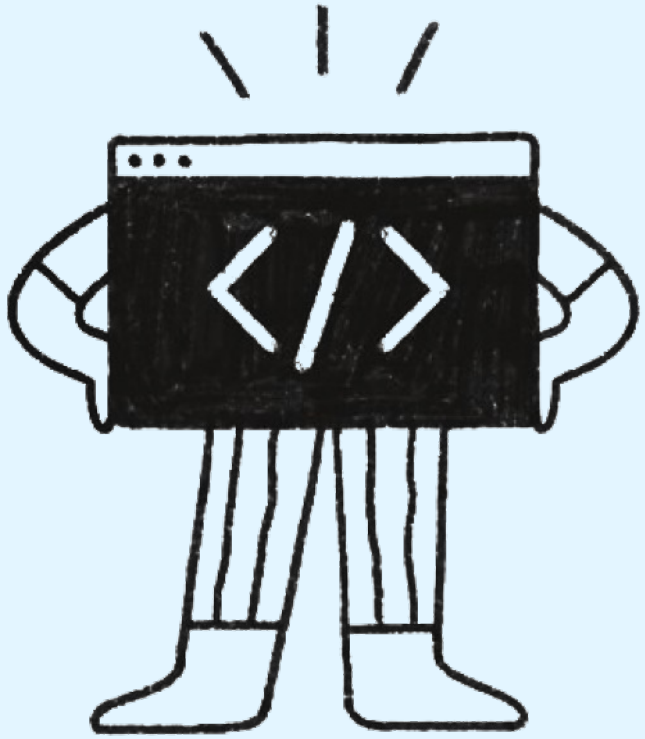
Vision Transformers

YOUNG &78 YANDEX

Радослав Нейчев
Выпускник и преподаватель ШАД и МФТИ,
руководитель группы ML-разработки в Яндексе,
основатель girafe ai

