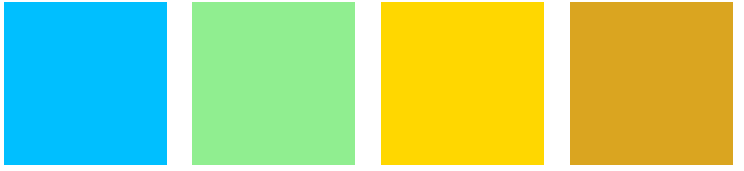
Recent Transformer based architectures for image analysis



Świerk Computing Centre

NATIONAL CENTRE
FOR NUCLEAR RESEARCH
ŚWIERK

Konrad Klimaszewski
20.01.2025, Warsaw

Quick Transformer Recap

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

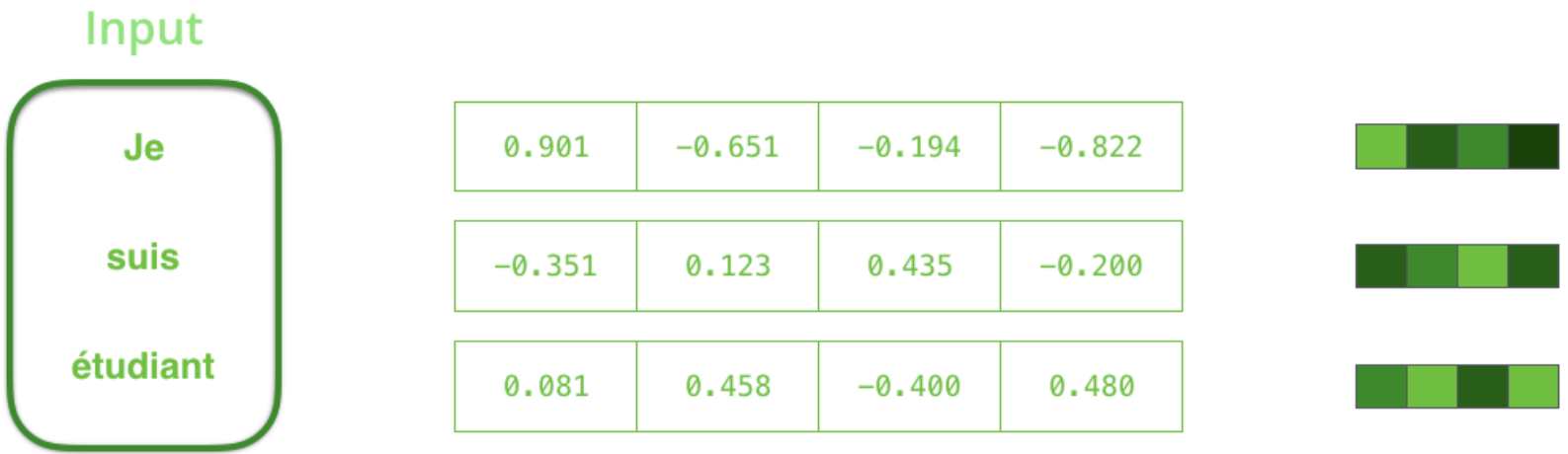
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems
Pages 6000 – 601

https://arxiv.org/abs/1706.03762

Word embedding

To transform a word into a vector, we turn to the class of methods called “word embedding” algorithms. These turn words into vector spaces that capture a lot of the meaning/semantic information of the words (e.g. king - man + woman = queen).

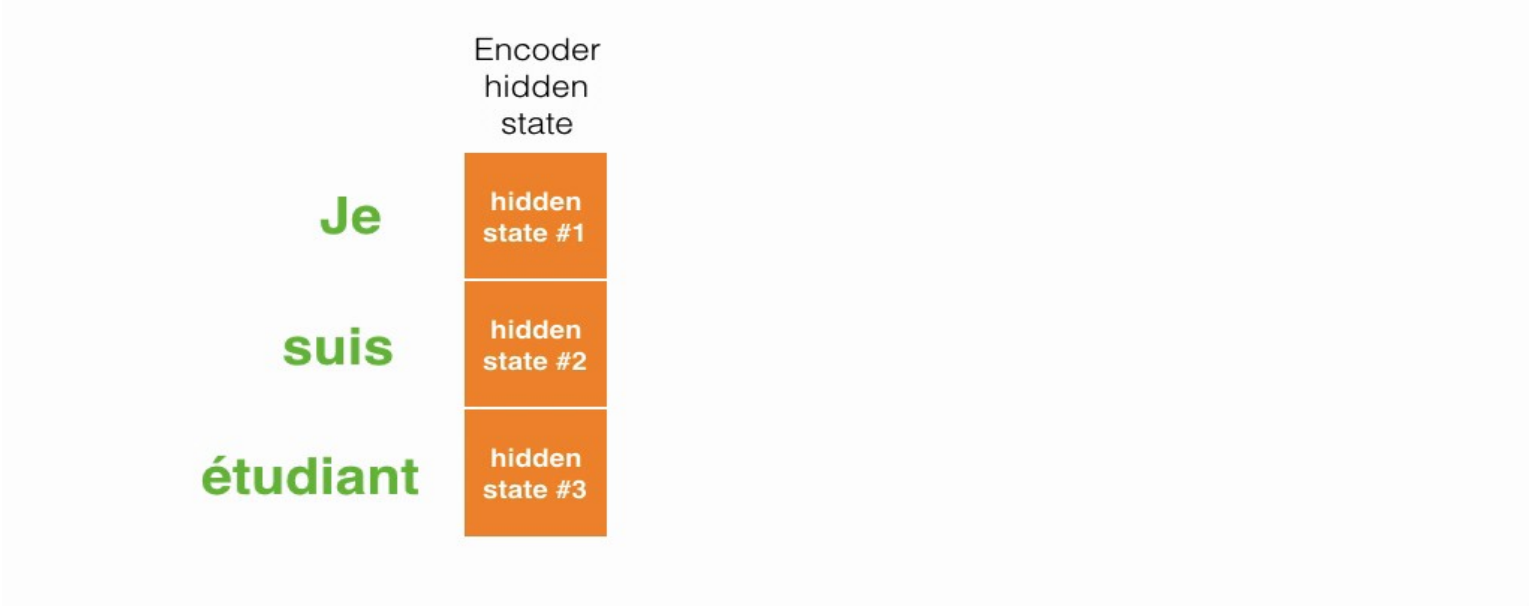


Diagrams from: https://jalammar.github.io/illustrated-transformer/

https://bbycroft.net/llm

Attention

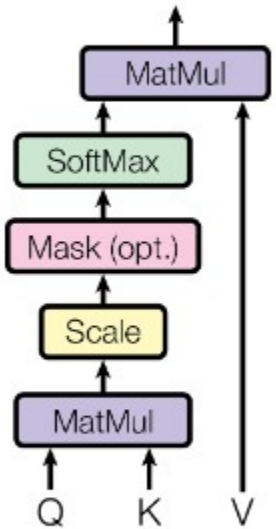
Attention


Encoder hidden state
Je
suis
étudiant
hidden state #1
hidden state #2
hidden state #3

Attention

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Scaled Dot-Product Attention



The Q, K, and V vectors:

Q: Query vector

K: Key vector

V: Value vector

Software analogy

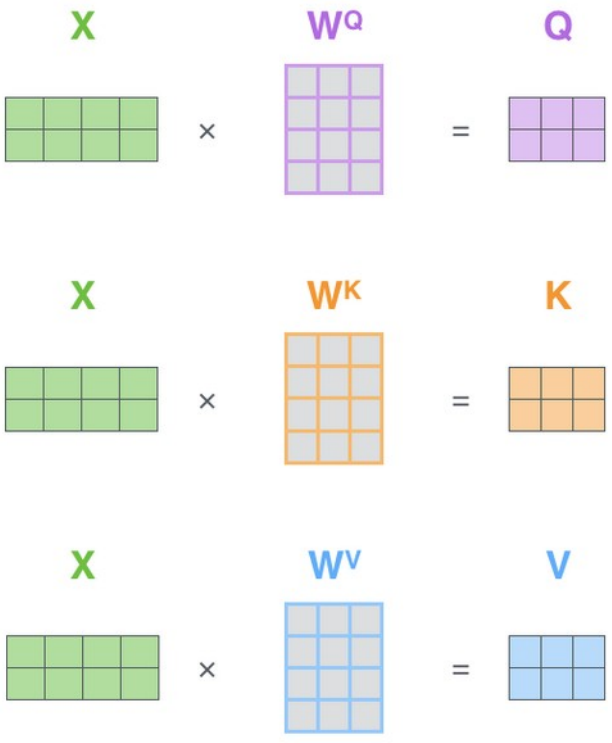
Lookup table:

