER-LIGHT	82.0	20.0M	$\mathcal{L}$	$\mathcal{L}$	32	$\mathcal{A}^{sa}$	$\mathcal{A}^{conv}$

Table 2: ImageNet accuracies for architecture variations (with convolutions, self-attention and MLP) enabled within the CONTAINER framework. As per our notation, C: num channels, M: num heads, C/M: head dimension. See Sec 3.3 and 4.1 for notation and model details.

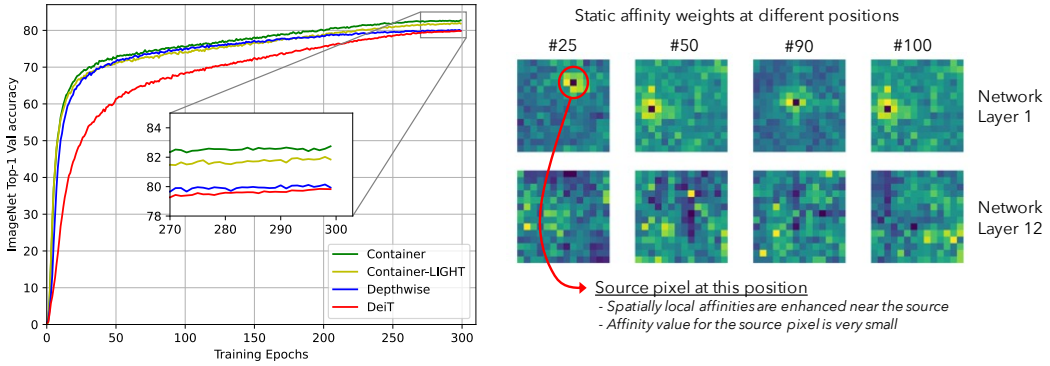

Figure 1: **(left)** Convergence speed comparison between CONTAINER, CONTAINER-LIGHT, Depthwise conv and DeiT. **(right)** Visualization of the static affinity weights at different positions and layers. Layer 1 displays the emergence of local affinities (resembling convolutions).

Data ratio	CONTAINER-LIGHT	DeiT
100 %	82.0 (+2.1)	79.9
80 %	81.1 (+2.6)	78.5
50 %	78.8 (+4.8)	74.0
10 %	61.8 (+22.5)	39.3

Table 3: ImageNet Top-1 Acc for CONTAINER-LIGHT and DeiT-S with varying training sizes.

Convergence Speed. Figure 1 (left) compares the convergence speeds of the two CONTAINER variants with a CNN and Transformer (DeiT) [49]. The inductive biases in the CNN enable it to converge faster than DeiT [49], but they eventually perform similarly at 300 epochs, suggesting that dynamic, long range context aggregation is powerful but slow to converge. CONTAINER combines the best of both and provides accuracy improvements with fast convergence. CONTAINER-LIGHT converges as fast with a slight accuracy drop.

Emergence of locality. Within our CONTAINER framework, we can easily add a static affinity matrix to the DeiT architecture. This simple change (1 line of code addition), can provide a +0.5 Top-1 improvement from 79.9% to 80.4%. This suggests that static and dynamic affinity matrices provide complementary information. As noted in Sec 3.3, we name this CONTAINER-PAM.

It is interesting to visualize the learnt static affinities at different network layers. Figure 1 (right) displays these for 2 layers. Each matrix represents the static affinities for a single position, reshaped to a 2-d grid to resemble the landscape of the neighboring regions. Within Layer 1, we interestingly observe the emergence of local operations via the enhancement of affinity values next to the source pixel (location). These are akin to convolution operations. Furthermore, the affinity value for the source pixel is very small, i.e. at each location, the context aggregator does not use its current feature. We hypothesize that this is a result of the residual connection [20], thereby alleviating the need to include the source feature within the context. Note that in contrast to dynamic affinity, the learnt static matrix is shared for all input images. Notice that Layer 12 displays a more global affinity matrix without any specific interpretable local patterns.

4.2 Detection with RetinaNet

Since the attention complexity for CONTAINER-LIGHT is linear at high image resolutions (initial layers) and then quadratic, it can be employed for downstream tasks such as object detection which usually require high resolution feature maps. Table 4 compares several backbones applied to the RetinaNet detector [33] on the COCO dataset [34]. Compared to the popular ResNet-50 [20], CONTAINER-LIGHT achieves 43.8 mAP, an improvement of 7.0, 7.2 and 10.4 on AP_S, AP_M, and AP_L with comparable parameters and cost. The significant increase for large objects shows the benefits of global attention via the dynamic global affinity matrix in our model. CONTAINER-LIGHT also surpasses the large convolution-based backbone X-101-64 [58] and pure Transformer models with similar number of parameters such as PVT-S [55], ViL-S [64], and SWIN-T [36] by large margins. Compared to large Transformer backbones such as ViL-M [64] and ViL-B [64], we achieve comparable performance with significantly fewer parameters and FLOPs.

