

Deep learning

Transformers

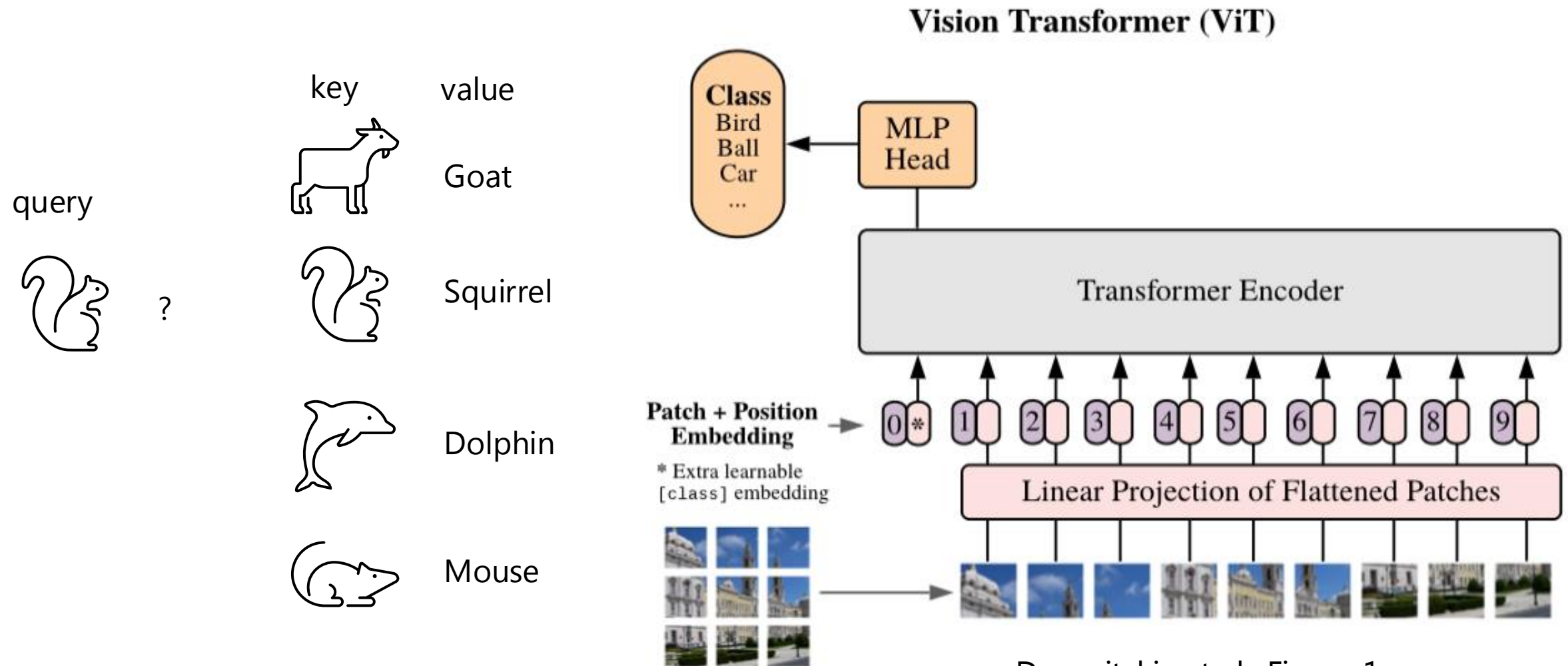
2024/12/18

Jon Sparring,
Department of Computer Science

UNIVERSITY OF COPENHAGEN

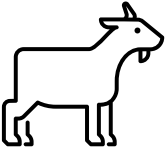


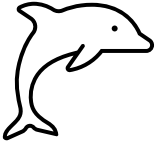



Transformer networks – Vision Transformer ViT: Vaswani ea (2017) & Dosovitskiy ea (2020)