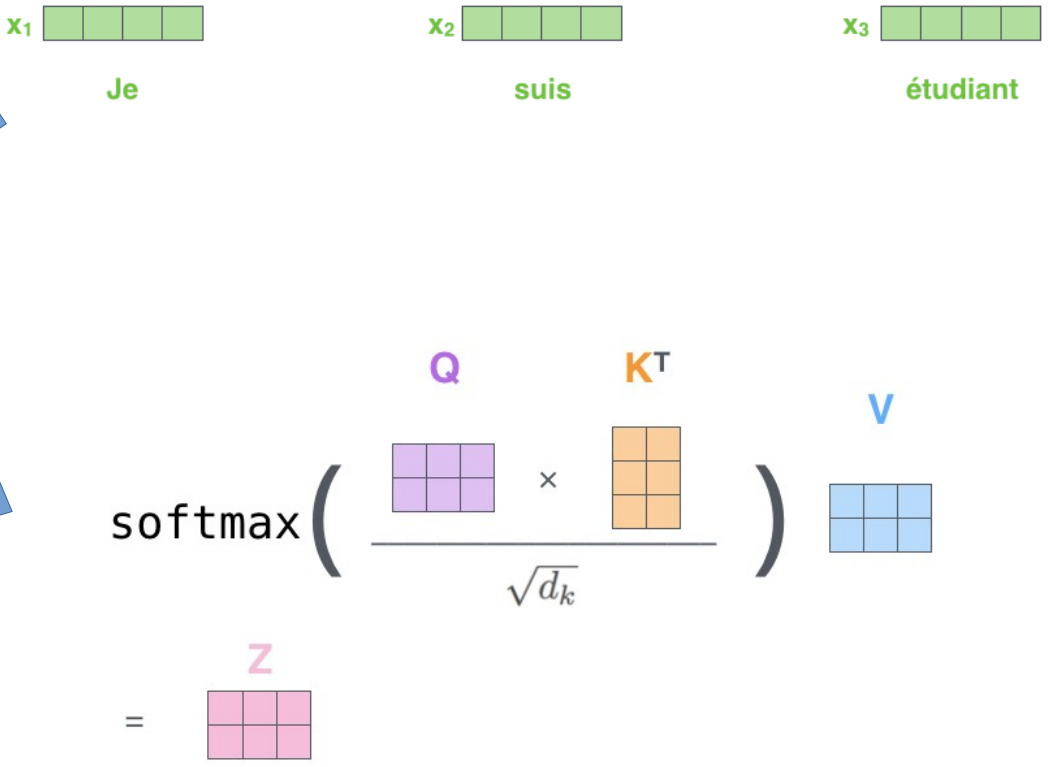
table = { "key0": "value0", "key1": "value1", ... }

Query Process:

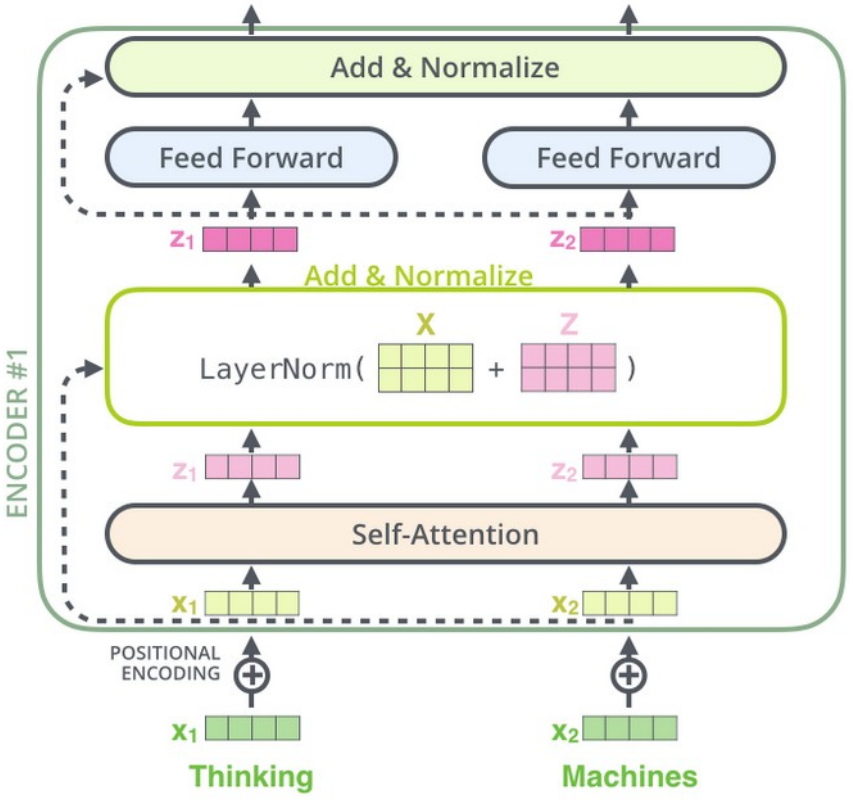
table["key1"] => "value1"

Self - attention





Quick Transformer Recap



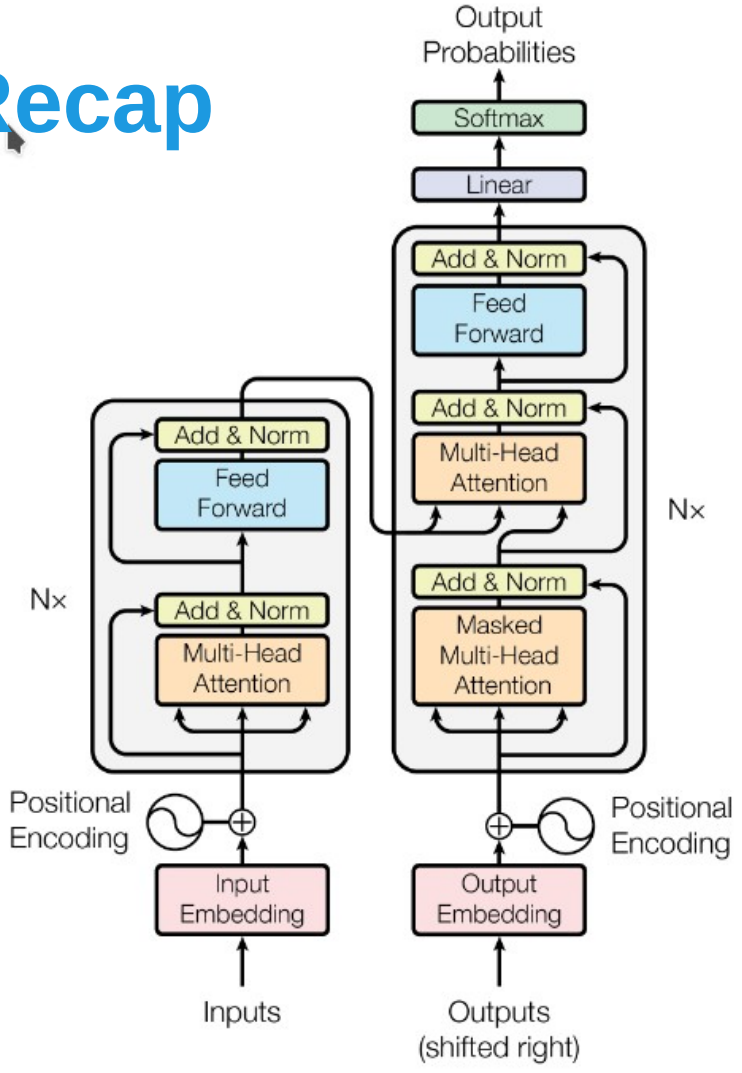
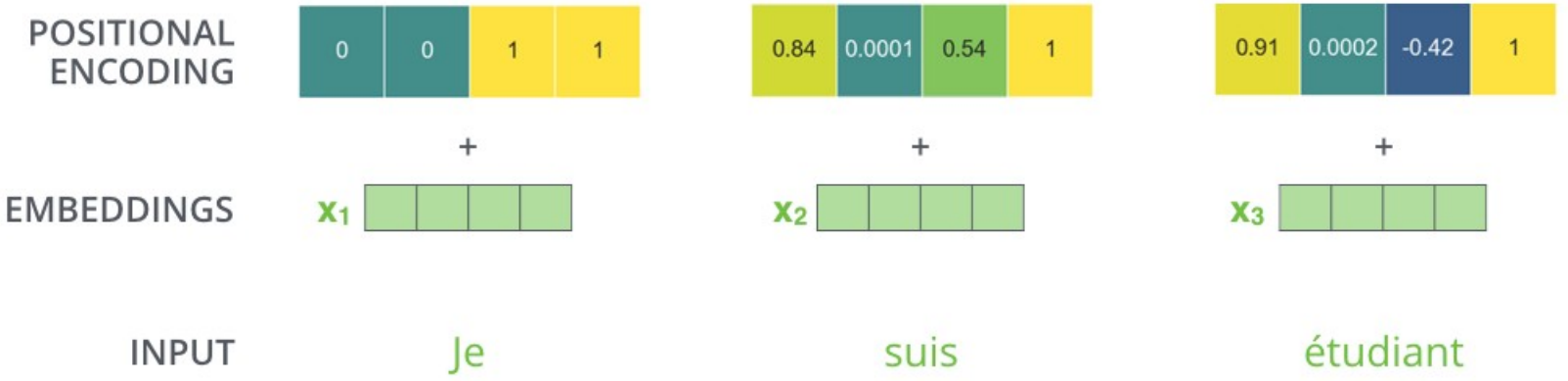
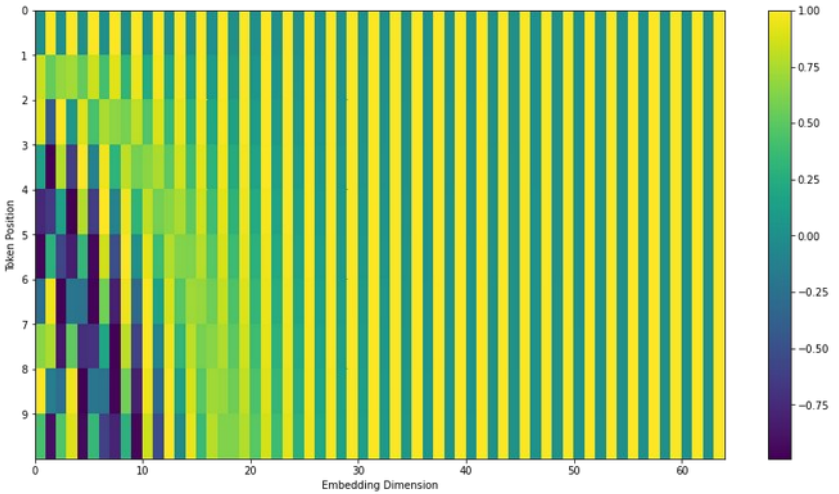


