Learned Queries for Efficient Local Attention

Moab Arar
Tel-Aviv University

Ariel Shamir
Reichman University

Amit H. Bermano
Tel-Aviv University

Abstract

Vision Transformers (ViT) serve as powerful vision models. Unlike convolutional neural networks, which dominated vision research in previous years, vision transformers enjoy the ability to capture long-range dependencies in the data. Nonetheless, an integral part of any transformer architecture, the self-attention mechanism, suffers from high latency and inefficient memory utilization, making it less suitable for high-resolution input images. To alleviate these shortcomings, hierarchical vision models locally employ self-attention on non-interleaving windows. This relaxation reduces the complexity to be linear in the input size; however, it limits the cross-window interaction, hurting the model performance. In this paper, we propose a new shift-invariant local attention layer, called query and attend (QnA), that aggregates the input locally in an overlapping manner, much like convolutions. The key idea behind QnA is to introduce learned queries, which allow fast and efficient implementation. We verify the effectiveness of our layer by incorporating it into a hierarchical vision transformer model. We show improvements in speed and memory complexity while achieving comparable accuracy with state-of-the-art models. Finally, our layer scales especially well with window size, requiring up to x10 less memory while being up to x5 faster than existing methods. The code is publicly available at https://github.com/moabarar/qna.

1. Introduction

Two key players take the stage when considering data aggregation mechanisms for image processing. Convolutions were the immediate option of choice. They provide *locality*, which is an established prior for image processing, and *efficiency* while doing so. Nevertheless, convolutions capture local patterns, and extending them to global context is difficult if not impractical. Attention-based models [56], on the other hand, offer an adaptive aggregation mechanism, where the aggregation scheme itself is input-dependent, or *spatially dynamic*. These models [4, 12] are the *de-facto* choice in the natural-language processing field and have recently blossomed for vision tasks as well.

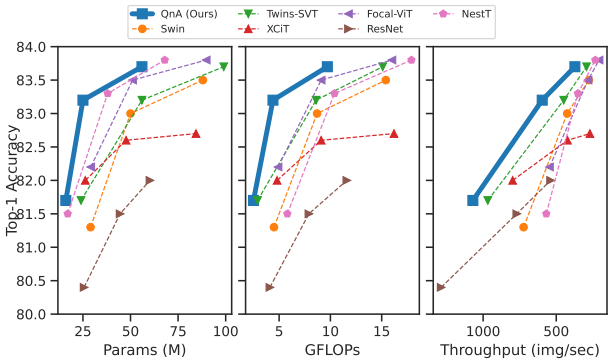


Figure 1. **Performance-Efficiency comparisons on 224^2 input size.** QnA-ViT (our method) demonstrates better accuracy-efficiency trade-off compared to state-of-the-art baselines. As suggested by Dehghani et al. [11], we report the ImageNet-1k [46] Top-1 accuracy (y-axis) trade-off with respect to parameter count (left), floating point operations (middle) and inference throughput (right). The throughput is measured using the timm [59] library, as tested on NVIDIA V100 with 16GB memory. Other metrics, are from the original publications [8, 14, 35, 60, 66, 70]

Earlier variants of the Vision Transformers (ViT) [13] provide global context by processing non-interleaving image patches as word tokens. For these models to be effective, they usually require a vast amount of data [13, 49], heavy regularization [48, 52] or modified optimization objectives [7, 16]. Even more so, it was observed that large scale-training drives the models to attend locally [44], especially for early layers, encouraging the notion that locality is a strong prior.

Local attention mechanisms are the current method of choice for better vision backbones. These backbones follow a pyramid structure similar to convolutional neural networks (CNNs) [8, 15, 58, 70], and process high-resolution inputs by restricting the self-attention to smaller windows, preferably with some overlap [55] or other forms of intercommunication [8, 35, 66]. The latter approaches naturally induce locality while benefiting from spatially dynamic ag-

gregation. On the other hand, these architectures come at the cost of computational overhead and, more importantly, are not shift-equivariant.

In this paper, we revisit the design of local attention and introduce a new aggregation layer called *Query and Attend* (QnA). The key idea is to leverage the locality and shift-invariance of convolutions and the expressive power of attention mechanisms.

In local self-attention, attention scores are computed between all window elements. This is a costly operation of quadratic complexity in the window size. We propose using learned queries to compute the aggregation weights, allowing linear memory complexity, regardless of the chosen window size. Our layer is also flexible, showing that it can serve as an effective up- or down-sampling operation. We Further observe that combining different queries allows capturing richer feature subspaces with minimal computational overhead. We conclude that QnA layers interleaved with vanilla transformer blocks form a family of hierarchical ViTs that achieve comparable or better accuracy compared to SOTA models while benefiting from up-to x2 higher throughput and fewer parameters and floating-point operations (see Figure 1).

Through rigorous experiments, we demonstrate that our novel aggregation layer holds the following benefits:

- QnA imposes locality, granting efficiency without compromising accuracy.
- QnA can serve as a general-purpose layer. For example, strided QnA allows effective down-sampling, and multiple-queries can be used for effective up-sampling, demonstrating improvements over alternative baselines.
- QnA naturally incorporates locality into existing transformer-based frameworks. For example, we demonstrate how replacing self-attention layers with QnA ones in an attention-based object-detection framework [5] is beneficial for precision, and in particular for small-scale objects.

