Photogrammetry & Robotics Lab

Machine Learning for Robotics and Computer Vision Tutorial

Transformer

Jens Behley

Pre-exam Q&A

- We offer an additional Q&A session:
  August 12, 2021@10:00-12:00

- Send us questions before the session, we will then discuss questions in the Q&A session
- (But will also answer ad hoc questions)

- All lectures & exercises relevant for the exam (invited talks are not relevant!)

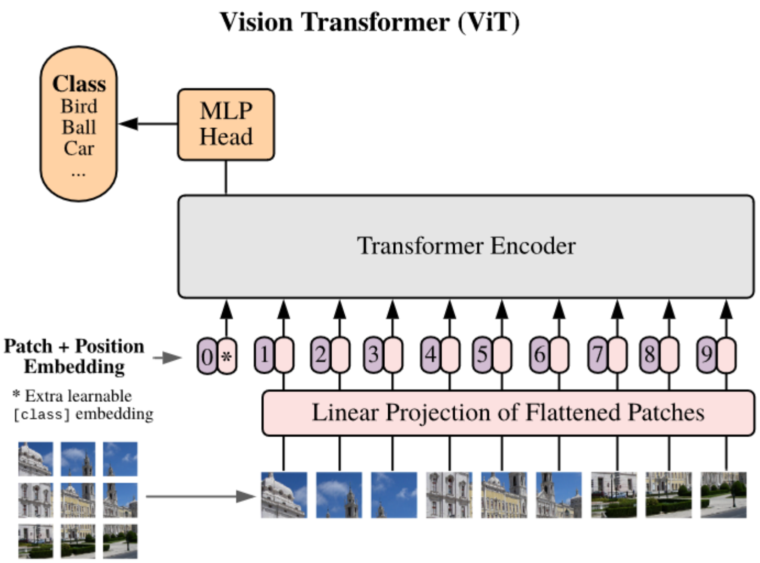
Exam Dates

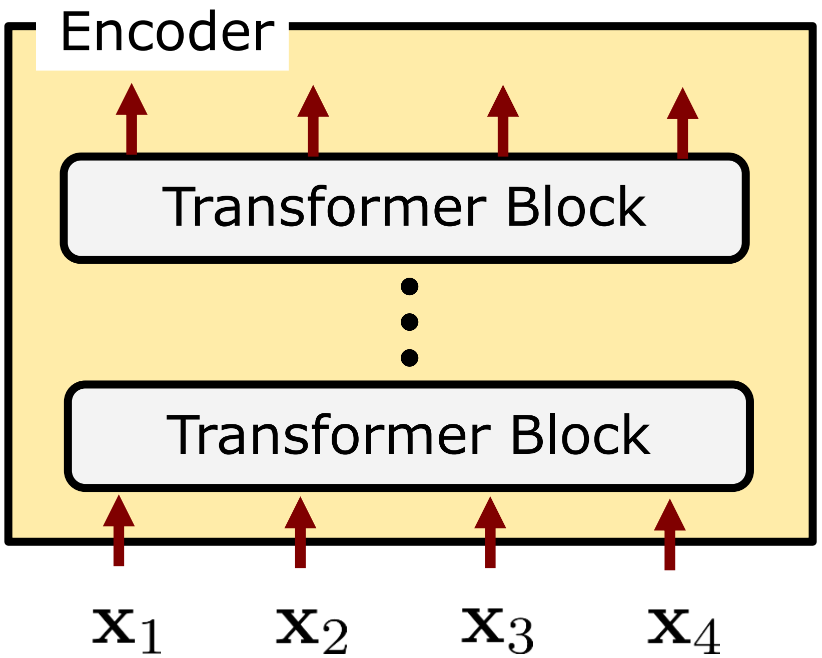
- Oral Exam via Zoom in English
- Webcam must be on all the time and alone in room
- No other windows besides Zoom open.

