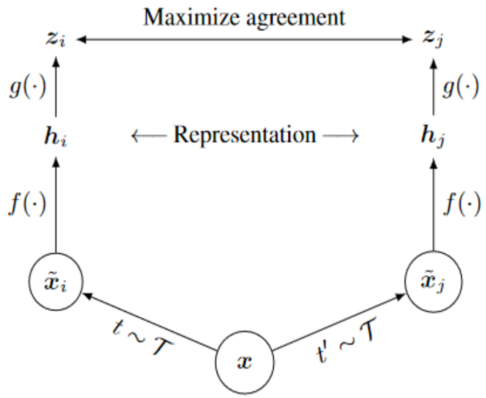
Photogrammetry & Robotics Lab

Machine Learning for Robotics and Computer Vision

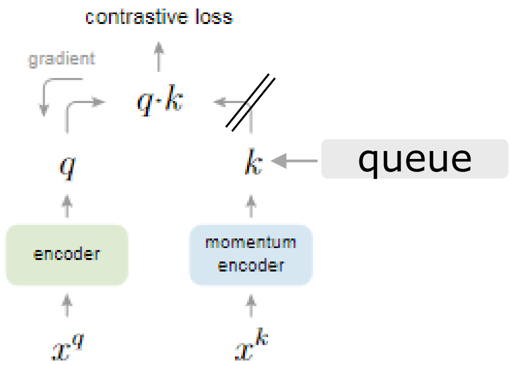
Beyond CNNs

Jens Behley

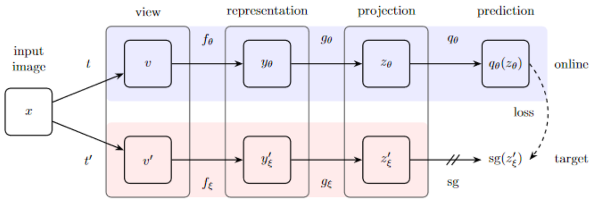
Last Lecture



SimCLR



MoCo

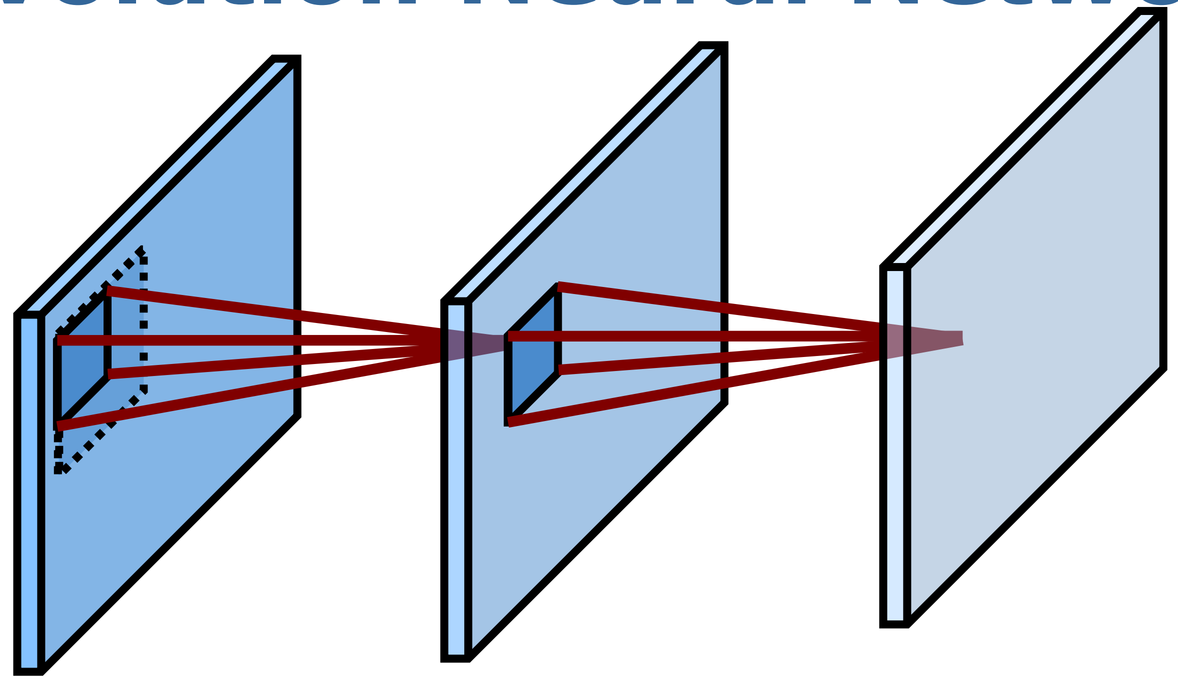


BYOL

- Labeling large amounts of data is expensive
- Discussed two paradigms to overcome lack of data:
  - Supervised pretraining on large existing datasets and fine-tuning of last layers on target dataset
  - Self-supervised pretraining on target dataset

- Discussed different state-of-the-art strategies: SimCLR, MoCo, and BYOL

Convolution Neural Networks



- Until now: Convolutions as main building block
- Inductive bias → spatial neighborhood of pixels and translation equivariance
- Deep architectures enable to have large receptive fields (long range dependencies)
- Are convolutions the only way to solve vision tasks?

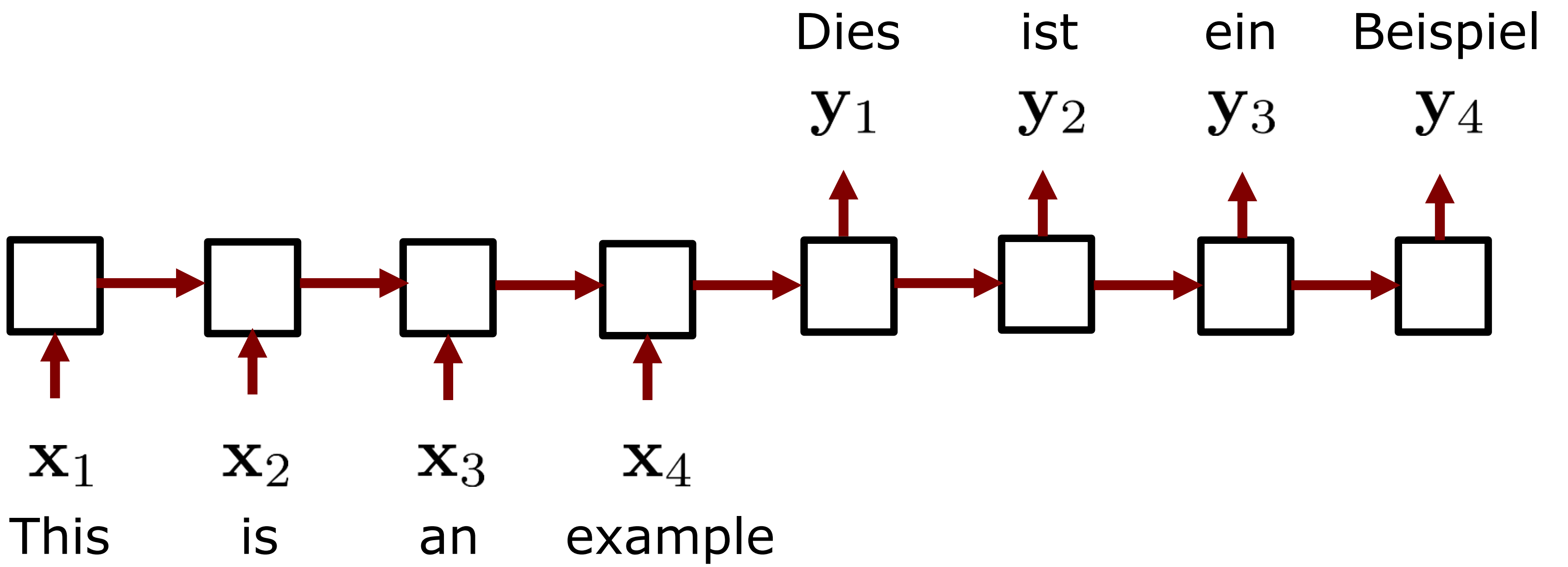
Transformer in NLP

- Since 2017, Transformer are the method of choice for Natural Language Processing (NLP) tasks
- Transformer architecture radically changed the way NLP is performed

- Very recently, Transformer were applied to a range of vision tasks with state-of-the-art performance

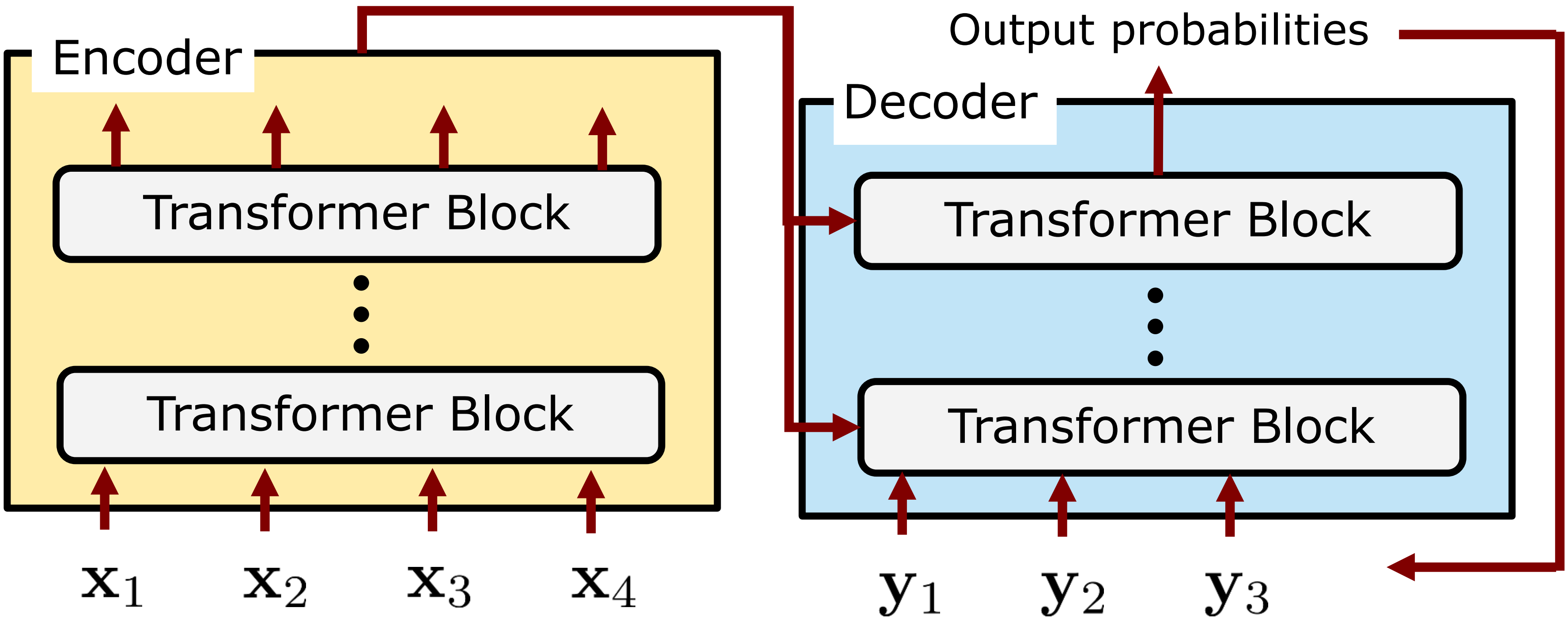
- Important: No convolutions involved!

