

An Image is 16×16 words: Transformers for Image Recognition at Scale

• [\[\[2010.11929.pdf\]\]](#)

• Basic Idea:

- Can we use Transformers for Vision task without much changes?
 - **P4** When considering the computational cost of pre-training the model, ViT performs very favourably, attaining state of the art on most recognition benchmarks at a lower pre-training cost.
- Challenges with Transformers wrt CNNs:
 - **P1** Transformers lack some of the inductive biases **P2** inherent to CNNs, such as translation equivariance and locality, and therefore do not generalize well when trained on insufficient amounts of data.
 - One way to overcome this: **P2** Parmar et al. (2018) applied the self-attention only in local neighborhoods for each query pixel instead of globally. Such local multi-head dot-product self attention blocks can completely replace convolutions
 - Words in NLP have semantic meanings, whereas pixels are just raw numbers. Hence, how to perform self-attention on pixels because quadratic cost of self-attention would be huge for pixels. Idea: Convert images into patches and consider each patch as a word.
- ViT-B/16 (86M parameters, 224 × 224 pre-training resolution, 384 × 384 fine-tuning resolution) performance (top1 accuracy) on ImageNet-1k (Table 5 of the paper):
 - Pre-trained on ImageNet-1k: 77.91
 - Pre-trained on ImageNet-21k: 83.97
 - Pre-trained on JFT-300M: 84.15

• Vision Transformer (ViT)

• **P3**

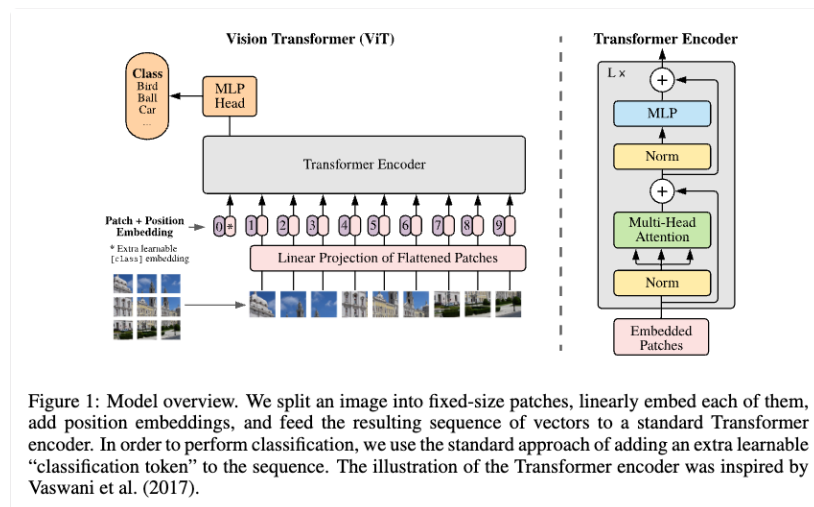


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable “classification token” to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

• Generating patches:

- Input image $x \in \mathbb{R}^{H \times W \times C} \implies$ Flatten to $x_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$ where
 - (H, W) is the input image resolution,
 - C is # channels,
 - (P, P) is the resolution of each image patch and
 - $N = HW/P^2$ is the # patches = Input sequence length to the Transformer.
- **P5** Note that the Transformer's sequence length is inversely proportional to the square of the patch size, thus models with smaller patch size are computationally more expensive
- **Patch Embeddings \mathbf{E}** : Linear projection of the patches into fixed dimensional latent vector of D dimension to be used throughout the Network. $\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$.
- **P3** Similar to BERT's [class] token, we prepend a learnable embedding to the sequence of embedded patches ($z_0 = \text{xclass}$), whose state at the output of the Transformer encoder (z_0 L) serves as the image representation y
- **Position Embeddings \mathbf{E}_{pos}** : $\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$

● P4

The MLP contains two layers with a GELU non-linearity.

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1 \mathbf{E}; \mathbf{x}_p^2 \mathbf{E}; \dots; \mathbf{x}_p^N \mathbf{E}] + \mathbf{E}_{\text{pos}}, \quad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \mathbf{E}_{\text{pos}} \in \mathbb{R}^{(N+1) \times D} \quad (1)$$

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \quad \ell = 1 \dots L \quad (2)$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \quad \ell = 1 \dots L \quad (3)$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0) \quad (4)$$

● Food for thought:

- Does the aspect ratio of the patches matter?
- Converting images into patches is similar to converting images into grid for YOLO. Therefore, using ViTs for Object Detection should be straight-forward.
- Patch Embeddings are similar to convolutions. Since, the weights are shared across patches, there is locality and translation equivariance.