- Date from the voting: **Wed, 25.08.2021**

- If this date still doesn’t fit, contact us and we provide one alternative date

This week's lecture

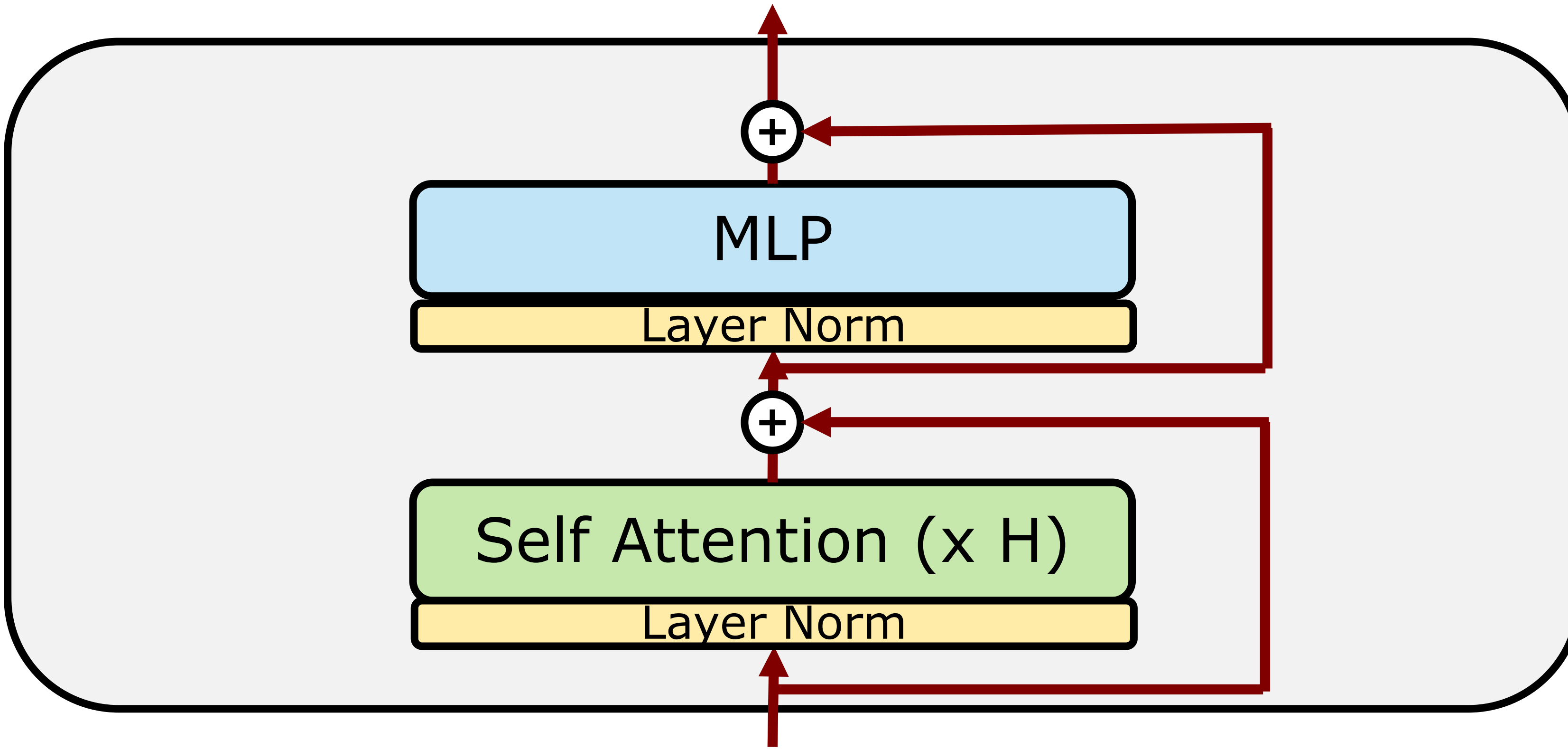




- Going beyond convolutions with Transformers
- Key building block: **Self-Attention**
- Promising results on various vision tasks

- Hot topic in computer vision & robotics

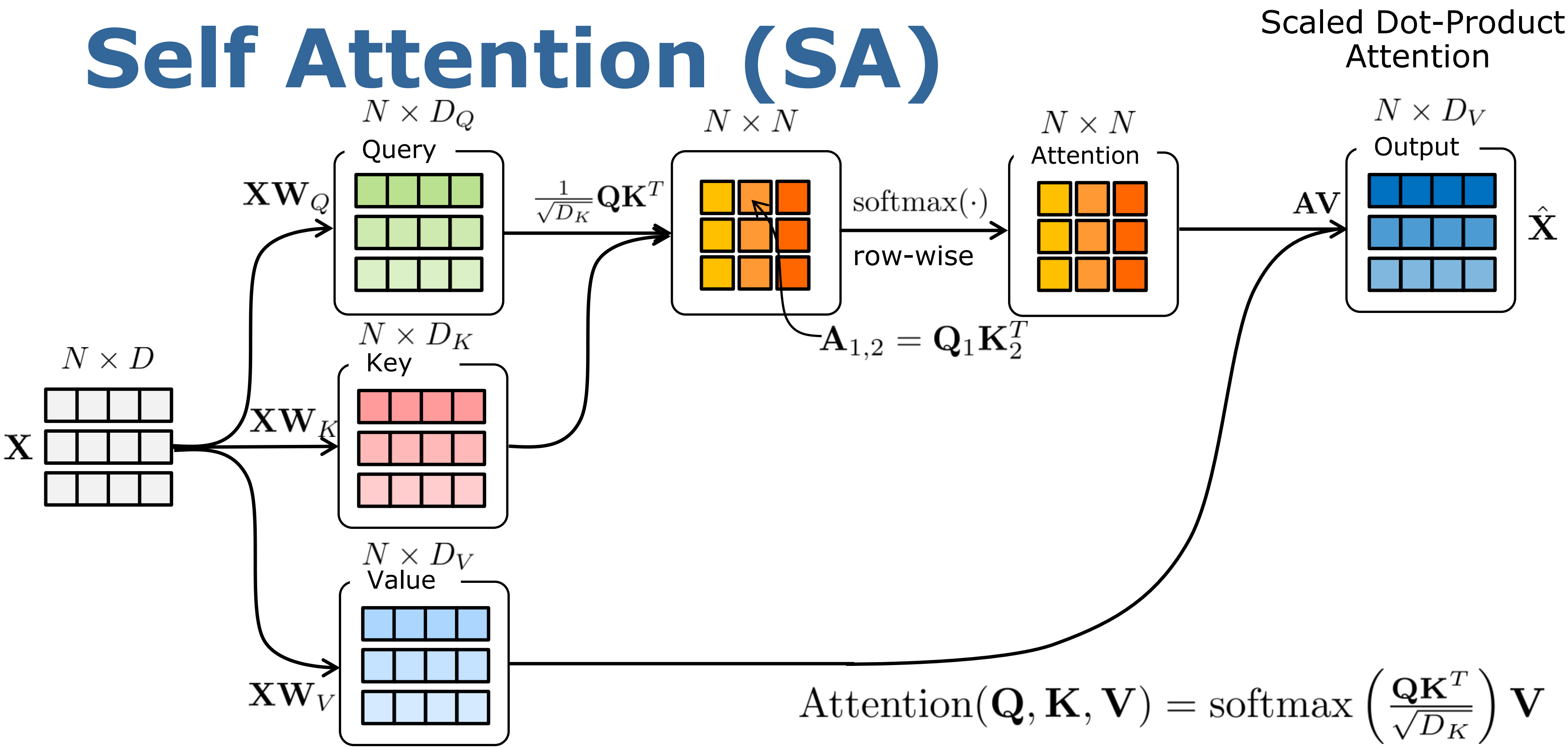
Transformer Block



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

[Vaswani, 2017]

Self Attention (SA)