NLP before 2017



- NLP was all about recurrent neural networks (RNN) → Long-term Short-Term Memory (LSTM)
- Sequence models with a memory → **Problem**: memory needs to capture all information from before
- Showed especially limitations for long sequences

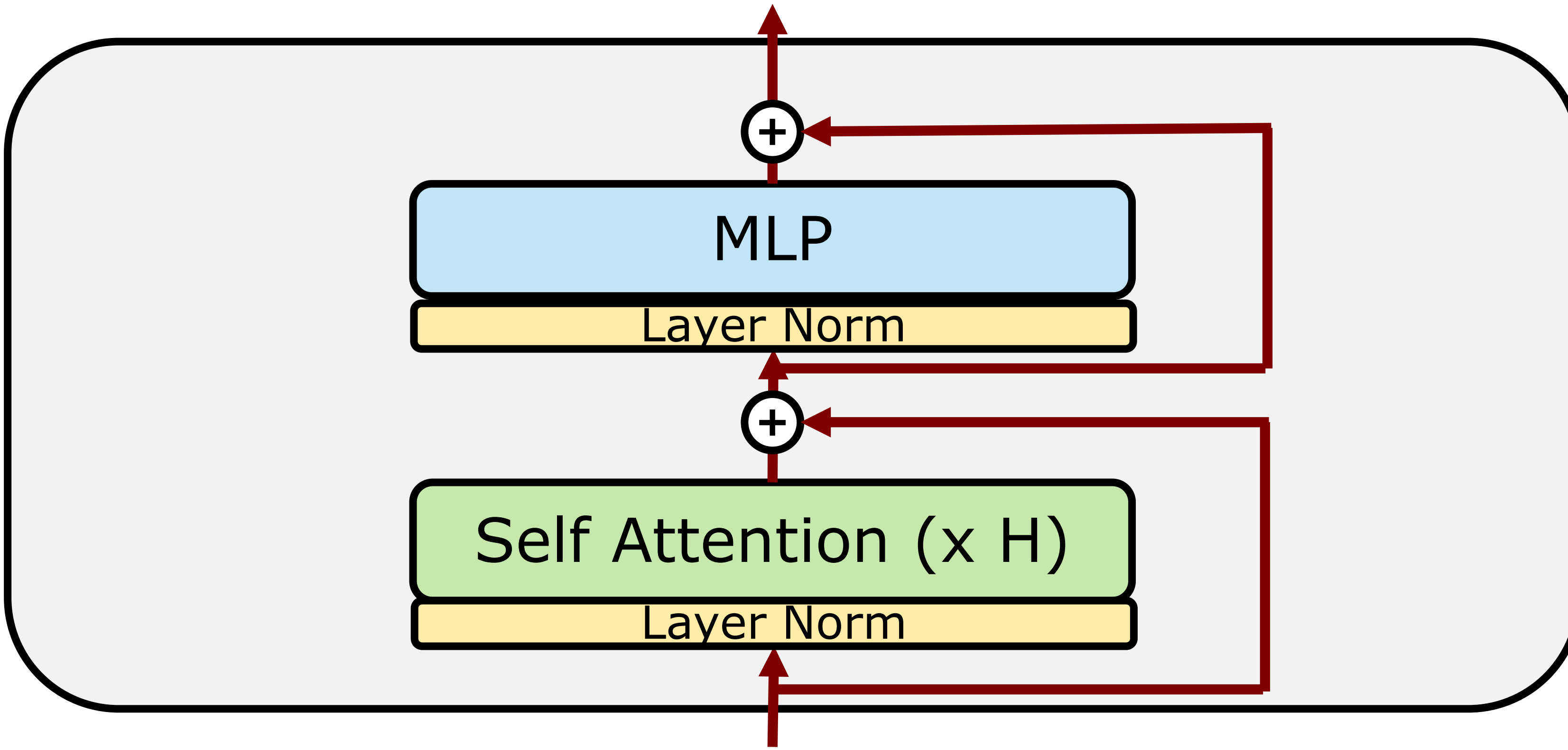
Transformer for Translation



- Now: **whole sequence** of tokens $\mathbf{x} \in \mathbb{R}^D$ as input
- For machine translation: produce token at a time and use previous output tokens as input to decoder
- Details see [Vaswani, 2017]

[Vaswani, 2017]

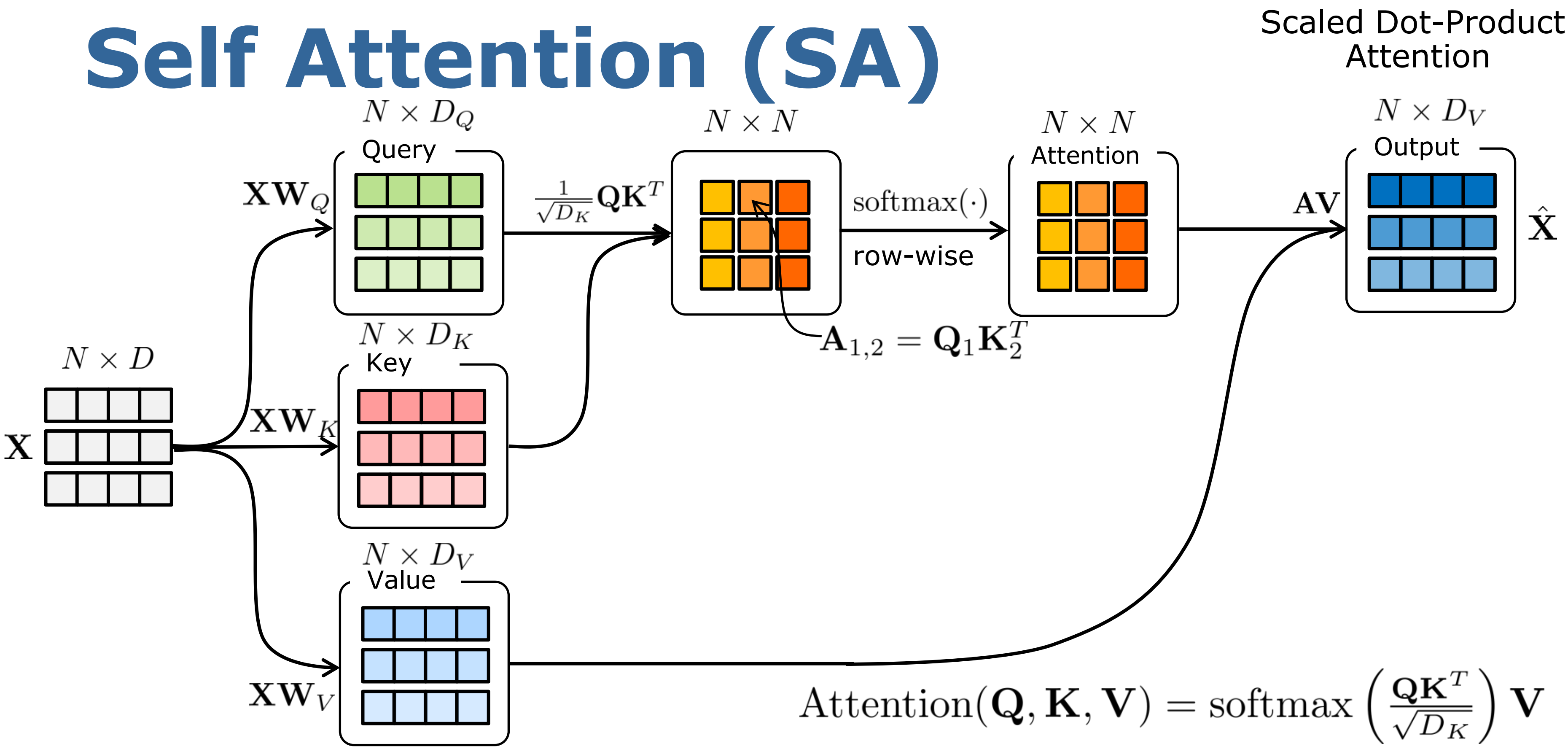
Transformer Block



- Each block consists of attention module and fully-connected layers with non-linearity (MLP)
- Skip-connections

[Vaswani, 2017]

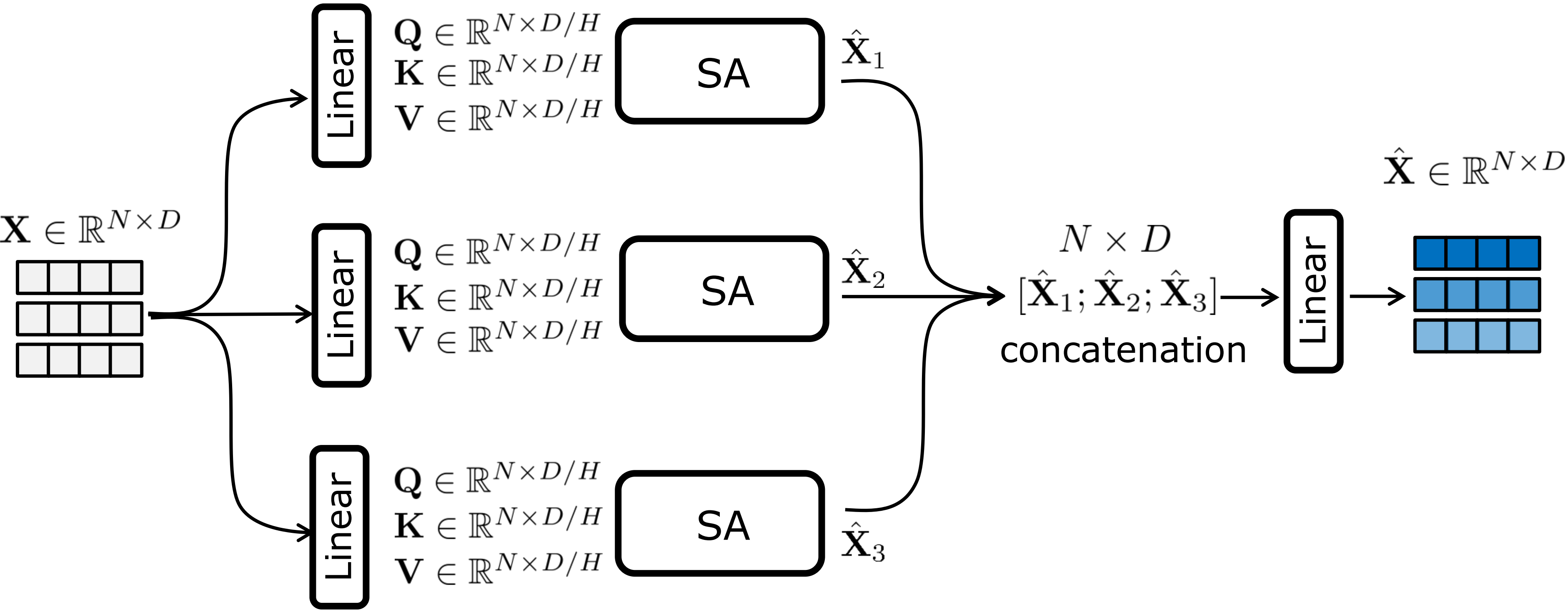
Self Attention (SA)