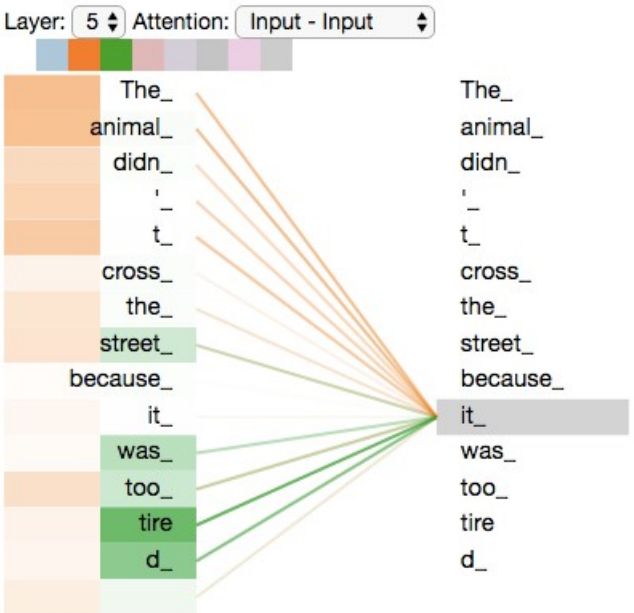
Figure 1: The Transformer - model architecture.

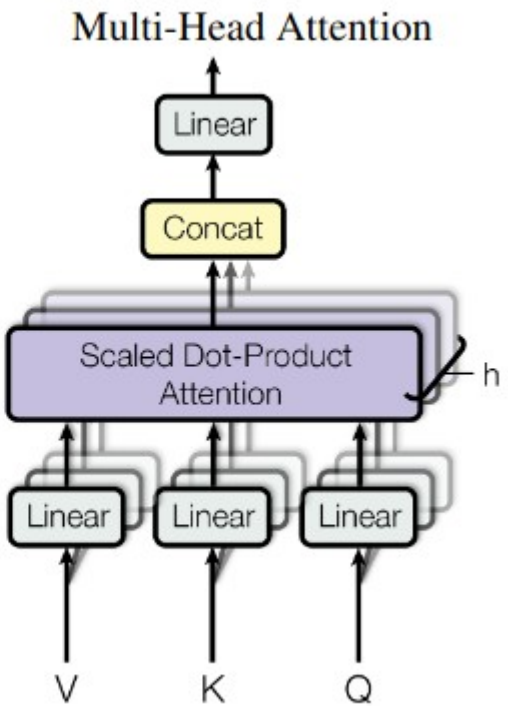
Positional encoding

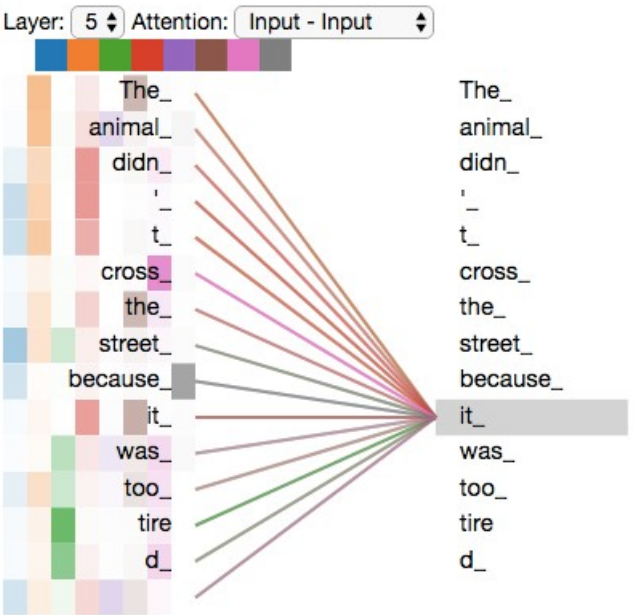




Multi Head Attention







Vision Transformer (ViT)

Published as a conference paper at ICLR 2021

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

https://arxiv.org/abs/2010.11929

https://github.com/google-research/vision_transformer

Vision Transformer (ViT)

- Divide image into N patches (PxP size)
- Image → flat sequence of patches
- Map patches to D dimension tokens via trainable linear projection
- Position embedding – learnable 1D mapping

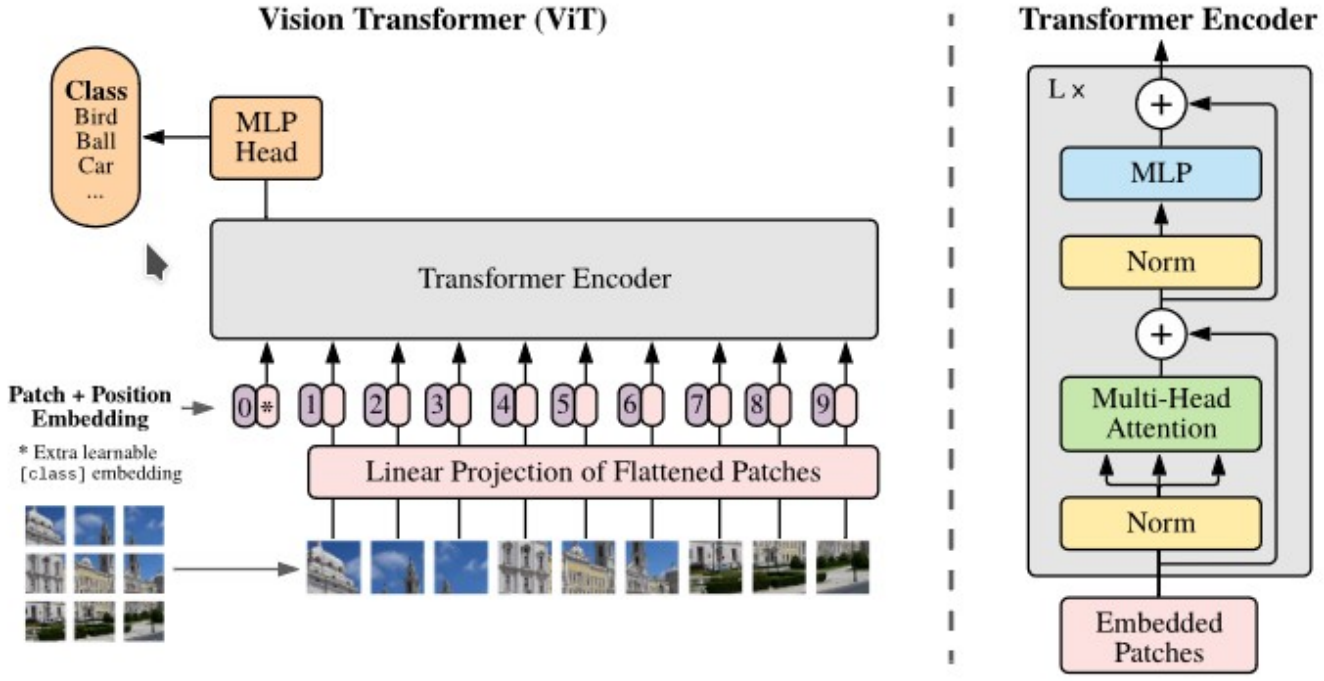


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification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