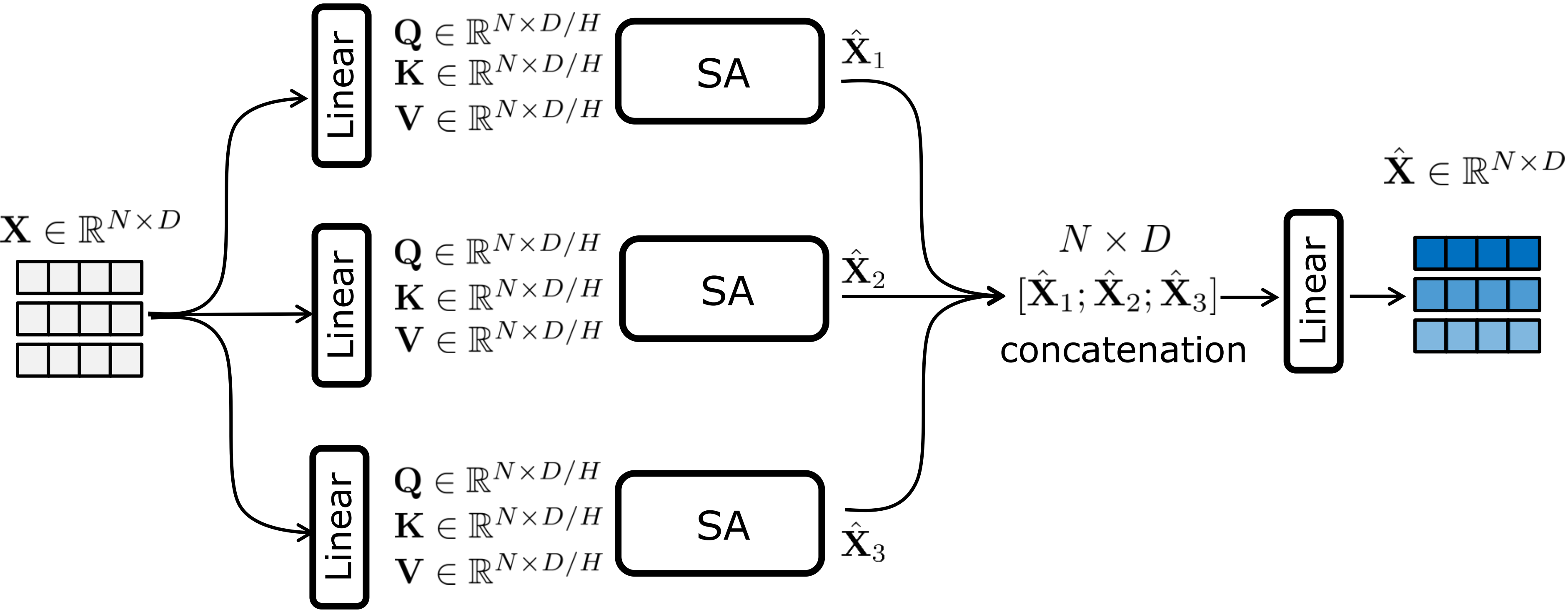


$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{D_K}}\right)\mathbf{V}$$

- 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

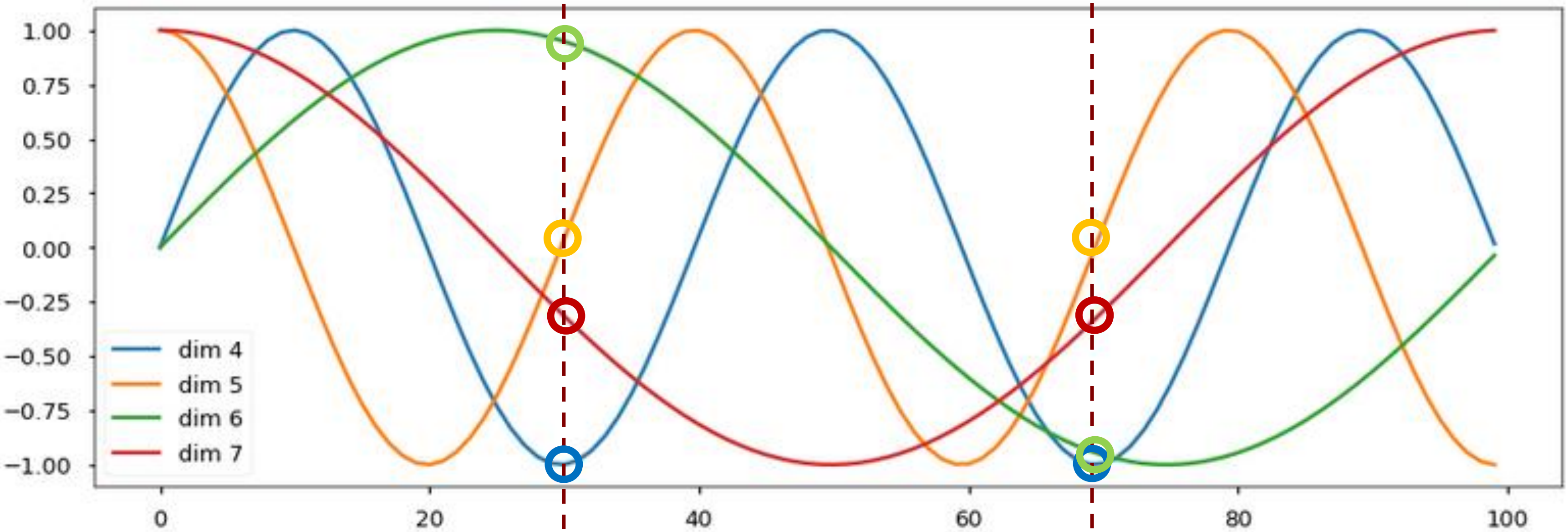
[Vaswani, 2017]

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)