- Weighted combination of the inputs (= complete sequence!)
- Enables to adapt compute on-the-fly depending on similarity between query and key
- Projections learn similarity function

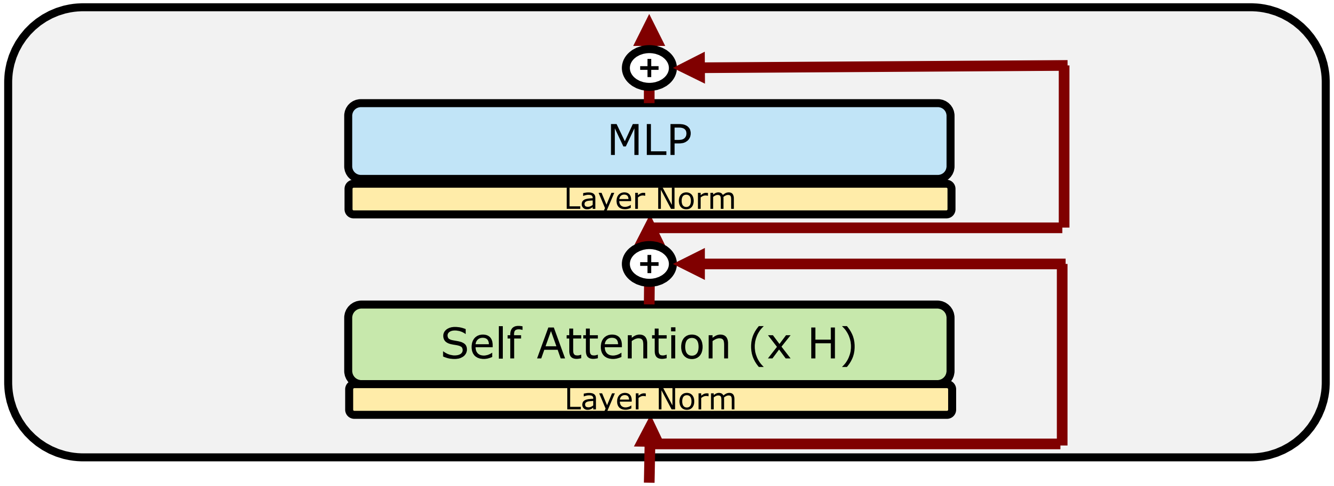
[Vaswani, 2017]

Multi-Head Attention



- Use multiple self attention blocks in parallel → multi-head attention (#heads = H)
- Use D/H as dimension of projections to keep compute independent of H
- Each SDA defines different attention pattern (similar to convolutional kernel)

Multi Layer Perceptron



- Fully-connected layers are applied to each of the N feature vectors of the N feature vectors:

$$MLP(\mathbf{X}) = \max(0, \mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W_2} + \mathbf{b}_1$$

$$\mathbf{W}_1 \in \mathbb{R}^{D \times D_{\text{ff}}}, \mathbf{W}_2 \in \mathbb{R}^{D_{\text{ff}} \times D}, b_1 \in \mathbb{R}^{D_{\text{ff}}}, b_2 \in \mathbb{R}^{D}$$

- In the NLP Transformer: $D = 512, D_{\text{ff}} = 2048$

[Vaswani, 2017]

Positional Encoding

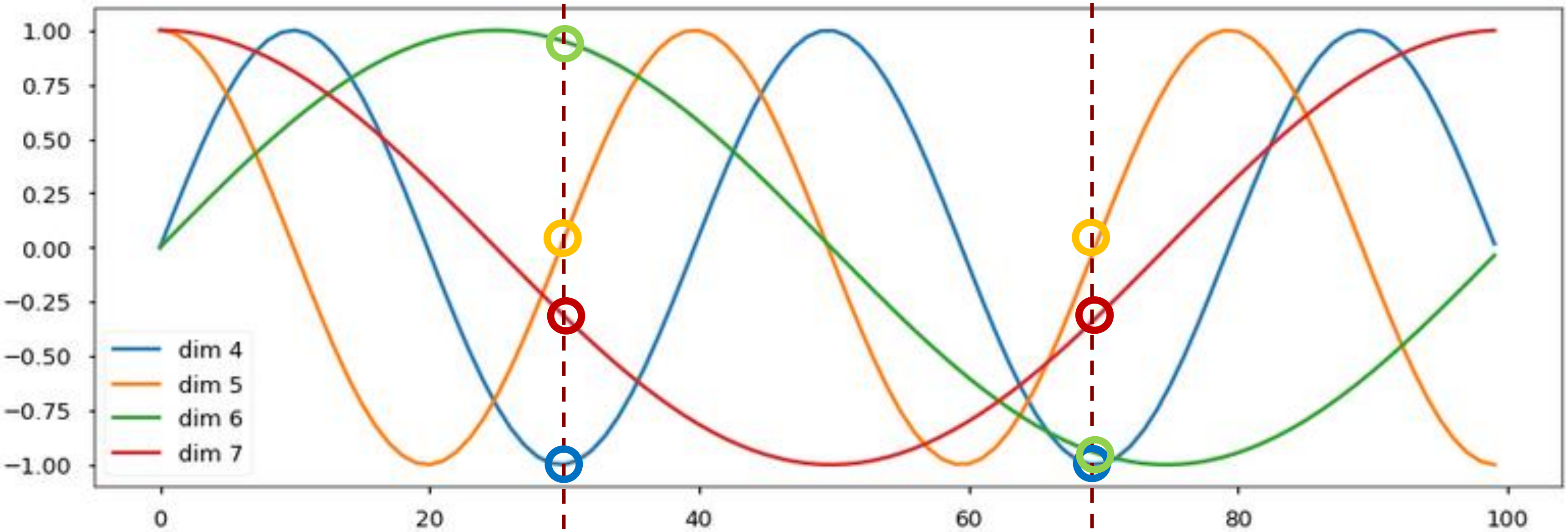
- Transformer has no notion of position → order of tokens does not matter!
- Introduce constants, i.e., **positional encoding** to provide positional information!

$$\mathrm{PE}(\mathrm{pos}, 2i) = \sin(\mathrm{pos}/10000^{2i/D})$$
$$\mathrm{PE}(\mathrm{pos}, 2i+1) = \cos(\mathrm{pos}/10000^{2i/D})$$

- Add PE to each token in the input sequence

[Vaswani, 2017]

Example: Positional Encoding



Pos = 28 Pos = 71

$$\mathbf{x}_{28} + \begin{pmatrix} \vdots \\ -0.98 \\ 0.01 \\ 0.98 \\ -0.26 \\ \vdots \end{pmatrix} \qquad \mathbf{x}_{71} + \begin{pmatrix} \vdots \\ -0.98 \\ 0.01 \\ -0.89 \\ -0.26 \\ \vdots \end{pmatrix}$$

[Vaswani, 2017]

Plot from The Annotated Transformer

Promising Results

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

- Transformer provided superior results for machine translation tasks

[Vaswani, 2017]

Table from [Vaswani, 2017]

Transformer in NLP

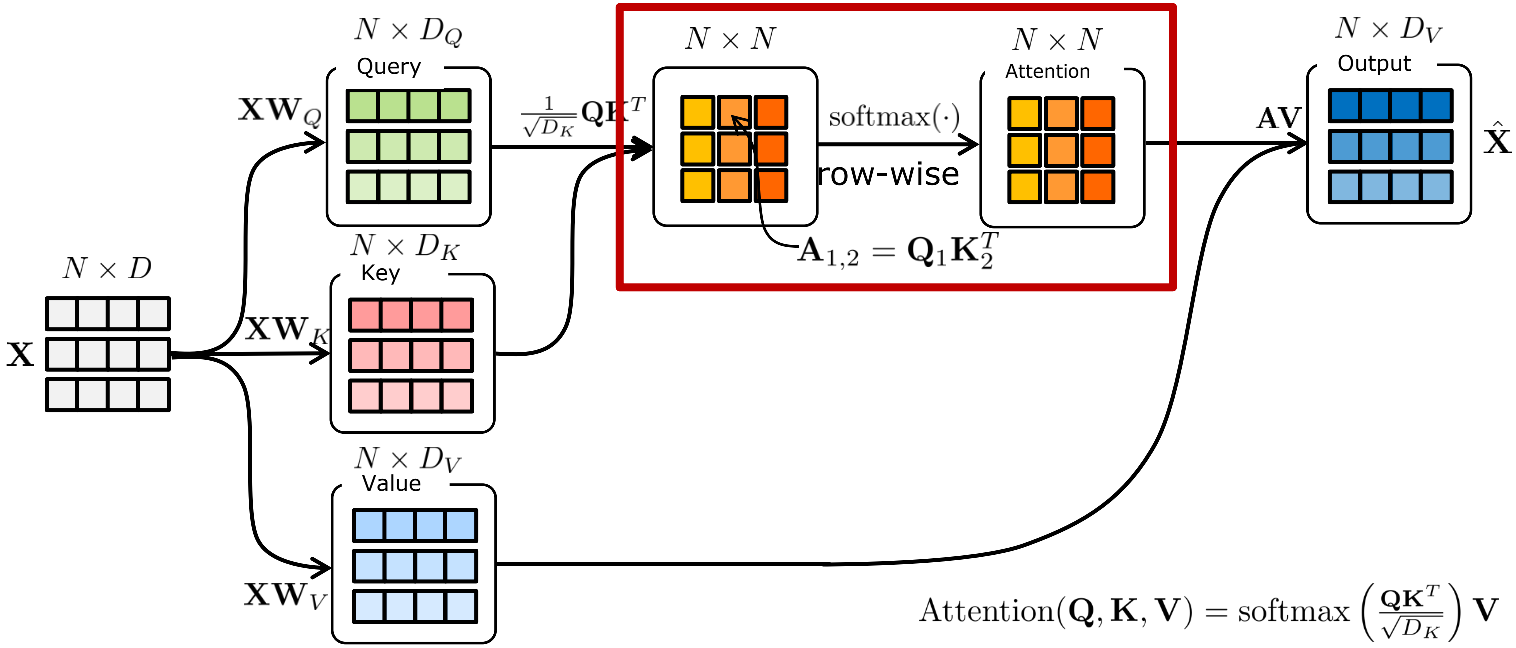
- Larger Transformer models with wide range of capabilities for different NLP tasks

- Interestingly, self-supervised pretrained Transformer models transfer well to novel tasks!