Vision Transformer (ViT)

- Optimizer: Adam
- Batch size: 4096
- Image size: 224x224
- Linear warmup + decay

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

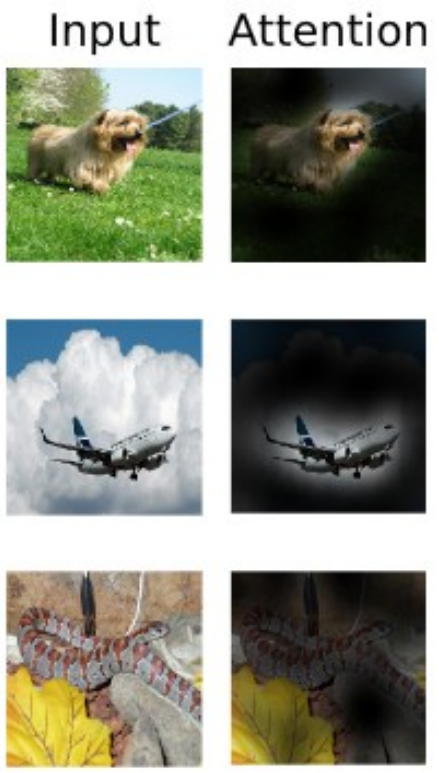


Figure 6: Representative examples of attention from the output token to the input space. See Appendix D.7 for details.

Vision Transformer (ViT)

- Trained on mid-sized datasets (ImageNet) yields results few percents below best CNNs
- **Trained on large datasets (JFT-300M ora ImageNet 21k) suprasses CNNs trained on the same data**

	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

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).

Swin Transformer

2021 IEEE/CVF International Conference on Computer Vision (ICCV)

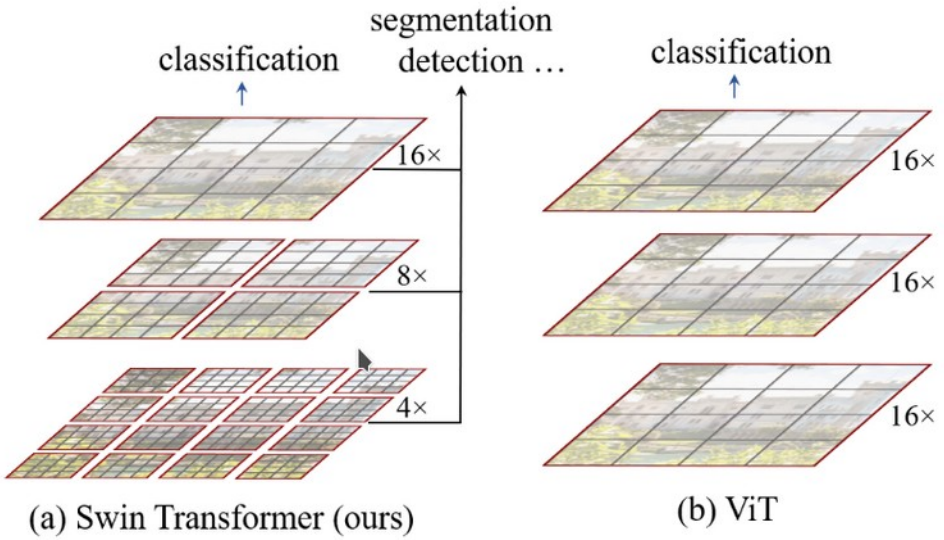
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Ze Liu[1,2†*] Yutong Lin[1,3†*] Yue Cao[1*] Han Hu[1*‡] Yixuan Wei[1,4†]
Zheng Zhang[1] Stephen Lin[1] Baining Guo[1]
[1]Microsoft Research Asia [2]University of Science and Technology of China
[3]Xian Jiaotong University [4]Tsinghua University
{v-zeliu1,v-yutlin,yuecao,hanhu,v-yixwe,zhez,stevelin,bainguo}@microsoft.com

https://arxiv.org/abs/2103.14030

https://github.com/microsoft/Swin-Transformer

Swin Transformer



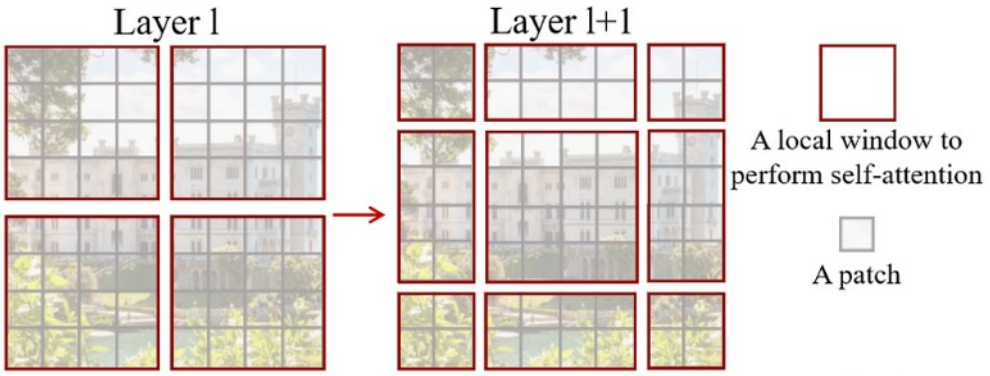


Figure 2. An illustration of the *shifted window* approach for computing self-attention in the proposed Swin Transformer architecture. In layer l (left), a regular window partitioning scheme is adopted, and self-attention is computed within each window. In the next layer $l + 1$ (right), the window partitioning is shifted, resulting in new windows. The self-attention computation in the new windows crosses the boundaries of the previous windows in layer l, providing connections among them.

Swin Transformer

- Self Attention computed in local windows with **masking**
- **W-MSA** – **window** based multihead self attention
- **SW-MSA** – **shifted-window** based multihead self attention

- Optimizer: AdamW
- Batch size: 1024
- Image size: 224^2 lub 384^2
- Linear warmup + cos decay

Self-attention in non-overlapped windows For efficient modeling, we propose to compute self-attention within local windows. The windows are arranged to evenly partition the image in a non-overlapping manner. Supposing each window contains $M \times M$ patches, the computational complexity of a global MSA module and a window based one on an image of $h \times w$ patches are[3]:

$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C, \quad (1)$$

$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC, \quad (2)$$

where the former is quadratic to patch number hw, and the latter is linear when M is fixed (set to 7 by default). Global self-attention computation is generally unaffordable for a large hw, while the window based self-attention is scalable.