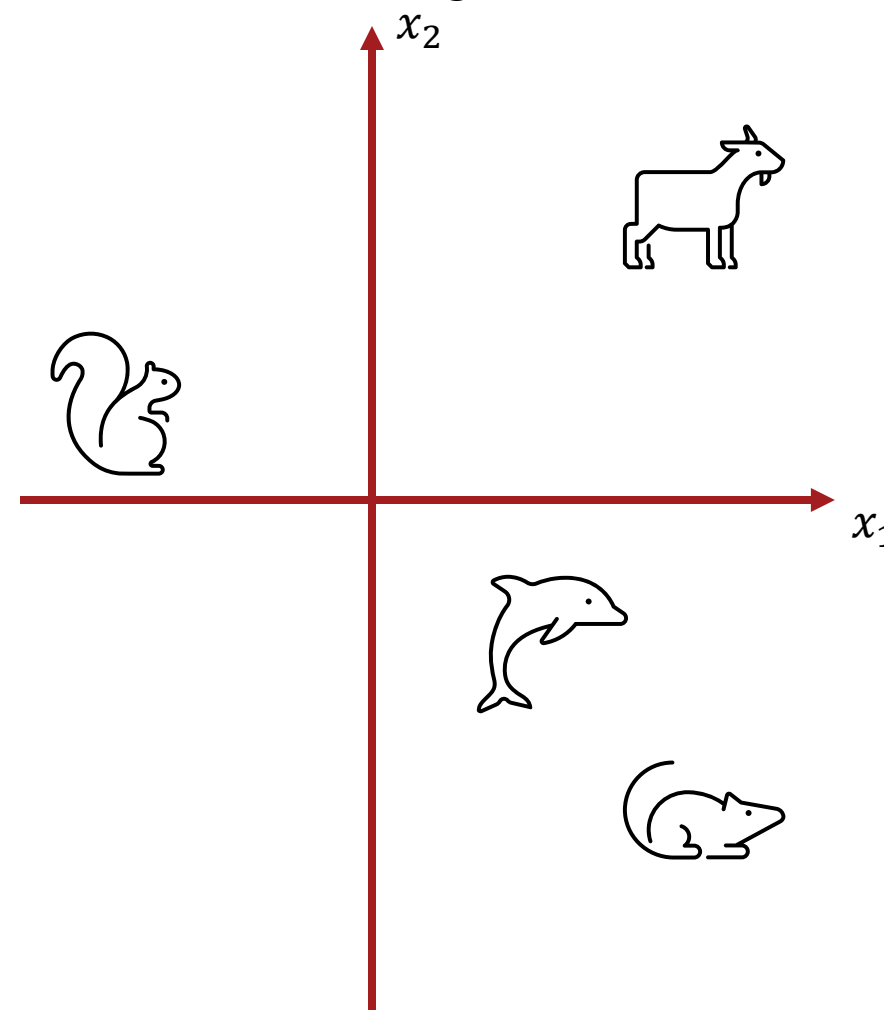


Dosovitskiy et al., Figure 1

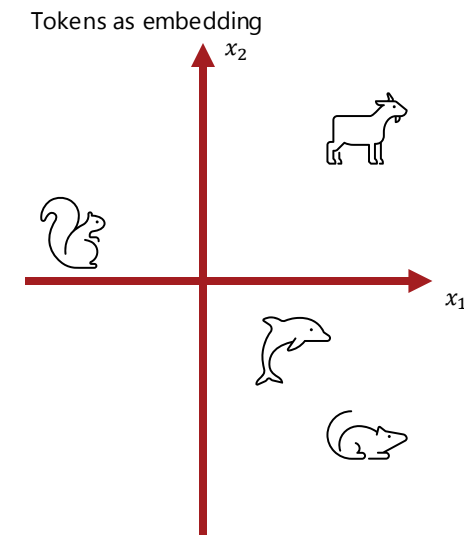
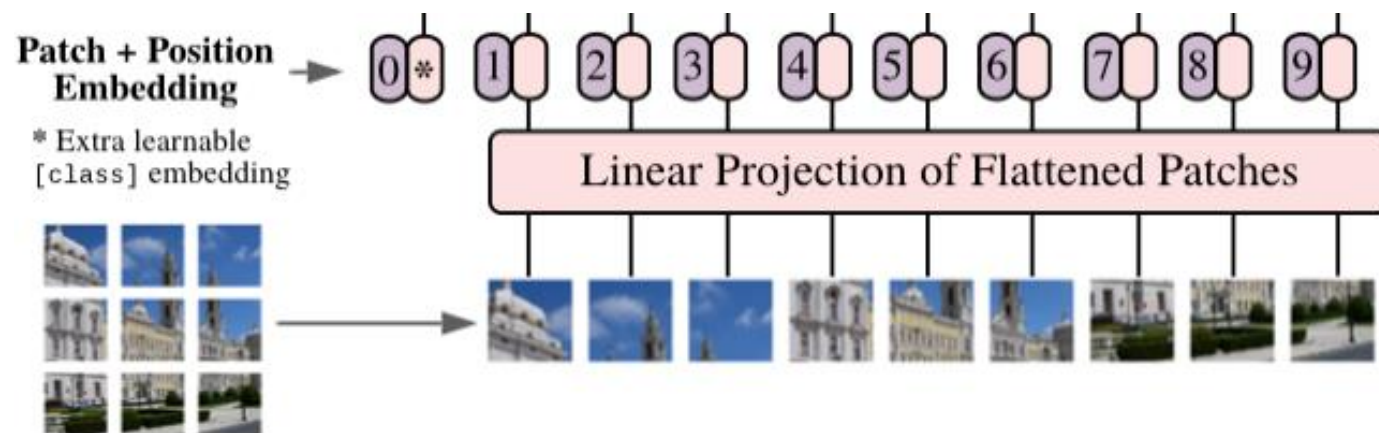
Tokens: Cut into patches, flatten and linearly embed in lower dimensional space

	key	value
query		Goat
 ?		Squirrel
		Dolphin
		Mouse

Tokens as embedding



Embeddings++



$$I \in \mathbb{R}^{H \times W \times C}, I_p \in \mathbb{R}^{M \times M \times C}, \text{Flatten: } \mathbb{R}^{M \times M \times C} \rightarrow \mathbb{R}^{M^2 C}, \text{Token: } \mathbb{R}^{M^2 C} \rightarrow \mathbb{R}^D$$

Convention: row vectors

$$\mathbf{x}_j = \text{Flatten}(I_j) \in \mathbb{R}^{M^2 C}$$

$$\mathbf{t}_j = \text{Token}(\mathbf{x}_j) = \mathbf{x}_j \mathbf{E} \in \mathbb{R}^D$$