- Bigger models got only better at providing compelling results (e.g. BERT, XLNet, GPT-3)

- Can we use Transformer for images?

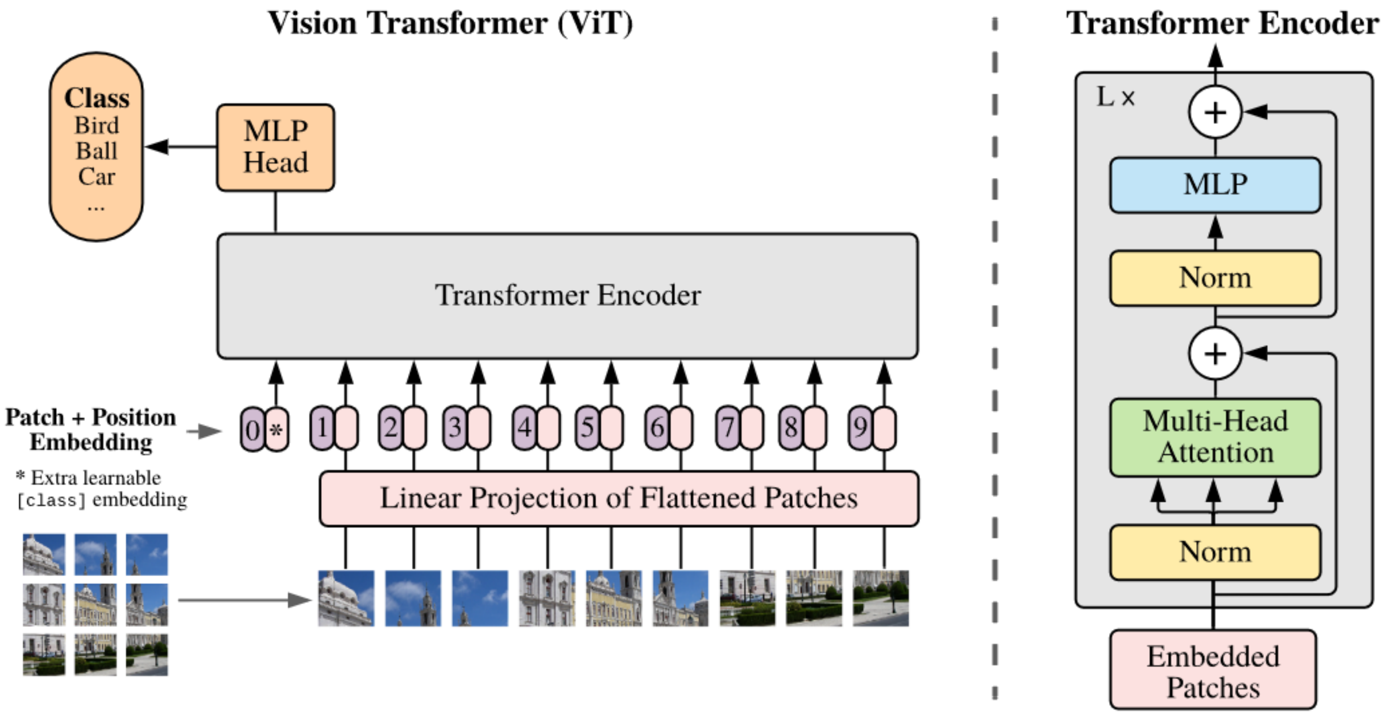
Complexity of Self Attention



- Attention weights are a $N \times N$ matrix (e.g., $O(N^2)$)
- Just taking an image as sequence of HW elements would result in N = 50,176 tokens (for 224x224 image)!

- Different way to employ Transformer for images?

Vision Transformer

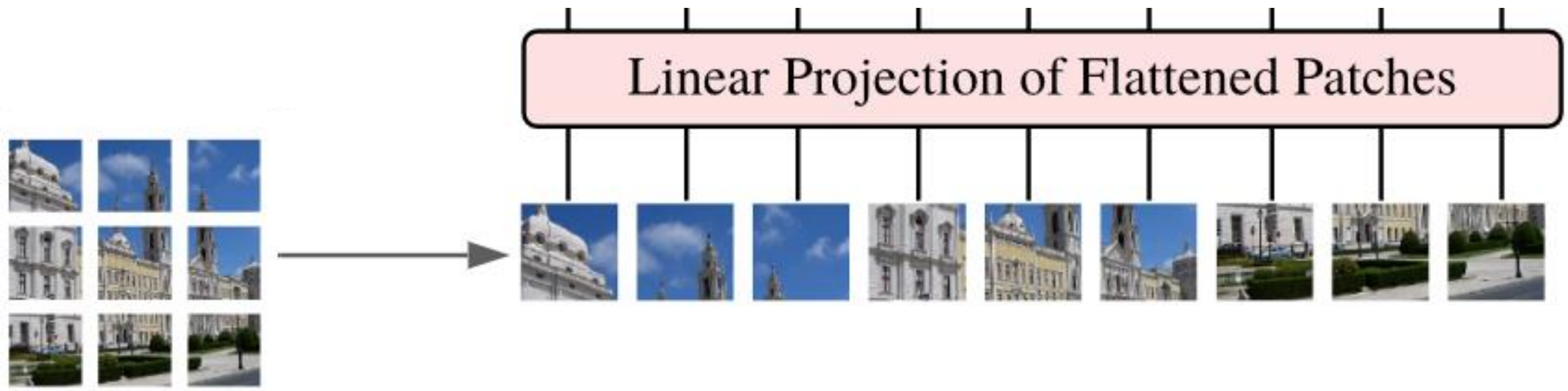


- Motivated by the success of Transformer in NLP, many works tried to use ideas for vision tasks
- Vision Transformer (ViT) achiev state-of-the-art results with minimal adjustments to the encoder

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Patches instead of Pixels

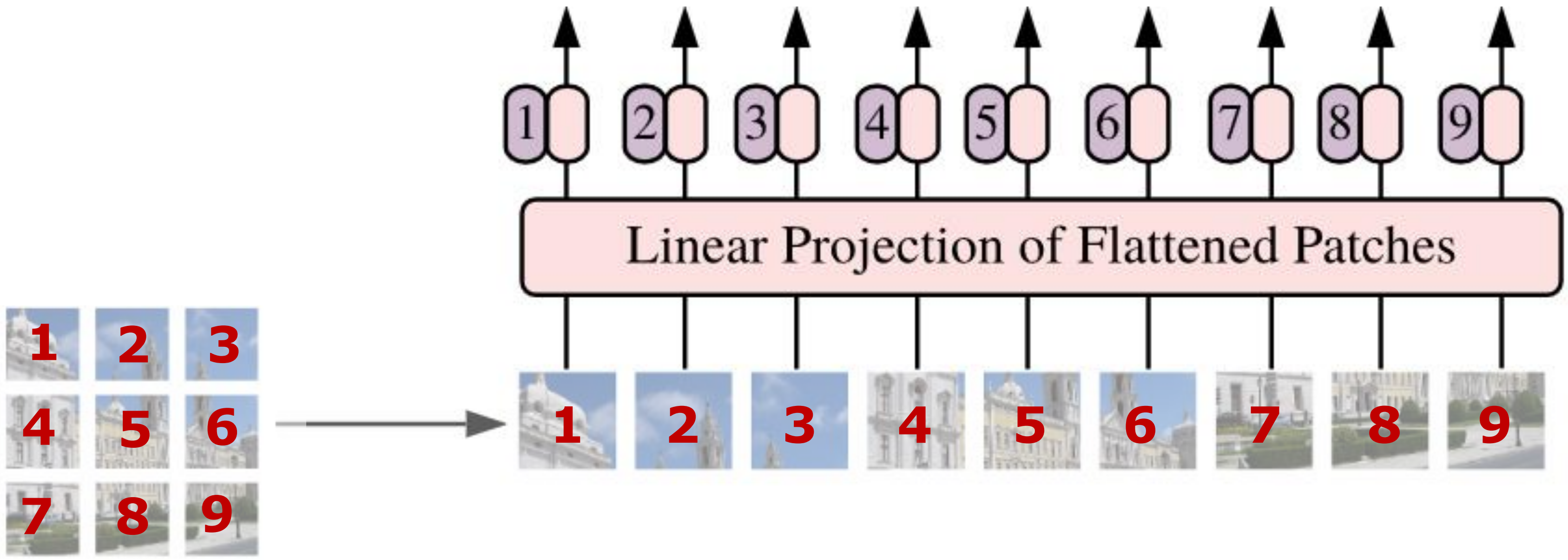


- Split image in patches of size 16×16
- Treat each image patch as $3 \cdot 16 \cdot 16$ vector and project to $D = 768/1024/1280$

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Positional Encoding

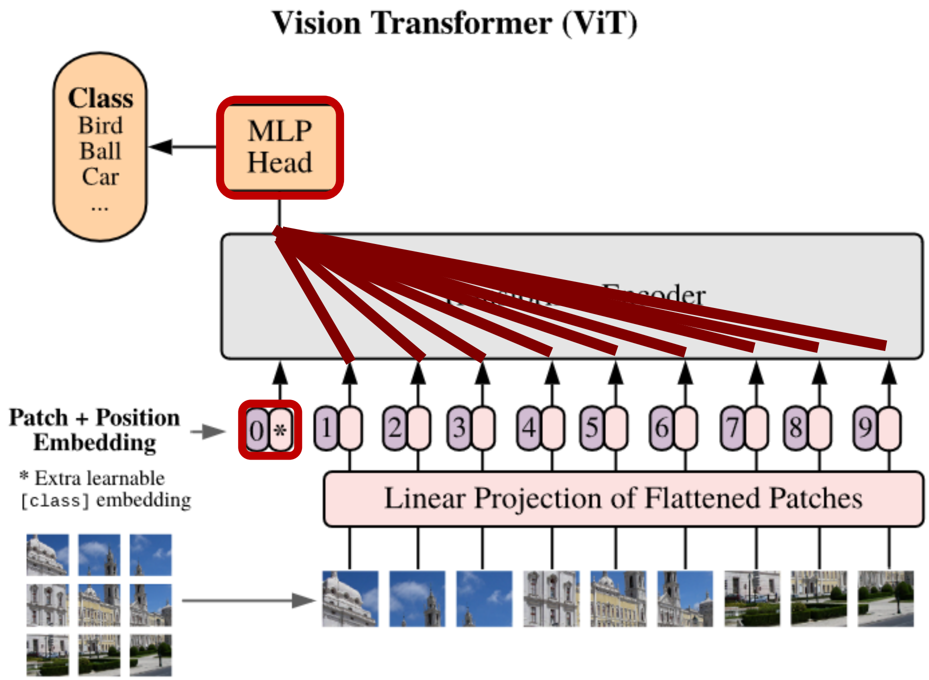


- Use 1D linear index as position with standard positional encoding

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Class Token



- Use special class token [CLS] as "aggregator" to gather information for classification
- Fully-connected layer (MLP) maps feature to classes