Swin Transformer

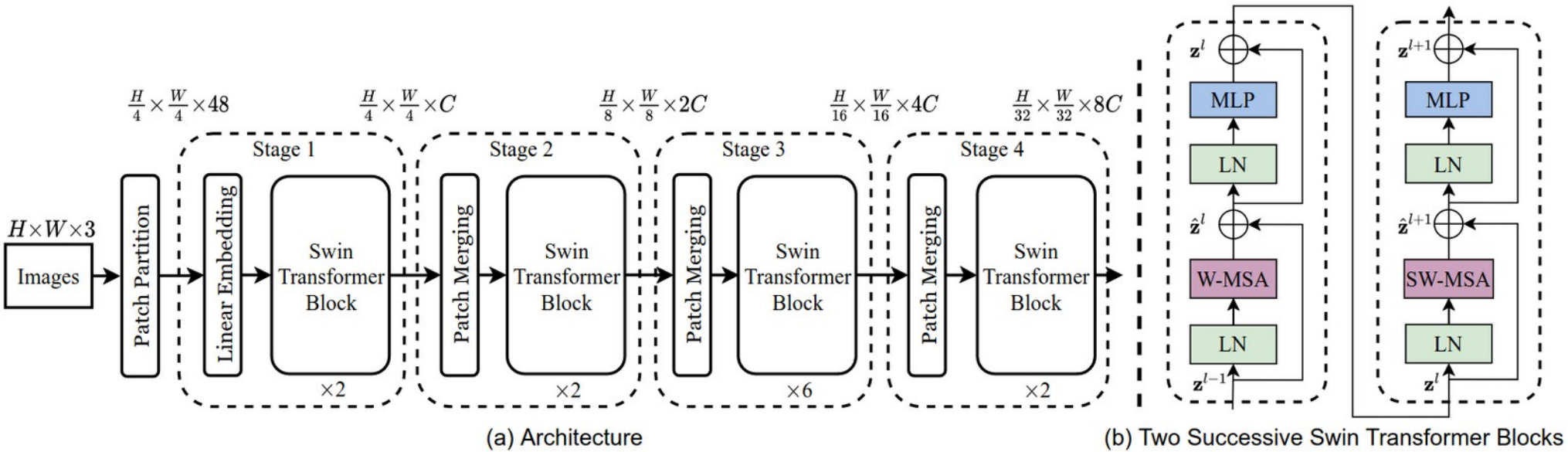
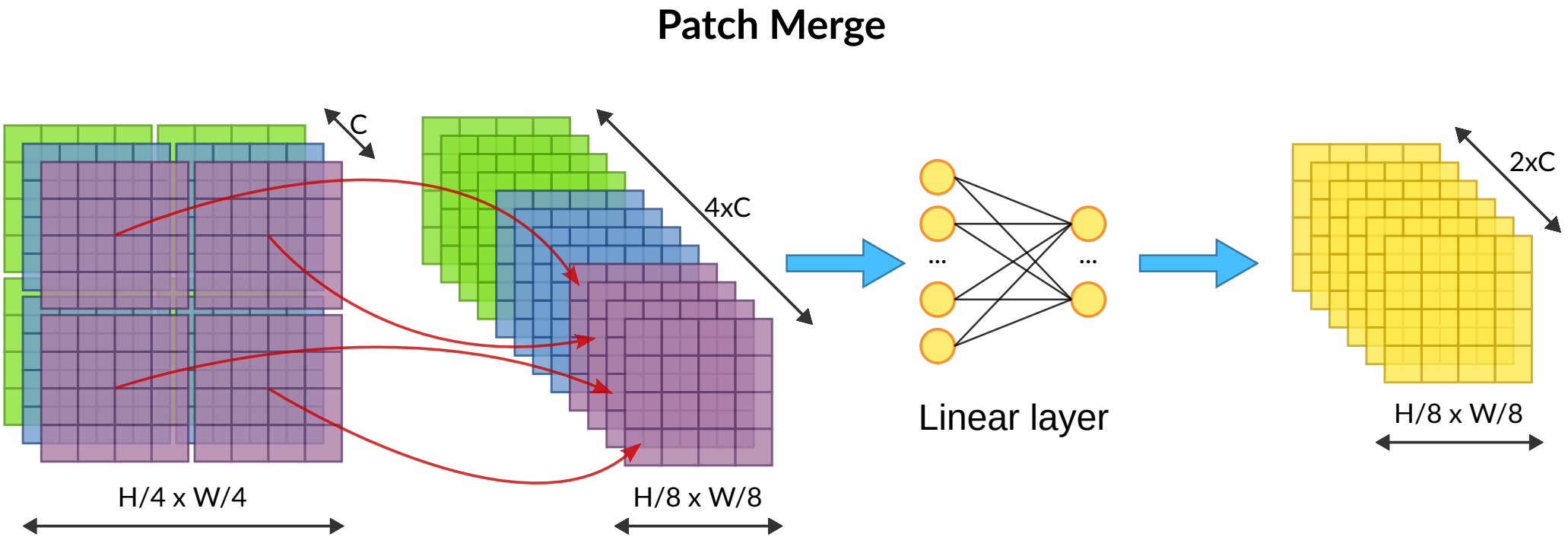


Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

Each patch is treated as a “token” and its feature is set as a concatenation of the raw pixel RGB values. In our implementation, we use a patch size of 4 × 4 and thus the feature dimension of each patch is 4 × 4 × 3 = 48.

Swin Transformer



Swin Transformer

Relative position bias In computing self-attention, we follow [45, 1, 29, 30] by including a relative position bias $B \in \mathbb{R}^{M^2 \times M^2}$ to each head in computing similarity:

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \quad (4)$$

where $Q, K, V \in \mathbb{R}^{M^2 \times d}$ are the *query*, *key* and *value* matrices; d is the *query/key* dimension, and M^2 is the number of patches in a window. Since the relative position along each axis lies in the range $[-M+1, M-1]$, we parameterize a smaller-sized bias matrix $\hat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$, and values in B are taken from $\hat{B}$.

Swin Transformer variants:

- Swin-T: $C = 96$, layer numbers = $\{2, 2, 6, 2\}$
- Swin-S: $C = 96$, layer numbers =$\{2, 2, 18, 2\}$
- Swin-B: $C = 128$, layer numbers =$\{2, 2, 18, 2\}$
- Swin-L: $C = 192$, layer numbers =$\{2, 2, 18, 2\}$

The same feature map resolutions as those of typical convolutional networks, e.g., VGG and ResNet → architecture can the backbone networks in existing methods for various vision tasks

Swin Transformer

ADE20K Method	Backbone	val mIoU	test score	#param.	FLOPs	FPS
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
DNL [65]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [67]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [63]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [67]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [11]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [11]	ResNeSt-200	48.4	-	88M	1381G	8.1
SETR [73]	T-Large‡	50.3	61.7	308M	-	-
UperNet	DeiT-S†	44.0	-	52M	1099G	16.2
UperNet	Swin-T	46.1	-	60M	945G	18.5
UperNet	Swin-S	49.3	-	81M	1038G	15.2
UperNet	Swin-B‡	51.6	-	121M	1841G	8.7
UperNet	Swin-L‡	**53.5**	**62.8**	234M	3230G	6.2

Table 3. Results of semantic segmentation on the ADE20K val and test set. † indicates additional deconvolution layers are used to produce hierarchical feature maps. ‡ indicates that the model is pre-trained on ImageNet-22K.

(a) Regular ImageNet-1K trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
RegNetY-4G [44]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [44]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [44]	224^2	84M	16.0G	334.7	82.9
ViT-B/16 [19]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [19]	384^2	307M	190.7G	27.3	76.5
DeiT-S [57]	224^2	22M	4.6G	940.4	79.8
DeiT-B [57]	224^2	86M	17.5G	292.3	81.8
DeiT-B [57]	384^2	86M	55.4G	85.9	83.1
Swin-T	224^2	29M	4.5G	755.2	81.3
Swin-S	224^2	50M	8.7G	436.9	83.0
Swin-B	224^2	88M	15.4G	278.1	83.5
Swin-B	384^2	88M	47.0G	84.7	84.5

(b) ImageNet-22K pre-trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
R-101x3 [34]	384^2	388M	204.6G	-	84.4
R-152x4 [34]	480^2	937M	840.5G	-	85.4
ViT-B/16 [19]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [19]	384^2	307M	190.7G	27.3	85.2
Swin-B	224^2	88M	15.4G	278.1	85.2
Swin-B	384^2	88M	47.0G	84.7	86.4
Swin-L	384^2	197M	103.9G	42.1	87.3

Table 1. Comparison of different backbones on ImageNet-1K classification. Throughput is measured using the GitHub repository of [62] and a V100 GPU, following [57].

Swin Transformer V2

2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)

Swin Transformer V2: Scaling Up Capacity and Resolution

Ze Liu[2*] Han Hu[1*†] Yutong Lin[3] Zhuliang Yao[4] Zhenda Xie[4] Yixuan Wei[4] Jia Ning[5]
Yue Cao[1] Zheng Zhang[1] Li Dong[1] Furu Wei[1] Baining Guo[1]
[1]Microsoft Research Asia [2]University of Science and Technology of China
[3]Xian Jiaotong University [4]Tsinghua University [5]Huazhong University of Science and Technology
{t-liuze,hanhu,t-yutonglin,t-zhuyao,t-zhxie,t-yixuanwei,v-jianing}@microsoft.com
{yuecao,zhez,lidong1,fuwei,bainguo}@microsoft.com

https://arxiv.org/abs/2111.09883

https://github.com/microsoft/Swin-Transformer

Swin Transformer V2

Scaled cosine attention In the original self-attention computation, the similarity terms of the pixel pairs are computed as a dot product of the *query* and *key* vectors. We find that when this approach is used in large visual models, the learnt attention maps of some blocks and heads are frequently dominated by a few pixel pairs, especially in the *res-post-norm* configuration. To ease this issue, we propose a *scaled cosine attention* approach that computes the attention logit of a pixel pair i and j by a scaled cosine function:

$$\text{Sim}(\mathbf{q}_i, \mathbf{k}_j) = \cos(\mathbf{q}_i, \mathbf{k}_j)/\tau + B_{ij}, \qquad (2)$$

where B_{ij} is the relative position bias between pixel i and j; τ is a learnable scalar, non-shared across heads and layers. τ is set larger than 0.01. The cosine function is naturally normalized, and thus can have milder attention values.