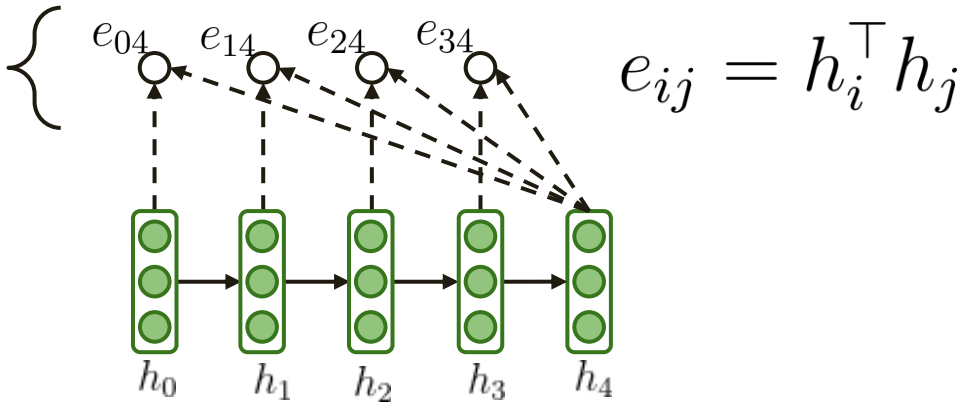
Attention for sequences



01

Attention для последовательности

Attention scores

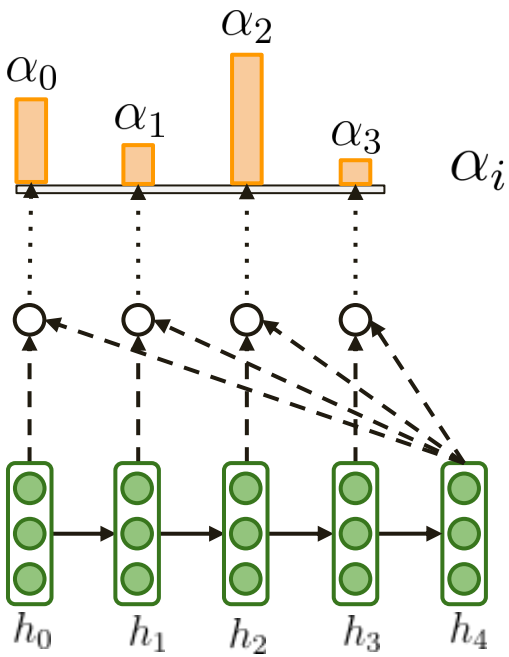


$$e_{ij} = h_i^\top h_j$$

Attention для последовательности

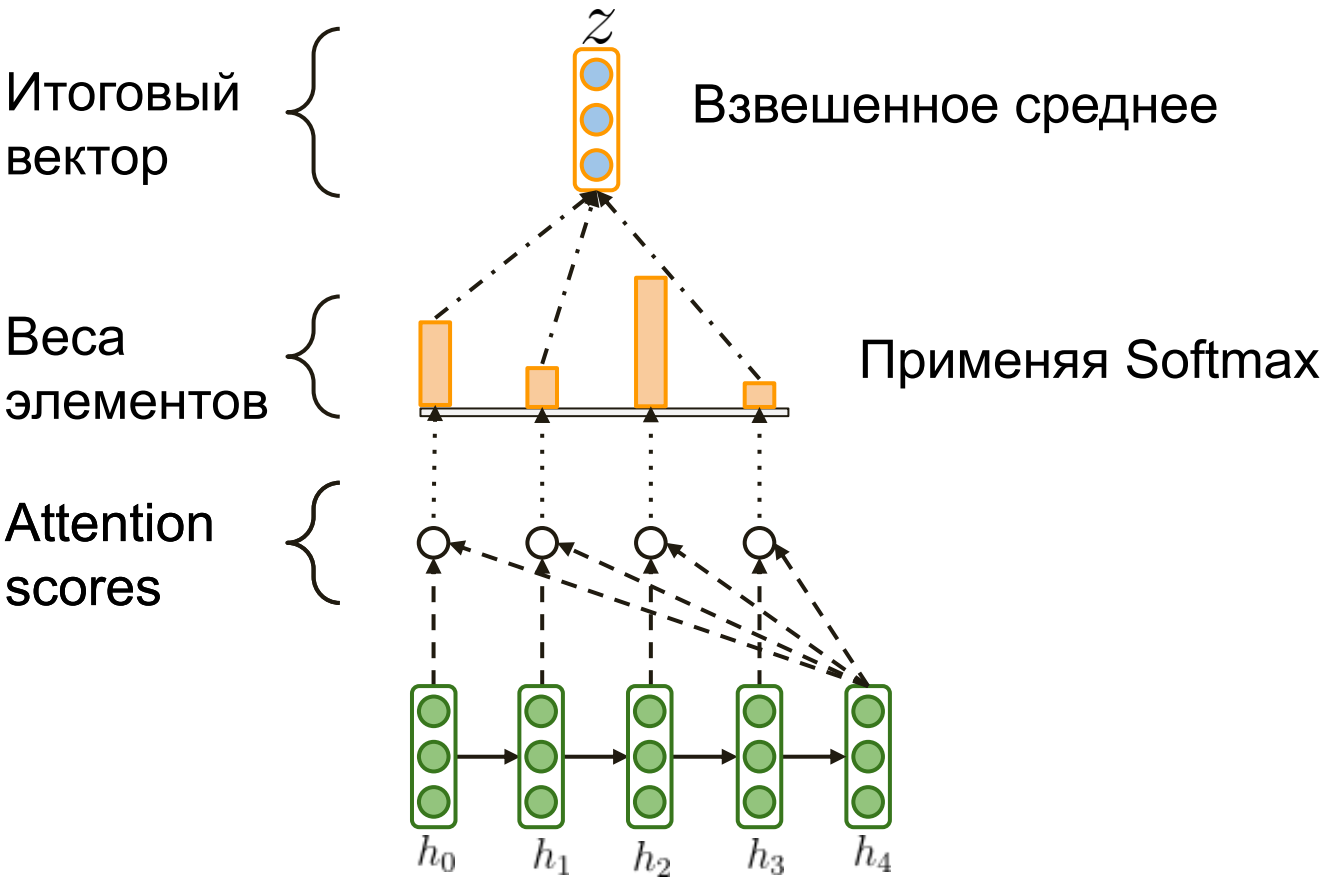
Веса элементов

Attention scores



$$\alpha_i = \mathrm{softmax}_i([e_{1j}, e_{2j}, \ldots e_{j-1\ j}])$$

Attention для последовательности

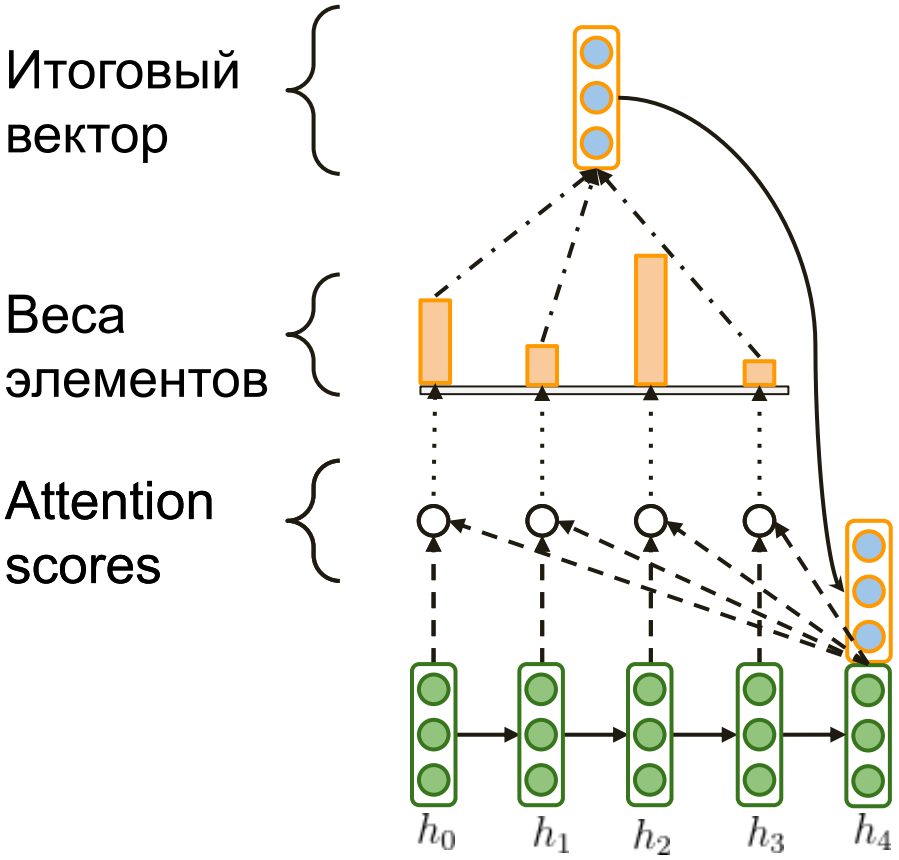


$$z = \sum_{i=1}^{T} \alpha_i h_i$$

$$\sum_{i=1}^{T} \alpha_i = 1$$

$$\forall i \quad \alpha_i \in [0; 1]$$

Attention для последовательности

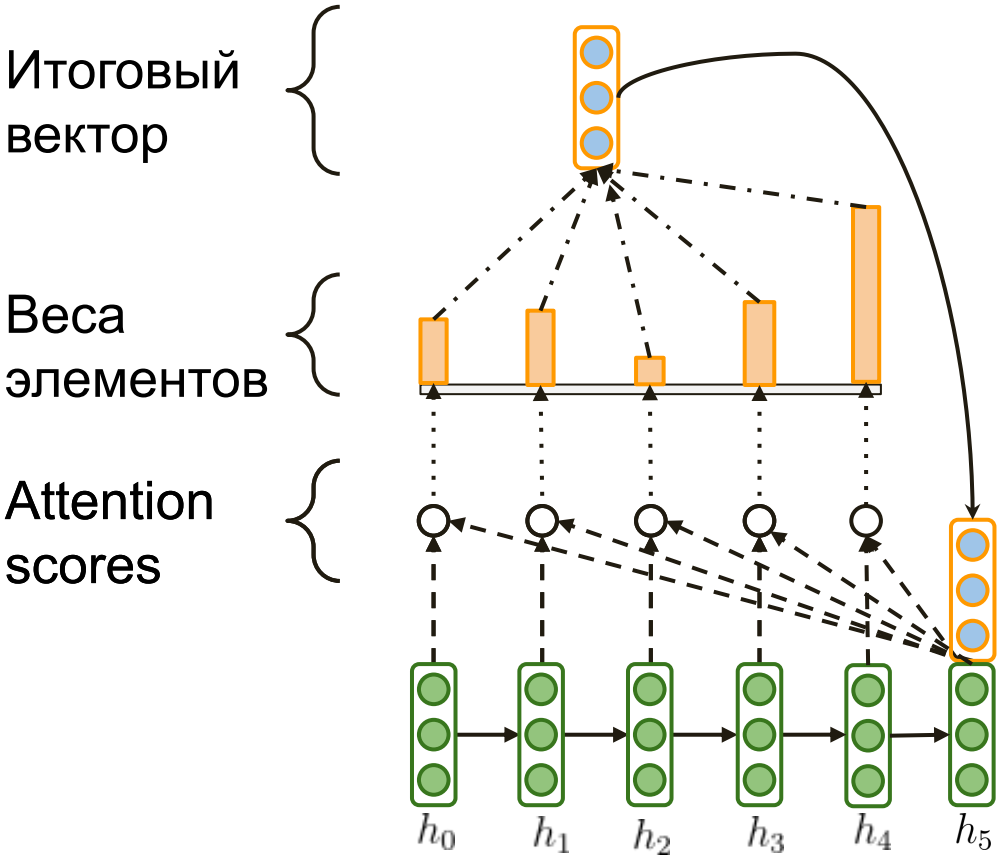


$$z = \sum_{i=1}^{T} \alpha_i h_i$$

$$\sum_{i=1}^{T} \alpha_i = 1$$

$$\forall i \quad \alpha_i \in [0; 1]$$

Attention для последовательности

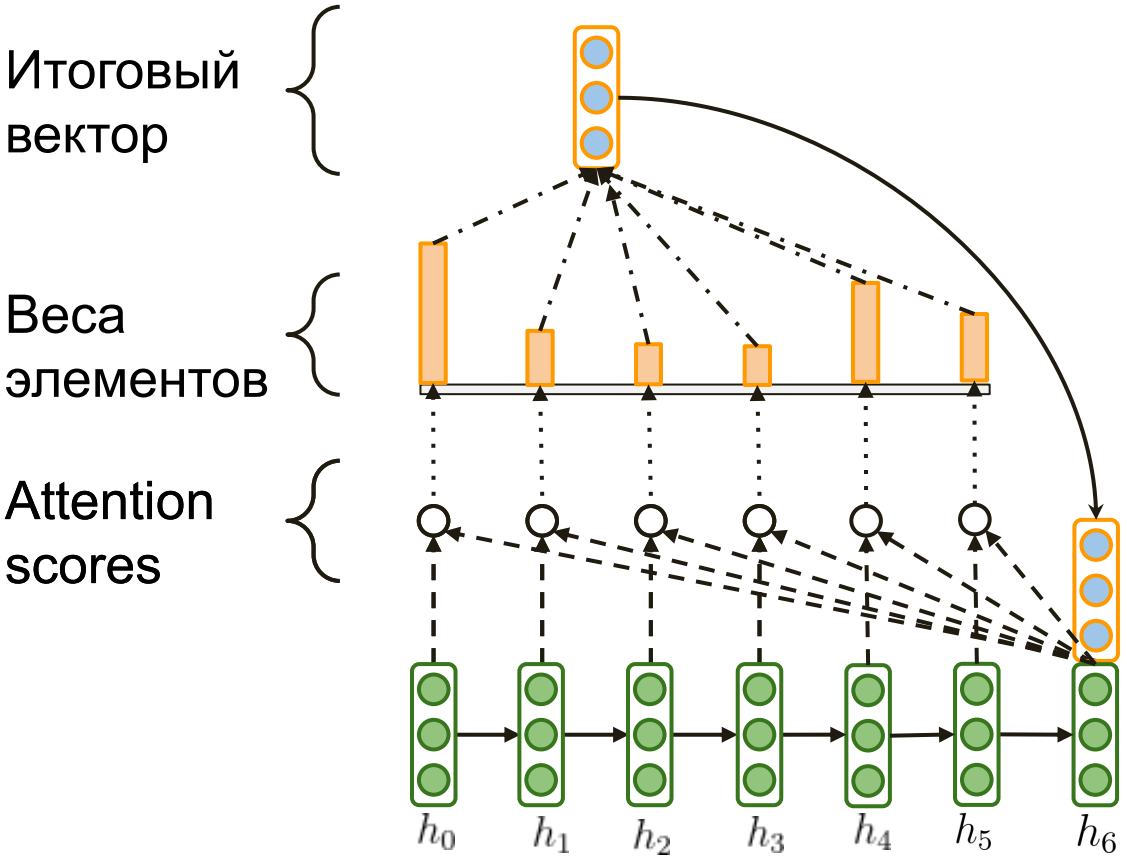


$$z = \sum_{i=1}^{T} \alpha_i h_i$$

$$\sum_{i=1}^{T} \alpha_i = 1$$

$$\forall i \quad \alpha_i \in [0; 1]$$

Attention для последовательности

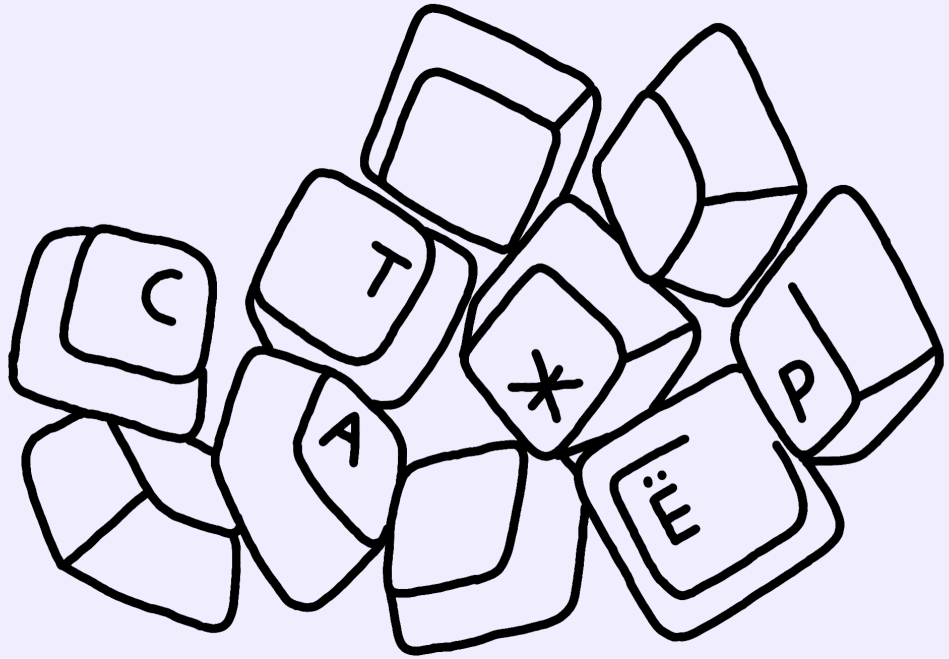


$$z = \sum_{i=1}^{T} \alpha_i h_i$$

$$\sum_{i=1}^{T} \alpha_i = 1$$

$$\forall i \quad \alpha_i \in [0; 1]$$

(Self-)Attention – механизм внимания



03

Self-Attention at a High Level

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

- What does “it” in this sentence refer to?

source: http://jalammar.github.io/illustrated-transformer/

Self-Attention at a High Level

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

- What does “it” in this sentence refer to?
- We want self-attention to associate “it” with “animal”

source: http://jalammar.github.io/illustrated-transformer/

Self-Attention at a High Level

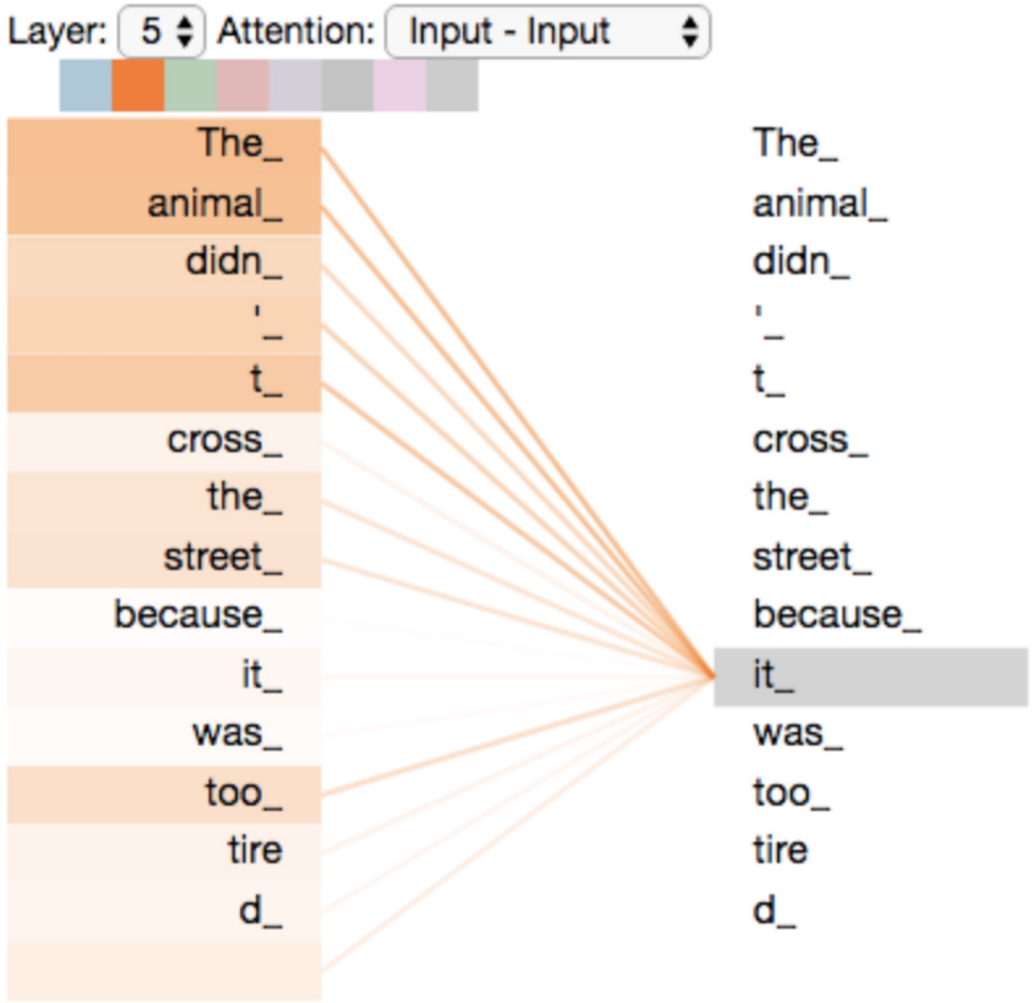
”The animal didn't cross the street because it was too tired”