4.3 Detection and Segmentation with Mask-RCNN

Table 4 also compares several backbones for detection and instance segmentation using the Mask R-CNN network [19]. As with the findings for RetinaNet [33], CONTAINER-LIGHT outperforms

	Mask R-CNN								RetinaNet					
Method	#P	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	#P	FLOPs	mAP	AP_S	AP_M	AP_L
ResNet50 [20]	44.2	180G	38.2	58.8	41.4	34.7	55.7	37.2	37.7	239G	36.5	20.4	40.3	48.1
ResNet101 [20]	63.2	259G	40.0	60.5	44.0	36.1	57.5	38.6	56.7	319G	38.5	21.7	42.8	50.4
X-101-32 [58]	62.8	259G	41.9	62.5	45.9	37.5	59.4	40.2	56.4	319G	39.9	22.3	44.2	52.5
X-101-64 [58]	101.9	424G	42.8	63.8	47.3	38.4	60.6	41.3	95.5	483G	41.0	23.9	45.2	54.0
PVT-S [55]	44.1	245G	40.4	62.9	43.8	37.8	60.1	40.3	34.2	226G	40.4	25.0	42.9	55.7
ViL-S [55]	45.0	174G	41.8	64.1	45.1	38.5	61.1	41.4	35.6	252G	41.6	24.9	44.6	56.2
SWIN-T [36]	48.0	267G	43.7	66.6	47.4	39.8	63.6	42.7	385	244G	41.5	26.4	45.1	55.7
ViL-M [55]	60.1	261G	43.4	65.9	47.0	39.7	62.8	42.1	50.7	338G	42.9	27.0	46.1	57.2
ViL-B [55]	76.1	365G	45.1	67.2	49.3	41.0	64.3	44.2	66.7	443G	44.3	28.9	47.9	58.3
BoT50 [43]	39.5	N/A	39.4	60.3	43.0	35.3	57	37.5	N/A	N/A	N/A	N/A	N/A	N/A
BoT50-(6x) [43]	39.5	N/A	43.7	64.7	47.9	38.7	61.8	41.1	N/A	N/A	N/A	N/A	N/A	N/A
CONTAINER-LIGHT	39.6	237G	45.1	67.3	49.5	41.3	64.2	44.5	29.7	218G	43.8	27.4	47.5	58.5

Table 4: Comparing the CONTAINER-LIGHT backbone with several previous methods at the tasks of object detection and instance segmentation using the Mask-RCNN and RetinaNet networks.

convolution and Transformer based approaches such as ResNet [20], X-101 [58], PVT [55], ViL [64] and recent state-of-the-art SWIN-T [36] and the recent hybrid approach BoT [43]. It obtains comparable numbers to the much larger ViL-B [64].

4.4 Detection with DETR

Table 5 shows that our model can consistently improve object detection performance compared to a ResNet-50 [20] backbone (comparable parameters and computation) on end-to-end object detection using DETR [5]. We demonstrate large improvements with DETR [5], DDETR [68] as well as SMCA-DETR [18]. See appendix for AP^S, AP^M, and AP^L numbers.

Method	mAP
DETR-ResNet50 [5]	32.3
DETR-CONTAINER-LIGHT	38.9
DDETR w/o multi-scale-ResNet50 [68]	39.3
DDETR w/o multi-scale-CONTAINER-LIGHT	43.0
SMCA w/o multi-scale-ResNet50 [18]	41.0
SMCA w/o multi-scale-CONTAINER-LIGHT	44.2

Table 5: CONTAINER-LIGHT and ResNet-50 backbones with DETR and variants for object detection.

4.5 Self supervised learning

We train DeiT [49] and CONTAINER-LIGHT for 100 epochs at the self supervised task of visual representation learning using the DINO framework [6]. Table 6 compares top-10 kNN accuracy for both backbones at different epochs of training. CONTAINER-LIGHT significantly outperforms DeiT with large improvements initially demonstrating more efficient learning.

Epochs →	20	40	60	80	100
DeiT [6]	52.0	63.3	66.5	68.9	69.6
CONTAINER-LIGHT	58.0	67.0	70.0	71.1	71.5

Table 6: CONTAINER-LIGHT and DeiT on DINO self-supervised learning.

5 Conclusion

In this paper, we have shown that disparate architectures such as Transformers, depth-wise CNNs and MLP-based methods are closely related via an affinity matrix used for context aggregation. Using this view, we have proposed CONTAINER, a generalized context aggregation building block that combines static and dynamic affinity matrices using learnable parameters. Our proposed networks, CONTAINER and CONTAINER-LIGHT show superior performance at image classification, object detection, instance segmentation and self-supervised representation learning. We hope that this unified view can motivate future research in the design of effective and efficient visual backbones.