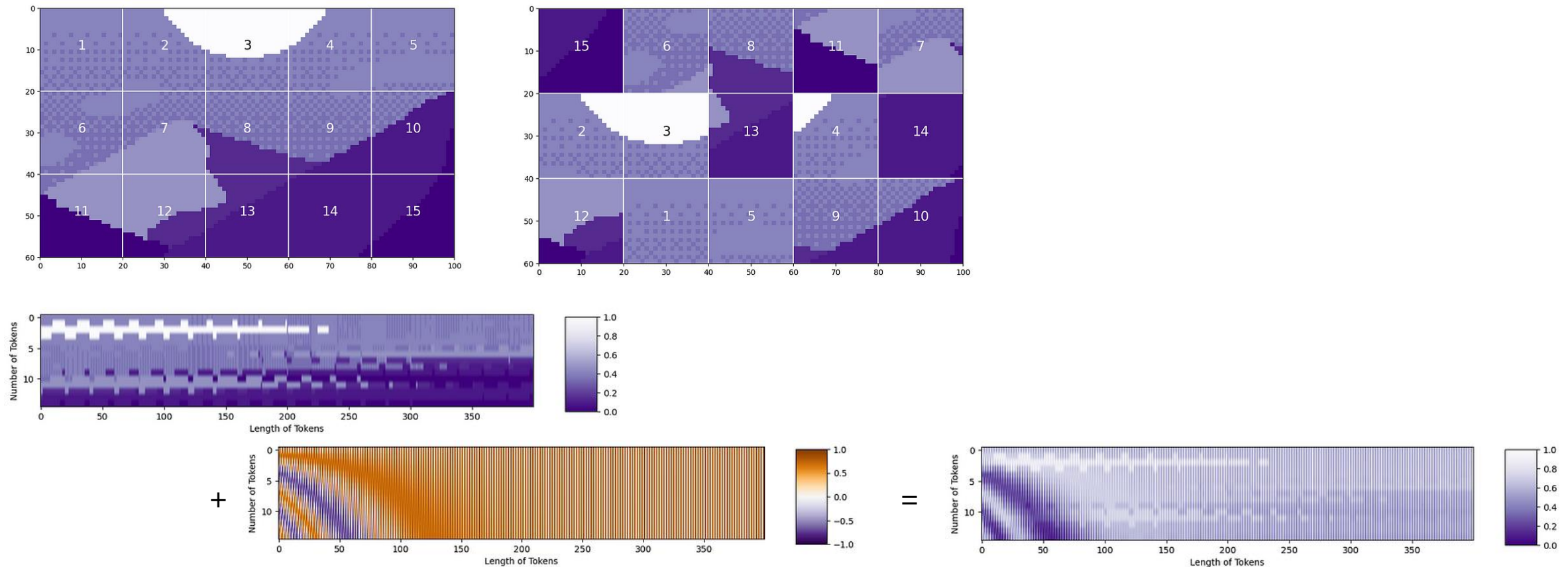
$$\mathbf{z}^0 = [\mathbf{x}_{\text{class}}; \mathbf{t}_1; \mathbf{t}_2; \dots; \mathbf{t}_N] + \text{PositionEmbedding}() \in \mathbb{R}^{(N+1) \times D}$$

Positional Embedding: Vision depends on position

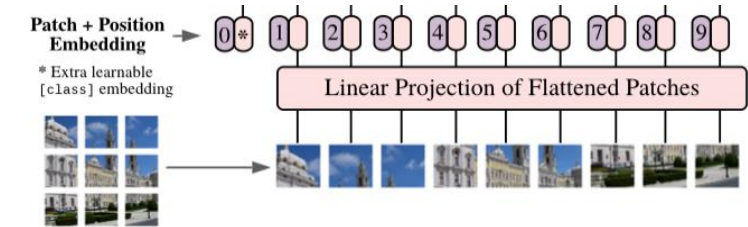
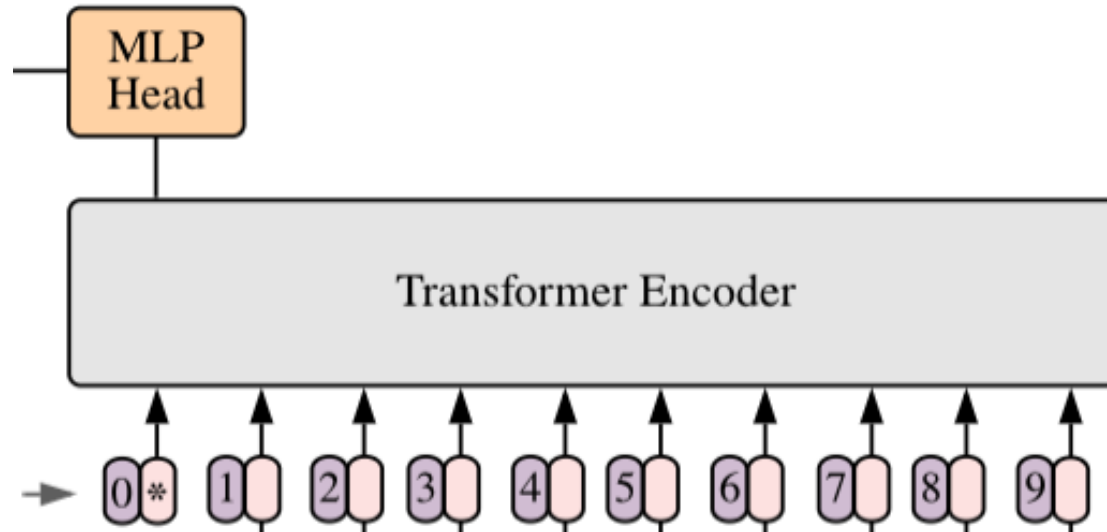
Yuan et al (2021). *Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet*.

Figures from

<https://towardsdatascience.com/position-embeddings-for-vision-transformers-explained-a6f9add341d5>



Encoding



A sequence of attention layers: $l = 1 \dots L$

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{t}_1; \mathbf{t}_2; \dots; \mathbf{t}_N] \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}'_l = \text{MSA}(\text{LN}(\mathbf{z}'_{l-1})) + \mathbf{z}'_{l-1} \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}_l = \text{MLP}_2(\text{LN}(\mathbf{z}_l)) + \mathbf{z}'_l \in \mathbb{R}^{(N+1) \times D}$$