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Pretraining with large datasets

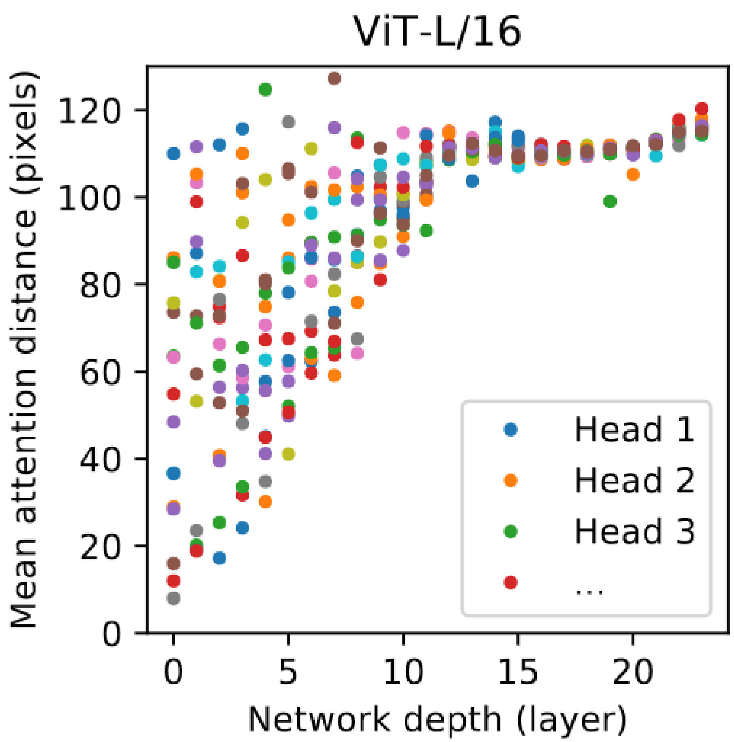
	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	—
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	—
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	—
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

- Essential for achieving state-of-the-art: pretraining with large-scale dataset → JTF dataset with 300M images for supervised pre-training
- ViT-Huge with 32 Transformer layers and 632M parameters

[Dosovitskiy, 2021]

Table from [Dosovitrskiy, 2021]

Receptive field of ViT



- Even in lower layers, attention weights cover a large range in the image
- Long-range dependencies can be exploited in early layers.

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Data-efficient training

Ablation on ↓	Pre-training	Fine-tuning	Rand-Augment	AutoAug	Mixup	CutMix	Erasing	Stoch. Depth	Repeated Aug.	Dropout	Exp. Moving Avg.	top-1 accuracy pre-trained 224^2	top-1 accuracy fine-tuned 384^2
none: DeiT-B	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✗	81.8 ±0.2	83.1 ±0.1
optimizer	SGD	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✗	74.5	77.3
	adamw	SGD	✓	✗	✓	✓	✓	✓	✓	✗	✗	81.8	83.1
data augmentation	adamw	adamw	✗	✗	✓	✓	✓	✓	✓	✗	✗	79.6	80.4
	adamw	adamw	✗	✓	✓	✓	✓	✓	✓	✗	✗	81.2	81.9
	adamw	adamw	✓	✗	✗	✓	✓	✓	✓	✗	✗	78.7	79.8
	adamw	adamw	✓	✗	✓	✗	✓	✓	✓	✗	✗	80.0	80.6
	adamw	adamw	✓	✗	✗	✗	✓	✓	✓	✗	✗	75.8	76.7
regularization	adamw	adamw	✓	✗	✓	✓	✗	✓	✓	✗	✗	4.3*	0.1
	adamw	adamw	✓	✗	✓	✓	✓	✗	✓	✗	✗	3.4*	0.1
	adamw	adamw	✓	✗	✓	✓	✓	✓	✗	✗	✗	76.5	77.4
	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✓	✗	81.3	83.1
	adamw	adamw	✓	✗	✓	✓	✓	✓	✓	✗	✓	81.9	83.1

- Essential for training with “smaller” datasets:
  1. Strong Data Augmentation: RandAugment, Mixup, Cutmix
  2. Better Regularization: Erasing, Stochastic Depth, Repeated Augmentation
- Transformers need to see more variation

[Touvron, 2021]

Table from [Touvron, 2021]

Training of Vision Transformer

How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers

Andreas Steiner*, Alexander Kolesnikov*, Xiaohua Zhai*
Ross Wightman†, Jakob Uszkoreit, Lucas Beyer*

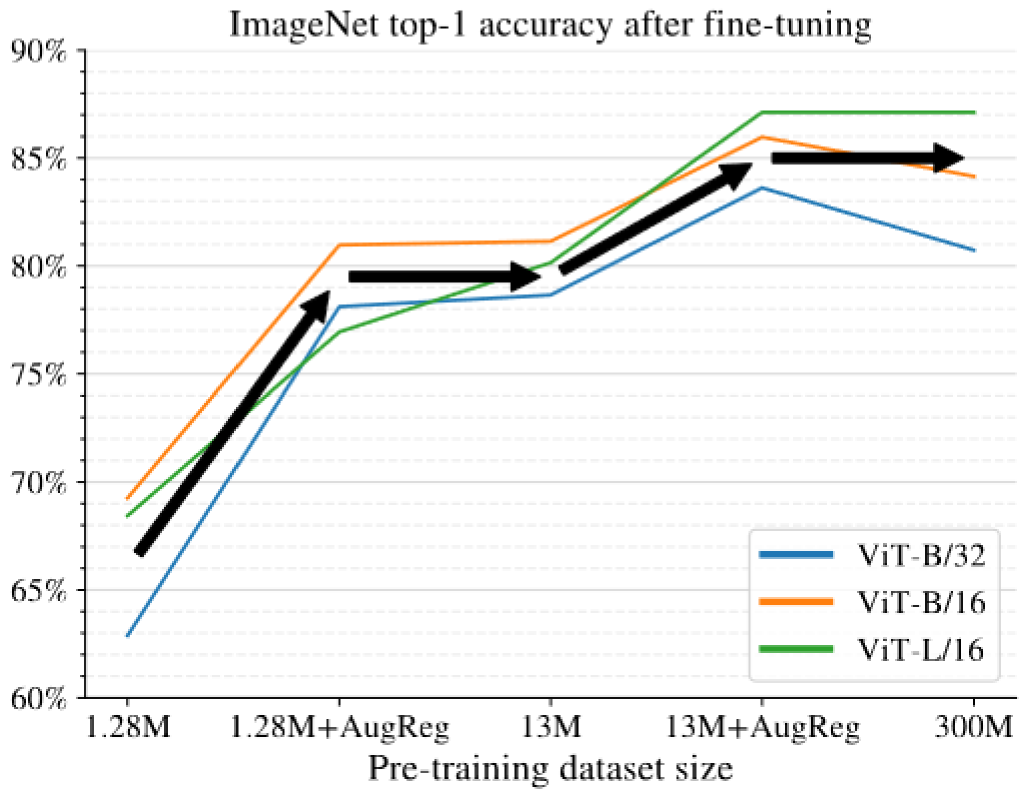
Google Research, Brain Team; †independent researcher

{andstein,akolesnikov,xzhai,usz,lbeyer}@google.com, rwightman@gmail.com

- Data Augmentation and Regularization key to achieve good performance
- Large-scale study on trade-offs between regularization, data augmentation, training data size and compute budget → over 50k experiments!

[Steiner, 2021]

AugReg vs. Pre-training size

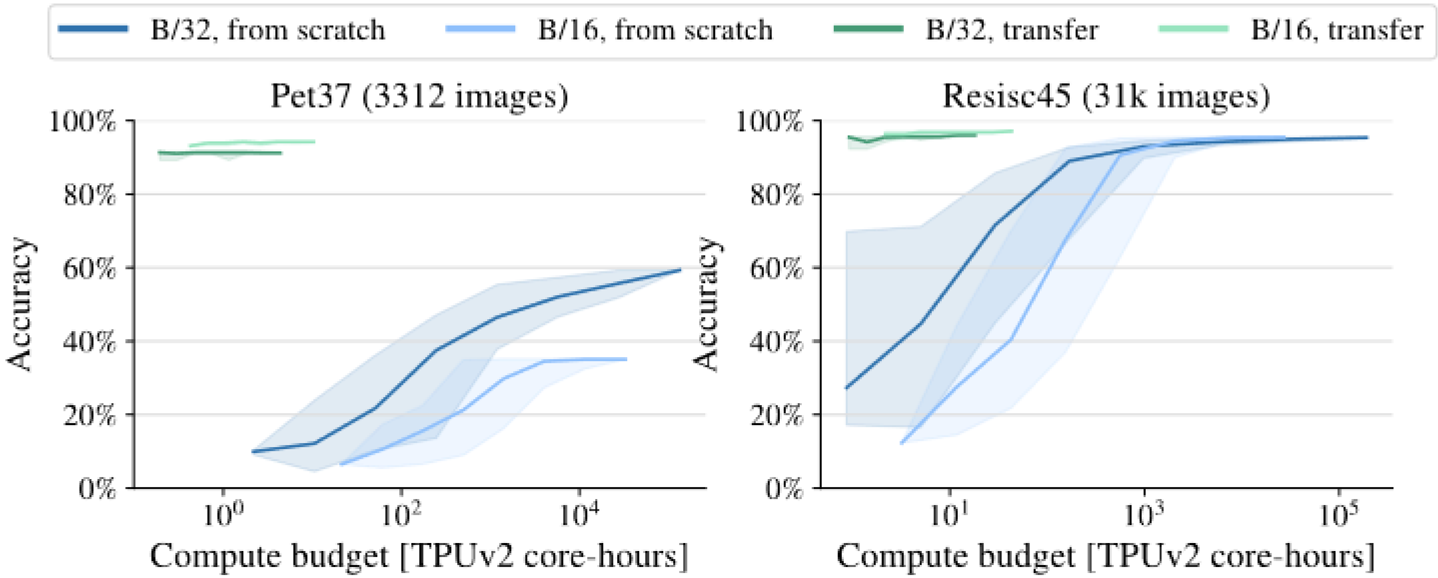


- Right amount of regularization and image augmentation leads to similar gains as increasing dataset size

