LOCAL-GLOBAL SHIFTING VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work has shown the potential of transformers for computer vision applications. An image is first partitioned into patches, which are then used as input tokens for the attention mechanism. Due to the expensive quadratic cost of the attention mechanism, either a large patch size is used, resulting in coarse-grained global interactions, or alternatively, attention is applied only on a local region of the image at the expense of long-ranged interactions. In this work, we propose an approach that allows for both coarse global interactions and fine-grained local interactions simultaneously. At the core of our method is the application of local and global attention layers. In the local attention layer, we apply attention to each patch and its local shifts, resulting in virtually located local patches, which are not bound to a single, specific location. These virtually located patches are then used in global attention layers, where global coarse interactions are learned, using a pyramid of attention layers applied on decreasing resolution inputs. The separation of the attention layer into local and global counterparts allows for a low computational cost in the number of patches, while still supporting data-dependent localization, as opposed to the static positioning in other visual transformers. Our method is shown to be superior to both convolutional and transformer-based methods for image classification on CIFAR10, CIFAR100, and ImageNet.

1 INTRODUCTION

Convolutional neural networks have dominated computer vision research and enabled significant breakthroughs in solving many visual tasks, such as image classification (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014) and semantic segmentation (Long et al., 2015). Typically, CNN architectures begin by applying convolutional layers of a small receptive field for low-level features, resulting in local dependencies between neighbouring image regions. As processing continues and the features become more semantic, the effective receptive field is gradually increased, capturing longer-ranged dependencies.

Inspired by the success of Transformers (Vaswani et al., 2017) for NLP tasks, a new set of attention-based approaches have emerged for vision-based processing. The Vision Transformer (ViT) (Dosovitskiy et al., 2020) is the first model to rely exclusively on the Transformer architecture to obtain competitive image classification performance. ViT divides the input image into patches of a fixed size and considers each patch as a token to which the transformer model is applied. The attention mechanism between these patches results in global dependencies between pixels, already at the first transformer layer. Due to the quadratic cost of the attention mechanism (Vaswani et al., 2017; Dosovitskiy et al., 2020) in the number of patches, a fixed-size partitioning is performed. As a result, ViT does not benefit from the built-in locality bias that is present in CNNs: neighbouring pixels within a patch may be highly correlated, but this bias is not encoded into the ViT architecture. That is, ViT encodes inter-patch correlations well, but not intra-patch correlations. Second, each image may require a different patch size and location, depending on the size and location of objects in the image.

Recent approaches (Liu et al., 2021; Zhang et al., 2021; Wang et al., 2021; Wu et al., 2021) attempt to alleviate the need for this fixed-size partition, thus enjoying some of the benefits of CNNs. PvT (Wang et al., 2021), for instance, applies attention in a pyramid-like fashion, with increasing patch size at each level. However, an initial fixed partition of the image into non-overlapping patches is still performed, so finer sub-patch correlations are still not captured. In another line

Method	Patch Size	Embedding	# Tokens	# Levels	Complexity	Region
ViT	16×16	Linear	$B = 14^2$	1	$O(B^2)$	Global
DeiT	16×16	Linear	$B = 14^2$	1	$O(B^2)$	Global
PvT	4×4	Linear	$B = 56^2$	4	$O(B^2)$	Global
CvT*	7×7	Convolution	$B = 56^2$	3	$O(B^2)$	Global
NesT†	4×4	Convolution	$B = 4^2$	3	$O(B^2)$	Local
Swin†	4×4	Convolution	$B = 7^2$	4	$O(B^2)$	Local
Ours	4×4	Convolution	$B = 56^2$, $T = 10$	4	$O(B \cdot T^2 + B^2)$	Global

Table 1: A comparison of our method to baselines on key elements. We assume an input resolution of 224×224, as used for ImageNet (see Tab. 2 for more details). We consider the *Patch Size* used for the division of the image, type of patch *Embedding*, number of tokens (*#Tokens*) resulting from the patch embedding (for the first level of the hierarchy), and number of levels (*#Levels*) used in the hierarchy. We consider the *Complexity* of the attention step in the number of tokens as well as the *Region* on which attention is applied - either over a local region or over the entire image (global). B stands for the number of tokens. In our method, we also consider T variants for each patch, which results in a total of $B \times T$ tokens. CvT* considers overlapping patches (a stride of 4 is used) and so the resulting number of tokens is large with respect to patch size. In NesT† and Swin†, attention is applied only to a local number of patches, so the number of tokens is small with respect to patch size.

of work NesT (Zhang et al., 2021) and Swin (Wang et al., 2021), apply attention in a localized fashion, over local regions in the image. This results in the inability to capture global correlations between distant patches in the image. CvT (Wu et al., 2021) considers overlapping patches, thus capturing both inter-patch correlations and intra-patch correlations. As a result, a large number of patches is considered, and CvT does not scale well to large images due to the quadratic cost of the attention mechanism in the number patches.