- What does “it” in this sentence refer to?
- We want self-attention to associate “it” with “animal”

- Self-attention is the method the Transformer uses to bake the “understanding” of other relevant words into the one we’re currently processing

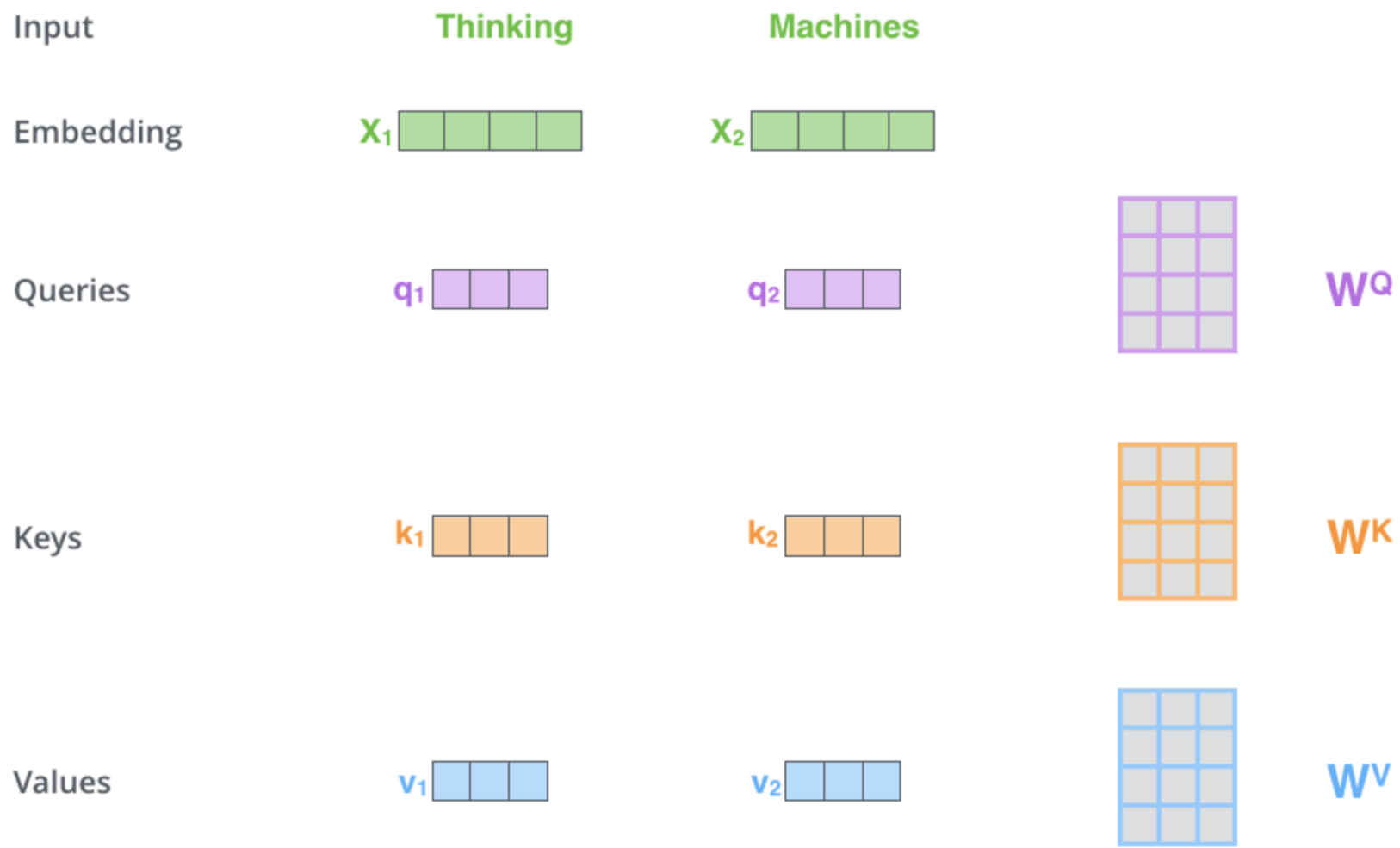
source: http://jalammar.github.io/illustrated-transformer/

Self-Attention at a High Level



source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: detailed explanation



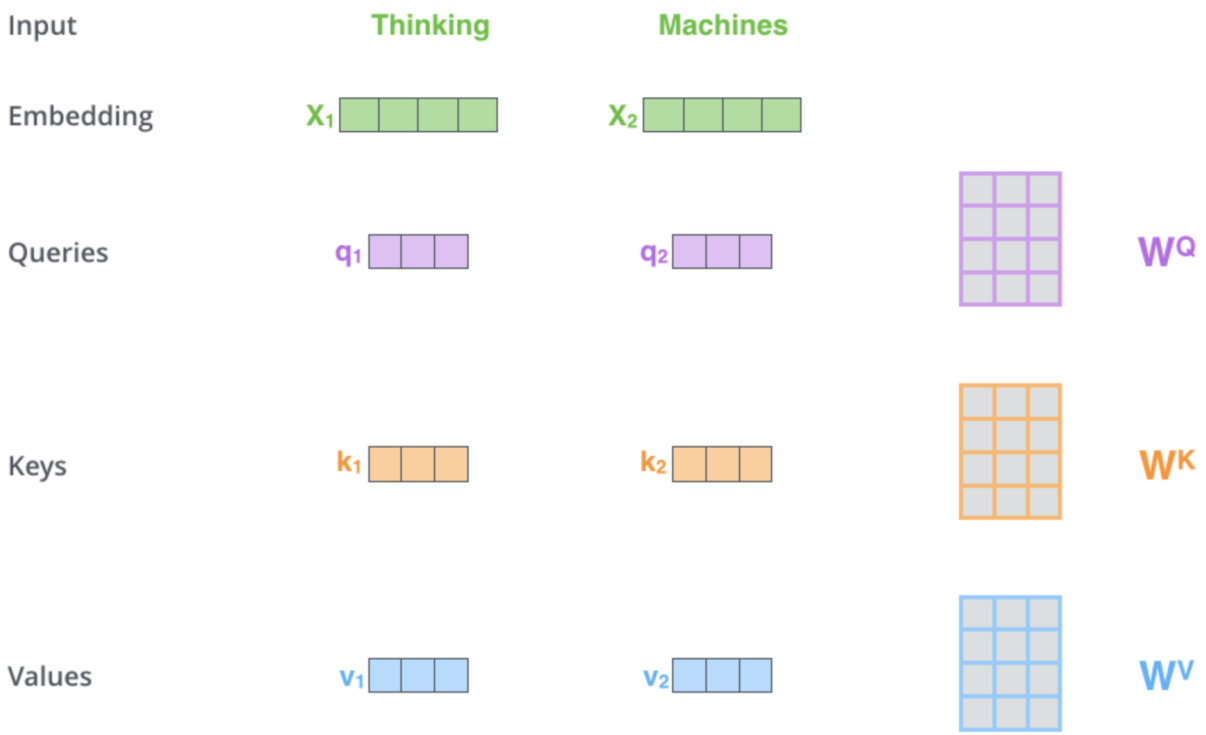
source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: detailed explanation

STEP 1:

create 3 vectors (Query, Key, Value)

from each of the encoder's input vectors



source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: detailed explanation

What are the “query”, “key”, and “value” vectors?

Self-Attention: detailed explanation

What are the “query”, “key”, and “value” vectors?

They’re abstractions that are useful for

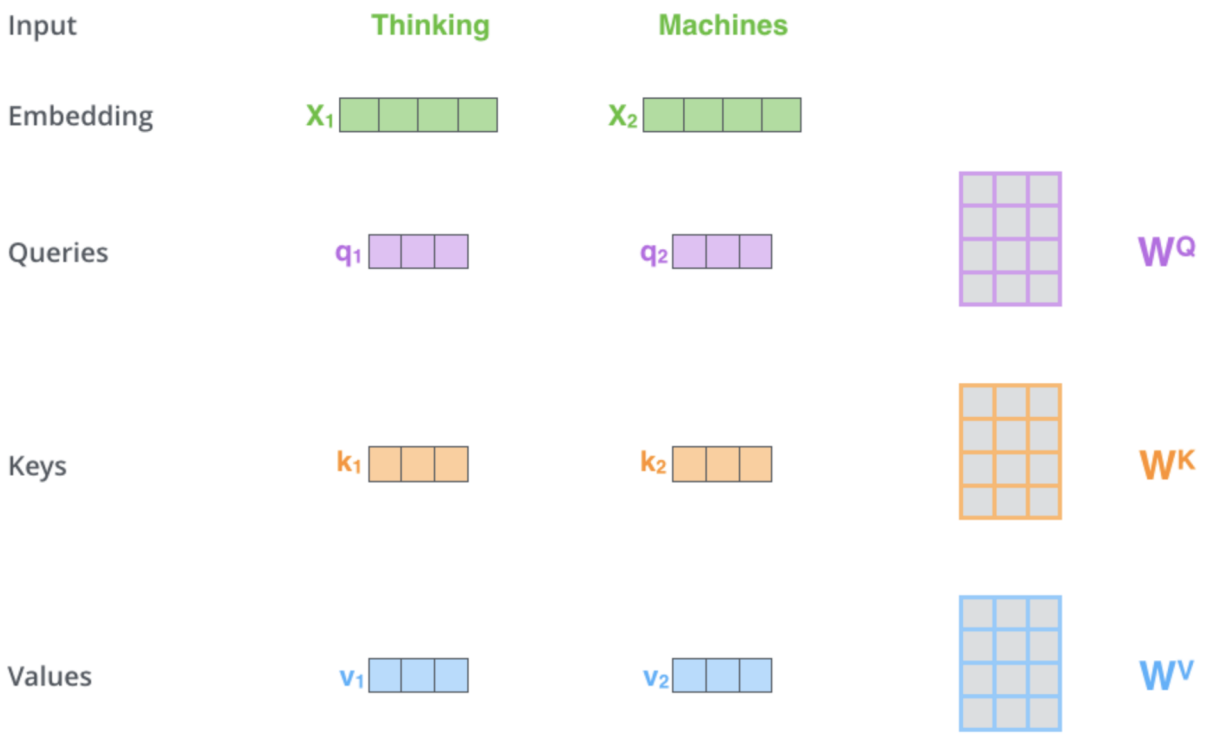
calculating and thinking about attention.