● Inductive Bias:

- CNNs: Locality, 2D neighbourhood structure, translation equivariance
- ViT: Only MLP layers are local and translation equivariant (Remember Position-wise feed forward neural networks in Transformers). This is because the weight matrix of this feed forward network is shared between the positions i.e., the MLP layers in Transformers are similar to convolution.
- P4 The two-dimensional neighborhood structure is used very sparingly: in the beginning of the model by cutting the image into patches and at fine-tuning time for adjusting the position embeddings for images of different resolution (as described below). Other than that, the position embeddings at initialization time carry no information about the 2D positions of the patches and all spatial relations between the patches have to be learned from scratch

● Hybrid Architecture:

- P4 Hybrid Architecture. As an alternative to raw image patches, the input sequence can be formed from feature maps of a CNN (LeCun et al., 1989). In this hybrid model, the patch embedding projection \mathbf{E} (Eq. 1) is applied to patches extracted from a CNN feature map. As a special case, the patches can have spatial size 1×1 , which means that the input sequence is obtained by simply flattening the spatial dimensions of the feature map and projecting to the Transformer dimension. The classification input embedding and position embeddings are added as described above
- Food for thought:** Connection of this hybrid architecture to the Conditional Random Fields (CRFs) based optimization used in early segmentation networks to improve the segmentation masks (DeepLab v1)?

● Fine-tuning

- While pre-training they use a hidden-layer followed by a classification layer (2 fc layers) on top of \mathbf{z}_L^0 (the image representation \mathbf{y}), whereas for fine-tuning they use a single fc layer which directly classifies into the target K classes. ● P4 we remove the pre-trained prediction head and attach a zero-initialized $D \times K$ feedforward layer, where K is the number of downstream classes
- Based on previous studies, they fine-tune at a higher resolution. When training at higher resolution, they keep the patch size to be the same, thereby increasing the sequence length. ● P4 The Vision Transformer can handle arbitrary sequence lengths (up to memory constraints), however, the pre-trained position embeddings may no longer be meaningful. We therefore perform 2D interpolation of the pre-trained position embeddings, according to their location in the original image. Note that this resolution adjustment and patch extraction are the only points at which an inductive bias about the 2D structure of the images is manually injected into the Vision Transformer
- Food for thought:**
 - Dynamic Patch Size Selection:* Can we dynamically select patch-sizes based on the input resolution? This can be important because if we keep the patch-size same, for high-resolution images the basic visual element (patch in this case) may not have any semantic information all together. Therefore, we would need bigger patch sizes to capture better semantic information.

● Results

- Notations:** ● P5 In what follows we use brief notation to indicate the model size and the input patch size: for instance, ViT-L/16 means the "Large" variant with 16×16 input patch size.

● Tricks of the Trade:

- **Batch Normalization vs Group Normalization:** P5 For the baseline CNNs, we use ResNet (He et al., 2016), but replace the Batch Normalization layers (Ioffe & Szegedy, 2015) with Group Normalization (Wu & He, 2018).
- **Standardized Convolutions:** P5 used standardized convolutions (Qiao et al., 2019). These modifications improve transfer (Kolesnikov et al., 2020), and we denote the modified model "ResNet (BiT)".
- Adam instead of SGD for fine-tuning ResNets
- **Optimization:** P5 Polyak & Juditsky (1992) averaging with a factor of 0.9999 (Ramachandran et al., 2019; Wang et al., 2020b).
- P13 Dropout, when used, is applied after every dense layer except for the the qkv-projections and directly after adding positional- to patch embeddings.
- P13 For ImageNet we found it beneficial to additionally apply gradient clipping at global norm 1.

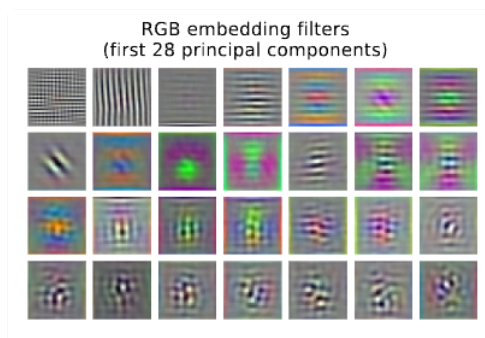
- **ViTs have fewer inductive biases than ResNets:** P7 the convolutional inductive bias is useful for smaller datasets, but for larger ones, learning the relevant patterns directly from data is sufficient, even beneficial.