Our approach combines the locality bias of CNNs, for both coarser and finer details, with the ability to attend globally to all patches in the image. The method scales well to large images, since it does not incur the prohibitive quadratic cost of considering all overlapping patches. It is based on the observation that the optimal location for each patch varies from image to image depending on object locations and sizes. Therefore, instead of considering a single patch at a given location, we consider an ensemble of patches. This ensemble consists of the conventional fixed patch location and of the patches obtained by small horizontal and vertical shifts of each patch. By employing this shift property, the ensemble can capture more precisely the finer details of the object patches, which are necessary for the downstream task.

To avoid the expensive quadratic cost of computing self-attention over all ensembles of patches, we split each attention layer to two consecutive attention operations, which accumulate both local and global information for each patch. In the local attention layer, we apply self-attention to each patch with its local shifts. This step allows the fixed patches to gain information from a rich collection of variations, where each variant represents an alternative to the location of the patch. This way, we construct a virtually located patch as a weighted sum of all possible shifts of the fixed patch. In the global attention layers, we utilize the virtually local patches and apply the standard global self-attention between them. This step allows each patch to gain global information from all the other patches, where each patch was optimized in the previous local layer by considering all of its local shifts. Global attention is applied in a number of layers, in a pyramid-like fashion, where at each layer a coarser resolution is considered. Our method obtains state-of-the-art performance in image classification on a variety of different model sizes for both CIFAR10, CIFAR100 (Krizhevsky, 2009) and ImageNet (Deng et al., 2009). At the same time, our method is efficient and can scale well to large image resolutions.

2 Related Work

The Transformer, first introduced in (Vaswani et al., 2017), revolutionized the field of NLP due to its ability to capture both local and global connections in an input sequence and its capac-

ity, compared to previous state-of-the-art models based on RNNs or CNNs. Inspired by the Transformer's success in NLP, multiple attempts have been made to incorporate such emerging attention-based techniques for image classification. The Vision Transformer (ViT) (Dosovitskiy et al., 2020) marked a turning point for Transformers in vision. ViT showed excellent results compared to existing CNN-based networks, while requiring fewer computational resources.

Later attempts incorporated the locality bias of CNNs within a transformer architecture. DeiT (Touvron et al., 2020) introduced a teacher-student strategy specifically for Transformers, using an additional distillation token, in which the teacher is a CNN. This enabled training vision Transfromers with the standard ImageNet dataset, removing the need of ViT to utilize a larger-scale pre-training dataset. PvT (Wang et al., 2021) fused the locality bias of CNNs with the ability to attend to all the patches of the image globally, at different scales. A pyramid of attention blocks is applied on coarser feature maps, thus enabling patches to capture a different set of pixels and for local dependencies between neighbouring pixels to emerge. However, an initial fixed partition of the image into non-overlapping patches limits the ability of the attention mechanism to capture finer sub-patch details.

Approaches such as the Swin Transformer (Liu et al., 2021) and NesT (Zhang et al., 2021) attempt to alleviate this issue by focusing on localized self-attention in non-overlapping blocks, and aggregating the blocks. This partition considers very small, even pixel-sized patches, so correlation between surrounding pixels can be considered, thus reinforcing the locality bias of CNNs. However, this comes at the expense of considering only a part of the image at a given scale, while ignoring global dependencies between all patches in the image.

CvT is a recent approach by (Wu et al., 2021) that applies the attention mechanism over overlapping patches of the image at different scales, thus capturing finer details along with global dependencies between distant patches. However, the number of overlapping patches in large images is significantly larger than the number of non-overlapping patches, so applying attention, whose cost increases quadratically with the number of patches, is prohibitively expensive.

In contrast, our method separates the attention mechanism into a local stage and a global stage. At the local stage, finer details can be aggregated over the local region of each patch, producing new patches, which incorporate fine details. At the global stage, a standard attention computation is applied between all newly aggregated patches. This aggregation is done in a pyramid-like fashion over a number of scales, resulting in an efficient transformer architecture, which can scale well to large images, and which incorporates the locality bias at both fine and coarse scales.

Tab. 1 shows the key differences of our method to previous work. Crucially, for B tokens resulting from the patch embedding, we can consider a large number of patch variants T (at most $\sqrt{B}$), while the overall complexity remains $O(B^2)$. This results in a much larger number of tokens being used in our attention mechanism, while also considering the entire image region globally.

3 METHOD

We first describe the formation of the patch embeddings. Then, we discuss the two types of attention layers: the local attention layer operates locally on each patch and its shifts, while the global attention is applied globally in an coarse-to-fine hierarchical manner. Finally we describe the implementation details.