Example: Positional Encoding



Pos = 28

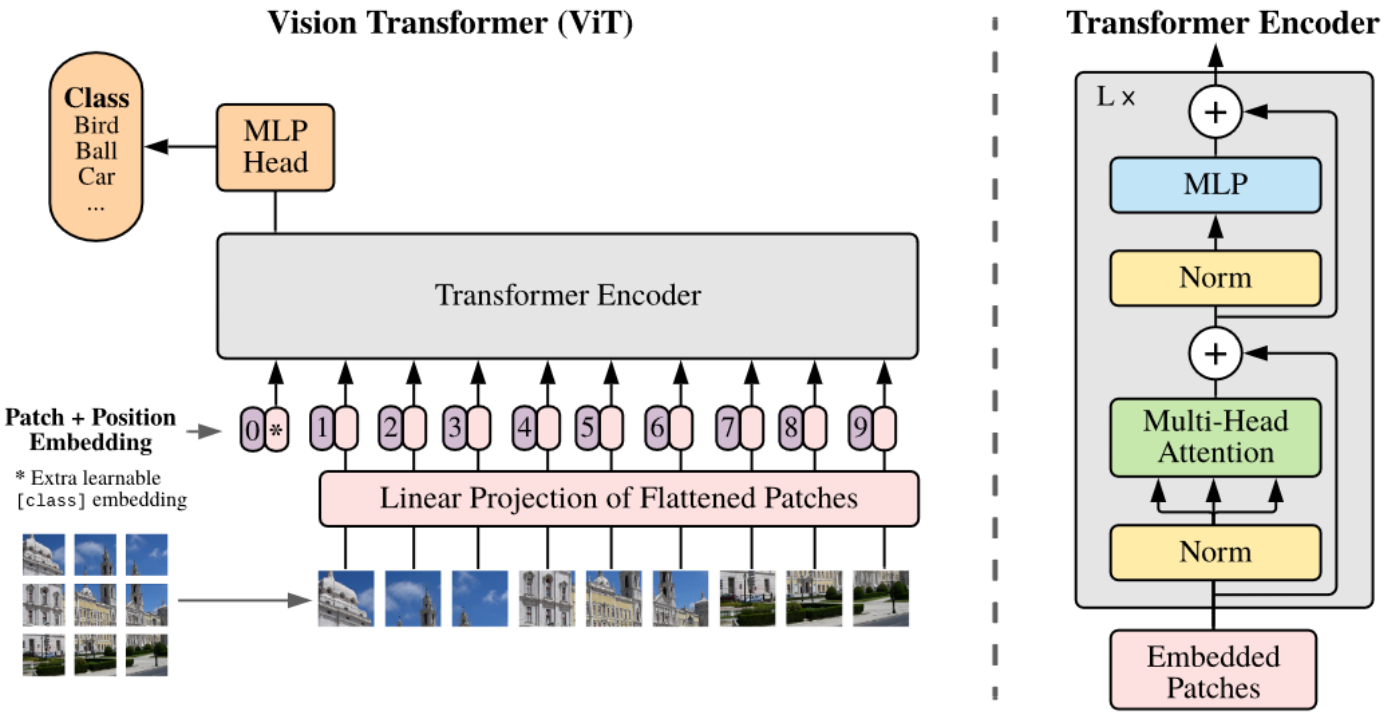
Pos = 71

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

[Vaswani, 2017]

Plot from The Annotated Transformer

Vision Transformer

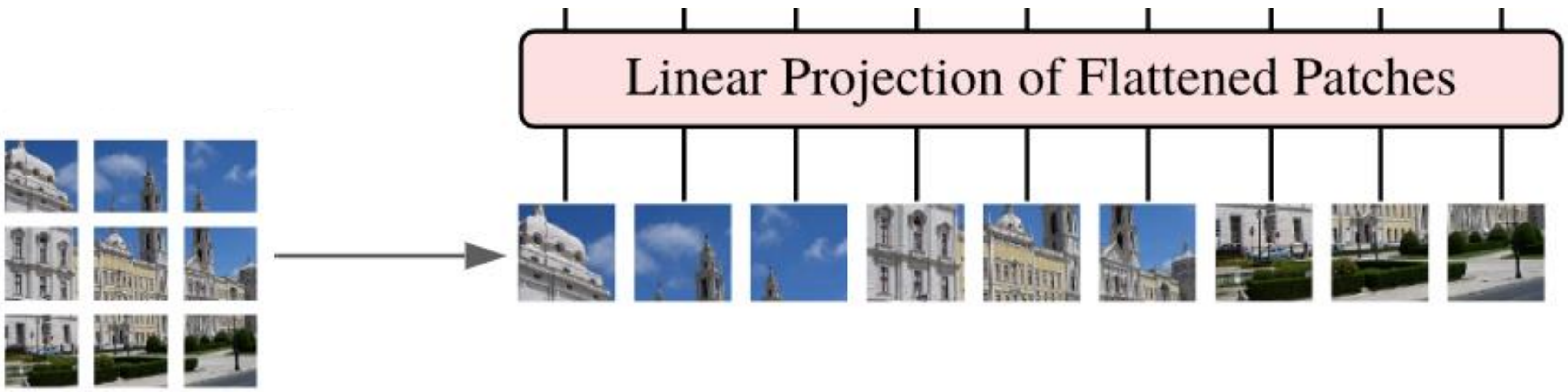


- Motivated by the success of Transformer in NLP, many works tried to use ideas for vision tasks
- Vision Transformer (ViT) achiev state-of-the-art results with minimal adjustments to the encoder