2. Related work

Convolutional Networks: CNNs have dominated the computer vision world. For several years now, the computer vision community has been making substantial improvements by designing powerful architectures [20, 22, 27, 28, 42, 47, 50, 51, 64]. A particularly related CNN-based work is RedNet [33], which introduces an involution operation. This operation extracts convolution kernels for every pixel through linear projection, enabling adaptive convolution operations. Despite its adaptive property, RedNet uses linear projections that lack the expressiveness of the self-attention mechanism.

Vision-Transformers: The adaptation of self-attention showed promising results in various vision tasks including image recognition [2, 37, 71], image generation [38, 69], object-detection [17, 72] and semantic-segmentation [17, 29, 57]. These models, however, did not place pure self-attention as a dominant tool for vision models. In contrast, vision transformers [13,52] brought upon a conceptual shift. Initially designed for image classification, these models use global self-attention on *tokenized* image-patches, where each token attends all others. T2T-ViT [67] further improves the tokenization process via light-weight self-attention at early layers. Similarly, carefully designing a Conv-based STEM-block [63] improves convergence rate and accuracy. CrossViT [6] propose processing at both a coarse- and fine-grained patch levels. TNT-ViT [21], on the other hand, splits coarse patches into locally attending parts. This information is then fused into global attention between patches. ConViT [10] improves performance by carefully initializing the self-attention block to encourage locality. LeViT [19] offers an efficient vision transformer through careful design that combines convolutions and extreme down-sampling. Common to all these models is that, due to memory considerations, expressive feature maps are extracted on very low resolutions, which is not favorable in downstream tasks such as object detection.

Local Self-Attention: Dense prediction tasks involve processing high-resolution images. Due to quadratic memory and computational requirements, global attention is not tractable in this setting. Instead, pyramid architectures employing local attention are used [8, 35, 55, 61, 66, 70]. Typically for such approaches, self-attention is performed within each window, with down-sampling usually applied for global context. Liu et al. [35] propose shifted windows, showing that communication between windows is preferable to independent ones [58]. Halo-Net [55] expands the neighborhood of each window to increase context and inter-window communication. Chu et al. [8] use two-stage self-attention. In the first stage, local attention is employed, while in the second stage, a global self-attention is applied on sub-sampled windows. These models, however, are not shift-invariant, which is a property we maintain. Closest to our work is the Stand-alone self-attention layer (SASA) [37]. As detailed in the text, this layer imposes restrictive memory overhead and is significantly slower, with similar accuracy compared to ours.

Learned Queries: The concept of learned queries has been explored in the literature in other settings [18, 30–32]. In Set Transformers [32], learned queries are used to project the input dimension to a smaller output dimension, either for computation consideration or decoding the output prediction. Similarly, the Perceiver networks family [30, 31]

use small latent arrays to encode information from the input array. Goyal et al. [18] propose a modification for transformer architectures where learned queries (shared workspace) serve as communication-channel between tokens, avoiding quadratic, pair-wise communication. Unlike QnA, the methods above use cross-attention on the whole input sequence. In QnA, the learned queries are shared across overlapping windows. The information is aggregated locally, leveraging the powerful locality priors that have been so well established through the vast usage of convolutions.

3. Method

Query and Attend is a context-aware local feature processing layer. The key design choice of QnA is a convolution-like operation in which aggregation kernels vary according to the context of the processed local region. The heart of QnA is the attention mechanism, where overlapping windows are efficiently processed to maintain shift invariance. Recall that three primary entities are deduced from the input features in self-attention: queries, keys, and values. The query-key dot product, which defines the attention weights, can be computationally pricey. To overcome this limitation, we detour from extracting the queries from the window itself but learn them instead (see Figure 2c). This process is conceptually similar to convolution kernels, as the learned queries determine how to aggregate token values, focusing on feature subspaces pre-defined by the network. We show that learning the queries maintains the expressive power of the self-attention mechanism and facilitates a novel efficient QnA implementation that uses only simple and fast operations. Finally, our layer can be extended to perform other functionalities (e.g., downsampling and upsampling), which are non-trivial in existing methods [37, 55].

Before the detailed explanation of QnA, we will briefly discuss the benefits and limitations of convolutions and self-attention. We let H and W be the height and width of the input feature maps, and denote D as the embedding dimension. Otherwise, throughout this section, we use upper-case notation to denote a matrix or tensor entities, and lower-case notation to denote scalars or vectors.

3.1. Convolution

The convolution layer aggregates information by considering a local neighborhood of each element (e.g., a pixel) of the input feature $X \in \mathbb{R}^{H\times W\times D}$. Specifically, given a kernel $W \in \mathbb{R}^{k\times k\times D\times D}$, the convolution output at location (i, j) is:

$$z_{i,j} = \sum_{\substack{(n,m)\in \\ \mathcal{N}_k(i,j)}} x_{n,m} \cdot W_{k+i-n,k+j-m}, \tag{1}$$

where the $k \times k$-spatial neighborhood of location (i, j) is

$$\mathcal{N}_k(i, j) = \{(n, m)| - k/2 < (i - n), (j - m) \leq k/2\}$$

(see Figure 2a). To simplify the notation, we omit k from Equation (1) and re-write it in matrix notation as:

$$z_{i,j} = X_{\mathcal{N}_{i,j}} \cdot W, \tag{2}$$

For brevity, we assume a stride 1 for all strided operations, and padding is applied to maintain spatial consistency.