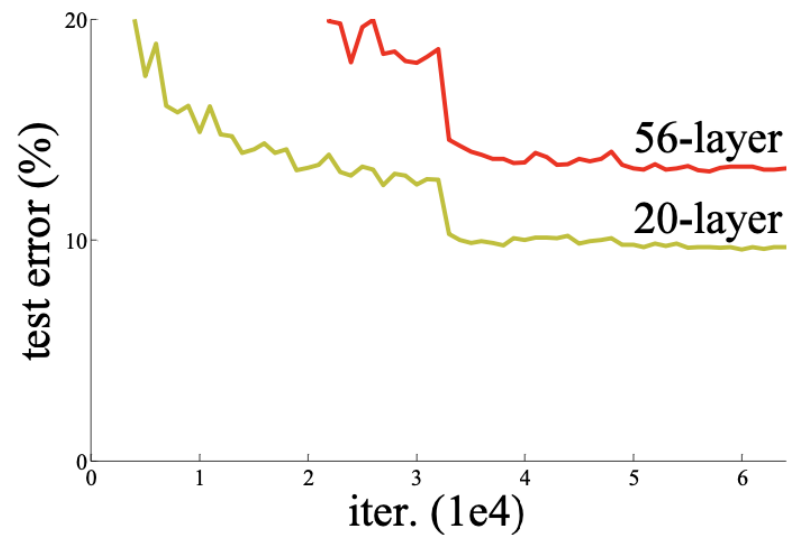
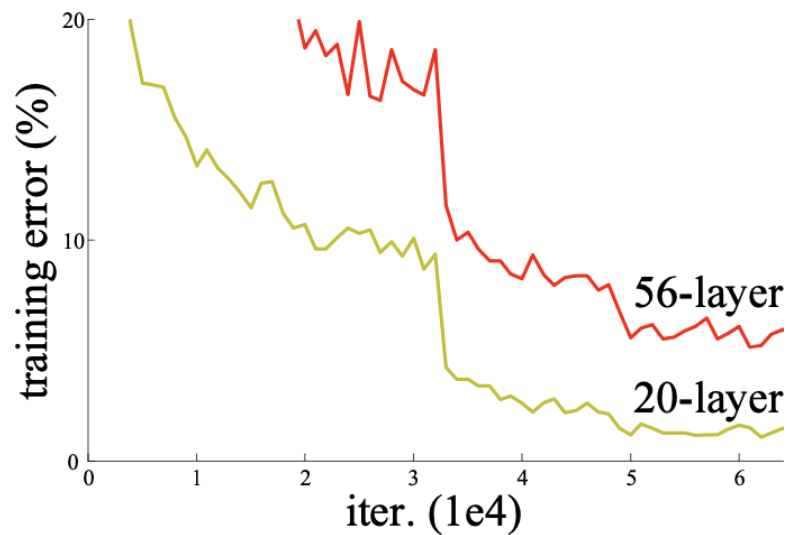
MSA: Multiheaded self-attention
 MLP_n : n-layer Multilayer perceptron
 LN: Layer normalization

Residual learning

Residual learning for optimizing deeper networks

He et al, "Deep Residual Learning for Image Recognition, CVPR 2016

Observation: Deeper networks = poorer convergence



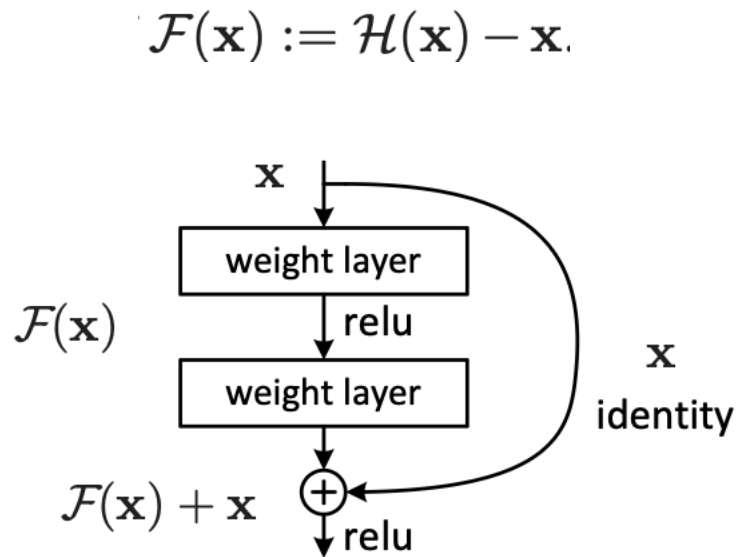
He et al. Figure 1

Residual learning for optimizing deeper networks

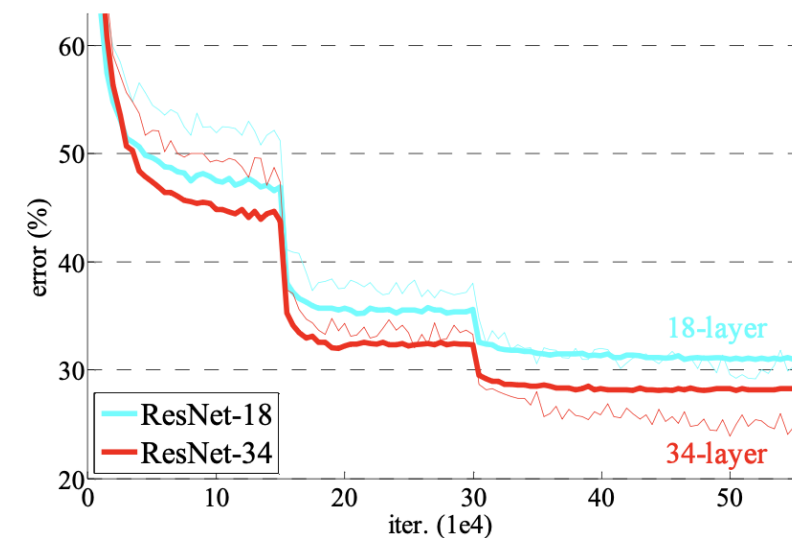
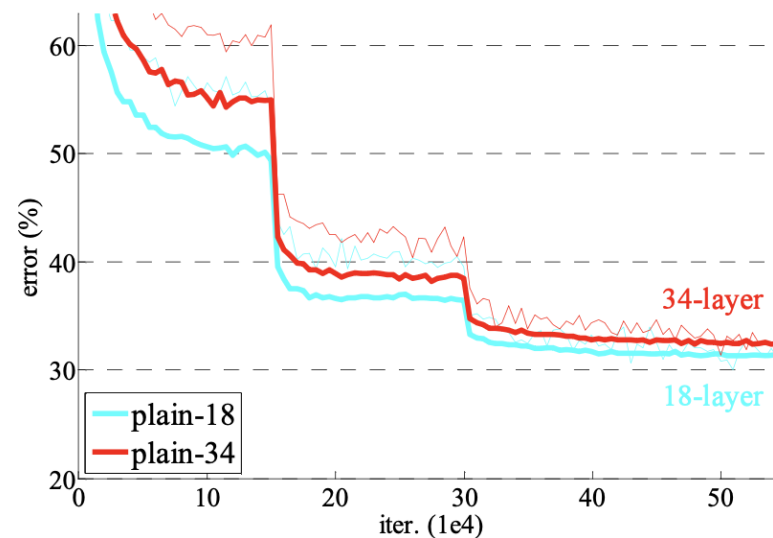
He et al, "Deep Residual Learning for Image Recognition, CVPR 2016