3.1 LOCAL SHIFT EMBEDDINGS

Given an image $I \in \mathbb{R}^{3\times H\times W}$, we consider a partition of the image to patches of size $S\times S$, resulting in a map of $\frac{H}{S}\times\frac{W}{S}$ patches, similarly to ViT (Dosovitskiy et al., 2020). For each patch i of this map, we consider T shifted patches. A patch is created by shifting patch i, P_x pixels horizontally and P_y pixels vertically where $|P_x| \le S$ and $|P_y| \le S$. Each such shift is identified by the pair (P_x, P_y).

This process results in $T\times\frac{H}{S}\times\frac{W}{S}$ patches of size $S\times S$. The subset of chosen shift variants among all possible shifts is a hyper-parameter described in Sec. 4; In Tab. 5, we report our results with subsets of varying size.

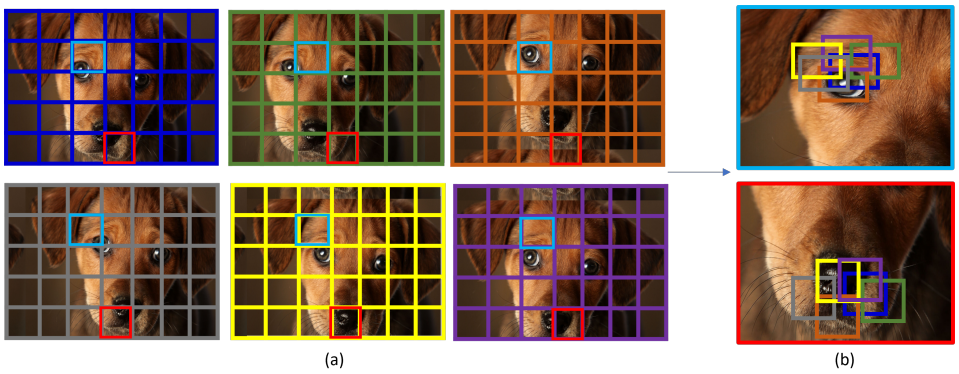

Figure 1: (a) Illustration of six image shifting variants out of the total T, as defined in Sec. 3.1. A grid is shown over each image shift variant. In each variant, the grid represents a different division of the image to patches. Patches shown in light blue and red indicate two different patch-shifting variants. The color of the grid is indicative of the variant index. (b) The same six variants are shown for the light blue (top) and red (bottom) patches in (a). The figure illustrates the sub-optimality of the fixed partition used in previous work. For example, the dog's left eye is fully captured only in the "orange partition" and the nose only in the "yellow partition". A fixed partition would only capture one of those concepts.

For each of the T variants of patch i, we construct a D dimensional embedding. These embeddings are obtained by shifting the entire image I by P_x and P_y, and applying circular padding at the edges. Each resulting image, referred to as image shifting variant, is then passed through a convolutional layer with an $S \times S$ kernel, stride S and D output channels. This results in T feature maps of size $D \times \frac{H}{S} \times \frac{W}{S}$. One can view each feature map as $\frac{H}{S} \cdot \frac{W}{S}$ tokens of size D. We define $O_i'^s$ to be the token corresponding to patch i and that shift index s, which corresponds to shift (P_x, P_y).

Positional embedding A learned positional embedding is added for each patch and variant convolutional embedding $O_i'^s$. That is, $O_i^s = O_i'^s + L_i^s$, where L_i^s is a learned positional embedding of size D. In Sec. 4.1, we consider a different strategy whereby a single D dimensional positional embedding L_i is learned for all variants of a given patch, which results in worse performance.

$O_i^s \in \mathbb{R}^D$, for $s = 1..T$, is then the set of embeddings for the shift-variants associated with each patch i of size $S \times S$. We assign indexes such that the first variant, O_i^1, is the embedding of the non-shifted (identity) patch, where $P_x = P_y = 0$.

3.2 Local-Global Attention

We first apply the local attention layer. This attention layer considers each patch i independently. For each i, we aggregate the embeddings of the T local shifted variants and using an attention mechanism. This results in an updated embedding of size D for each patch. Subsequently global attention layers are applied using a pyramid structure.

Local Attention An illustration of our local attention layer is shown in Fig. 2. Denote by $B = \frac{H}{S} \cdot \frac{W}{S} = \frac{HW}{S^2}$, the number of patches for each variant. The first step is to calculate, for each non-shifted patch embedding O_1^i, a query value:

$$q = XU_q \qquad X \in \mathbb{R}^{B \times D},\ U_q \in \mathbb{R}^{D \times D} \tag{1}$$

where X is a matrix constructed from all non-shifted patch embeddings O_1^i for $i = 1..B$ and U_q is a learned query matrix. Next, we calculate keys and values for each patch embedding and variant:

$$[kv] = ZU_{kv} \qquad Z \in \mathbb{R}^{(TB) \times D},\ U_{kv} \in \mathbb{R}^{D \times 2D} \tag{2}$$

That is, Z is a matrix constructed from all patch embeddings O_s^i over all variants. U_{kv} is a learned key and value matrix. We note that, for each patch, while the keys and values are computed from all T variants, the queries are obtained only from the non-shifted variant.