- **Scaling Study:** Keeping the dataset size fixed, assess performance versus pre-training cost of each model:
 - P8 First, Vision Transformers dominate ResNets on the performance/compute trade-off. ViT uses approximately 2 – 4× less compute to attain the same performance (average over 5 datasets)
 - P8 Second, hybrids slightly outperform ViT at small computational budgets, but the difference vanishes for larger models. This result is somewhat surprising, since one might expect convolutional local feature processing to assist ViT at any size
 - P8 Third, Vision Transformers appear not to saturate within the range tried, motivating future scaling efforts
 - Scaling depth is better than scaling width of the network.
 - P16 Decreasing the patch size and thus increasing the effective sequence length shows surprisingly robust improvements without introducing parameters.

Visual Inspection:

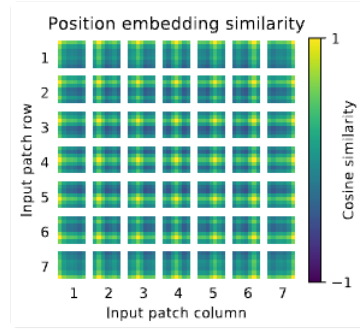
Patch Embeddings:

P9



- Patch embeddings can be thought of as convolutions. $\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$, therefore, this layer can be thought of as learning D filters, each of shape $P \times P \times C$. Now, usually D is as high as 768, therefore, to visualize the important filters, we can do a PCA and pick first 28 filters.
- P8 The components resemble plausible basis functions for a low-dimensional representation of the fine structure within each patch.

Position Embeddings:



● P9 Tiles show the cosine similarity between the position embedding of the patch with the indicated row and column and the position embeddings of all other patches.

- P8 model learns to encode distance within the image in the similarity of position embeddings, i.e. closer patches tend to have more similar position embeddings.
- P8 Further, the row-column structure appears; patches in the same row/column have similar embeddings.
- P8 Finally, a sinusoidal structure is sometimes apparent for larger grids (Appendix D). That the position embeddings learn to represent 2D image topology explains why hand-crafted 2D-aware embedding variants do not yield improvements (Appendix D.4).

• Self-Attention:

● P13

Standard \mathbf{qkv} self-attention (SA, Vaswani et al. (2017)) is a popular building block for neural architectures. For each element in an input sequence $\mathbf{z} \in \mathbb{R}^{N \times D}$, we compute a weighted sum over all values \mathbf{v} in the sequence. The attention weights A_{ij} are based on the pairwise similarity between two elements of the sequence and their respective query \mathbf{q}^i and key \mathbf{k}^j representations.

$$[\mathbf{q}, \mathbf{k}, \mathbf{v}] = \mathbf{z} \mathbf{U}_{qkv} \quad \mathbf{U}_{qkv} \in \mathbb{R}^{D \times 3D_h}, \quad (5)$$

$$A = \text{softmax} \left(\frac{\mathbf{q} \mathbf{k}^\top}{\sqrt{D_h}} \right) \quad A \in \mathbb{R}^{N \times N}, \quad (6)$$

$$\text{SA}(\mathbf{z}) = A \mathbf{v}. \quad (7)$$

Multihead self-attention (MSA) is an extension of SA in which we run k self-attention operations, called “heads”, in parallel, and project their concatenated outputs. To keep compute and number of parameters constant when changing k , D_h (Eq. 5) is typically set to D/k .

$$\text{MSA}(\mathbf{z}) = [\text{SA}_1(\mathbf{z}); \text{SA}_2(\mathbf{z}); \dots; \text{SA}_k(\mathbf{z})] \mathbf{U}_{msa} \quad \mathbf{U}_{msa} \in \mathbb{R}^{k \cdot D_h \times D} \quad (8)$$

- Attention Distance: ● P8 the average distance in image space across which information is integrated, based on the attention weights. This is similar to receptive field size in CNNs.
- ● P8 Self-attention allows ViT to integrate information across the entire image even in the lowest layers. ● P8 We find that some heads attend to most of the image already in the lowest layers, showing that the ability to integrate information globally is indeed used by the model.
- ● P8 This highly localized attention is less pronounced in hybrid models that apply a ResNet before the Transformer (Figure 7, right), suggesting that it may serve a similar function as early convolutional layers in CNNs.