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Patches instead of Pixels

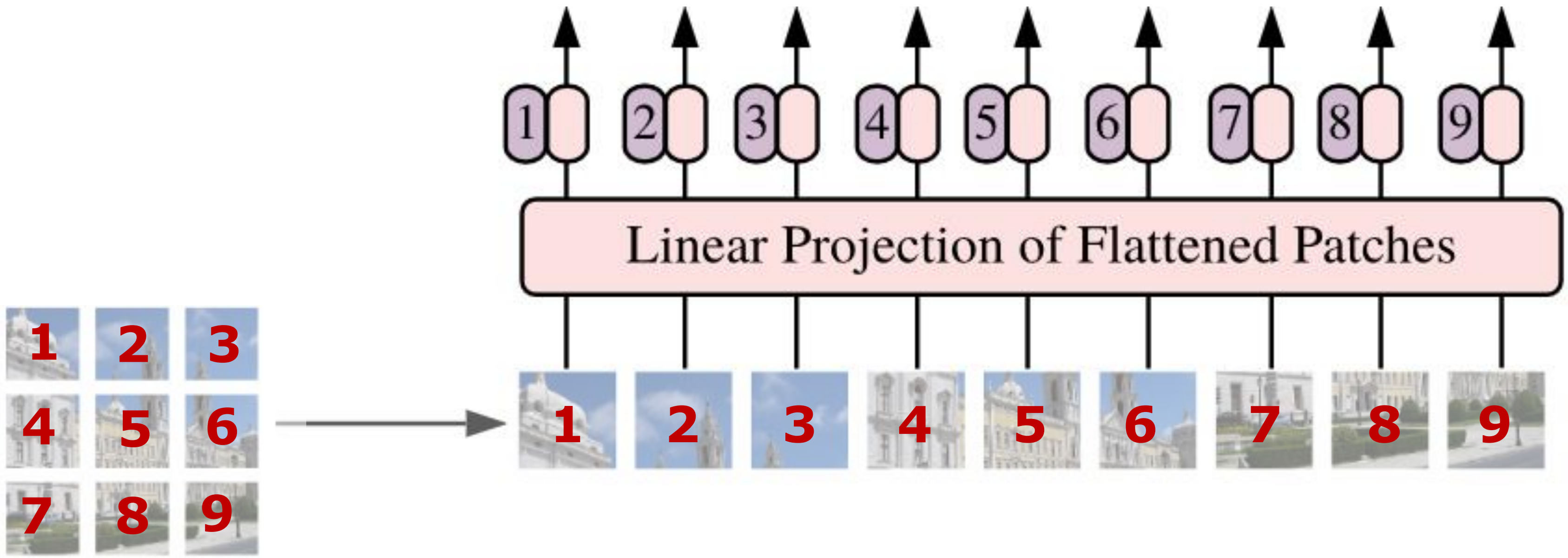


- Split image in patches of size 16×16
- Treat each image patch as $3 \cdot 16 \cdot 16$ vector and project to $D = 768/1024/1280$

[Dosovitskiy, 2021]

Figure from [Dosovitrskiy, 2021]

Positional Encoding

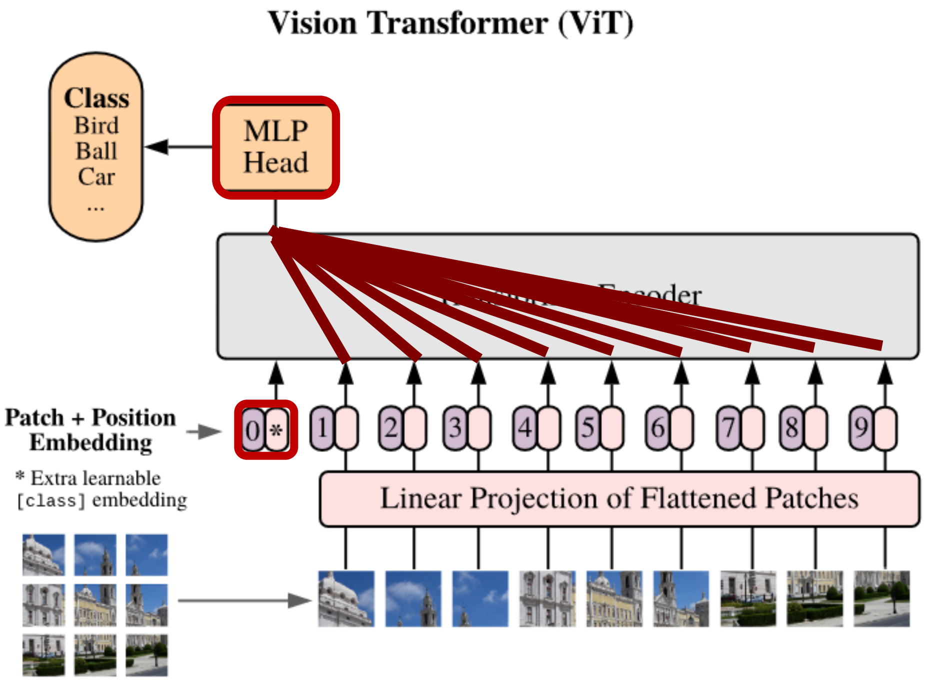


- Use 1D linear index as position with standard positional encoding

[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

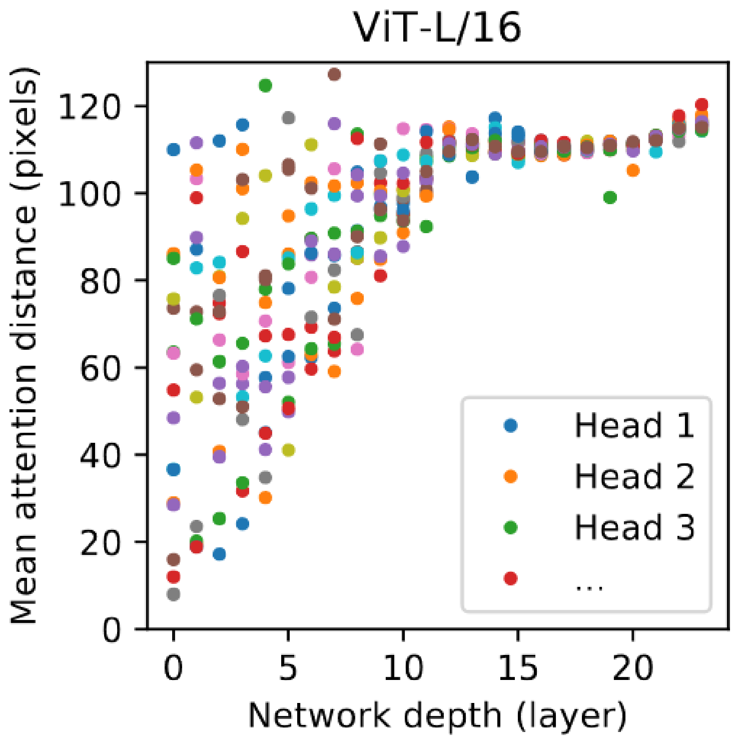
	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 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]