Self-Attention: detailed explanation

STEP 2:

calculate a score

(score each word of the input sentence against the current word)



source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: detailed explanation

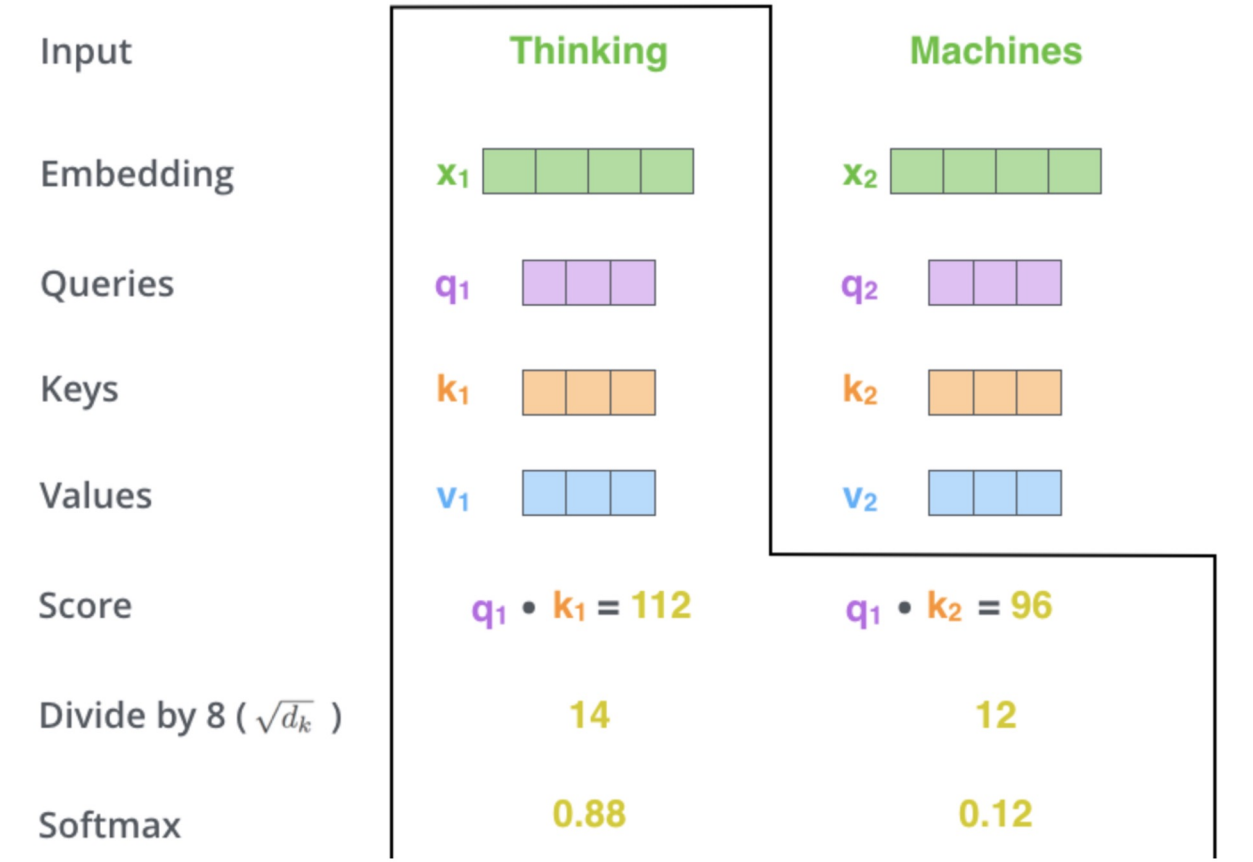
STEP 3:

divide the scores by 8

(the square root of the dimension of the key vectors)

STEP 4:

softmax



source: http://jalammar.github.io/illustrated-transformer/

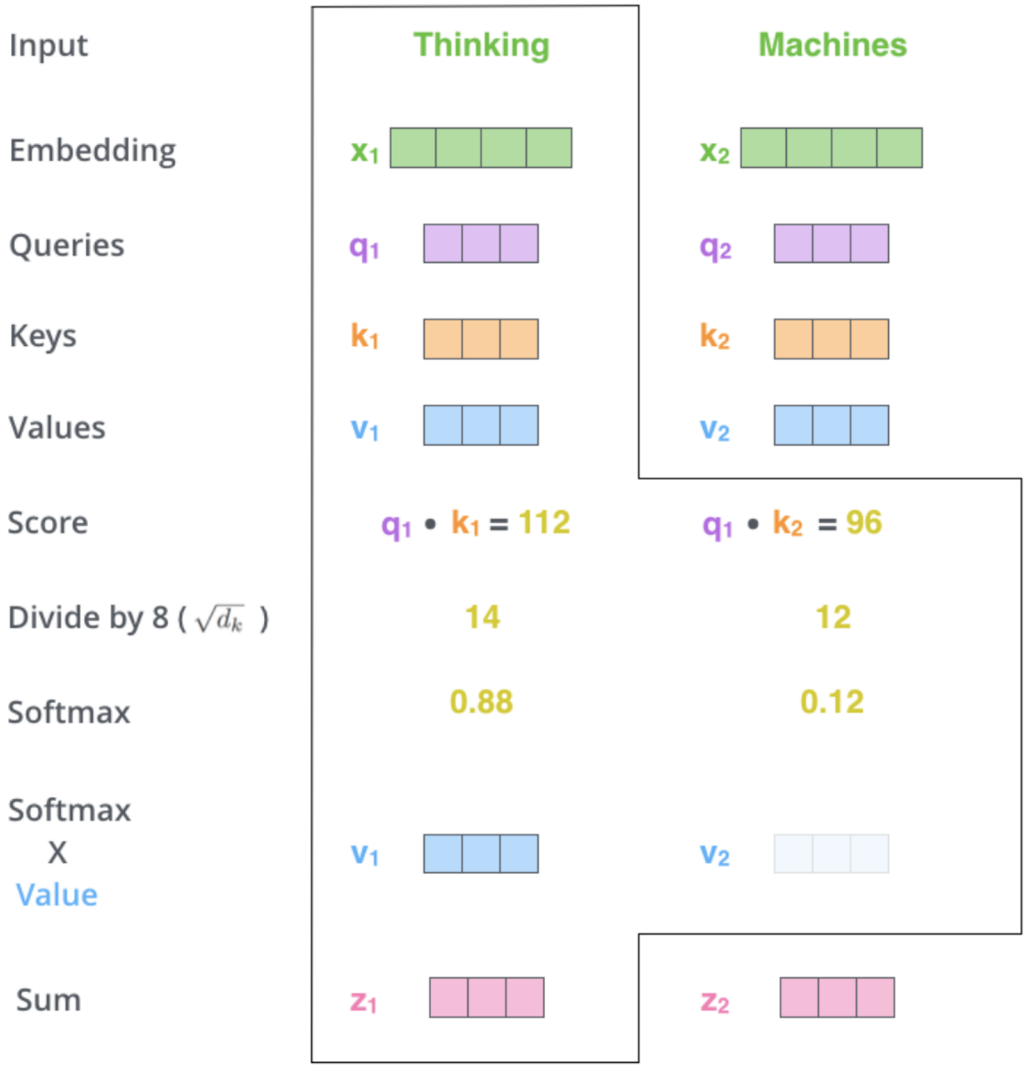
Self-Attention: detailed explanation

STEP 5:

multiply each value vector by the softmax score

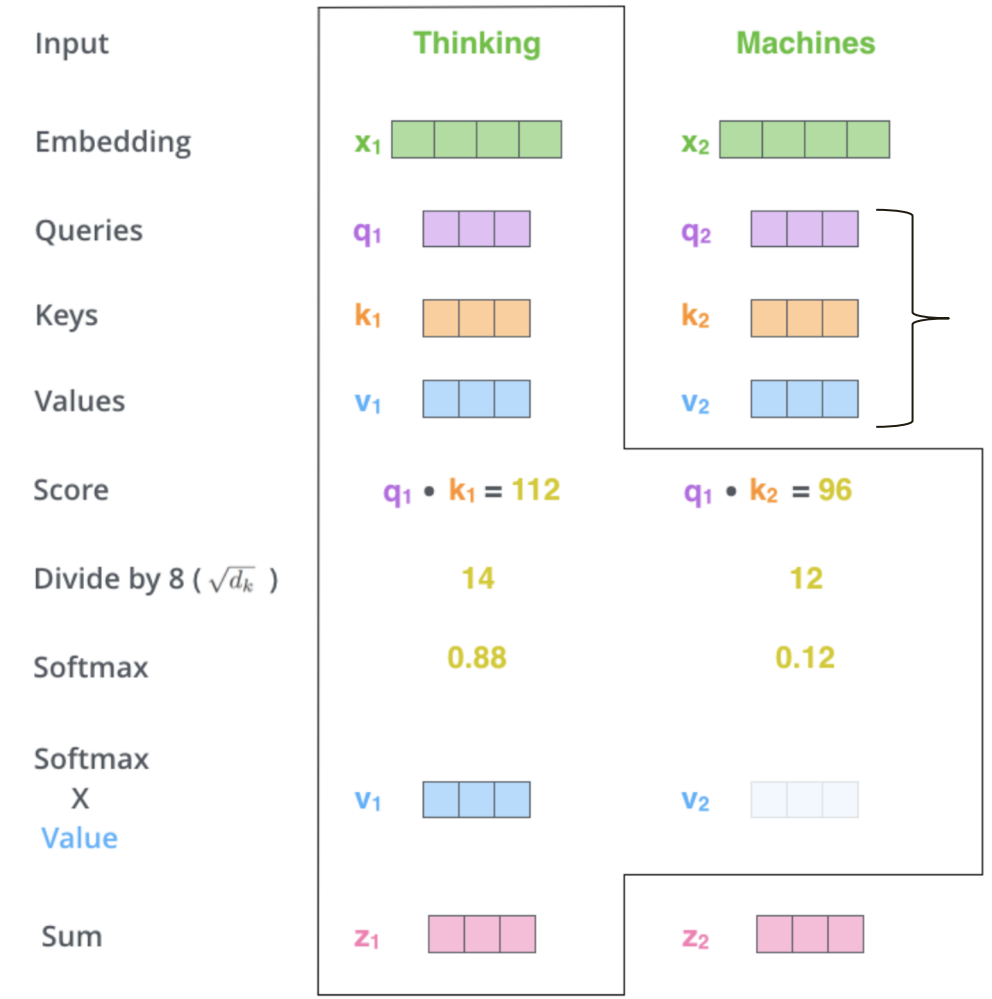
STEP 6:

sum up the weighted value vectors



source: http://jalammar.github.io/illustrated-transformer/

Self-Attention



STEP 1: create Query, Key, Value

STEP 2: calculate scores

STEP 3: divide by $\sqrt{d_k}$

STEP 4: softmax

STEP 5: multiply each value vector by the softmax score

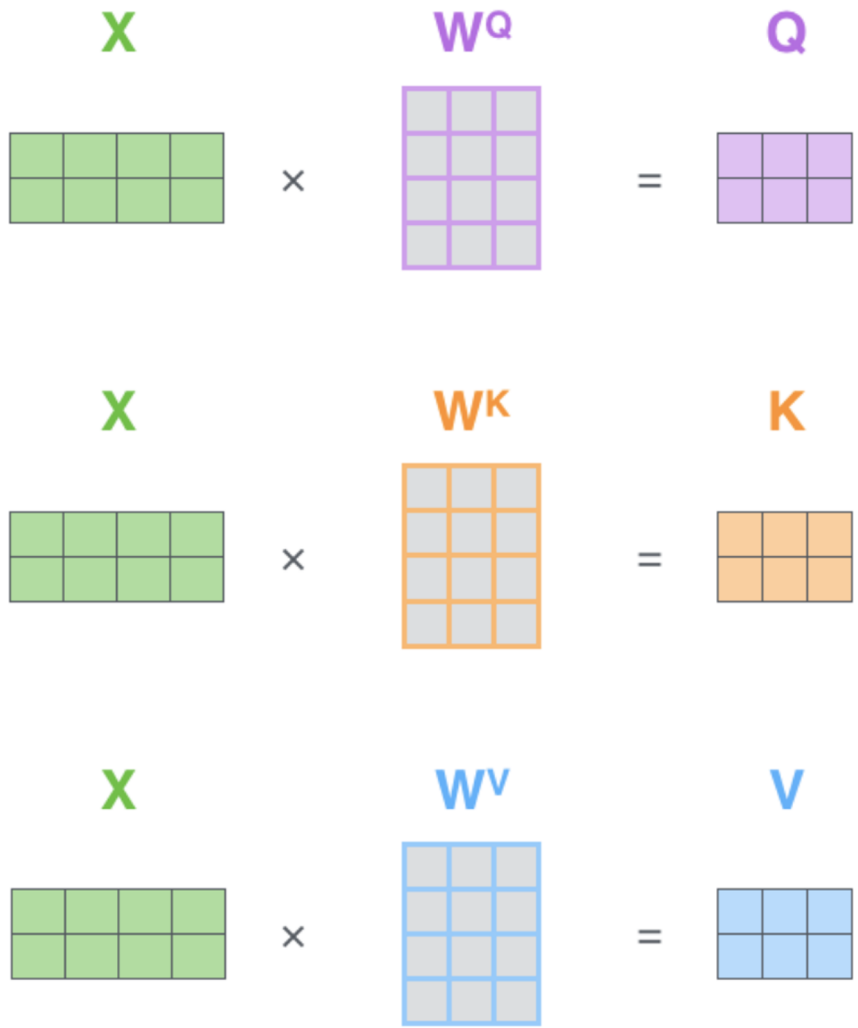
STEP 6: sum up the weighted value vectors

source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: Matrix Calculation

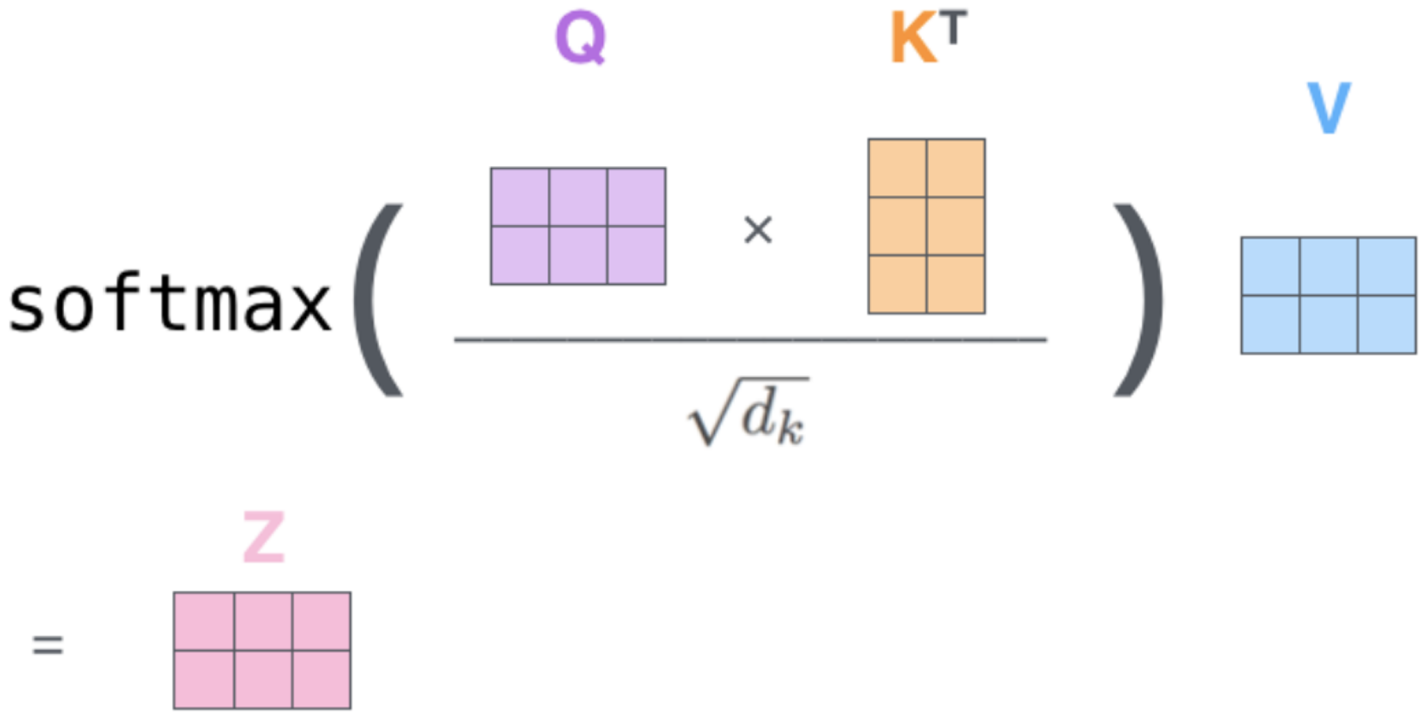
Pack embeddings into matrix **X**

Multiply **X** by weight matrices we've trained (**Wk, Wq, Wv**)



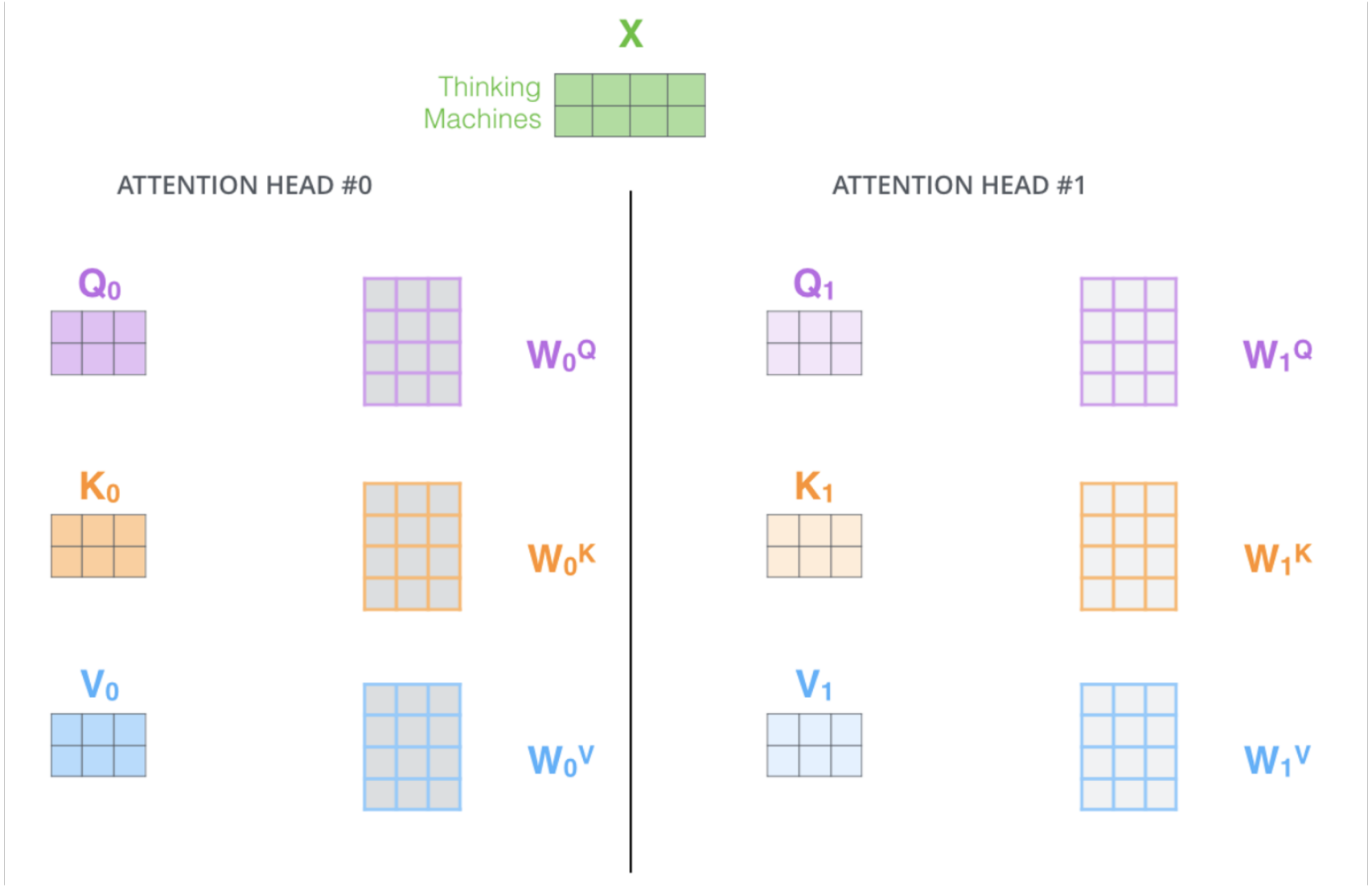
source: http://jalammar.github.io/illustrated-transformer/