Observation: Deeper networks = poorer convergence

Solution: Train on residuals



He et al. Figure 2



He et al. Figure 4

Multilayer perceptron (MLP) and Layer Normalization (LN)

Ba, Kiros, and Hinton, Layer normalization, 2016

$$\text{LN}(\mathbf{z}) = \frac{\mathbf{z} - \bar{\mathbf{z}}}{\sqrt{\text{Var}(\mathbf{z}) + \epsilon}}$$

A sequence of attention layers: $l = 1 \dots L$

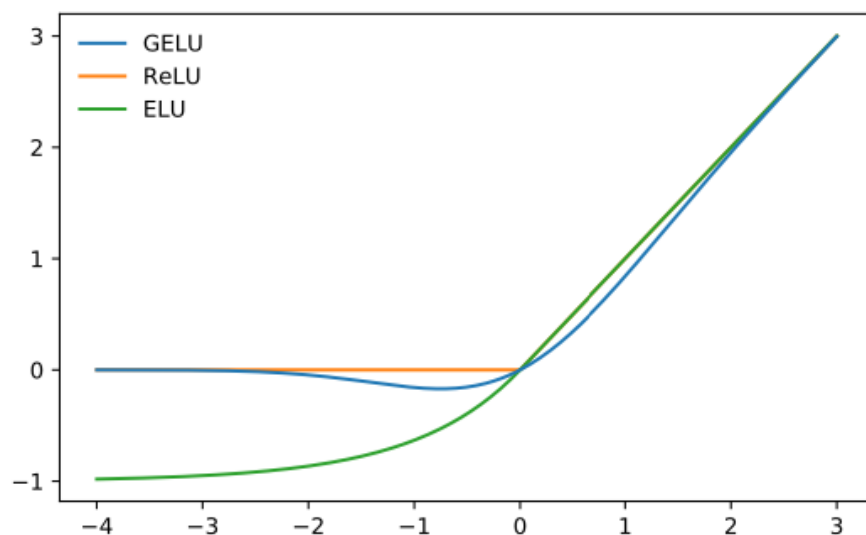
$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{t}_1; \mathbf{t}_2; \dots; \mathbf{t}_N] \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}'_l = \text{MSA}(\text{LN}(\mathbf{z}'_{l-1})) + \mathbf{z}'_{l-1} \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}_l = \text{MLP}(\text{LN}(\mathbf{z}'_l)) + \mathbf{z}'_l \in \mathbb{R}^{(N+1) \times D}$$

MLP: Single layer with two GELU

GELU: Hendryks & Gimpel, Gaussian Error Linear Units (GELU)

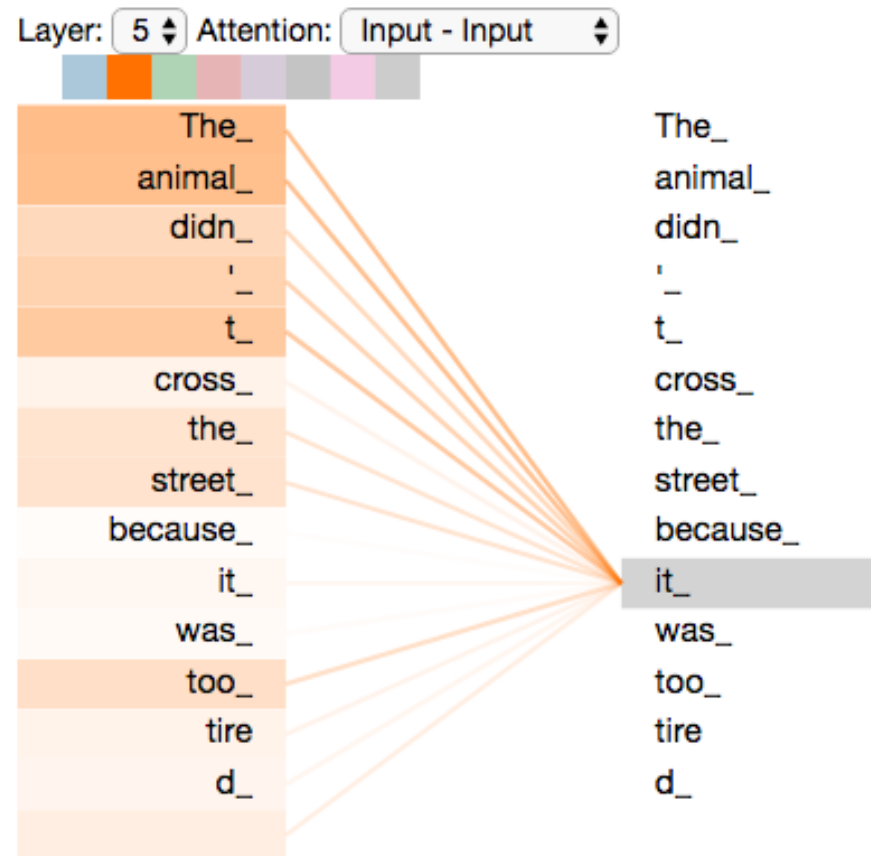


Hendryks & Gimpel, Figure 1

$$\begin{aligned} \text{GELU}(x) &= \frac{x}{2} \left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right) \right] \\ &\simeq x/2 \left(1 + \tanh\left((x + 0.044715x^3) \sqrt{\frac{2}{\pi}} \right) \right) \end{aligned}$$