[Steiner, 2021]

Figure from [Steiner, 2021]

Transfer is the better option

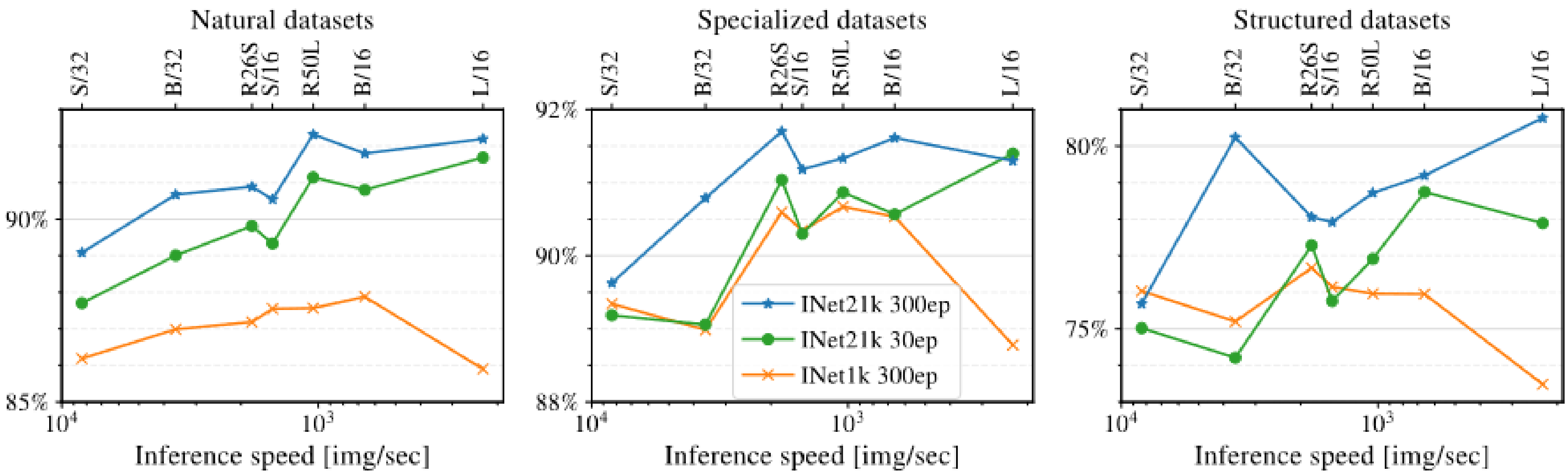


- Transfer learning leads to better performance with less compute
- **Warning:** For small datasets training from scratch will not result in models as good as transfer!

[Steiner, 2021]

Figure from [Steiner, 2021]

Better transfer with more data

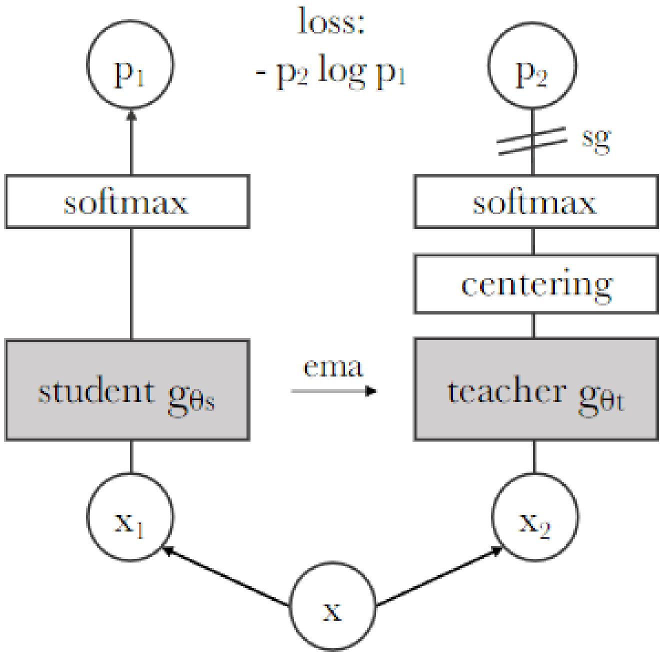


- Pretraining on more data yields more transferable models
- Again: more variations allow to “induce” inductive biases from CNNs.

[Steiner, 2021]

Figure from [Steiner, 2021]

Self-supervision for ViT



- Student and teacher have same architecture
- Student tries to replicate outputs of teacher of augmented views
- As in MoCo and BYOL, teacher parameters are updated via momentum

[Caron, 2021]

Figure from [Caron, 2021]

Results of self-supervised pretraining

- Superior performance of pre-training scheme
- Large Transformer on par or better then CNNs

Method	Arch.	Param.	im/s	Linear	k-NN
Supervised	RN50	23	1237	79.3	79.3
SCLR [12]	RN50	23	1237	69.1	60.7
MoCov2 [15]	RN50	23	1237	71.1	61.9
InfoMin [67]	RN50	23	1237	73.0	65.3
BarlowT [81]	RN50	23	1237	73.2	66.0
OBoW [27]	RN50	23	1237	73.8	61.9
BYOL [30]	RN50	23	1237	74.4	64.8
DCv2 [10]	RN50	23	1237	75.2	67.1
SwAV [10]	RN50	23	1237	**75.3**	65.7
DINO	RN50	23	1237	**75.3**	**67.5**
Supervised	ViT-S	21	1007	79.8	79.8
BYOL* [30]	ViT-S	21	1007	71.4	66.6
MoCov2* [15]	ViT-S	21	1007	72.7	64.4
SwAV* [10]	ViT-S	21	1007	73.5	66.3
DINO	ViT-S	21	1007	**77.0**	**74.5**
Comparison across architectures					
SCLR [12]	RN50w4	375	117	76.8	69.3
SwAV [10]	RN50w2	93	384	77.3	67.3
BYOL [30]	RN50w2	93	384	77.4	–
DINO	ViT-B/16	85	312	78.2	76.1
SwAV [10]	RN50w5	586	76	78.5	67.1
BYOL [30]	RN50w4	375	117	78.6	–
BYOL [30]	RN200w2	250	123	79.6	73.9
DINO	ViT-S/8	21	180	79.7	**78.3**
SCLRv2 [13]	RN152w3+SK	794	46	79.8	73.1
DINO	ViT-B/8	85	63	**80.1**	77.4

[Caron, 2021]

Table from [Caron, 2021]

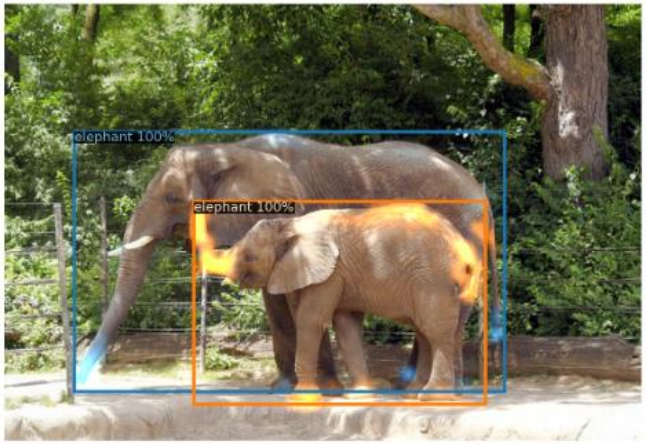
Emerging Properties of ViT

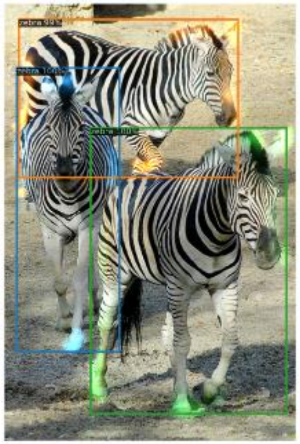
- Interestingly, self-supervised training leads to class-specific features
- Visualization of attention from [CLS] token leads to unsupervised object segmentation

[Caron, 2021]

Figure from [Caron, 2021]

Transformer for other Vision Tasks

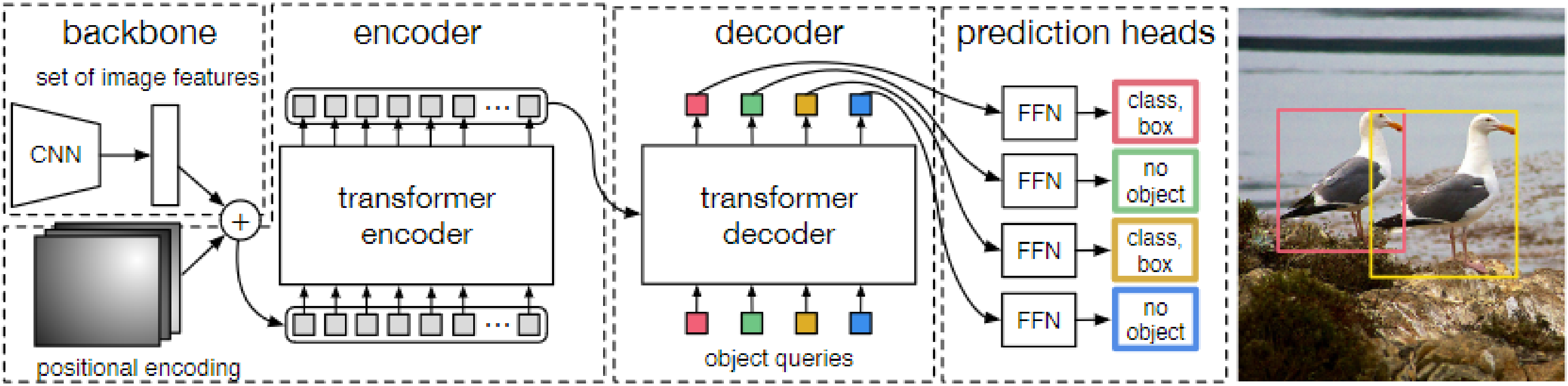





- Results on image classification motivated investigation of other vision tasks
- Here two examples: Object Detection and Semantic Segmentation

Images from [Carion, 2020] and [Zheng, 2021]

Transformer for Detection



- DETR uses Transformer encoder and decoder to generate object detections
- Predictions head produce N object/no object predictions
- No non-maximum suppression needed!