We now wish to construct an attention matrix W. We consider each patch separately. That is, given $q \in \mathbb{R}^{B\times D}$ we extract the patch-specific query vector $q_i \in \mathbb{R}^D$. Similarly, we can view k and v as tensors in $\mathbb{R}^{B\times T\times D}$. For each patch, we consider the patch-specific key and value matrices $k_i, v_i \in \mathbb{R}^{T\times D}$. We now apply multi-head attention *separately* for each patch:

$$W_i = softmax(q_i k_i^T / \sqrt{D_h}) \qquad q_i \in \mathbb{R}^D,\ k_i \in \mathbb{R}^{T\times D}, \tag{3}$$

where D_h is the dimension of each attention head. For each patch, this results in a pseudo-probability vector $W_i \in \mathbb{R}^T$ indicating the weight of each patch variant. The pooled value for patch i is given by:

$$A_i = W_i v_i \qquad\qquad W_i \in \mathbb{R}^T,\ v_i \in \mathbb{R}^{T\times D} \tag{4}$$

$A \in \mathbb{R}^{B\times D}$ is then constructed by pooling all aggregated patch embeddings A_i.

Following the local attention layer, we apply a feed-forward fully-connected network (FFN) with skip-connection (He et al., 2016) and LayerNorm (LN) (Ba et al., 2016). The application of all of these components together is referred to as a local attention block.

Global Attention Following the local attention block, we are given B tokens (for each patch) with embedding of size D. As in the standard setting of transformers (Touvron et al., 2021), a multi-head self-attention (Vaswani et al., 2017) is applied to the tokens. The global attention block consists of a multihead-attention layer followed by a feed-forward fully-connected network (FFN) with skip-connection (He et al., 2016) and LayerNorm (LN) (Ba et al., 2016).

Pyramid structure Recall that $B = \frac{H}{S} \cdot \frac{W}{S}$ and so one can view the input as having a height of $B_h = \frac{H}{S}$, width of $B_w = \frac{W}{S}$ and D channels. We would like to consider patches at coarser scales, so we apply a sequence (pyramid) of global attention blocks, but with the output of each block being downsampled before the application of the next attention block.

The *downsampling* operation consists of the application of a convolutional layer with a 3×3 kernel, stride 1 and padding of 1 followed by a max-pooling operation with a 3×3 kernel, stride 2 and padding of 1. Assuming the output channel of the convolutional layer is D^{down}, downsampling results in $\frac{B}{4}$ tokens of dimension D^{down} on which the next global attention block is applied.

We continue this way for K times (chosen as a hyperparameter), obtaining an output with lower and lower resolution. Lastly, we apply global average pooling over the spatial dimension, resulting in a final vector of dimension D^{final}. This is followed by a linear layer that outputs C (number of classes) logits, on which standard softmax classification is applied.

3.3 Implementation Details

In Tab. 2 we provide the exact architectures for two different image resolutions - 224×224 (used for ImageNet (Deng et al., 2009)) and 32×32 (used for CIFAR10 and CIFAR100 (Krizhevsky, 2009) datasets). For 32×32 resolution (CIFAR10/CIFAR100) images we consider three model variants provided - tiny, small and base, with increasing number of parameters. For 224×224 (ImageNet) resolution, tiny and small variants are considered [1].

We use 300 epochs for all experiments and use the same set of data augmentation and regularization strategies used by (Touvron et al., 2021) but exclude repeated augmentations (Hoffer et al., 2020) and exponential moving average (Polyak & Juditsky, 1992). The initial learning rate is set to $5\cdot 10^{-4}$. We apply a linear warm-up of 20 epochs for ImageNet and 5 epochs for CIFAR10/CIFAR100. We scale the learning rate (lr) according to the batch size (bs) as: $\frac{lr}{512}\times bs$. We use the AdamW (Kingma & Ba, 2014) optimizer with a cosine learning rate scheduler. The weight decay is set to 0.05 and the maximal gradient norm is clipped to 5.0.