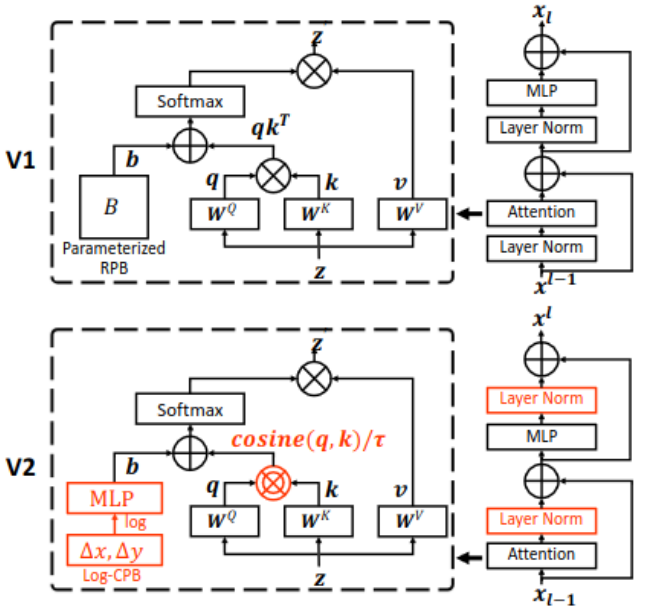


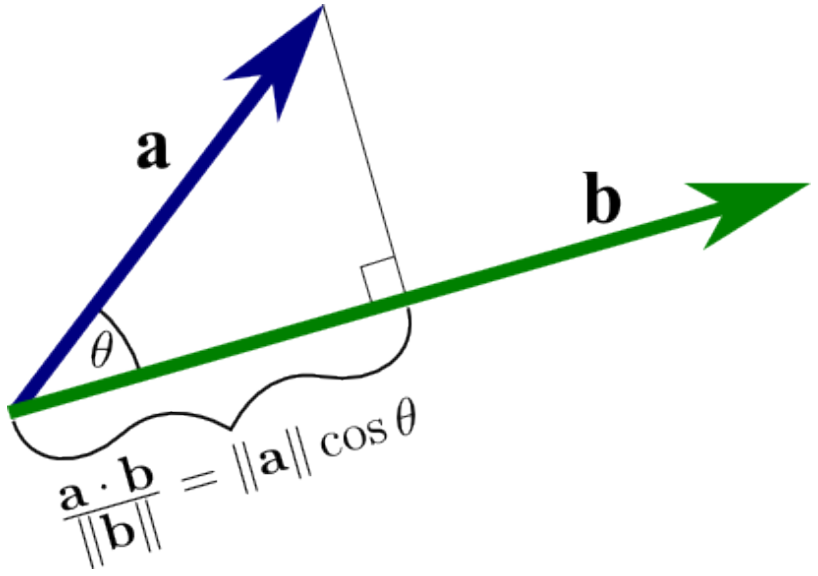
Figure 1. To better scale up model capacity and window resolution, several adaptions are made on the original Swin Transformer architecture (V1): 1) A *res-post-norm* to replace the previous *pre-norm* configuration; 2) A *scaled cosine attention* to replace the original *dot product attention*; 3) A *log-spaced continuous* relative position bias approach to replace the previous *parameterized* approach. Adaptions 1) and 2) make it easier for the model to scale up capacity. Adaption 3) makes the model to be transferred more effectively across window resolutions. The adapted architecture is named Swin Transformer V2.

Swin Transformer V2

Scaled cosine attention In the original self-attention computation, the similarity terms of the pixel pairs are computed as a dot product of the *query* and *key* vectors. We find that when this approach is used in large visual models, the learnt attention maps of some blocks and heads are frequently dominated by a few pixel pairs, especially in the *res-post-norm* configuration. To ease this issue, we propose a *scaled cosine attention* approach that computes the attention logit of a pixel pair i and j by a scaled cosine function:

$$\text{Sim}(\mathbf{q}_i, \mathbf{k}_j) = \cos(\mathbf{q}_i, \mathbf{k}_j)/\tau + B_{ij}, \qquad (2)$$

where B_{ij} is the relative position bias between pixel i and j; τ is a learnable scalar, non-shared across heads and layers. τ is set larger than 0.01. The cosine function is naturally normalized, and thus can have milder attention values.



$$similarity(A,B) = cos(\theta) = \frac{A \cdot B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}.$$

Swin Transformer V2

Continuous relative position bias Instead of directly optimizing the parameterized biases, the *continuous* position bias approach adopts a small meta network on the relative coordinates:

$$B(\Delta x, \Delta y) = \mathcal{G}(\Delta x, \Delta y), \tag{3}$$

where $\mathcal{G}$ is a small network, e.g., a 2-layer MLP with a ReLU activation in between by default.

The meta network $\mathcal{G}$ generates bias values for arbitrary relative coordinates, and thus can be naturally transferred to fine-tuning tasks with arbitrarily varying window sizes. In inference, the bias values at each relative position can be pre-computed and stored as model parameters, such that the inference is the same as the original parameterized bias approach.

3.6. Model configurations

We maintain the stage, block, and channel settings of the original Swin Transformer for 4 configurations of Swin Transformer V2:

- SwinV2-T: $C = 96$, #. block = $\{2, 2, 6, 2\}$
- SwinV2-S/B/L: C=$96/128/192$, #.block=$\{2, 2, 18, 2\}$

with C the number of channels in the first stage.

We further scale up Swin Transformer V2 to its huge size and giant size, with 658 million parameters and 3 billion parameters, respectively:

- SwinV2-H: $C = 352$, #. block = $\{2, 2, 18, 2\}$
- SwinV2-G: $C = 512$, #. block = $\{2, 2, 42, 4\}$

Swin Transformer V2

Method	param	pre-train images	pre-train length (#im)	pre-train im size	pre-train time	fine-tune im size	ImageNet-1K-V1 top-1 acc	ImaegNet-1K-V2 top-1 acc
SwinV1-B	88M	IN-22K-14M	1.3B	224^2	$<30^{\dagger}$	384^2	86.4	76.58
SwinV1-L	197M	IN-22K-14M	1.3B	224^2	$<10^{\dagger}$	384^2	87.3	77.46
ViT-G [66]	1.8B	JFT-3B	164B	224^2	>30k	518^2	90.45	83.33
V-MoE [44]	14.7B*	JFT-3B	-	224^2	16.8k	518^2	90.35	-
CoAtNet-7 [10]	2.44B	JFT-3B	-	224^2	20.1k	512^2	**90.88**	-
SwinV2-B	88M	IN-22K-14M	1.3B	192^2	$<30^{\dagger}$	384^2	87.1	78.08
SwinV2-L	197M	IN-22K-14M	1.3B	192^2	$<20^{\dagger}$	384^2	87.7	78.31
SwinV2-G	3.0B	IN-22K-ext-70M	3.5B	192^2	$<0.5k^{\dagger}$	640^2	90.17	**84.00**

Table 2. Comparison with previous largest vision models on ImageNet-1K V1 and V2 classification. * indicates the sparse model; the "pre-train time" column is measured by the TPUv3 core days with numbers copied from the original papers. † That of SwinV2-G is estimated according to training iterations and FLOPs.

Swin Transformer V2

Method	train I(W) size	test I(W) size	mIoU
SwinV1-L [35]	640(7)	640(7)	53.5*
MaskFormer [7]	640(7)	640(7)	55.6*
FaPN [22]	640(7)	640(7)	56.7*
BEiT [3]	640(40)	640(40)	58.4*
SwinV2-L (UperNet)	640(40)	640(40)	55.9*
SwinV2-G (UperNet)	640(40)	640(40)	59.1
		896 (56)	59.3
		896 (56)	**59.9***

Table 4. Comparison with previous best results on ADE20K semantic segmentation. * indicates multi-scale testing is used.