Training of Vision Transformer

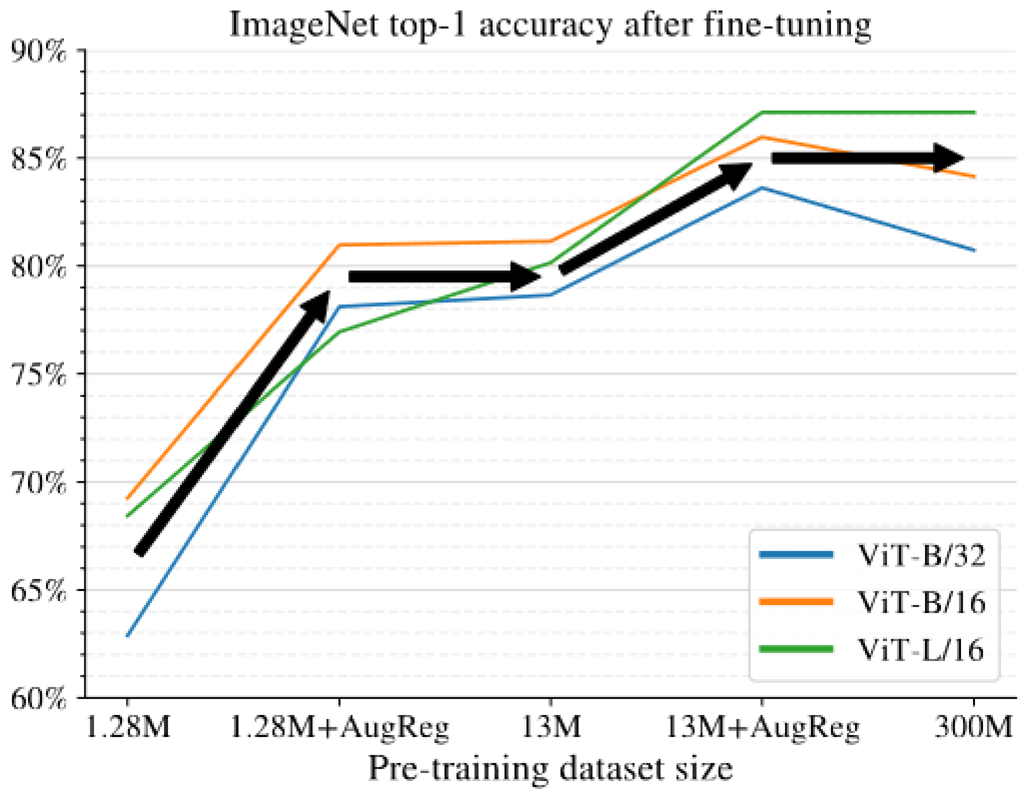

How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers

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

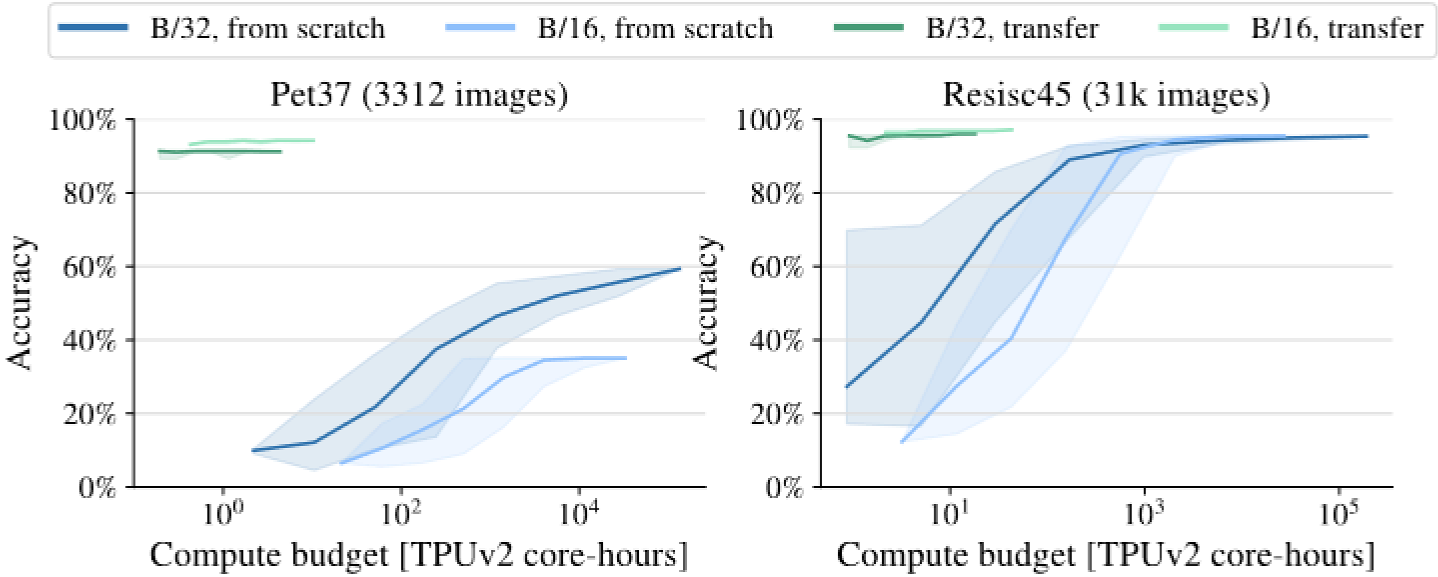


- Right amount of regularization and image augmentation leads to similar gains as increasing dataset size

[Steiner, 2021]

Figure from [Steiner, 2021]

Transfer is the better option



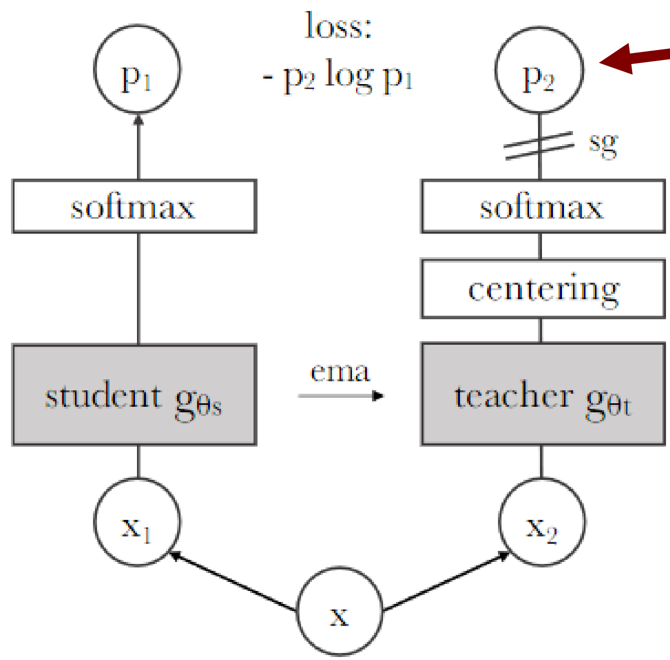
- Transfer learning leads to better performance with less compute
- **Warning:** For small datasets training from scratch will not result in models as good as transfer!