Limitations: CONTAINER is very effective at image classification but cannot be directly applied to high resolution inputs. The efficient version CONTAINER-LIGHT, can be used for a variety of tasks. However, its limitation is that it is partially hand-crafted – the dynamic affinity matrix is switched off in the first 3 stages. Future work will address how to learn this using the task at hand.

Negative societal impact: This research does not have a direct negative societal impact. However, we should be aware that powerful neural networks, particularly image classification networks can be used for harmful applications like face and gender recognition.

References

[1] Peter Anderson, X. He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018. 1

[2] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*, 2020. 3

[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv*, 2020. 3

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. 1

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 2, 3, 9

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv*, 2021. 2, 3, 9

[7] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised visual transformers. *arXiv*, 2021. 3

[8] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 2, 3, 4, 7

[9] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *ICLR*, 2021. 3

[10] Giannis Daras, Nikita Kitaev, Augustus Odena, and Alexandros G Dimakis. Smyrf: Efficient attention using asymmetric clustering. In *NeurIPS*, 2020. 3

[11] Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. *arXiv*, 2021. 3, 7

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 7

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1, 3

[14] 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. In *ICLR*, 2021. 1, 2, 3, 6, 7

[15] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 3

[16] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019. 3

[17] Peng Gao, Haoxuan You, Zhanpeng Zhang, Xiaogang Wang, and Hongsheng Li. Multi-modality latent interaction network for visual question answering. In *ICCV*, 2019. 3

[18] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. *arXiv*, 2021. 2, 9

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 2, 8

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 3, 4, 5, 7, 8, 9

[21] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv*, 2021. 2

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8), 1997. 3

[23] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv*, 2017. 3

[24] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018. 3

[25] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 3

[26] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, 2019. 3

[27] Lukasz Kaiser, Aidan N Gomez, and Francois Chollet. Depthwise separable convolutions for neural machine translation. *arXiv*, 2017. 4

[28] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020. 3

[29] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. 3

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012. 1, 3

[31] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. 1, 2, 3

[32] Yawei Li, Kai Zhang, Jiezhang Cao, R. Timofte, and L. Gool. Localvit: Bringing locality to vision transformers. *arXiv*, 2021. 2

[33] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 2, 8

[34] 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 *ECCV*, 2014. 8

[35] Yicheng Liu, Jinghuai Zhang, Liangji Fang, Qinhong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. In *CVPR*, 2021. 3

[36] 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*, 2021. 2, 6, 7, 8, 9

[37] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2017. 3

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *arXiv*, 2021. 1

[39] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv*, 2015. 1, 3

[40] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10428–10436, 2020. 6, 7

[41] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016. 1

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[43] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *arXiv*, 2021. 2, 7, 9

[44] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 3

[45] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 6, 7

[46] Mingxing Tan and Quoc V. Le. Efficientnetv2: Smaller models and faster training. *arXiv*, 2021. 3

[47] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. *arXiv*, 2021. 1, 2, 3, 4, 5, 6, 7

[48] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv*, 2021. 7

[49] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv*, 2020. 1, 2, 3, 6, 7, 8, 9

[50] Ashish Vaswani, Prajit Ramachandran, A. Srinivas, Niki Parmar, Blake A. Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. *arXiv*, 2021. 2

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3, 4, 6

[52] Apoorv Vyas, Angelos Katharopoulos, and François Fleuret. Fast transformers with clustered attention. In *NeurIPS*, 2020. 3

[53] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. 1

[54] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv*, 2020. 3

[55] 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. *arXiv*, 2021. 3, 5, 6, 7, 8, 9

[56] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 1, 3

[57] Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. In *ICLR*, 2019. 3

[58] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 2, 3, 8, 9

[59] Weijian Xu, Yifan Xu, Tyler Chang, and Zhuowen Tu. Co-scale conv-attentional image transformers. *arXiv*, 2021. 2

[60] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *arXiv*, 2020. 3

[61] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv*, 2021. 3

[62] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *ECCV*, 2020. 3

[63] Manzil Zaheer, Guru Guruganesh, 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. 3

[64] Pengchuan Zhang, X. Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *arXiv*, 2021. 2, 8, 9

[65] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *arXiv preprint arXiv:2103.15358*, 2021. 7

[66] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv*, 2020. 3

[67] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 3

[68] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 9

[69] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *ICCV*, 2019. 3

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Sec 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Please see Sec 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Provided in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Provided in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Training the large neural networks in this paper is very expensive. Given our limited computational resources, we are unfortunately unable to provide error bars for our experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Provided in the supplementary material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]