Pros:

- Best in class segementation
- Improved image size scaling
- Large potential for improvement

Cons:

- Requires large pre-training
- Local window + positional bias + cosine attention are **NOT compatible** with optimized attention kernels (e.g. **Flash Attention**)

Swin U-Net

Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation

Hu Cao[1†], Yueyue Wang[2†], Joy Chen[1], Dongsheng Jiang[3*], Xiaopeng Zhang[3*], Qi Tian[3*], and Manning Wang[2]

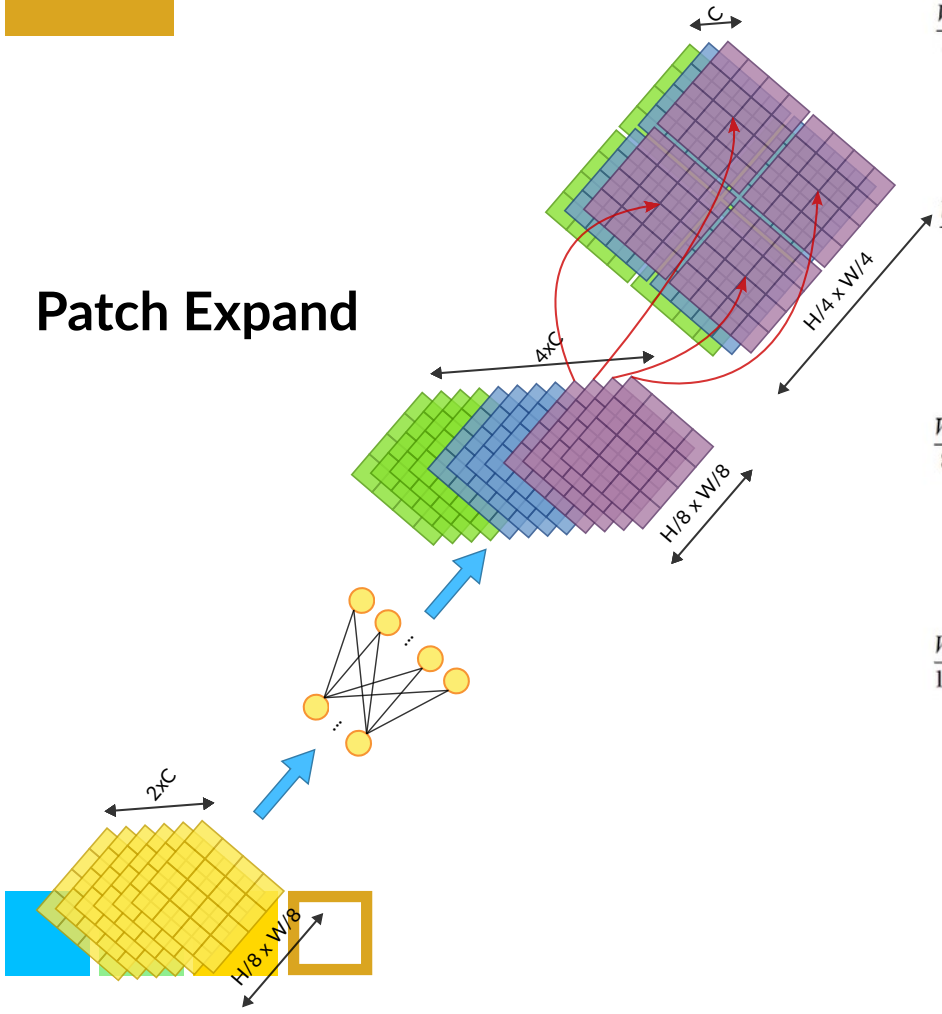
[1] Technische Universität München, München, Germany
[2] Fudan University, Shanghai, China
[3] Huawei Technologies, Shanghai, China

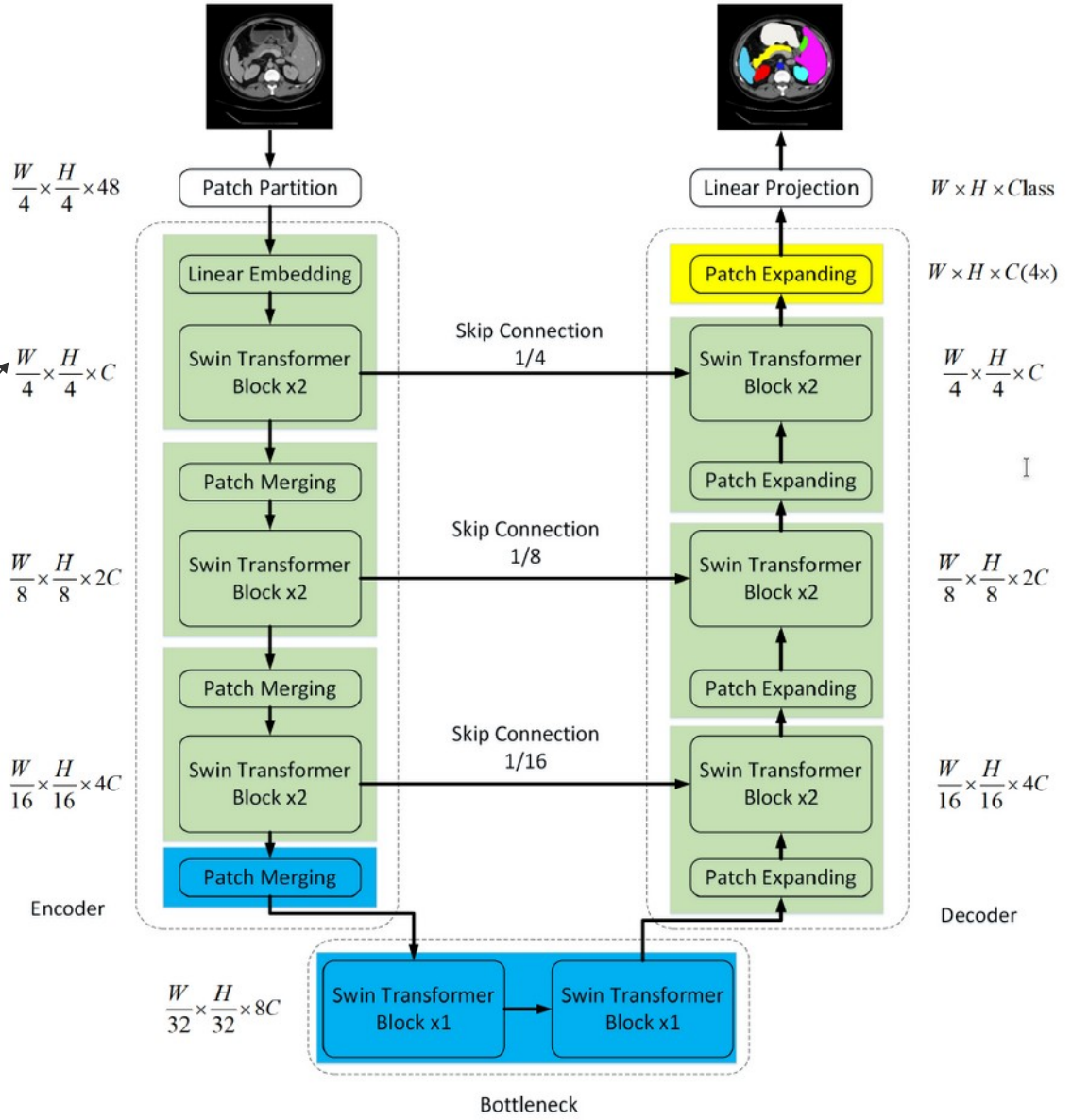
Computer Vision – ECCV 2022 Workshops. ECCV 2022.
Lecture Notes in Computer Science, vol 13803

https://arxiv.org/abs/2105.05537

Swin U-Net

Patch Expand





Swin U-Net

- 3779 clinical CT images
- Image resolution: 224 x 224
- Loss: Dice + Cross Entropy
- Optimizer: SGD
- Batch size: 24