[1] Note that while some previous works report results also for larger models, we were unable to allocate the resources needed for such experiments. Specifically, running our larger model on ImageNet would require

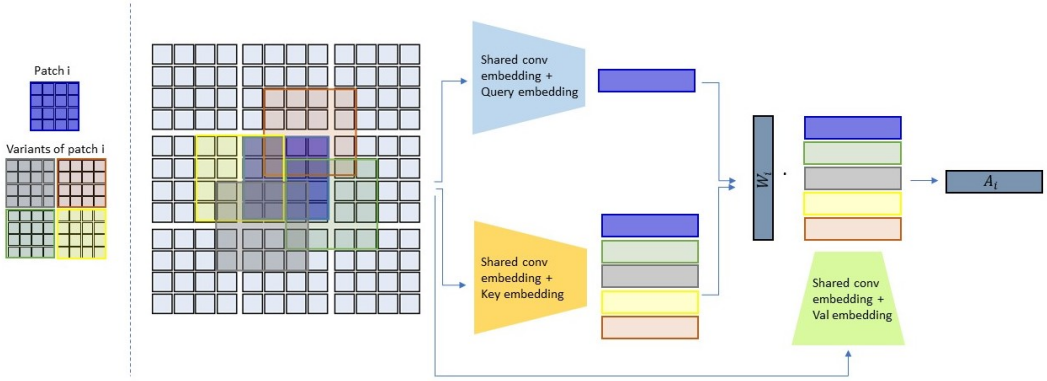

Figure 2: Illustration of our local attention layer. Patch i is represented in blue; Variants of i are shown in different colors. All patches are first encoded using a convolutional layer, which, together with the positional encoding, results in a D dimensional embedding O_i^s for each patch i and variant s. Given O_i^s, a query embedding is generated as in Eq. 1. For each patch variant, a key embedding is generated as in Eq. 2. Both the query and key embeddings are used to create W_i using Eq. 3, a softmax probability vector indicating the importance of each patch variant. Finally, W_i is multiplied by the value embeddings (generated using Eq. 2 for all patch variants) as in Eq. 4, to generate the final output A_i.

4 EXPERIMENTS

We present multiple image classification experiments. Results are reported on three datasets: CIFAR10 (Krizhevsky, 2009), CIFAR100 (Krizhevsky, 2009), and ImageNet (Deng et al., 2009). Evaluation on the CIFAR10 and CIFAR100 datasets demonstrates the effectiveness of our method on low-resolution 32×32 images, while evaluation on ImageNet demonstrates the effectiveness of our method on a higher resolution of 224×224.

We consider state-of-the-art convolution-based baselines as well as transformer-based baselines. Beyond DeiT (Touvron et al., 2020), we also consider baselines that use a pyramid-like architecture: PVT (Wang et al., 2021), Swin (Liu et al., 2021) and Nest (Zhang et al., 2021). CvT (Wu et al., 2021) is also considered for ImageNet (CIFAR10/CIFAR100 values not reported). Convolutional baselines include Pyramid-164-48 (Han et al., 2017) and WRN28-10 (Zagoruyko & Komodakis, 2016) for CIFAR10/CIFAR100 and ResNet50, ResNet101 (He et al., 2016), ResNetY-4GF and ResNetY-8GF (Radosavovic et al., 2020) for ImageNet. We also perform an extensive ablation study for evaluating the each of our contributions.

CIFAR10/CIFAR100 As noted previously by (Zhang et al., 2021), previous transformer-based methods usually perform poorly on such datasets. This is because self-attention methods are typically data-intensive, while in these datasets both the resolution and the number of samples is relatively small. In contrast, our method reintroduced the locality bias of CNNs by first applying local attention over neighbouring patches. By subsequently applying global attention in a pyramid-like fashion our method benefits from the introduction of correlations between distant patches gradually, at different scales. For CIFAR10 and CIFAR100 datasets, we use $T = 18$ possible local variants: {(0,0), (1,0), (0,1), (-1,0), (0,-1), (2,0), (0,2), (-2,0), (0,-2), (1,1), (1,2), (1,-1), (-1,1), (-1,2), (-1,-1), (2,1), (2,2), (2,-1)}.

The results for CIFAR10 and CIFAR100 experiments are shown in Table 3. As can be seen, our method is superior to both CNN-based and Transformer-based baselines for each model size (tiny, small and base). Already at a small model size of 10.2M parameters our model achieves superior performance for both CIFAR10 and CIFAR100 on most baselines, which have a significantly larger number of parameters.

more than 20 GPU months, using 32GB GPUs. This is not more demanding than previous work. e.g, (Wang et al., 2021; Liu et al., 2021; Zhang et al., 2021). However, these resources are not at our disposal at this time.