The number of convolutional parameters is quadratic in kernel size, inhibiting usage of large kernels, therefore limiting the ability to capture global interactions. In addition, reusing convolutional filters across different locations does not allow adaptive content-based filtering. Nevertheless, the locality and shift-invariance properties of convolutions benefit vision tasks. For this reason, convolutions are widely adopted in computer vision networks, and deep learning frameworks support hardware-accelerated implementation of Equation (1).

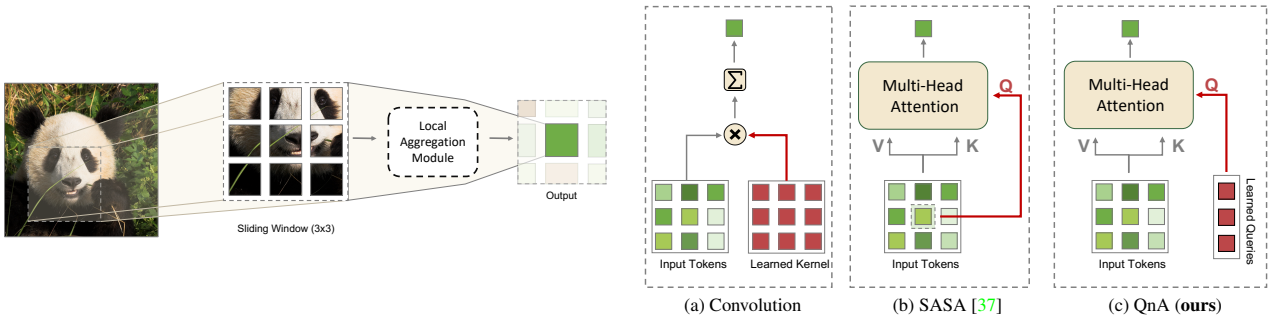
3.2. Self-Attention

A vision transformer network processes a sequence of D-dimensional vectors, $X \in \mathbb{R}^{N\times D}$, by mixing the sequence of size N through the self-attention mechanism. These vectors usually encode some form of image patches where $N = H \times W$ and H, W are the number of patches in each spatial dimension. Specifically, the input vectors are first projected into keys $K = XW_K$, values $V = XW_V$ and queries $Q = XW_Q$ via three linear projection matrices $W_K, W_V, W_Q \in \mathbb{R}^{D\times D}$. Then, the output of the self-attention operation is defined by:

$$\begin{aligned} \textbf{SA}(X) &= \textbf{Attention}\,(Q, K) \cdot V \\ &= \textbf{Softmax}\left(QK^T/\sqrt{D}\right) \cdot V, \end{aligned} \tag{3}$$

where $\textbf{Attention}\,(Q, K)$ is an attention score matrix of size $N \times N$ which is calculated using Softmax that is applied over each row.

Unlike convolutions, self-attention layers have a global receptive field and can process the whole input sequence, without affecting the number of learned parameters. Furthermore, every output of the self-attention layer is an input-dependent linear combination of the V values, whereas in convolutions the aggregation is the same across the spatial dimension. However, the self-attention layer suffers from quadratic run-time complexity and inefficient memory usage, which makes it less favorable for processing high-resolution inputs. Furthermore, it has been shown that vanilla transformers don't attend locally very well [10, 13, 44, 52], which is a desired prior for downstream tasks. These models tend to become more local in nature only after a long and data-hungry training process [44].



(a) Convolution (b) SASA [37] (c) QnA (**ours**)

Figure 2. **QnA overview.** Local layers operate on images by considering non-overlapping windows (left), where the output is computed by aggregating information within each window: (a) Convolutions apply aggregation by learning weighted filters that are applied on each window. (b) Stand-Alone-Self-Attention (SASA) combines the window tokens via self-attention [37] — a time and memory consuming operation. (c) Instead of attending all window elements with each other, we employ learned queries that are shared across windows. This allows linear space complexity, while maintaining the expressive power of the attention mechanism.

3.3. Query-and-Attend

To devise a high-powered layer, we will adapt the self-attention mechanism into a convolution-like aggregation operation. The motivation behind this is that, as it has already been shown [9, 13] the self-attention layer has better capacity than the convolution layer, yet, the inductive bias of convolutions allows better transferability and generalization capability [9]. Specifically, the locality and shift-invariance priors (for early stages) impose powerful guidance in the image domain.

We begin by revisiting the Stand-Alone-Self-Attention approach (SASA) [37], where attention is computed in small overlapping $k \times k$-windows, much like a convolution. The output $z_{i,j}$ of SASA is defined as:

$$z_{i,j} = \textbf{Attention}\left(q_{i,j}, K_{\mathcal{N}_{i,j}}\right) \cdot V_{\mathcal{N}_{i,j}}, \tag{4}$$

where $q_{i,j} = X_{i,j}W_Q$. In other words, in order to aggregate tokens locally, self-attention is applied between the tokens of each local window, and a single query is extracted from the window center (see Figure 2b).