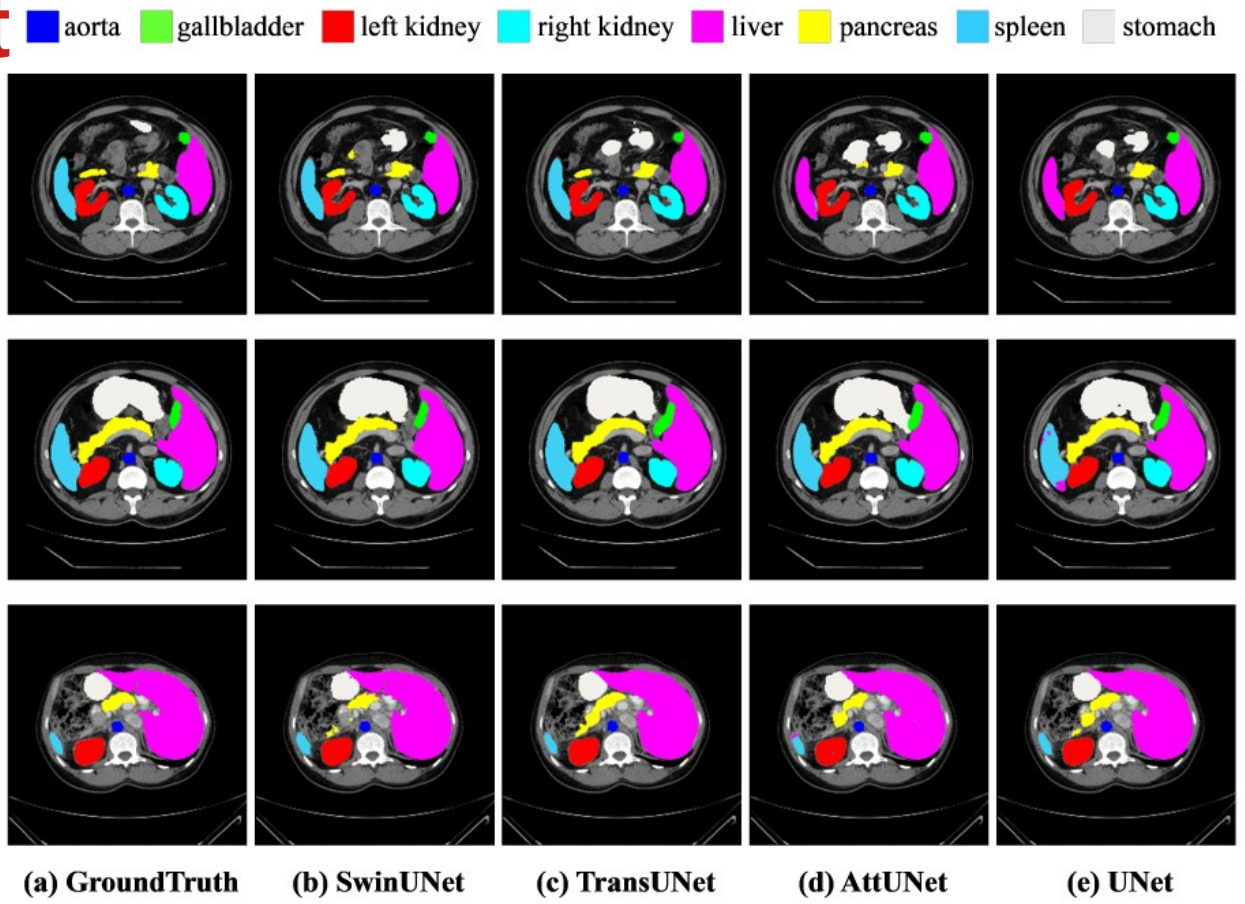


Fig. 3. The segmentation results of different methods on the Synapse multi-organ CT dataset.

Swin U-Net

Table 1. Segmentation accuracy of different methods on the Synapse multi-organ CT dataset.

Methods	DSC↑	HD↓	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach
V-Net [35]	68.81	-	75.34	51.87	77.10	**80.75**	87.84	40.05	80.56	56.98
DARR [36]	69.77	-	74.74	53.77	72.31	73.24	94.08	54.18	89.90	45.96
R50 U-Net [2]	74.68	36.87	87.74	63.66	80.60	78.19	93.74	56.90	85.87	74.16
U-Net [3]	76.85	39.70	89.07	**69.72**	77.77	68.60	93.43	53.98	86.67	75.58
R50 Att-UNet [2]	75.57	36.97	55.92	63.91	79.20	72.71	93.56	49.37	87.19	74.95
Att-UNet [37]	77.77	36.02	**89.55**	68.88	77.98	71.11	93.57	**58.04**	87.30	75.75
R50 ViT [2]	71.29	32.87	73.73	55.13	75.80	72.20	91.51	45.99	81.99	73.95
TransUnet [2]	77.48	31.69	87.23	63.13	81.87	77.02	94.08	55.86	85.08	75.62
SwinUnet	**79.13**	**21.55**	85.47	66.53	**83.28**	79.61	**94.29**	56.58	**90.66**	**76.60**

Swin U-Net

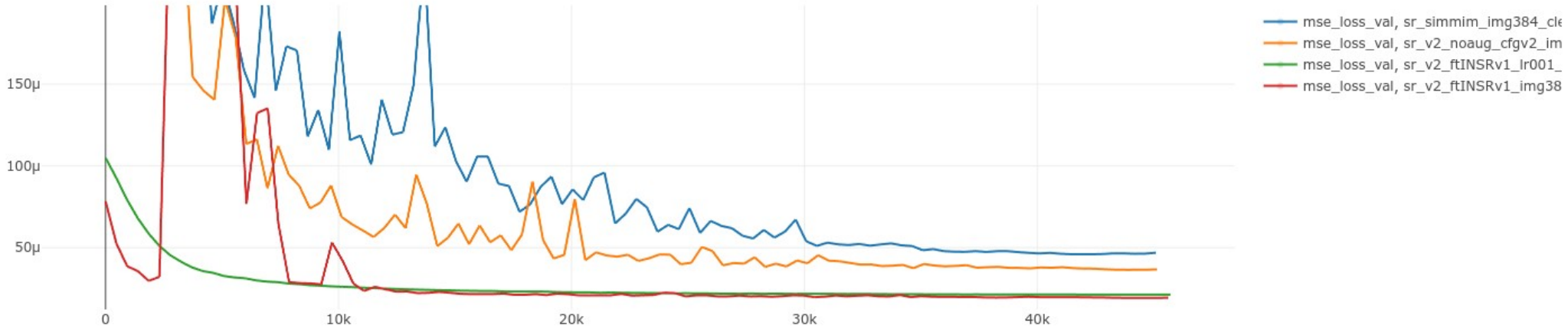
Table 3. Ablation study on the impact of the up-sampling

Up-sampling	DSC	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach
Bilinear interpolation	76.15	81.84	66.33	80.12	73.91	93.64	55.04	86.10	72.20
Transposed convolution	77.63	84.81	65.96	82.66	74.61	**94.39**	54.81	89.42	74.41
Patch expand	**79.13**	**85.47**	**66.53**	**83.28**	**79.61**	94.29	**56.58**	**90.66**	**76.60**

Table 4. Ablation study on the impact of the number of skip connection

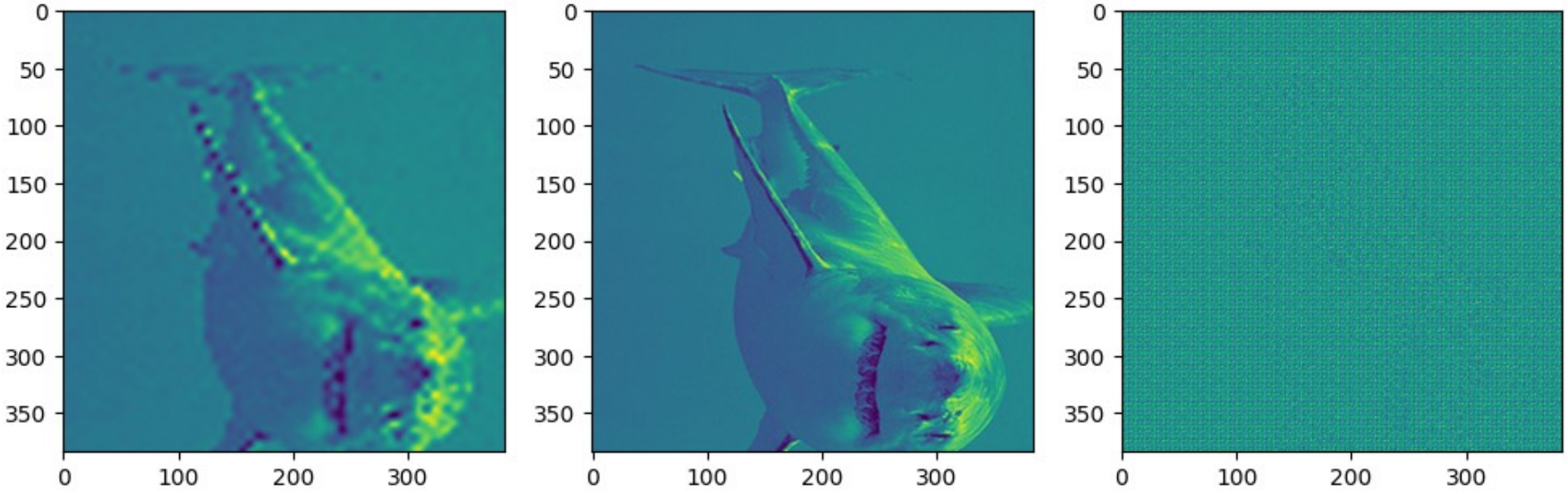
Skip connection	DSC	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach
0	72.46	78.71	53.24	77.46	75.90	92.60	46.07	84.57	71.13
1	76.43	82.53	60.44	81.36	79.27	93.64	53.36	85.95	74.90
2	78.93	**85.82**	66.27	**84.70**	**80.32**	93.94	55.32	88.35	**76.71**
3	**79.13**	85.47	**66.53**	83.28	79.61	**94.29**	**56.58**	**90.66**	76.60

Swin U-Net – pre training



blue / orange – only Swin Transformer pretrained on Imagenet 1k
red / green – Swin U-Net pretrained on Imagenet 1k

Swin U-Net – artifacts



Swin U-Net – artifacts