Attention:

The animal didn't cross the street because it was too tired



Self-attention head

$$I \in \mathbb{R}^{H \times W \times C}, I_p \in \mathbb{R}^{M \times M \times C}, \text{Flatten: } \mathbb{R}^{M \times M \times C} \rightarrow \mathbb{R}^{M^2 C}, \text{Token: } \mathbb{R}^{M^2 C} \rightarrow \mathbb{R}^D$$

$$\mathbf{z} \in \mathbb{R}^{(N+1) \times D}$$

Query $\mathbf{q}_j = \mathbf{z}_{j*} U_q \in \mathbb{R}^{D_h}$
 Key $\mathbf{k}_j = \mathbf{z}_{j*} U_k \in \mathbb{R}^{D_h}$
 Value $\mathbf{v}_j = \mathbf{z}_{j*} U_v \in \mathbb{R}^{D_h}$

$U_* \in \mathbb{R}^{D \times D_h}$ learnable

$$\mathbf{a}_{ij} = \text{softmax}\left(\frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{D_h}}\right)$$

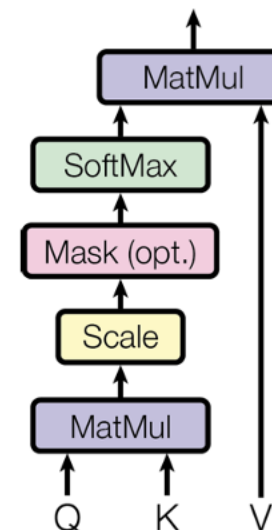
$$\text{softmax}(\{s_j\}) = \left\{ \frac{e^{s_j}}{\sum_{j=1}^n e^{s_j}} \right\}$$

$$\text{SA}(\mathbf{z}_{i*}) = \sum_j \mathbf{a}_{ij} \mathbf{v}_j \in \mathbb{R}^{D_h}$$

Query how one embedding (\mathbf{q}) matches keys (\mathbf{k}), which are all other embeddings.

Softmax select the best matching queries to their cosine difference

Output is now computed as a weighted sum of values (\mathbf{v}).



Vaswani, Fig 2

Multihead self-attention

$I \in \mathbb{R}^{H \times W \times C}, I_p \in \mathbb{R}^{M \times M \times C}, \text{Flatten: } \mathbb{R}^{M \times M \times C} \rightarrow \mathbb{R}^{M^2 C}, \text{Token: } \mathbb{R}^{M^2 C} \rightarrow \mathbb{R}^D$

$\mathbf{z} \in \mathbb{R}^{(N+1) \times D}$

Query $\mathbf{Q} = \{\mathbf{q}_j\} \in \mathbb{R}^{(N+1) \times D_h}$
 Key $\mathbf{K} = \{\mathbf{k}_j\} \in \mathbb{R}^{(N+1) \times D_h}$
 Value $\mathbf{V} = \{\mathbf{v}_j\} \in \mathbb{R}^{(N+1) \times D_h}$

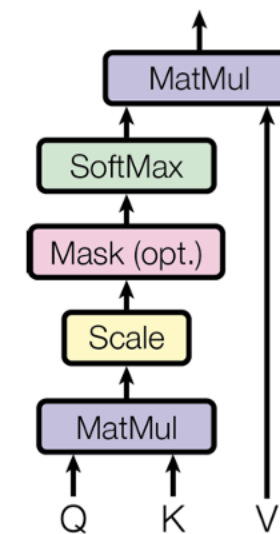
$\mathbf{U}_* \in \mathbb{R}^{D \times D_h}$ learnable

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D_h}}\right) \in \mathbb{R}^{(N+1) \times (N+1)}$$

$$\text{SA}(\mathbf{z}) = \mathbf{A}\mathbf{V} \in \mathbb{R}^{(N+1) \times D_h}$$

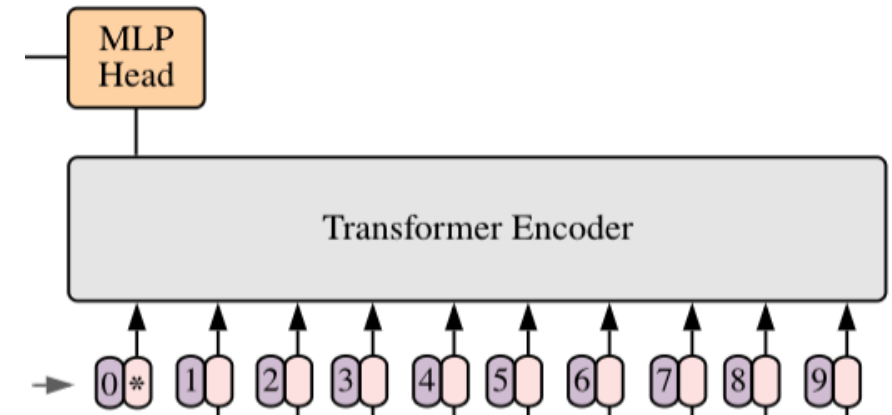
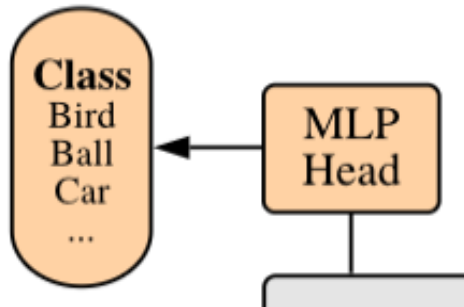
$$\mathbf{U}_{msa} \in \mathbb{R}^{kD_h \times D}$$

$$\text{MSA}(\mathbf{z}) = [\text{SA}_1(\mathbf{z}), \text{SA}_2(\mathbf{z}), \dots, \text{SA}_k(\mathbf{z})]\mathbf{U}_{msa} \in \mathbb{R}^{(N+1) \times D}$$