While SASA [37] enjoy expressiveness and locality, through an input-adaptive convolution-like operation, it demands heavy memory usage. Specifically, to the best of our knowledge, all publicly available implementations use an unfolding operation that extracts patches from the input tensor. This operation expands the memory requirement by k^2 if implemented naively. Vaswani et al. [55] improved the memory-requirement of SASA [37] using local attention with halo expansion. Nevertheless, this implementation requires x3-x10 more memory than QnA while being x5-x8 slower, depending on k (see Figure 3). This limitation makes the SASA layer infeasible for processing high-resolution images, employing larger kernels, or using sizable batches.

3.3.1 QnA - single query

To alleviate the compute limitation of SASA [37], we redefine the key-query dot product in Equation (4) by introducing learned queries. As we will later see, this modification leverages the weight-sharing principle (just like convolutions) and enables the efficient implementation of the QnA layer (see Section 3.4).

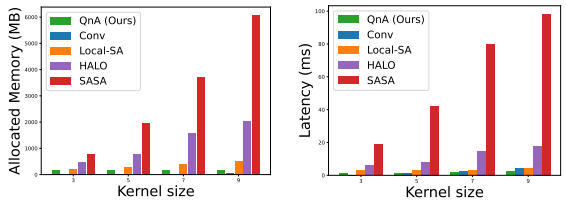
We begin by first replacing the queries $q_{i,j}$ from Equation (4) with a single D-dimensional vector $\tilde{q}$, that is learned during training. More particularly, we define the output of the QnA layer at location (i, j) to be:

$$z_{i,j} = \textbf{Attention}\left(\tilde{q}, K_{\mathcal{N}_{i,j}}\right) \cdot V_{\mathcal{N}_{i,j}}. \tag{5}$$

Through the above modification, we interpret the query-key dot product as the scalar-projection of the keys onto D-dimensional query directions. Therefore, the token values are aggregated according to their relative orientation with the query vectors. Intuitively, the keys can now be extracted such that relevant features' keys will be closely aligned with $\tilde{q}$. This means that the network can optimize the query direction to detect contextually related features.

3.3.2 QnA - multiple queries

As it turns out, performance can be further pushed forward under our paradigm, with minimal computational overhead and negligible additional memory. The naive approach is to add channels or attention heads when considering multi-head attention. While this enhances expressiveness, additional heads induce a larger memory footprint and computational overhead. To improve the layer expressiveness, we can use L-different queries $\tilde{Q} \in \mathbb{R}^{L \times D}$ instead of one. Nevertheless, simply plugging in $\tilde{Q}$ in Equation (5) leads to $L \times D$ output, which expands the memory usage by L (also known as cross-attention). Instead, we weight-sum the attention maps learned by the queries into a single attention



(a) Memory allocation (b) Latency

Figure 3. **Single layer computational complexity during forward pass.** QnA outperforms SASA [37], HaloNet [55], and local self-attention baselines in terms of speed and memory consumption. In particular, during forward pass, HaloNet [55] requires at least x3 additional memory allocation while being x5 slower. For larger kernels, the computation overhead becomes significant where up-to x10 additional memory allocation is needed. Convolutional layers are the most memory efficient, however they are x1.8 slower compared to QnA for large kernels. All experiments tested with PyTorch [39], on input size $256 \times 256 \times 64$.

map (for each attention head) and use it to aggregate the values. Therefore our QnA output becomes:

$$z_{i,j} = \left(\sum_{i \in [L]} \mathcal{W}_i * \textbf{Attention} \left(\tilde{Q}_i, K_{\mathcal{N}_{i,j}} \right) \right) \cdot V_{\mathcal{N}_{i,j}}, \quad (6)$$

where $\mathcal{W} \in \mathbb{R}^{L \times k^2}$ is a learned weight matrix, and $*$ is the element-wise multiplication operation. The overall extra space used in this case is $\mathcal{O}(L \times k^2)$, which is relatively small, as opposed to the naive solution, which requires $\mathcal{O}(L \times D)$ extra space.

3.3.3 QnA variants

Our layer naturally accommodates the improvements made for the vanilla self-attention layer [56]. Specifically, we use relative-positional embedding [1, 25, 26, 43, 62] and multi-head attention in all our models (further details can be found in the supplemental material).

Upsampling & Downsampling Using QnA downsampling can be trivially attained using strided windows. To up-scale tokens by a factor s, we can use a QnA layer with $L = s^2$ learned queries. Assigning the result of each query as an entry in the upsampled output, we effectively construct a spatially dynamic upsampling kernel of size $s \times s$. We define the upsampling operation more formally in the supplemental material. We show that QnA could be used to efficiently perform the upsampling function (Section 4.4) with improved performance, suggesting it can be incorporated into other vision tasks such as image synthesis.

3.4. Implementation & complexity analysis

The shared-learned queries across windows allow us to implement QnA using efficient operations that are available in existing deep-learning frameworks (e.g., Jax [3]). In particular, the query-key dot product can be calculated once for the entire input sequence, avoiding extra space allocation. Then, we can use window-based operations to effectively calculate the softmax operation across different windows, leading to a linear time-and-space complexity (see Figure 3). Full-implementation details of our method are in the supplemental material, along with a code snippet in Jax/Flax [3, 23].