[Steiner, 2021]

Figure from [Steiner, 2021]

Self-supervision for ViT

K=65536



$$c \leftarrow mc + (1 - m)\frac{1}{B}\sum_{i=1}^{B} g_{\theta_t}(x_i),$$

- 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]

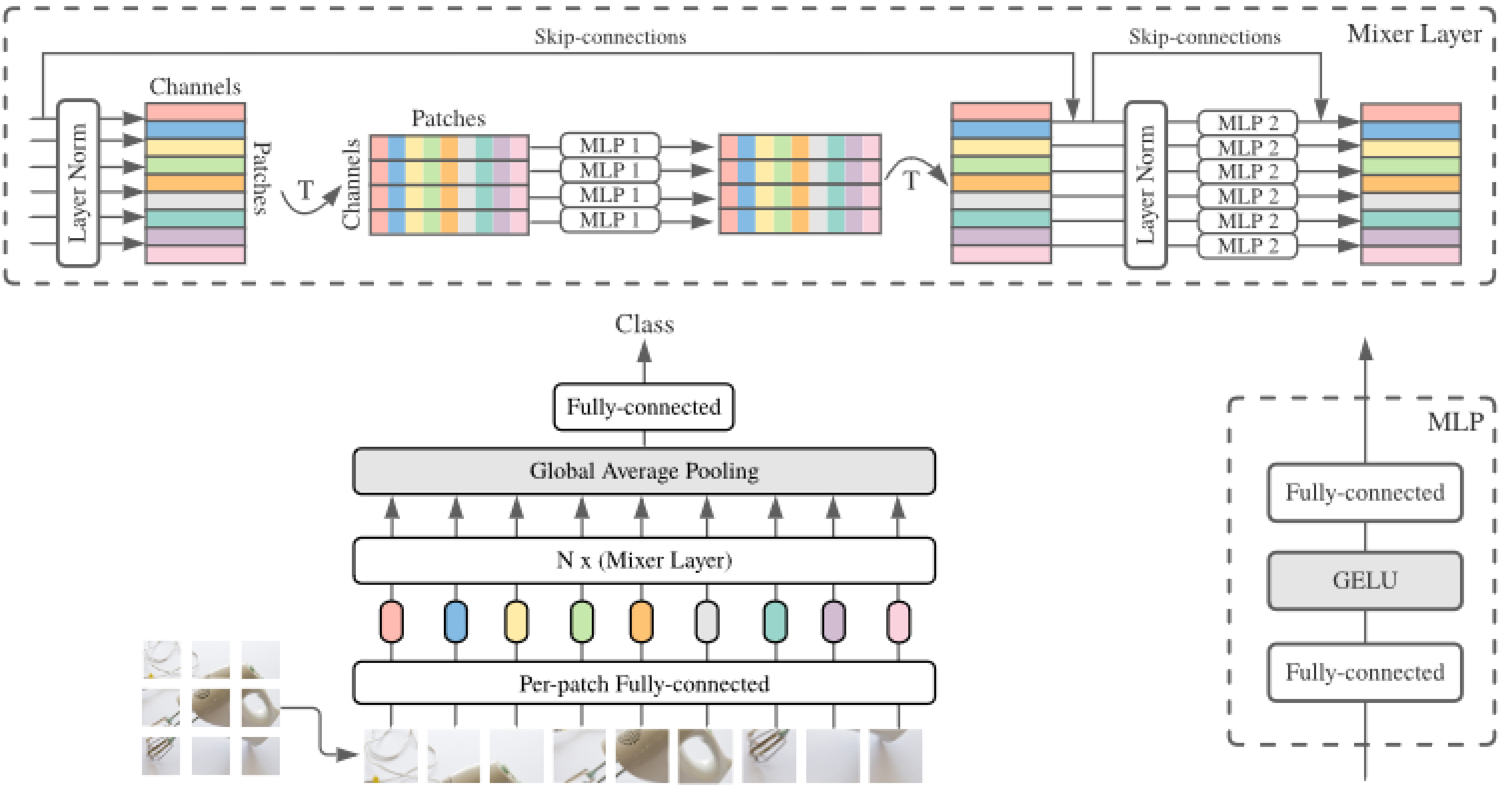
Emerging Properties of ViT

- 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]