Self-Attention: Matrix Calculation



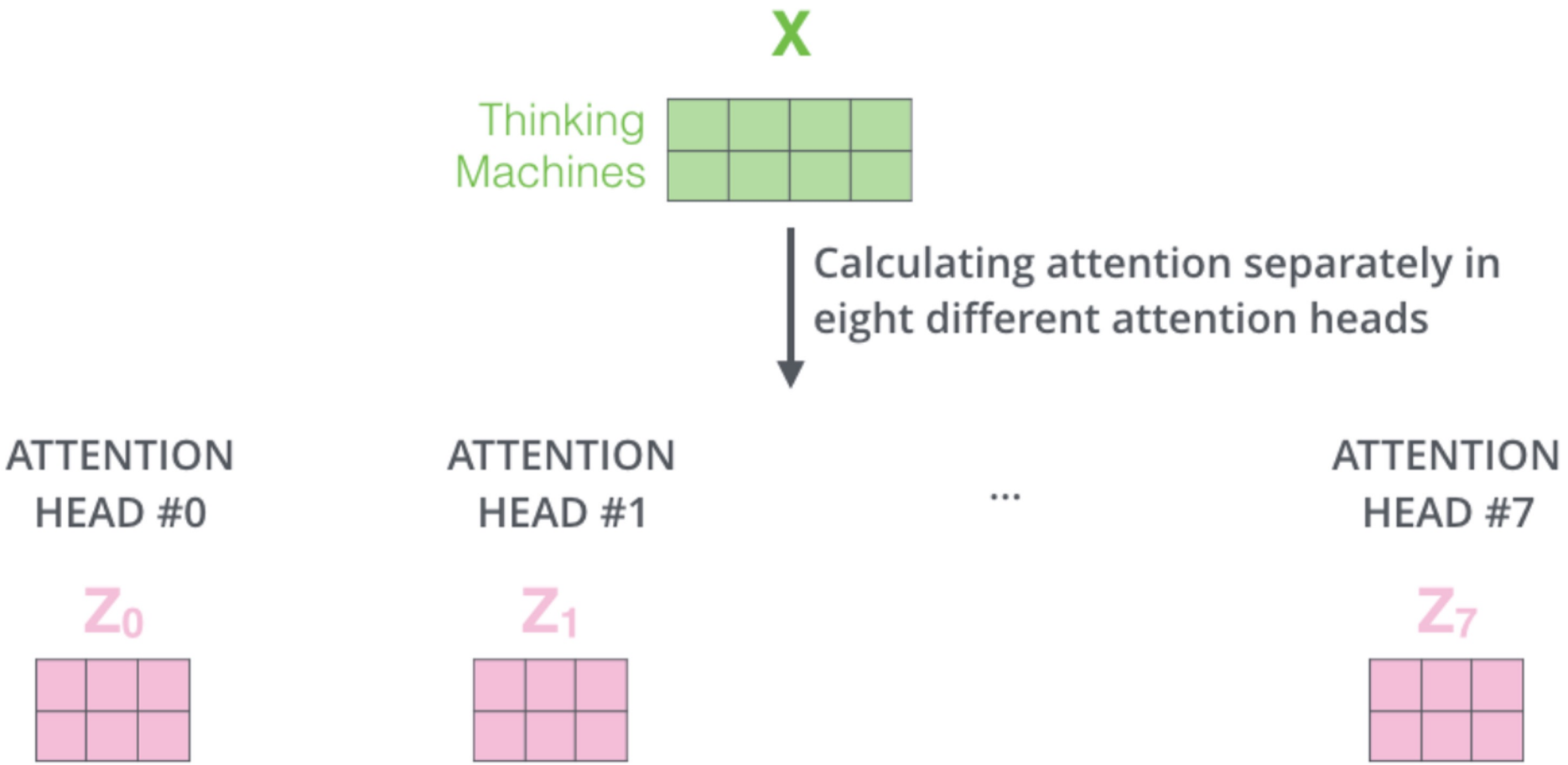
source: http://jalammar.github.io/illustrated-transformer/

Multi-Head Attention



source: http://jalammar.github.io/illustrated-transformer/

Multi-Head Attention



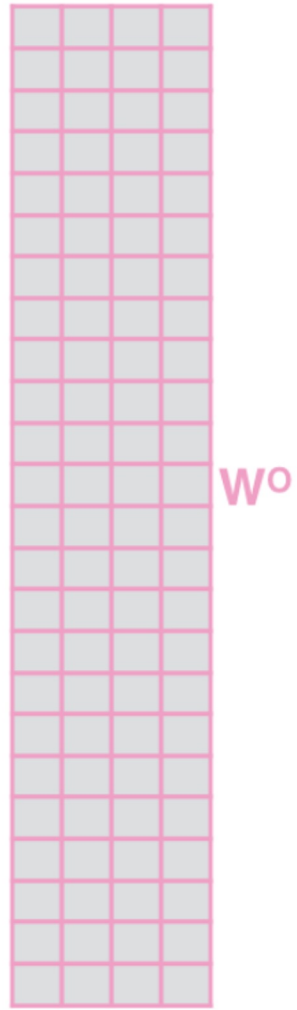
source: http://jalammar.github.io/illustrated-transformer/

Multi-Head Attention

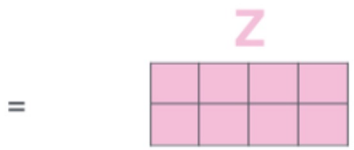
1) Concatenate all the attention heads

2) Multiply with a weight matrix W^O that was trained jointly with the model

X



3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN



source: http://jalammar.github.io/illustrated-transformer/

1) This is our input sentence*

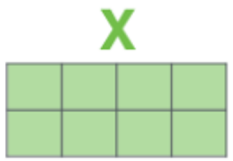
2) We embed each word*

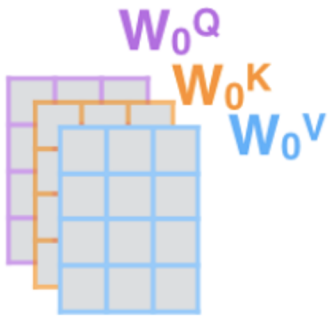
3) Split into 8 heads. We multiply X or R with weight matrices

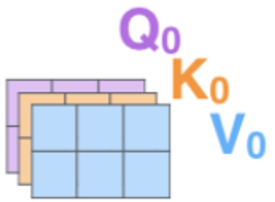
4) Calculate attention using the resulting Q/K/V matrices

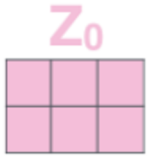
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