[Carion, 2020]

Figure from [Carion, 2020]

Results of DETR

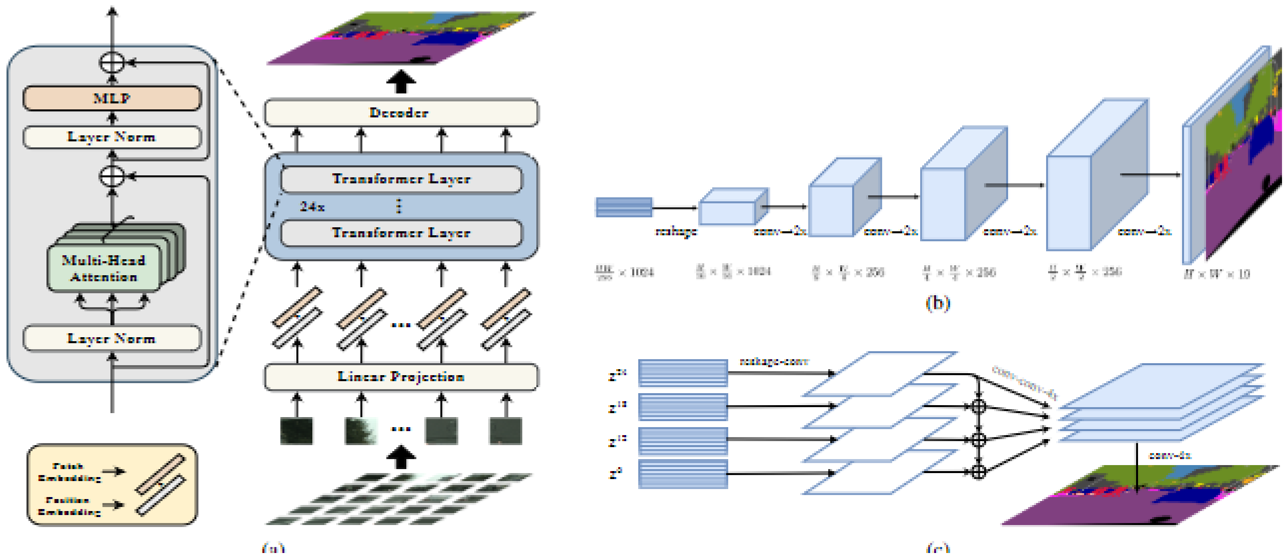
Model	GFLOPS/FPS	#params	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	62.1	45.5	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	63.9	**47.8**	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	187/12	41M	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	**64.7**	47.7	23.7	**49.5**	**62.3**

- Highly competitive results for object detection on COCO

[Carion, 2020]

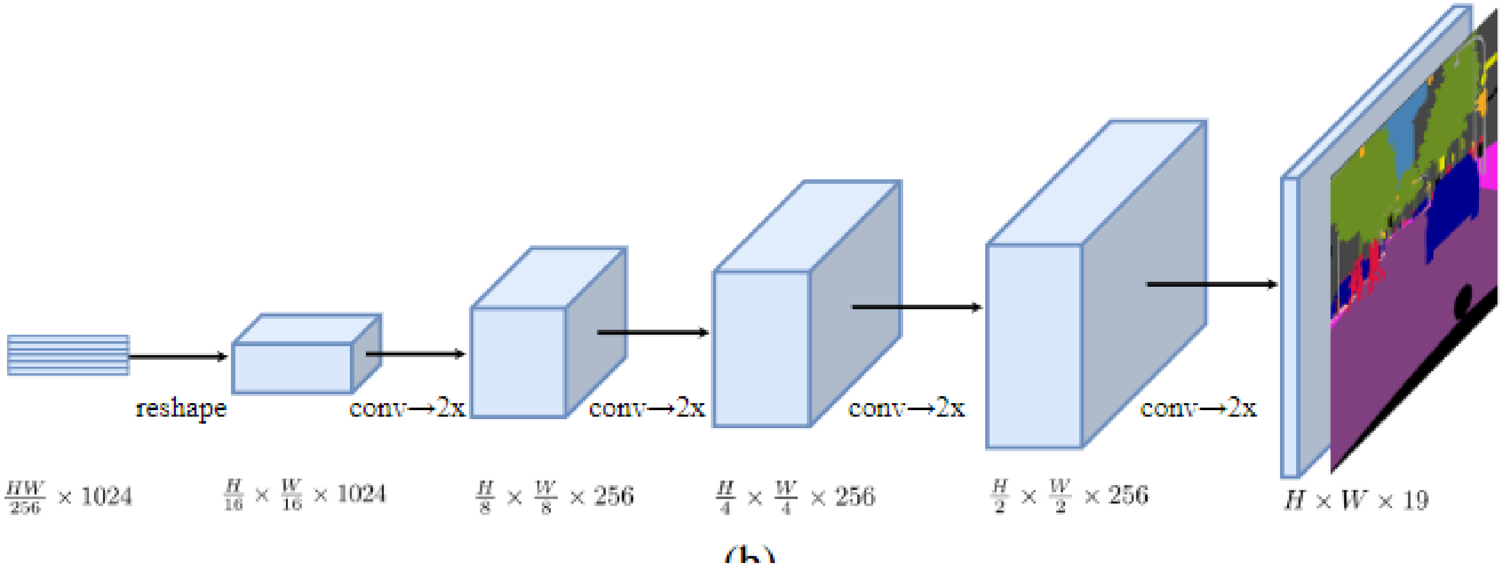
Table from [Carion, 2021]

Transformer for Segmentation



- **Se**gmentation **Tr**ansformer (SETR) uses patch-wise encoder to extract patch features
- Investigates two decoders to upsample patch features

Progressive Upsampling in SETR

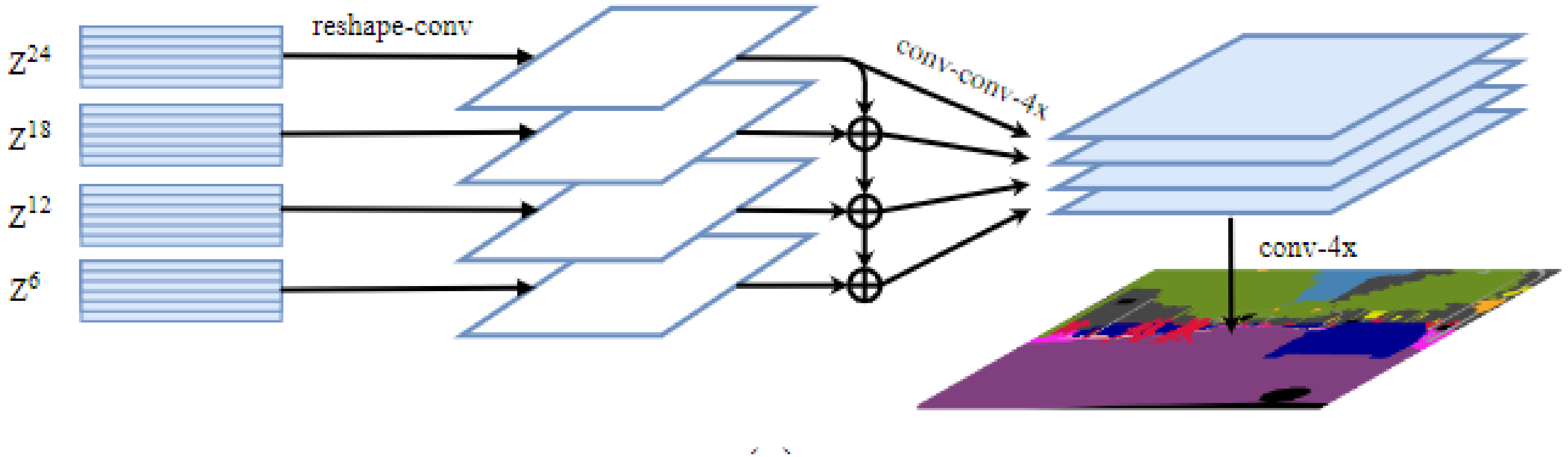


- Upsample 16x16 patch features to full resolution via convolutions and bilinear upsampling

[Zheng, 2021]

Figure from [Zheng, 2021]

Multi-level Feature Aggregation



- Use patch features from different Transformer layer
- Convolutional combination of upsampled feature maps

[Zheng, 2021]

Figure from [Zheng, 2021]

Results of SETR

Method	Backbone	mIoU	Pixel Acc.
FCN (16, 160k, SS) [39]	ResNet-101	39.91	79.52
FCN (16, 160k, MS) [39]	ResNet-101	41.40	80.65
EncNet [54]	ResNet-101	44.65	81.69
PSPNet [59]	ResNet-269	44.94	81.69
DMNet [18]	ResNet-101	45.50	-
CCNet [25]	ResNet-101	45.22	-
Strip pooling [23]	ResNet-101	45.60	82.09
APCNet [19]	ResNet-101	45.38	-
OCNet [53]	ResNet-101	45.45	-
SETR-*Naïve* (16, 160k, SS)	T-Large	48.06	82.40
SETR-*Naïve* (16, 160k, MS)	T-Large	48.80	82.92
SETR-*PUP* (16, 160k, SS)	T-Large	48.58	82.90
SETR-*PUP* (16, 160k, MS)	T-Large	50.09	**83.58**
SETR-*MLA* (16, 160k, SS)	T-Large	48.64	82.64
SETR-*MLA* (16, 160k, MS)	T-Large	**50.28**	83.46

ADE20K

Method	Backbone	mIoU
PSPNet [59]	ResNet-101	78.40
DenseASPP [49]	DenseNet-161	80.60
BiSeNet [51]	ResNet-101	78.90
PSANet [60]	ResNet-101	80.10
DANet [17]	ResNet-101	81.50
OCNet [53]	ResNet-101	80.10
CCNet [25]	ResNet-101	81.90
Axial-DeepLab-L [47]	Axial-ResNet-L	79.50
Axial-DeepLab-XL [47]	Axial-ResNet-XL	79.90
SETR-*PUP* (100k)	T-Large	81.08
SETR-*PUP*‡	T-Large	81.64