	Input Resolution: 224 × 224				Input Resolution: 32 × 32				
	Output size	Layer	Tiny	Small	Output size	Layer	Tiny	Small	Base
1	$\mathrm{T} \times E_1 \times \frac{H}{S} \times \frac{W}{S}$ $\mathrm{T} \times (\frac{H}{S} \cdot \frac{W}{S}) \times E_1$ $(\frac{H}{S} \cdot \frac{W}{S}) \times E_1$	Conv. emb. Proj. Local Att.	$K = S = 7$ $P = 0$ $\Big\} \times T$	$K = S = 4$ $P = 0$ $\Big\} \times T$	$\mathrm{T} \times E_1 \times H \times W$ $\mathrm{T} \times (H \cdot W) \times E_1$	Conv. emb. Proj. Local Att.	$K = 3$ $S = P = 1$ $\Big\} \times T$		
2	$(\frac{H}{S} \cdot \frac{W}{S}) \times E_1$ $E_2 \times \frac{H}{2S} \times \frac{W}{2S}$ $(\frac{H}{2S} \cdot \frac{W}{2S}) \times E_2$	Global Att. Down. Proj.	$E_1 = 192$ $H_1 = 4$ $\Big\} \times 2$	$E_1 = 64$ $H_1 = 2$ $\Big\} \times 2$	$(H \cdot W) \times E_1$ $E_2 \times \frac{H}{2} \times \frac{W}{2}$ $(\frac{H}{2} \cdot \frac{W}{2}) \times E_2$	Global Att. Down. Proj.	$E_1 = 192$ $H_1 = 3$ $\Big\} \times 2$	$E_1 = 384$ $H_1 = 6$ $\Big\} \times 2$	$E_1 = 768$ $H_1 = 12$ $\Big\} \times 3$
3	$(\frac{H}{2S} \cdot \frac{W}{2S}) \times E_2$ $E_3 \times \frac{H}{4S} \times \frac{W}{4S}$ $(\frac{H}{4S} \cdot \frac{W}{4S}) \times E_3$	Global Att. Down. Proj.	$E_2 = 192$ $H_2 = 4$ $\Big\} \times 4$	$E_2 = 192$ $H_2 = 6$ $\Big\} \times 2$	$(\frac{H}{2} \cdot \frac{W}{2}) \times E_2$ $E_3 \times \frac{H}{4} \times \frac{W}{4}$ $(\frac{H}{4} \cdot \frac{W}{4}) \times E_3$	Global Att. Down. Proj.	$E_2 = 192$ $H_2 = 3$ $\Big\} \times 4$	$E_2 = 384$ $H_2 = 6$ $\Big\} \times 4$	$E_2 = 768$ $H_2 = 12$ $\Big\} \times 3$
4	$(\frac{H}{4S} \cdot \frac{W}{4S}) \times E_3$ $E_4 \times \frac{H}{8S} \times \frac{W}{8S}$ $(\frac{H}{8S} \cdot \frac{W}{8S}) \times E_4$	Global Att. Down. Prof.	$E_3 = 192$ $H_3 = 4$ $\Big\} \times 4$	-	$(\frac{H}{4} \cdot \frac{W}{4}) \times E_3$ $E_4 \times \frac{H}{8} \times \frac{W}{8}$ $(\frac{H}{8} \cdot \frac{W}{8}) \times E_4$	Global Att. Down. Proj.	$E_3 = 192$ $H_3 = 3$ $\Big\} \times 4$	$E_3 = 384$ $H_3 = 6$ $\Big\} \times 4$	$E_3 = 768$ $H_3 = 12$ $\Big\} \times 3$
5	$(\frac{H}{F} \cdot \frac{W}{F}) \times E_4$ E_4 # classes	Global Att. Avg Pool Linear.	$F = 8S$ $E_4 = 192$ $H_4 = 4$ $\Big\} \times 4$	$F = 4S$ $E_4 = 384$ $H_4 = 12$ $\Big\} \times 10$	$(\frac{H}{8} \cdot \frac{W}{8}) \times E_4$ E_4 # classes	Global Att. Avg Pool Linear.	$E_4 = 192$ $H_4 = 3$ $\Big\} \times 4$	$E_4 = 384$ $H_4 = 6$ $\Big\} \times 4$	$E_4 = 768$ $H_4 = 12$ $\Big\} \times 3$

Table 2: Architectures used for images with input resolution 224 × 224 (ImageNet) and for input resolution 32 × 32 (CIFAR10 and CIFAR100) for three model types: tiny, small and base. In step 1, we begin by applying a convolutional embedding (Conv emb.) to the input image. A kernel size K, stride S and padding P are used and the output has E_1 channels. A separate convolutional embedding is performed for each of the T image shifting variants (indicated beside the curly bracket) and positional encoding is added for each patch and variant embedding. Subsequently, we project the image (Proj.) into a three dimensional tensor, which is then used as input for our local attention block (Local Att.). From step 2 onward, we begin by applying a global attention block (Glob Att.). We subsequently un-project the input and apply a downsampling operation (Down), as described in Sec. 3.2, where the resulting dimension is E_i. H_i denotes the number of heads used. A projection layer is then applied, converting the input into a two-dimensional input that can be used for subsequent global attention blocks. Each such step (Global Att. - Down. - Proj.) may be applied a number of times, as indicated by the number beside the curly bracket. In the last step, the downsampling operation is replaced by average pooling (Avg. Pool) and the projecting operation is replaced by a linear layer (Linear), resulting in a logit vector. The final step (not shown) is a standard apllication of softmax, followed by a cross-entropy loss.