3.5. The QnA-ViT architecture

The QnA-ViT architecture is composed of vision transformer blocks [13] (for global context) and QnA blocks (for local context). The QnA block shares a similar structure with the ViT block, except we replace the multi-head self-attention layer with the QnA layer. We present a family of architectures that follow the design of ResNet [22]. Specifically, we use a 4-stage hierarchical architecture. The base dimension D varies according to the model size. Below we indicate how many layers we use in each stage (T stands for ViT-blocks and Q stands for QnA-blocks):

- Tiny: $D, T, Q = \{64, [0, 0, 4, 2], [3, 4, 3, 0]\}$
- Small: $D, T, Q = \{64, [0, 0, 12, 2], [3, 4, 7, 0]\}$
- Base: $D, T, Q = \{96, [0, 0, 12, 2], [3, 4, 7, 0]\}$

For further details, please refer to the supplemental material.

4. Experiments

4.1. Image Recognition & ImageNet-1K Results

Setting: we evaluate our method using the ImageNet-1K [46] benchmark, and follow the training recipe of DEiT [52], except we omit EMA [41] and repeated augmentations [24]. Additionally, the query vectors are normalized to be unit vectors. For full-training details please refer to the supplemental material.

Results: A summary comparison between different models appears in Table 1. As shown from the table, most transformer-based vision models outperform CNN-based ones in terms of the top-1 accuracy, even when the CNN models are trained using a strong training procedure. For example, ResNet50 [22] with standard ImageNet training achieves 76.6% top-1 accuracy. However, as argued in [60], with better training, its accuracy sky-rockets to 80.4%. Indeed, this is a very impressive improvement, yet it falls short behind transformer models. In particular, our model

Method	Params	GFLOPS	Throughput	Top-1 Acc.
ResNet50 [22, 60]	26M	4.1	**1287**	80.4
ResNet101 [22, 60]	45M	7.9	770	81.5
ResNet152 [22, 60]	60M	11.6	539	82.0
DeiT-S [52]	22M	4.6	940	79.8
DeiT-B [52]	86M	17.5	292	81.8
Swin-Tiny [35]	29M	4.5	723	81.3
Swin-Small [35]	50M	8.7	425	83.0
Swin-Base [35]	88M	15.4	277	83.5
Swin-Base [35]$_{\uparrow 384}$	88M	47.0	85	84.5
NestT-Tiny [70]	17M	5.8	568	81.5
NestT-Small [70]	38M	10.4	352	83.3
NestT-Base [70]	68M	17.9	233	83.8
Focal-Tiny [35]	29M	4.9	546	82.2
Focal-Small [35]	51M	9.1	282	83.5
Focal-Base [35]	90M	16.0	207	83.8
QnA-Tiny	**16M**	2.5	1060	81.7
QnA-Tiny$_{7\times7}$	16M	2.6	895	82.0
QnA-Small	25M	4.4	596	83.2
QnA-Base	56M	9.7	372	83.7
QnA-Base$_{\uparrow 384}$	56M	30.6	177	**84.8**

Table 1. **ImageNet-1K [46] pre-training results.** All models were pre-trained and tested on input size 224×224. Models marked with $\uparrow$ 384 are later also fine-tuned and tested on 384^2 resolution, following [54]. The Accuracy, parameter count, and floating point operations are as reported in the corresponding publication. Throughput was calculated using the timm [59] library, on a single NVIDIA V100 GPU with 16GB memory. For QnA$_{7\times7}$, a 7×7 window size was used instead of 3×3. Our model achieves comparable results to state-of-the-art models, with fewer parameters and better computation complexity.

(the tiny version) improves upon ResNet by 1.3% with 40% fewer parameters and FLOPs.

In terms of speed, CNNs are very fast and have a smaller memory footprint (see Figure 3). The throughput gap can be evident by investigating the vision transformers reported in Table 1. A particular strong ViT is the Focal-ViT [66]; in its tiny version, it improves upon ResNet101 by 0.7% while the latter enjoys x1.4-times better throughput. Nonetheless, our model stands out in terms of the speed-accuracy trade-off. Comparing QnA-Tiny with Focal-Tiny, we achieve only 0.5% less accuracy while having x2-times better throughput, parameter-count, and flops. We can even reduce this gap by training the QnA with a larger receptive field. For example, setting the receptive field of the QnA to be 7x7, instead of 3x3, achieve 82.0% accuracy, with negligible effect on the model speed and size.

Finally, we notice that most Vision Transformers achieve similar Top-1 accuracy. More specifically, tiny models (in terms of parameters and number of FLOPs) achieve roughly the same Top-1 accuracy of 81.2-82.0%. The accuracy difference is even less significant in larger models (e.g., base variants accuracy differs by only 0.1%), and this accuracy difference can be easily tipped to either side by many factors, even by choosing a different seed [40]. Nonetheless, our model is faster, all while using fewer resources.

	SASA	QnA $L=1$	QnA $L=2$	QnA $L=3$	QnA $L=4$
Top-1 Acc.	**80.86**	80.3	80.7	80.76	80.81
Params (M).	16.440	**16.182**	16.188	16.192	16.200.
FLOPS (G)	2.620	**2.378**	2.400	2.420	2.442

Table 2. **Multiple queries effect.** We compare the performance of SASA [37] to QnA with a varying amount of queries. As can be seen, using multiple queries improves QnA, reaching comparable performance, using an order of magnitude less memory.