Vaswani, Fig 2

Encoding



A sequence of attention layers: $l = 1 \dots L$

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{t}_1; \mathbf{t}_2; \dots; \mathbf{t}_N] \in \mathbb{R}^{2(N+1) \times D}$$

$$\mathbf{z}'_l = \text{MSA}(\text{LN}(\mathbf{z}'_{l-1})) + \mathbf{z}'_{l-1} \in \mathbb{R}^{2(N+1) \times D}$$

$$\mathbf{z}_l = \text{MLP}_2(\text{LN}(\mathbf{z}'_l)) + \mathbf{z}'_l \in \mathbb{R}^{2(N+1) \times D}$$

$\text{Class} = \text{MLP}_1(\mathbf{z}_L)$

Inductive bias [Dosovitskiy]

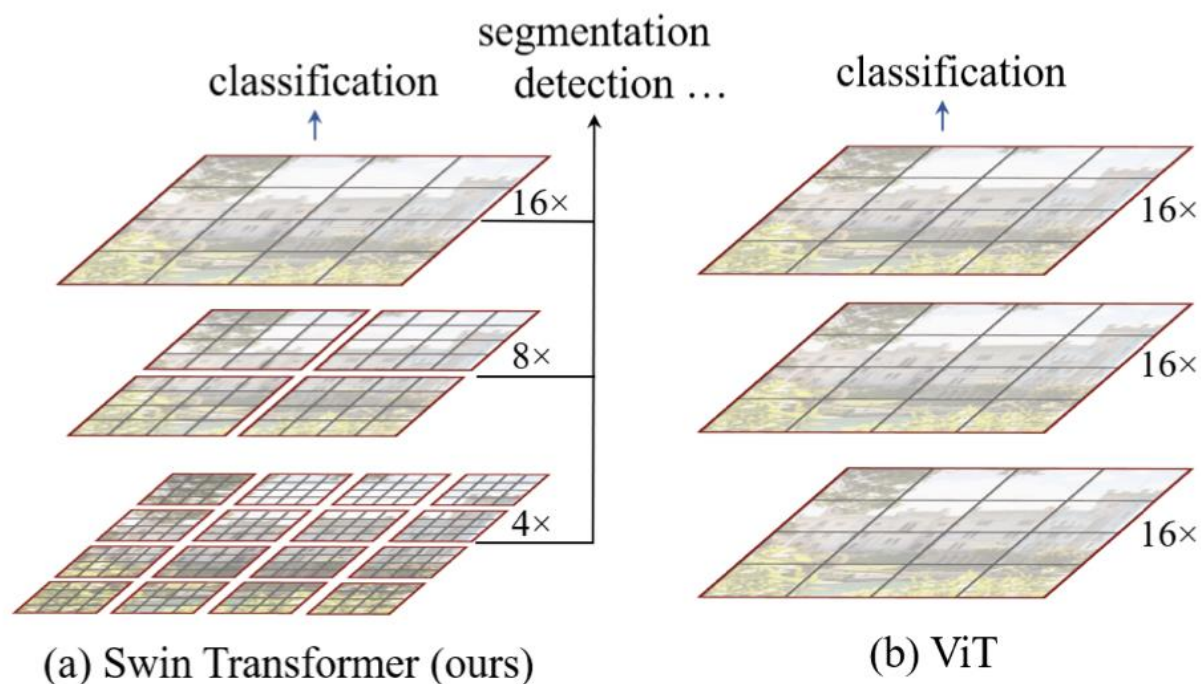
Inductive bias. We note that Vision Transformer has much less image-specific inductive bias than CNNs. In CNNs, locality, two-dimensional neighborhood structure, and translation equivariance are baked into each layer throughout the whole model. In ViT, only MLP layers are local and translationally equivariant, while the self-attention layers are global. The two-dimensional neighborhood

ImageNet: <https://image-net.org/>
14*10⁶ images, 2*10⁴ categories, 10⁶ images with
bounding boxes