ImageNet In Tab. 4, we consider a comparison of our method to baselines on the ImageNet dataset, which consists of a much larger number of higher-resolution (224 × 224) images. Since the image resolution is 224 × 224, a fixed partition of the image into patches of size $S \times S$ results in $\frac{224}{S} \cdot \frac{224}{S}$ patches. As the attention mechanism is quadratic in the number of tokens B, S is either chosen to be large (ViT, DeiT), or a subset of patches is used as tokens (Nest, Swin). In contrast, our method considers, for each patch, $T = 10$ possible local variants: (0,0), (1,0), (2,0), (3,0), (0,1), (0,2), (0,3), (1,1), (2,2), (3,3). As a result, it can correctly capture details from its local neighborhood. Our local attention layer results in quadratic computation only in the number of variants. That is, the cost is $O(B \cdot T^2)$. The cost of subsequent global attention layers is $O(B^2)$. Therefore, as long as $T \leq \sqrt{B}$, our method results in lower or equal computation cost, and it also considers local variants of each patch. As can be seen in Tab. 4, this results in a superior performance of our method in comparison to baselines for two different model sizes.

4.1 Ablation Study

An ablation study is performed for our method. The results are summarized in Tab. 5. To demonstrate the superiority of our method, we consider the current state-of-the-art as baseline (Zhang et al., 2021). First, we examine the importance of shifting i.e. introducing shifting variants as input to the model and attending them with a local attention layer. By removing the use of shifting variants and local attention layers, our method is simplified to only applying hierarchical global attention the as described in Sec. 3.2. As can be seen (variant A), the tiny setting our method sur-

Type	Method	#Params (M)	Throughput	CIFAR10(%)	CIFAR100(%)
CNNs	Pyramid-164-48	1.7	3715.9	95.97	80.70
	WRN28-10	36.5	1510.8	95.83	80.75
Transformers	Deit-T	5.3	1905.3	88.39	67.52
	Deit-S	21.3	734.7	92.44	69.78
	Deit-B	85.1	233.7	92.41	70.49
	PVT-T	12.8	1478.1	90.51	69.62
	PVT-S	24.1	707.2	92.34	69.79
	PVT-B	60.9	315.1	85.05	43.78
	Swin-T	27.5	1372.5	94.46	78.07
	Swin-S	48.8	2399.2	94.17	77.01
	Swin-B	86.7	868.3	94.55	78.45
	Nest-T	6.2	627.9	96.04	78.69
	Nest-S	23.4	1616.9	96.97	81.70
	Nest-B	90.1	189.8	97.20	82.56
	Our-T	10.2	1700.2	**97.00**	**81.80**
	Our-S	36.2	807.6	**97.64**	**83.66**
	Our-B	115.8	276.8	**97.75**	**84.70**

Table 3: Classification accuracy on CIFAR10 and CIFAR100. The number of parameters (in millions), and inference throughput (images per second) on a single GPU are shown.

Type	Method	#Params (M)	GFLOPs (G)	Throughput	ImageNet(%)
CNNs	ResNet-50	25	3.9	1226.1	76.2
	ResNet-101	45	7.9	753.6	77.4
	RegNetY-4GF	21	4.0	1156.7	80.0
	RegNetY-8GF	39	8.0	591.6	81.7
Transformers	Deit-T	5.7	1.3	2536.5	72.2
	Deit-S	22	4.6	940.4	79.8
	PVT-T	13.2	1.9	-	75.1
	PVT-S	24.5	3.8	-	79.8
	Swin-T	29	4.5	755.2	81.3
	Swin-S	50	8.7	436.9	83.0
	Nest-T	17	5.8	633.9	81.5
	Nest-S	38	10.4	374.5	83.3
	CVT-T	20	4.5	-	81.6
	CVT-S	32	7.1	-	82.5
	Our-T	9.9	3.1	797.2	**75.4**
	Our-S	22.0	7.6	298.9	**82.2**

Table 4: Classification accuracy on the ImageNet validation set. The number of parameters (in millions), GFLOPs, and inference throughput (images per second) on single GPU are shown. Throughput is given in baselines in which this value is reported.

passes the baseline even without using shifting variants. Comparing the shiftless variant A to our full method (variant F) demonstrates the performance gain achieved by adding shifting variants.