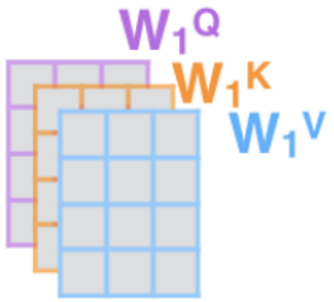
Thinking Machines

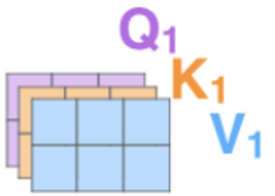


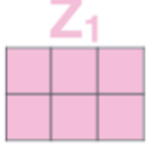


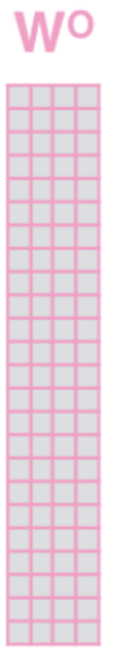


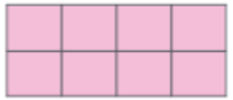




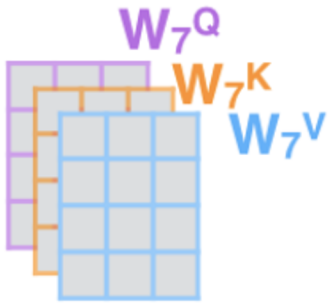


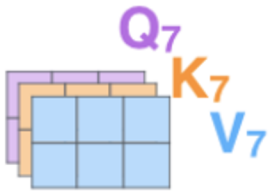


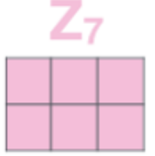




* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

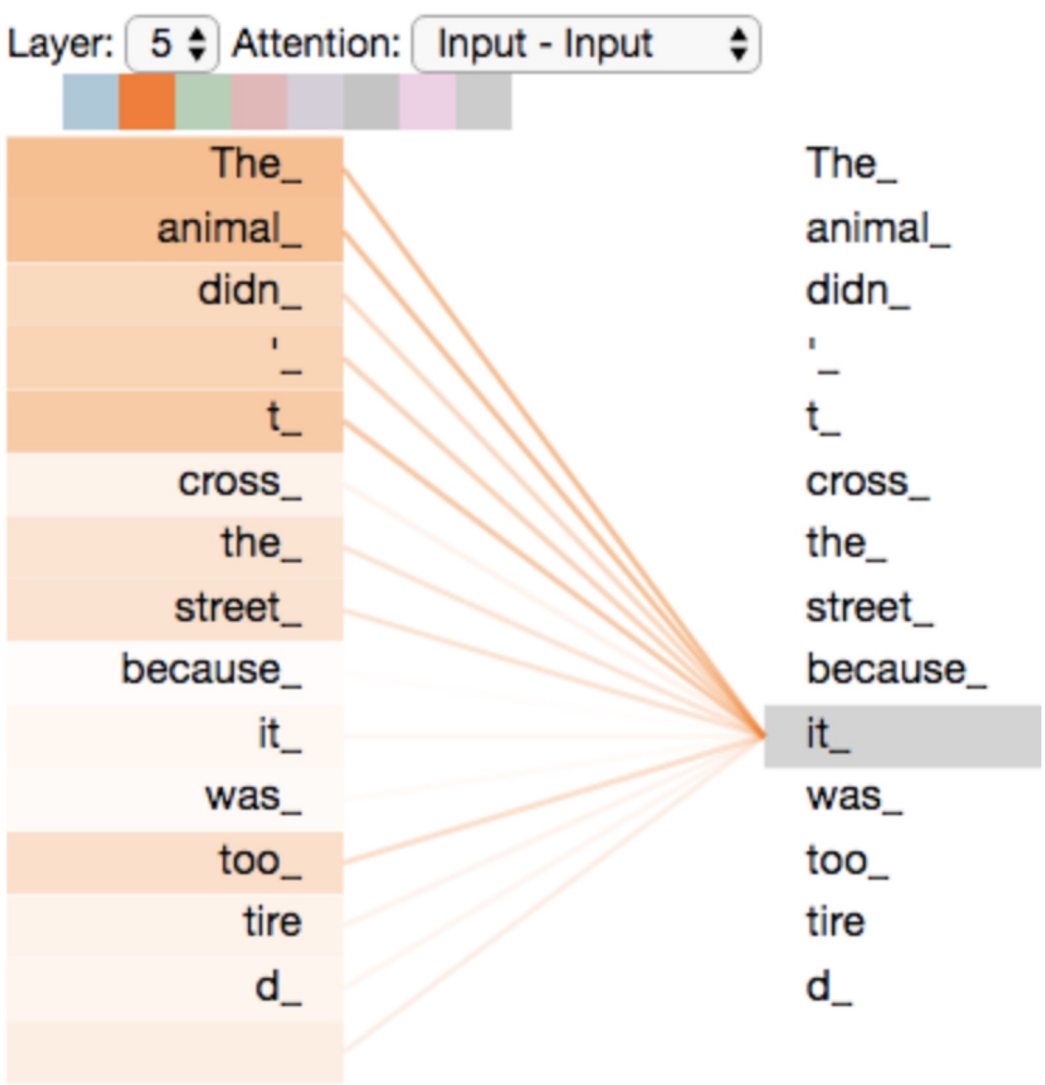


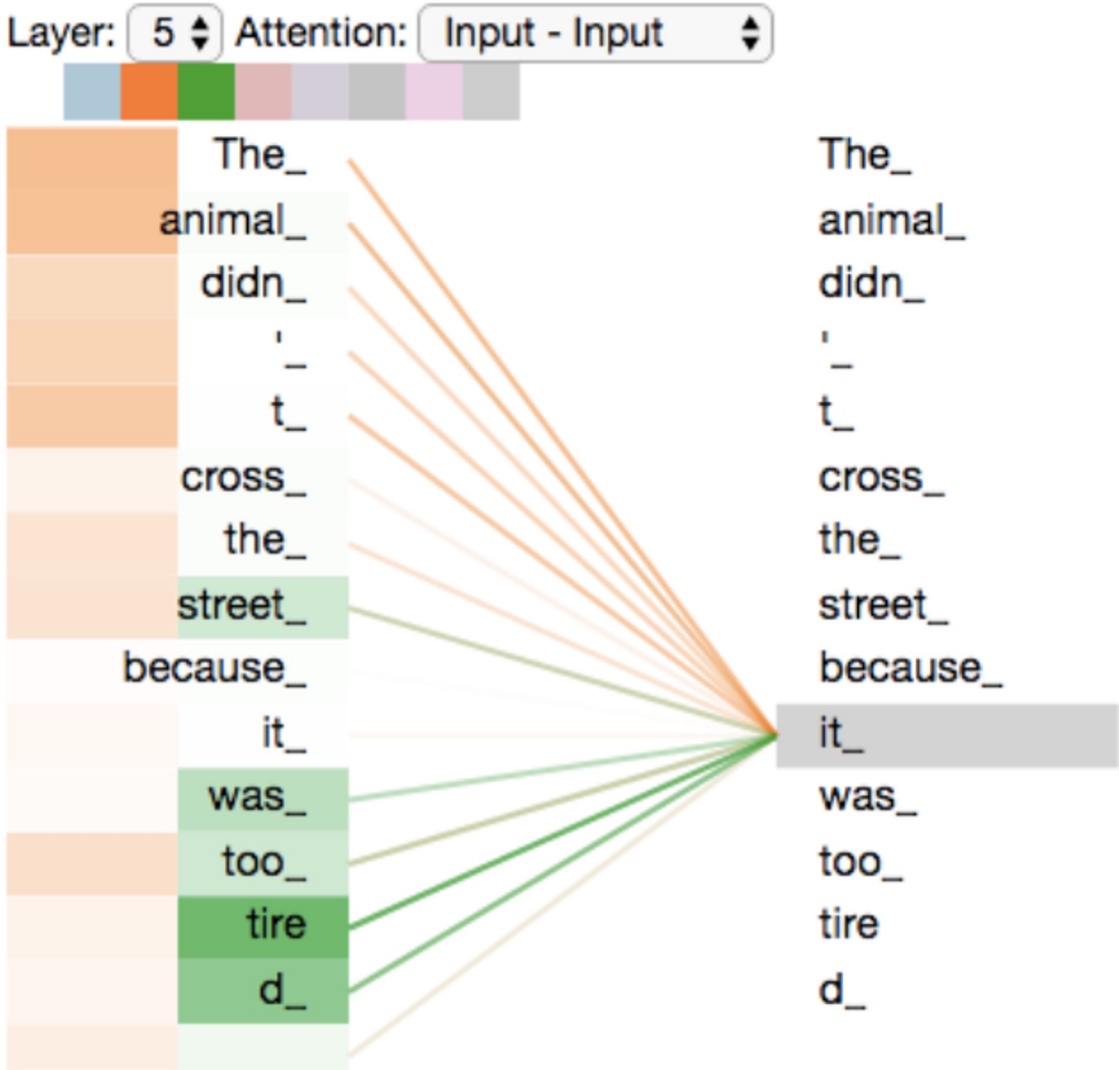




source: http://jalammar.github.io/illustrated-transformer/

Multi-Head Attention





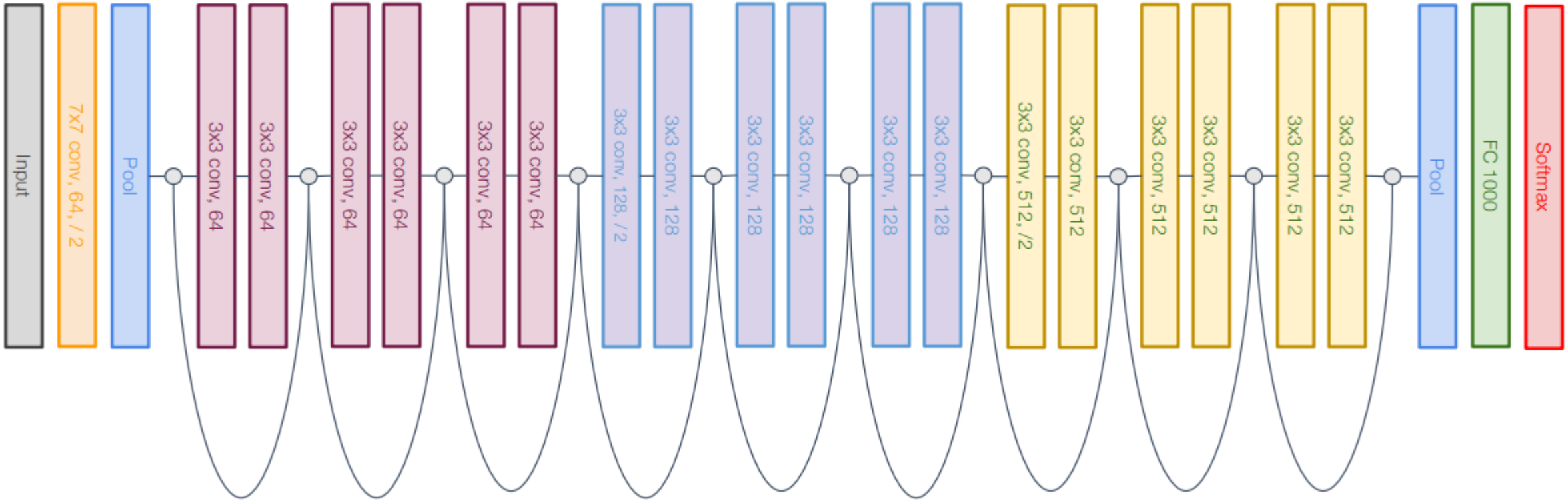
source: http://jalammar.github.io/illustrated-transformer/

Attention для изображений

04

Idea #1: Add attention to existing CNNs

Start from standard CNN architecture (e.g. ResNet)



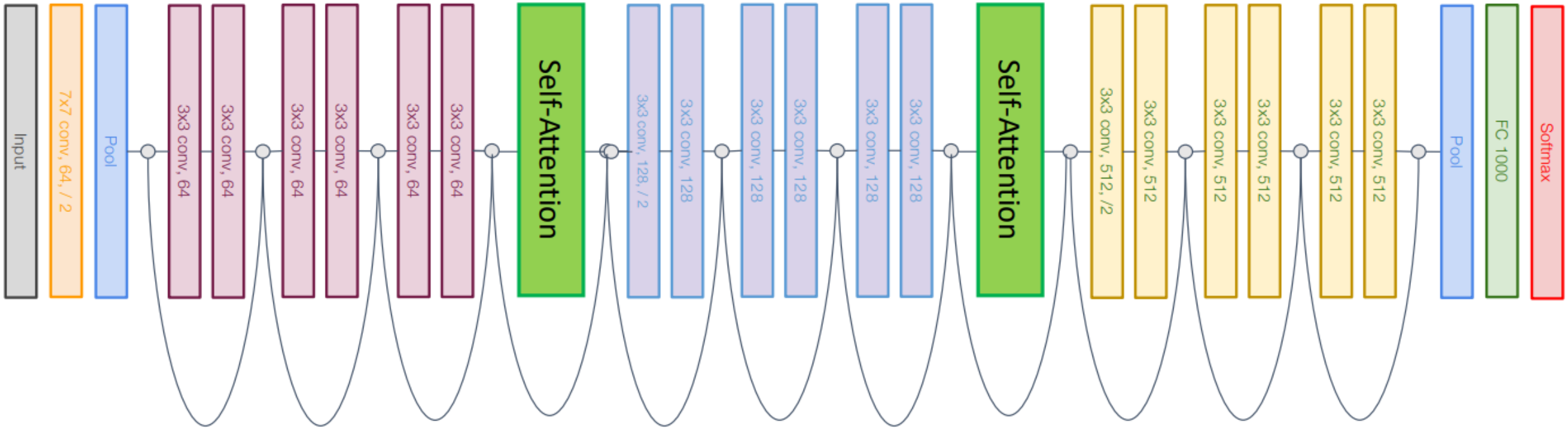
Zhang et al, ”Self-Attention Generative Adversarial Networks”, ICML 2018
Wang et al, ”Non-local Neural Networks”, CVPR 2018

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #1: Add attention to existing CNNs

Start from standard CNN architecture (e.g. ResNet)

Add Self-Attention blocks between existing ResNet blocks



Zhang et al, ”Self-Attention Generative Adversarial Networks”, ICML 2018
Wang et al, ”Non-local Neural Networks”, CVPR 2018