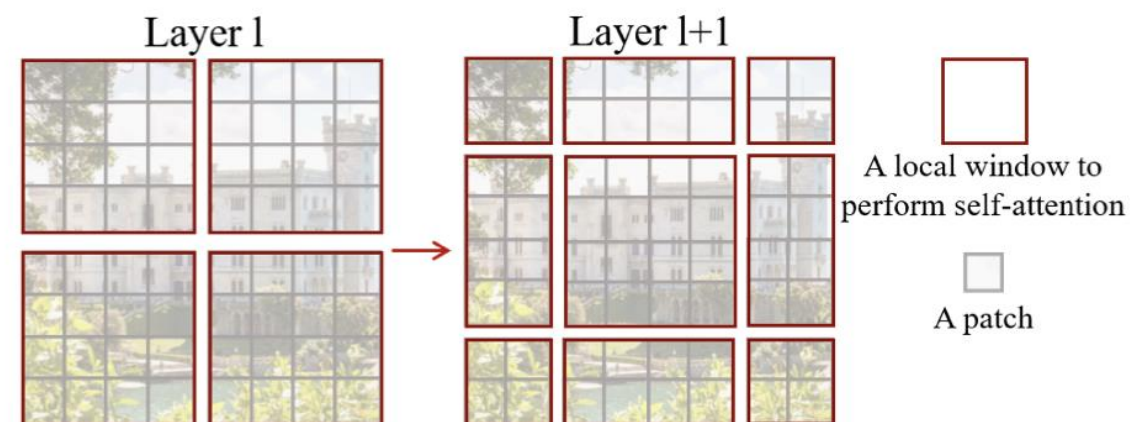


State-of-the-art 2024

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows



xuan Wei[†]
o



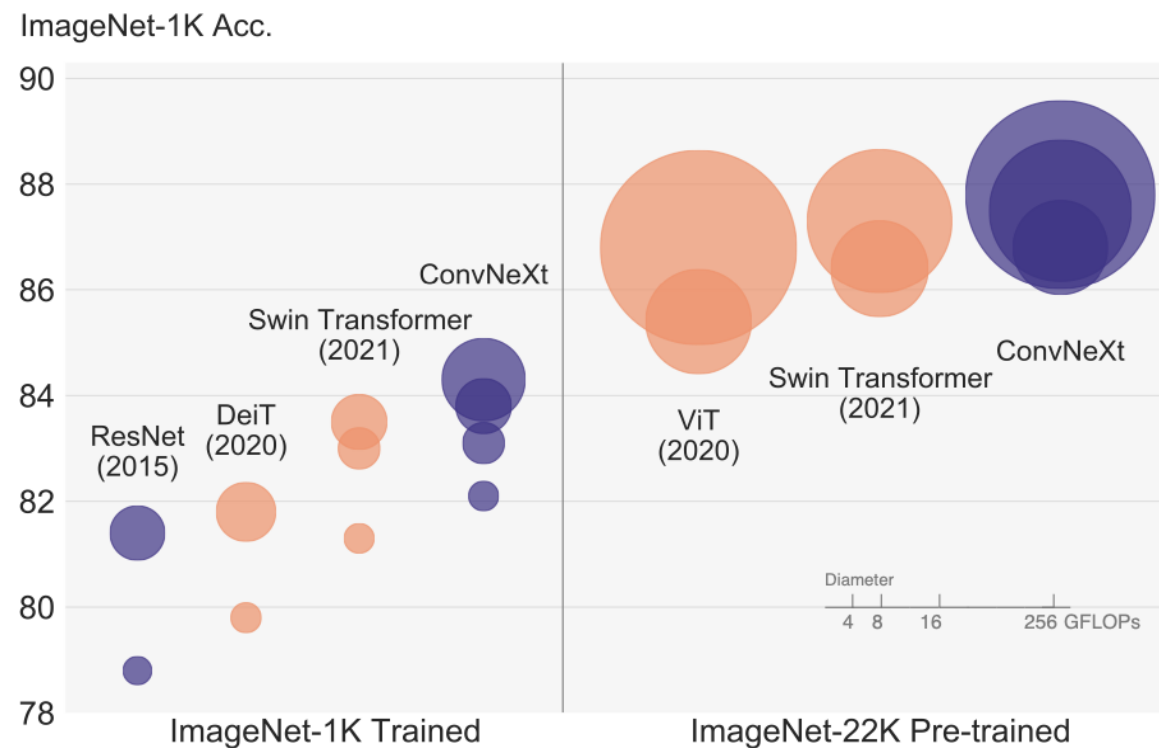
ConvNeXt

A ConvNet for the 2020s

Zhuang Liu^{1,2*} Hanzi Mao¹ Chao-Yuan Wu¹ Christoph Feichtenhofer¹ Trevor Darrell² Saining Xie^{1†}

¹Facebook AI Research (FAIR) ²UC Berkeley

Code: <https://github.com/facebookresearch/ConvNeXt>



Papers with code <https://paperswithcode.com/>

The image displays three overlapping browser windows from the Papers with Code website. The leftmost window shows the 'Vision Transformer' model page by rwightman / pytorch-image-models, last updated on Feb 14, 2021. It features a dropdown menu for 'vit_large_patch16_224' and a table of model parameters and training details. The middle window shows the 'Swin Transformer: Hierarchical Vision Transformer using Shifted Windows' paper by Ze Liu et al., published at ICCV 2021. The rightmost window shows the 'ConvNeXt' paper by Liu et al., introduced in 'A ConvNet for the 2020s'. It includes a 'Backbone Architectures' tag and buttons for 'Read Paper' and 'See Code'.

Vision Transformer

rwightman / pytorch-image-models
Last updated on Feb 14, 2021

vit_large_patch16_224

Parameters	FLOPs	File Size
304 Million	119 Billion	1.16 GB

Training Data	Training Resources	Training Time
JFT-300M, ImageNet	TPUv3	

Training Techniques	Architecture
SGD with Momentum, Cosine Annealing, Gradient Clip	Layer Normalization, Multi-Head Attention, Tanh Activation, Connections, Attention Dropout, Dropout, Scaled Dot-Attention, GELU, Convolution

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

ICCV 2021 · Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, et al.
Edit social preview

This paper presents a new vision Transformer, called Swin Transformer, that can be used as a general-purpose backbone for computer vision. Challenges in adapting Transformer to vision arise from differences between the two domains, such as large variations in visual entities and the high resolution of pixels in images compared to words. To address these differences, we propose a hierarchical Transformer whose representation is composed of shifted windows. The shifted windowing scheme brings great benefits by limiting self-attention computation to non-overlapping local windows while maintaining a window connection. This hierarchical architecture has the flexibility to model vision at different scales. It has linear computational complexity with respect to image size. These qualities make it compatible with a broad range of vision tasks, including image classification, object detection, and semantic segmentation. The Swin Transformer achieves 51.1 mask AP on COCO test-dev and semantic segmentation (53.5 mIoU on ADE20K), performance surpasses the previous state-of-the-art by a large margin of +2.7 mask AP on COCO, and +3.2 mIoU on ADE20K, demonstrating the potential of Transformer models as vision backbones. The hierarchical design and the shifted window are beneficial for all-MLP architectures. The code and models are publicly available at github.com/microsoft/Swin-Transformer.

ConvNeXt

Introduced by Liu et al. in [A ConvNet for the 2020s](#)

Source: [A ConvNet for the 2020s](#)

[Read Paper](#) [See Code](#)

Papers

Search for a paper or author

Paper	Code	Results	Date	Stars ↑
DenseNets Reloaded: Paradigm Shift Beyond ResNets and ViTs	Code	Results	28 Mar 2024	32.623