The reason behind better accuracy-efficiency trade-off: QnA-ViT achieves a better accuracy-efficiency trade-off for several reasons. First, QnA is fast, which is crucial for better throughput. Further, most of the vision transformer's parameter count is due to the linear projection matrices. Our method reduces the number of linear projections by omitting the query projections (i.e., the W_q matrix is replaced with 2-learned queries). Furthermore, the feed-forward network requires $\times 2$ more parameters than the self-attention. Our model uses smaller embedding dimensions than existing models without sacrificing accuracy. Namely, NesT-Tiny [70] uses an embedding dimension of 192, while Swin-Tiny [35] and Focal-Tiny [66] use 96 embedding dimensions. On the other hand, our method achieves a similar feature representation capacity, with a lower dimension of 64.

Finally, other parameter efficient methods achieve low parameter count by training on larger input images [51, 55]. This is shown to improve image-classification accuracy [54]. However, it comes at the cost of lower-throughput and more FLOPs. For example, EfficientNet-B5 [51], which was trained and tested on images of 456×456 resolution, achieves 83.6% accuracy while using only 30M parameters. Nonetheless, the network's throughput is 170 images/sec, and it uses 9.9 GFLOPs. Compared to our base model, QnA achieves similar accuracy with twice the throughput. Also, it is important to note that these models were optimized via Neural Architecture search, an automated method for better architecture design. We believe employing methods with similar purpose [65] would even further optimize our models' parameter count.

4.2. Ablation & design choices

Number of queries: Using multiple queries allows us to capture different feature subspaces. We consider SASA [37] as our baseline, which extracts the self-attention queries from the window elements. Due to its heavy memory footprint, we cannot consider SASA variants similar to QnA-ViT. Instead, we consider a lightweight variant that

Global Attention	QnA	Downsampling	Params	FLOPs	Top1-Acc.
Different downsampling choices					
[3,3,6,2]	[0,0,0,0]	Nest [70]	16.8M	3.7	81.2
[3,3,6,2]	[0,0,0,0]	Swin [35]	**16.0M**	**3.1**	81.2
[3,3,6,2]	[1,1,1,0]	QnA	**16.0M**	3.2	**81.9**
Number of QnA blocks vs Transformer blocks					
[0,0,0,0]	[4,4,7,2]	QnA	14.9M	2.4	80.9
[3,3,6,2]	[1,1,1,0]	QnA	16.0M	3.2	**81.9**
[0,0,4,2]	[4,4,3,0]	QnA	**15.8M**	**2.6**	**81.9**
Deeper Models					
[0,0,8,2]	[3,4,11,0]	QnA	**24.7M**	**4.2**	82.7
[0,0,10,2]	[3,4,9,0]	QnA	24.8M	4.3	83.0
[0,0,12,2]	[3,4,7,0]	QnA	25.0M	4.4	**83.2**
[0,0,16,2]	[3,4,3,0]	QnA	25.3M	4.6	83.1

Table 3. **Ablation studies and design choices.** In the first two columns we specify the number of global-attention and QnA layers used in each stage. See section 4.2 for further details, and the supp. materials for more configurations.

combines local self-attention with SASA. All SASA layers use a 3x3 window size. Downsampling is performed similar to QnA-ViT, except that we replace QnA with SASA. Finally, the local-self attention layers use a 7x7 window size without overlapping (see supplementary). The results are summarized in Table 2. As can be seen, we achieved comparable results to SASA. In addition, two queries outperform one, but this improvement saturates quickly. We hence recommend using two queries, as it enjoys efficiency and expressiveness.

Number of heads: Most vision transformers use large head dimension (e.g., $\geq$ 32) [53]. However, we found that the QnA layer enjoys more heads. We trained various models based on QnA and self-attention layers with different training setups to verify this. Our experiments found that a head dimension $d = 8$ works best for QnA layers. Similar to previous work [53], in hybrid models, where both self-attention and QnA layers are used, we found that self-attention layers still require a large head dimension (i.e., $d = 32$). Moreover, we found that using more heads for QnA is considerably better (up to 1% improvement) for small networks. Moreover, this performance gap is more apparent when training the models for fewer epochs without strong augmentations (see supplemental material for further details). Intuitively, since the QnA layer is local, it benefits more from local pattern identifications, unlike global context, which requires expressive representation.

How many QnA layers do you need? In order to verify the expressive power of QnA, we consider a dozen different models. Each model consists of four stages. In each stage, we consider using self-attention and QnA layers. A summary report can be found in Table 3 (for the full report, please see supplementary material). In our experiments, we conclude that the QnA layer is effective in the early stages and can replace global attention without affecting the model's performance. QnA is fast and improves the model's efficiency. Finally, the QnA layer is a very effective down-sampling layer. For example, we considered two baseline architectures which are mostly composed of transformer blocks, (1) one model uses simple 2x2 strided-convolution to reduce the feature maps (adopted in [35]), and the (2) other is based on the down-sampling used in NesT [70], which is a 3x3 convolution, followed by a layer-norm and max-pooling layer. These two models achieve similar accuracy, which is 81.2%. On the other hand, when merely replacing the downsampling layers with the QnA layer, we witness a 0.7% improvement without increasing the parameter count and FLOPs. Note, global self-attention is still needed to achieve good performance. However, it can be diminished by local operations, e.g., QnA.