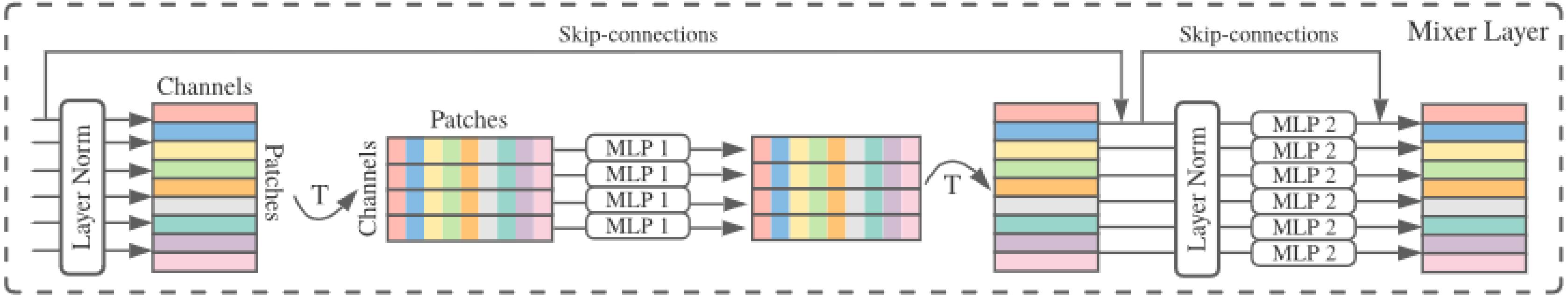
Cityscapes

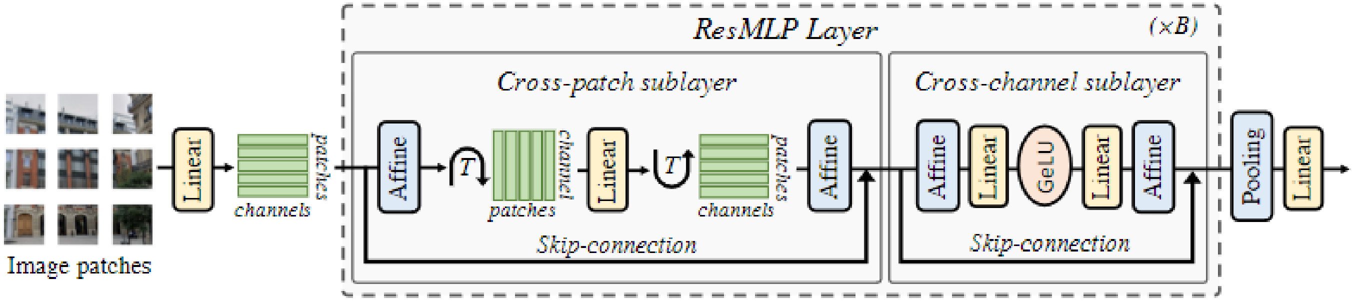
- Strong results on ADE20K and Cityscapes

[Zheng, 2021]

Tables from [Zheng, 2021]

Self Attention Needed?

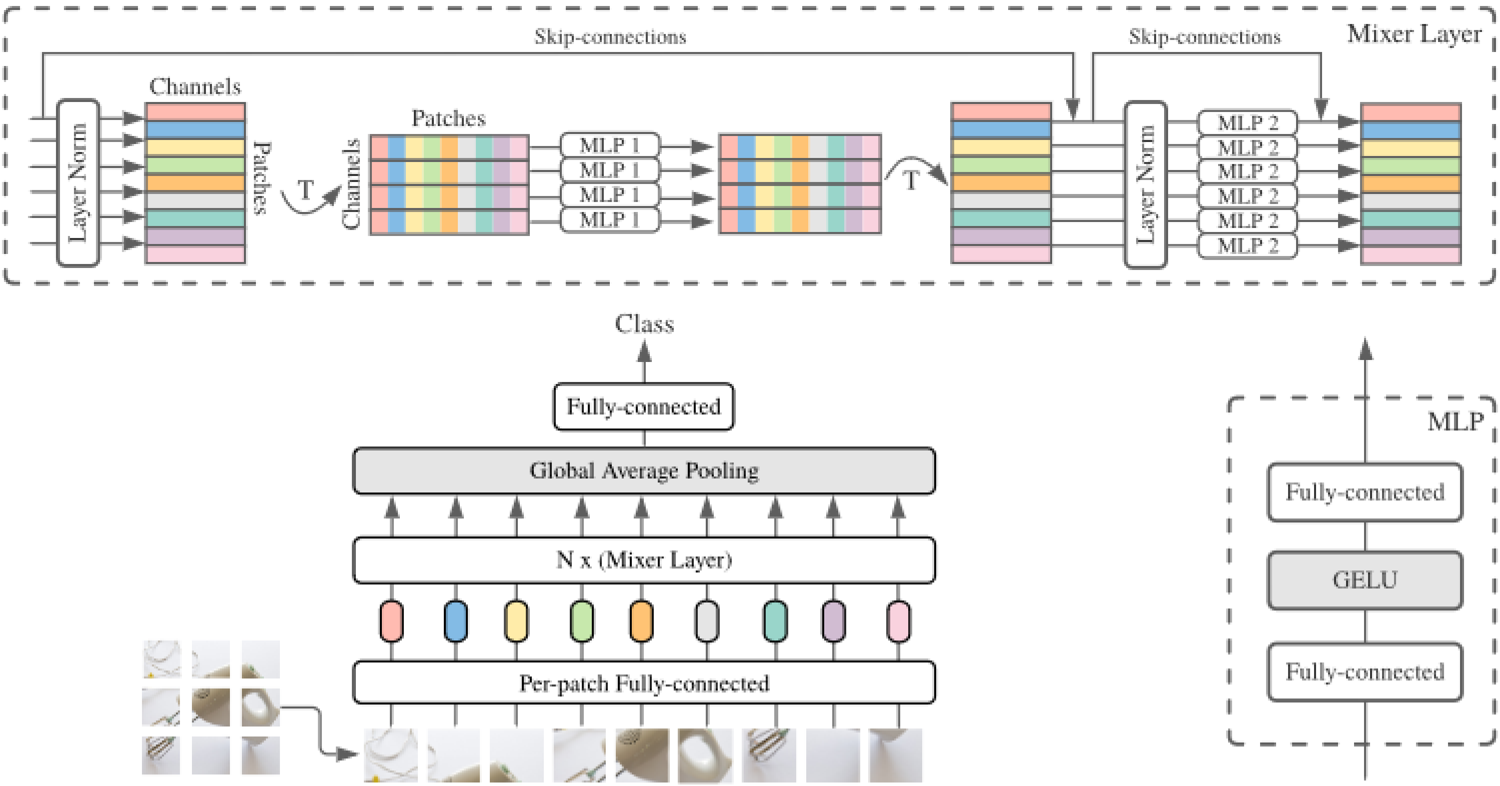




- Another line of research investigated to replace self-attention with MLPs

[Tolstinkhin, 2021] [Touvron, 2021] [Melas-Kyriazi, 2021]

MLP-Mixer



- Replace self-attention with MLP on transposed feature vectors
- All operations are MLPs on image patches

[Tolstinkhin, 2021]

Figure from [Tolstinkhin, 2021]

Results of MLP Mixer

	ImNet top-1	ReaL top-1	Avg 5 top-1	VTAB-1k 19 tasks	Throughput img/sec/core	TPUv3 core-days
Pre-trained on ImageNet-21k (public)						
● HaloNet [51]	85.8	—	—	—	120	0.10k
● Mixer-L/16	84.15	87.86	93.91	74.95	105	0.41k
● ViT-L/16 [14]	85.30	88.62	94.39	72.72	32	0.18k
● BiT-R152x4 [22]	85.39	—	94.04	70.64	26	0.94k
Pre-trained on JFT-300M (proprietary)						
● NFNet-F4+ [7]	89.2	—	—	—	46	1.86k
● Mixer-H/14	87.94	90.18	95.71	75.33	40	1.01k
● BiT-R152x4 [22]	87.54	90.54	95.33	76.29	26	9.90k
● ViT-H/14 [14]	88.55	90.72	95.97	77.63	15	2.30k
Pre-trained on unlabelled or weakly labelled data (proprietary)						
● MPL [34]	90.0	91.12	—	—	—	20.48k
● ALIGN [21]	88.64	—	—	79.99	15	14.82k

- Slightly worse results then competing Vision Transformers

[Tolstinkhin, 2021]

Table from [Tolstinkhin, 2021]

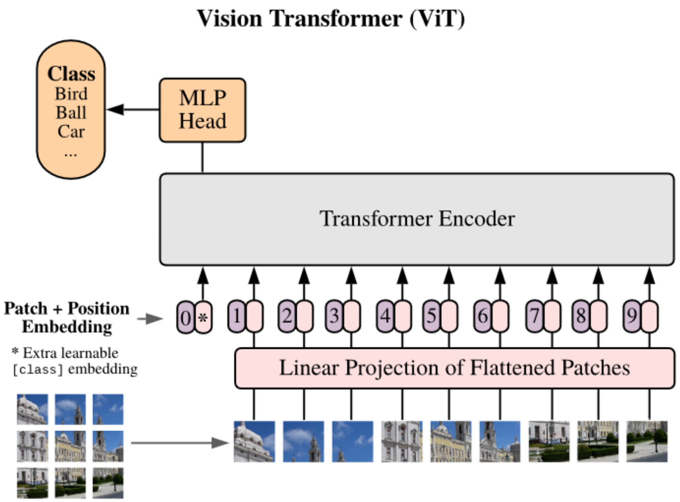
Outlook

- Highly active research area
- Combination of CNNs (early layers) and Transformer shows promising results

- Other directions in Transformer research:
  - Deeper Transformer architectures (e.g. CaiT)
  - Reduce cost of self-attention (e.g. Perceiver)
  - Hierarchical Vision Transformer (e.g. PVT)
  - Better decoder for segmentation(e.g., SegFormer)

[Xiao, 2021][Touvron, 2021][Jaegle, 2021][Wang, 2021][Xie, 2021]

Summary



- Success of Transformer in NLP motivated investigation for vision tasks
- Transformer have less inductive bias and produce promising results
- Paradigm shift for vision tasks?

See you next year!

References

- Chen et al. An Empirical Study of Training Self-supervised Vision Transformers, arxiv, 2021.
- Carion et al. End-to-End Object Detection with Transformers, ECCV, 2020.
- Caron et al. Emerging Properties in Self-Supervised Vision Transformers, arxiv, 2021.
- Dosovitskiy et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. ICLR, 2021.
- Jaegle et al. Perceiver: General Perception with Iterative Attention, ICML, 2021.
- Melas-Kyriazi. Do You Even Need Attention? A Stack of Feed-Forward Layers Does Surprisingly Well on ImageNet, arxiv, 2021.
- Steiner et al. How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers, arxiv, 2021.
- Tolstikhin et al. MLP-Mixer: An all-MLP Architecture for Vision, arxiv, 2021.
- Touvron et al. Training data-efficient image transformers & distillation through attention, ICML, 2021.
- Touvron et al. ResMLP: Feedforward networks for image classification with data-efficient training, arxiv, 2021.
- Touvron et al. Going deepter with Image Transformers, arxiv, 2021.
- Vaswani et al. Attention is all you need. NeurIPS, 2017.
- Wang et al. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions, arxiv, 2021.
- Xiao et al. Early Convolutions Help Transformers See Better, arxiv, 2021.
- Xie et al. SegFormer: Simple and Efficient Design for Segmentation with Transformers, arxiv, 2021.
- Zheng et al. Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers, arxiv, 2021.