MLP-Mixer

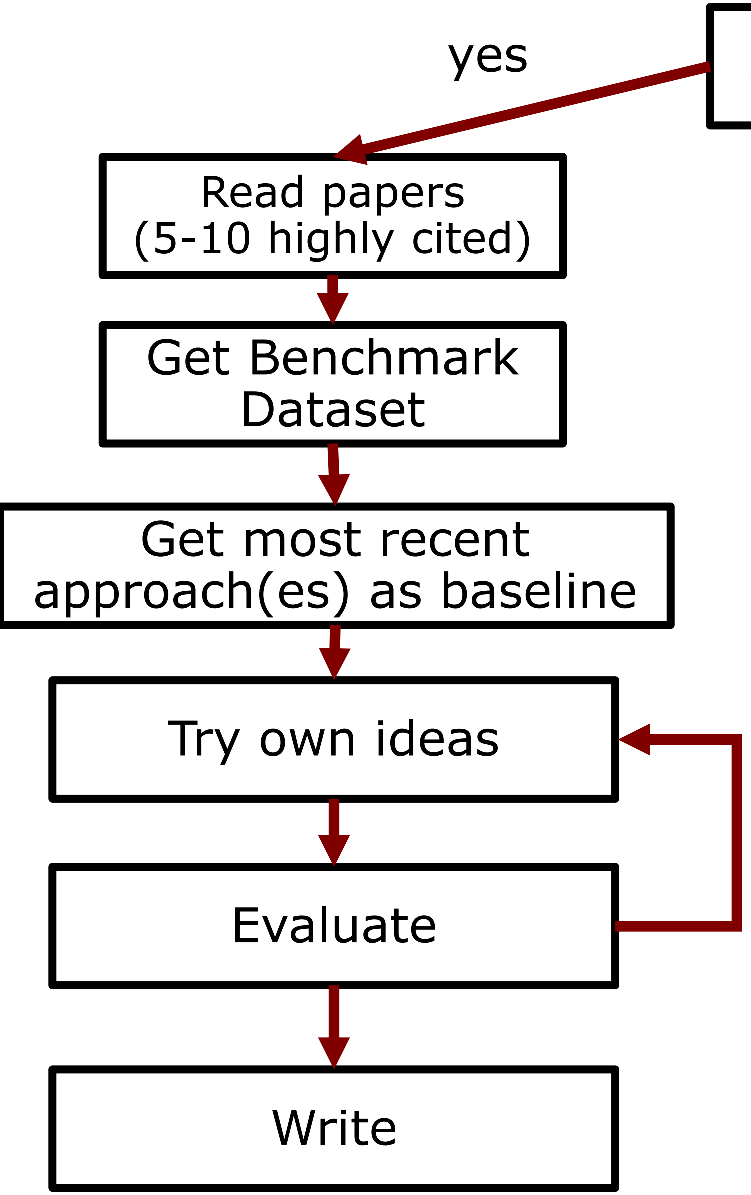


- Replace self-attention with MLP on transposed feature vectors
- All operations are MLPs on image patches

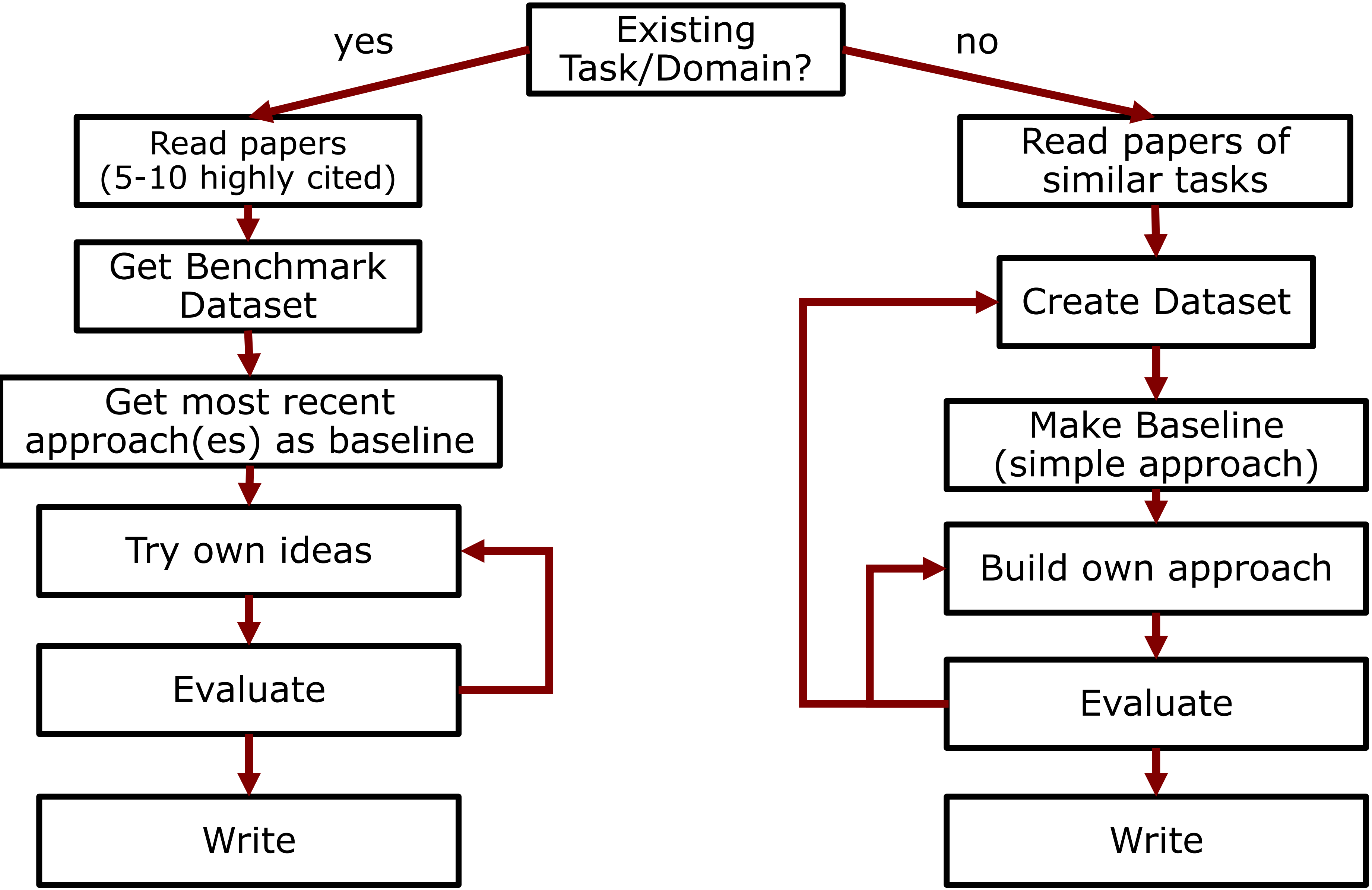
[Tolstinkhin, 2021]

Figure from [Tolstinkhin, 2021]

How to start a project?



How to start a project?



How to create own approach?

1. Start simple, small! Take existing architectures.
2. Test/steal one idea at a time! (Look always at validation error)
3. Evaluate progress. Try to understand why something works/not works. Does it support your hypothesis?
4. Not only metrics. Visualize results.
5. Add data augmentation/mor reularization

See you next week!