Model	Backbone	AP_{50}	AP_{75}	AP_L	AP_M	AP_S	AP
DETR	R50	55.4	36.6	53.2	38.0	15.1	35.3
	QnA-Ti	58.9	38.6	56.8	40.6	16.0	37.5
	QnA-Ti7	**59.6**	39.3	**57.6**	41.2	16.0	37.9
DETR-QnA	QnA-Ti	**59.6**	**39.7**	57.4	**41.8**	**18.2**	**38.5**

Table 4. **DETR [5] based Object detection on the COCO dataset [34].** Incorporating QnA-ViT-Tiny with DETR substantially improves upon the ResNet50 backbone (by up to 3.2). QnA with receptive field 7x7 improves the average precision on large objects (AP_L), and incorporating QnA into the DETR network improves performance on smaller objects, indicating locality.

Deep models: To scale-up our model, we chose to increase the number of layers in the network's third stage (as typical in previous works [22]). This design choice is adapted mainly for efficiency reasons, where the spatial and feature dimension are manageable in the third stage. In particular, we increase the total number of layers in the third stage from 7 to 19 and consider four configurations where each configuration varies by the number of QnA layers used. The models' accuracies are reported in Table 3. As seen from the table, the model's accuracy can be maintained by reducing the number of global attention. This indicates that while self-attention can capture global information, it is beneficial to a certain degree, and local attention could be imposed by the architecture design for efficiency consideration.

4.3. Object Detection

Setting: To evaluate the representation quality of our pre-trained networks, we use the DETR [5] framework, which is a transformer-based end-to-end object detection framework. We use three backbones for our evaluations; ResNet50 [22], and two variants of QnA-ViT, namely, QnA-ViT-Tiny, and

(a) Bilinear (b) ConvTransposed (c) QnA

Figure 4. **Qualitative Auto-Encoder results.** We train a simple Autoencoder using convolution layers (a-b), and (c) QnA layers. We show reconstructed images from the CelebA test set [36]. QnA shows preferable reconstructions. See Section 4.4 for more details.

Figure 5. **Quantitative Auto-Encoder results** are reported, compared to the same convolutional baselines as in Figure 4. We compare our method (gray) to bilinear upsampling (green) and transposed covolution-based upsampling (pink). We show consistent improvement across epochs (horizontal axes) in the L1 loss (top left), the pSNR (top right), the SSIM (bottom left), and MSE (bottom right) metrics.

QnA-ViT-Tiny-7x7, which uses a 7x7 receptive for all QnA layers (instead of 3x3). Complete training details are provided in the supplemental material.

Revisiting DETR transformer design: DETR achieves comparable results to CNN-based frameworks [45]. However, it achieves less favorable average precision when tested on smaller objects. The DETR model uses a vanilla transformer encoder to process the input features extracted from the backbone network. As argued earlier, global attention suffers from locality issues. To showcase the potential of incorporating QnA in existing transformer-based networks, we propose DETR-QnA architecture, in which two transformer blocks are replaced with four QnA blocks.

Results: We report the results in Table 4. As can be seen, DETR trained with QnA-Tiny achieves +2.2 better AP compared to the ResNet50 backbone. Using a larger receptive field (7×7) further improves the AP by 0.4. However, much improvement is due to better performance on large objects (+0.7). Finally, when incorporating QnA into the DETR encoder, we gain an additional +0.6AP (and +1.0AP relative to using the DETR model). More particularly, incorporating QnA with DETR achieves an impressive +2.2 AP improvement on small objects, indicating the benefits of QnA's locality.

4.4. QnA as an upsampling layer

We suggest that QnA can be adapted to other tasks besides classification and detection. To demonstrate this, we train an autoencoder network on the CelebA [36] dataset, using the L_1 reconstruction loss. We consider two simple baselines that are convolution-based. In particular, one baseline uses bilinear up-sampling to upscale the feature maps, and another baseline uses the transposed convolution layer [68]. Qualitative and quantitative results appear in Figure 4 and Figure 5, respectively. The figures show that the QnA-based auto-encoder achieves better qualitative and quantitative results and introduces fewer artifacts (further details are available in the supplemental material).

5. Limitations & conclusion

We have presented QnA, a novel local-attention layer with linear complexity that is also shift-invariant. As demonstrated in the experiments, the introduced layer could serve as a general-purpose layer. We showed how to improve the efficiency of vision transformers without compromising on the accuracy part. Furthermore, we evaluated our method in the object-detection setting and improved upon the existing self-attention-based method. Our layer could also be used as an up-sampling layer, which we believe is essential for incorporating transformers in other tasks, such as image generation. Finally, we would like the reader to note that our layer is attention-based. Hence it requires additional intermediate memory, whereas convolutions operate seamlessly, requiring no additional allocation. Nonetheless, QnA has more expressive power than convolution. In addition, global self-attention blocks are more powerful in capturing global context. Therefore, we believe that our layer mitigates the gap between self-attention and convolutions and that future works should incorporate all three layers to achieve the best performance networks.

Acknowledgment

Research supported with Cloud TPUs from Google's TPU Research Cloud (TRC).