Next, we analyzed the effect of shifting the image with different number of variants (#Shifting). As can be seen (variants A to B to F), increasing the number of shifting variants improves the model's performance. To reduce the number of shifts, we focus on shifts that are either on the horizontal or vertical lines, i.e, the 9 shifts: $(0,0),(0,1),(0,2),(1,0),(2,0),(-1,0),(-2,0),(0,-1),(0,-2)$, or a subset of the shifts in which we include diagonal lines as well, i.e., the 18 shifts out of the possible

Method	Model Variant	#Shifting	Conv. Variations	Pos. Embed.	#Params (M)	CIFAR10 (%)
Baseline	nested-T	0	-	-	6.2	96.04
Ours w/o shifting	A	0	-	-	6.8	96.50
Ours	B	9	✓	✓	8.4	96.70
	C	18	×	×	6.8	96.69
	D	18	×	✓	10.1	96.77
	E	18	✓	×	7.6	96.85
	F	18	✓	✓	10.2	97.00

Table 5: Ablation analysis for examining the contribution of: (1) shifting variants, (2) number of shifting variants, (3) shifting variants generated using learned convolutional filters, and, (4) employing different positional embeddings for each shifting variant.

25: $(0,0)$, $(1,0)$, $(0,1)$, $(-1,0)$, $(0,-1)$, $(2,0)$, $(0,2)$, $(-2,0)$, $(0,-2)$, $(1,1)$, $(1,2)$, $(1,-1)$, $(-1,1)$, $(-1,2)$, $(-1,-1)$, $(2,1)$, $(2,2)$, $(2,-1)$.

In addition, we compare two approaches for shifting: (i) constructing T shifting variants as a pre-processing step, using image translation with reflection padding and (ii) passing the original image as input to the network, and applying T convolutional layers as in Sec. 3.1 (as is done in our method). In Tab. 5, "Conv. Variations" indicates applying (ii) as opposed to (i). As can be seen, comparing variant C with variant E, and comparing variant D with variant F, generating shifting variants using learned convolutional layers improves the model performance.

Finally, we checked whether it is necessary to add different positional embeddings to each shifting variant rather than simply learning one set of positional embeddings for all T variants. As can be seen, comparing variant C with variant D and variant E with variant F, there is a trade-off between parameters and accuracy. Adding positional embeddings for each variant improves the performance, but the number of parameters increases. In our experiments we apply different positional embeddings to each shifting variant.

As can be seen, our complete method, variant F, outperforms the baseline. Furthermore, our tiny model outperforms the small model of the baseline - as can be seen in Tab. 3.

5 Conclusion

While convolutional layers typically employ small strides, resulting in heavily overlapping patches, the token-based approach of recent transformer-based techniques has led to a grid view of the input image. This leads to the loss of the translation invariance which played a major role in the development of neural networks in computer vision as well as in the study of biological vision models (Bouvrie et al., 2009).

In this work, we reintroduce the locality bias of CNNs into a transformer-based architecture in order to benefit from the ability to model fine-detailed local correlations in addition to the coarse-detail global correlations that transformers model well. We employ two types of attention layers. The local attention layer models the correlation of a patch with its local shifting variants, thus modeling fine grained correlations. The global attention layer, applied in a pyramid-like manner, with decreasing input resolution, models long-range correlations. The use of local-global attention layers as opposed to a single attention layer is crucial for introducing the desired locality bias and capturing the correlation between neighbouring local shifts of each patch. This is especially useful for smaller datasets, with low resolution images, such as CIFAR10/CIFAR100. Nevertheless, our method also scales well to large images as is the case for ImageNet.

We demonstrate the superiority of our method on both small resolution inputs of 32×32 (CIFAR10/CIFAR100) and larger resolution inputs of 224×224 (ImageNet). Our method achieves superior accuracy to other convolutional and transformer-based state-of-the-art methods with a comparable number of parameters.

REPRODUCIBILITY STATEMENT

The full implementation of our method and scripts for reproducing the experiments will be made publicly available. This will include a README file as well as requirements for reproducing all experiments.

Our experimental comparison in Sec. 4 includes important statistics, including memory, throughput and GFLOPs. Datasets used in our work are publicly available. A detailed depiction of our training architectures as well as implementation details are provided in Tab. 2 and Sec. 3.3 respectively. We further compare our method architecture in relation to baselines in Tab. 1.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Jake Bouvrie, Lorenzo Rosasco, and Tomaso Poggio. On invariance in hierarchical models. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper/2009/file/2ab56412b1163ee131e1246da0955bd1-Paper.pdf.

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

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

Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5927–5935, 2017.

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

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*, pp. 8129–8138, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. pp. 32–33, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

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:2103.14030*, 2021.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

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

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*, pp. 10428–10436, 2020.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021.

Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers, 2021.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zizhao Zhang, Han Zhang, Long Zhao, Ting Chen, and Tomas Pfister. Aggregating nested transformers. In *arXiv preprint arXiv:2105.12723*, 2021.