References

[1] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and Hsiao-Wuen Hon. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 642–652. PMLR, 2020. 5

[2] Irwan Bello, Barret Zoph, Quoc Le, Ashish Vaswani, and Jonathon Shlens. Attention augmented convolutional networks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 3285–3294. IEEE, 2019. 2

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 5

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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 Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 1

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2020. 2, 7

[6] Chun-Fu (Richard) Chen, Quanfu Fan, and Rameswar Panda. CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[7] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pretraining or strong data augmentations. *CoRR*, abs/2106.01548, 2021. 1

[8] 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. In *NeurIPS 2021*, 2021. 1, 2

[9] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 4

[10] Stéphane d'Ascoli, Hugo Touvron, Matthew L. Leavitt, Ari S. Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2286–2296. PMLR, 2021. 2, 3

[11] Mostafa Dehghani, Anurag Arnab, Lucas Beyer, Ashish Vaswani, and Yi Tay. The efficiency misnomer. *CoRR*, abs/2110.12894, 2021. 1

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. 1

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1, 2, 3, 4, 5

[14] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Xcit: Cross-covariance image transformers. *CoRR*, abs/2106.09681, 2021. 1

[15] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *CoRR*, abs/2104.11227, 2021. 1

[16] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1

[17] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 3146–3154. Computer Vision Foundation / IEEE, 2019. 2

[18] Anirudh Goyal, Aniket Rajiv Didolkar, Alex Lamb, Kartikeya Badola, Nan Rosemary Ke, Nasim Rahaman, Jonathan Binas, Charles Blundell, Michael Curtis Mozer, and Yoshua Bengio. Coordination among neural modules through a shared global workspace. In *International Conference on Learning Representations*, 2022. 2, 3

[19] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Herve Jegou, and Matthijs

Douze. Levit: A vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12259–12269, October 2021. 2

[20] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6307–6315. IEEE Computer Society, 2017. 2

[21] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. In *NeurIPS*, 2021. 2

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. 2, 5, 6, 7

[23] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020. 5

[24] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8129–8138, 2020. 5

[25] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3588–3597. Computer Vision Foundation / IEEE Computer Society, 2018. 5

[26] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 3463–3472. IEEE, 2019. 5

[27] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7132–7141. Computer Vision Foundation / IEEE Computer Society, 2018. 2

[28] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017. 2

[29] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 603–612. IEEE, 2019. 2

[30] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs. *CoRR*, abs/2107.14795, 2021. 2

[31] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and João Carreira. Perceiver: General perception with iterative attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 2021. 2

[32] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753. PMLR, 2019. 2

[33] Duo Li, Jie Hu, Changhu Wang, Xiangtai Li, Qi She, Lei Zhu, Tong Zhang, and Qifeng Chen. Involution: Inverting the inherence of convolution for visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 12321–12330. Computer Vision Foundation / IEEE, 2021. 2

[34] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 7

[35] 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. *International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 6, 7

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738. IEEE Computer Society, 2015. 8

[37] Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 68–80, 2019. 2, 3, 4, 5, 6

[38] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4052–4061. PMLR, 2018. 2

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming

Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[40] David Picard. Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2021. 6

[41] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992. 5

[42] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10425–10433. Computer Vision Foundation / IEEE, 2020. 2

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. 5

[44] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *CoRR*, abs/2108.08810, 2021. 1, 3

[45] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015. 8

[46] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 1, 5, 6

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 2

[48] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *CoRR*, abs/2106.10270, 2021. 1

[49] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 843–852. IEEE Computer Society, 2017. 1

[50] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9. IEEE Computer Society, 2015. 2

[51] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 2, 6

[52] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 2021. 1, 2, 3, 5, 6

[53] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *CoRR*, abs/2103.17239, 2021. 7

[54] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8250–8260, 2019. 6

[55] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake A. Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 12894–12904. Computer Vision Foundation / IEEE, 2021. 1, 2, 3, 4, 5, 6

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. 1, 5

[57] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan L. Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28,*

2020, Proceedings, Part IV, volume 12349 of *Lecture Notes in Computer Science*, pages 108–126. Springer, 2020. 2

[58] 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. *CoRR*, abs/2102.12122, 2021. 1, 2

[59] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 1, 6

[60] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *CoRR*, abs/2110.00476, 2021. 1, 5, 6

[61] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22–31, October 2021. 2

[62] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. *CoRR*, abs/2107.14222, 2021. 5

[63] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross B. Girshick. Early convolutions help transformers see better. *CoRR*, abs/2106.14881, 2021. 2

[64] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995. IEEE Computer Society, 2017. 2

[65] Huanrui Yang, Hongxu Yin, Pavlo Molchanov, Hai Li, and Jan Kautz. Nvit: Vision transformer compression and parameter redistribution. *CoRR*, abs/2110.04869, 2021. 6

[66] 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, 2021. 1, 2, 6

[67] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis E.H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 558–567, October 2021. 2

[68] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Robert Fergus. Deconvolutional networks. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 2528–2535. IEEE Computer Society, 2010. 8

[69] Weichao Zhang, Guanjun Wang, Mengxing Huang, Hongyu Wang, and Shaoping Wen. Generative adversarial networks for abnormal event detection in videos based on self-attention mechanism. *IEEE Access*, 9:124847–124860, 2021. 2

[70] Zizhao Zhang, Han Zhang, Long Zhao, Ting Chen, and Tomas Pfister. Aggregating nested transformers. In *arXiv preprint arXiv:2105.12723*, 2021. 1, 2, 6, 7

[71] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10073–10082. Computer Vision Foundation / IEEE, 2020. 2

[72] Xizhou Zhu, Dazhi Cheng, Zheng Zhang, Stephen Lin, and Jifeng Dai. An empirical study of spatial attention mechanisms in deep networks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6687–6696. IEEE, 2019. 2