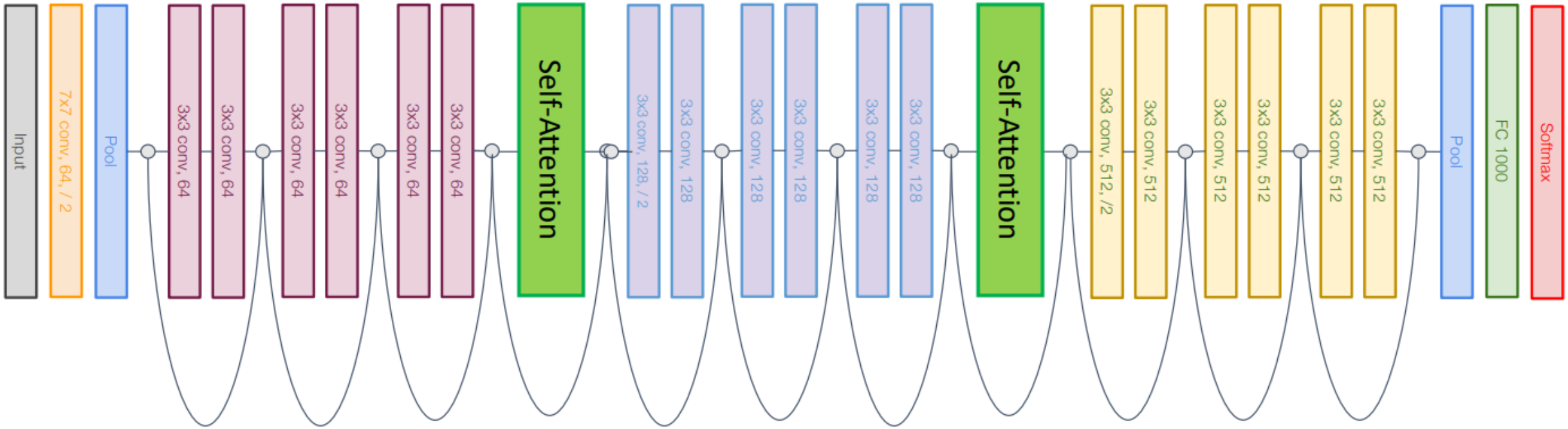
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #1: Add attention to existing CNNs

Model is still a CNN!
Can we replace convolution entirely?

Start from standard CNN architecture (e.g. ResNet)

Add Self-Attention blocks between existing ResNet blocks

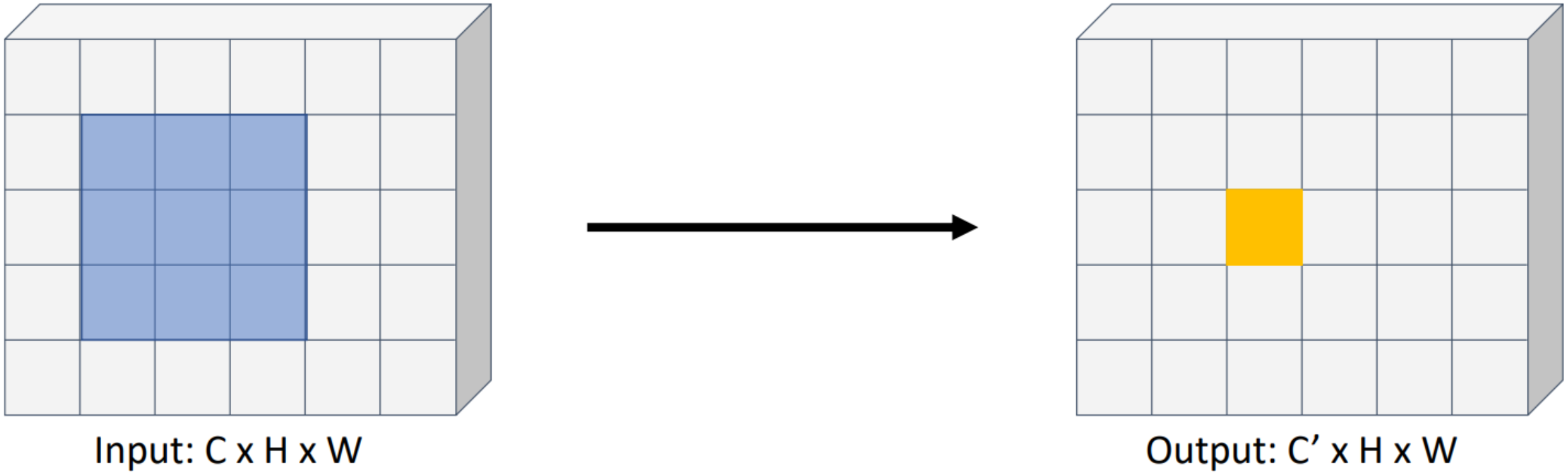


Zhang et al, "Self-Attention Generative Adversarial Networks", ICML 2018
Wang et al, "Non-local Neural Networks", CVPR 2018

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #2: Replace Convolution with “Local Attention”

Convolution: Output at each position is inner product of conv kernel with receptive field in input

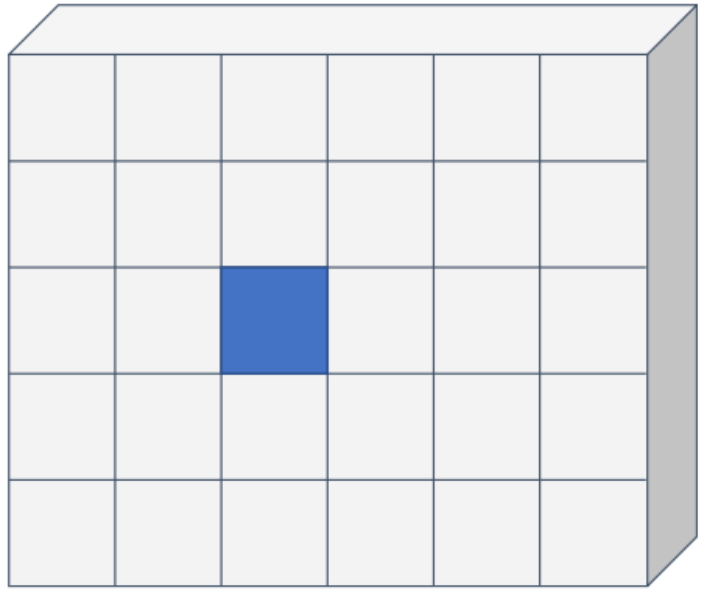


Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

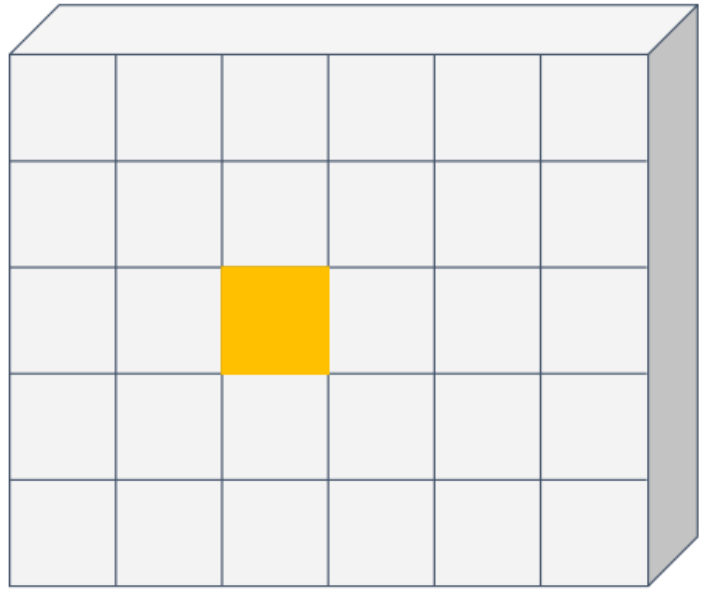
Idea #2: Replace Convolution with “Local Attention”

Map center of receptive field to **query**



Input: C x H x W

Query: D_Q



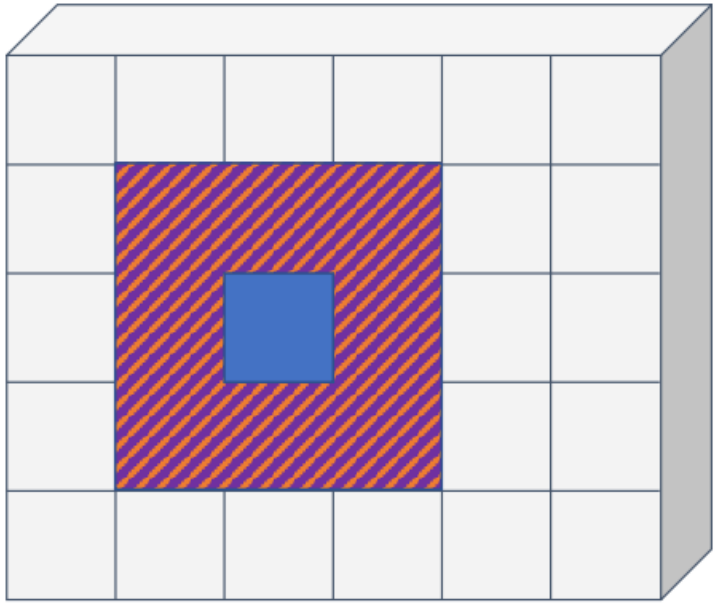
Output: C’ x H x W

Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

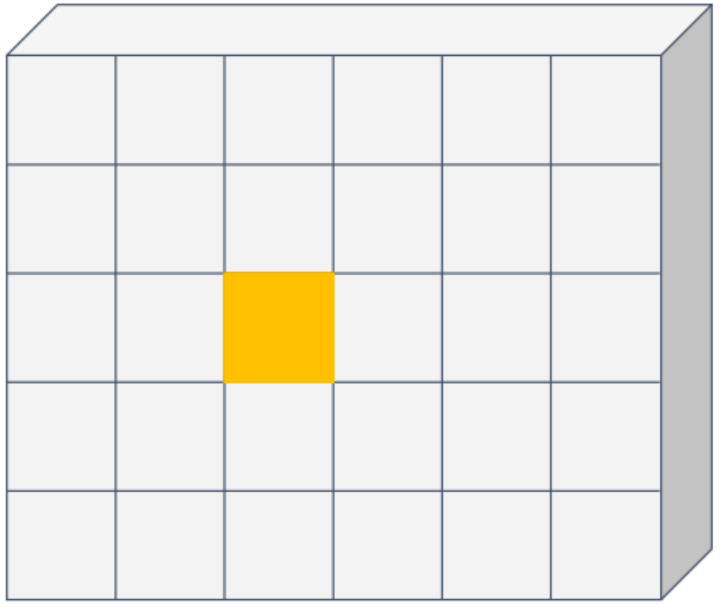
Idea #2: Replace Convolution with “Local Attention”

Map center of receptive field to **query**
Map each element in receptive field to **key** and **value**



Input: C x H x W

Query: D_Q
Keys: R x R x D_Q
Values: R x R x C’



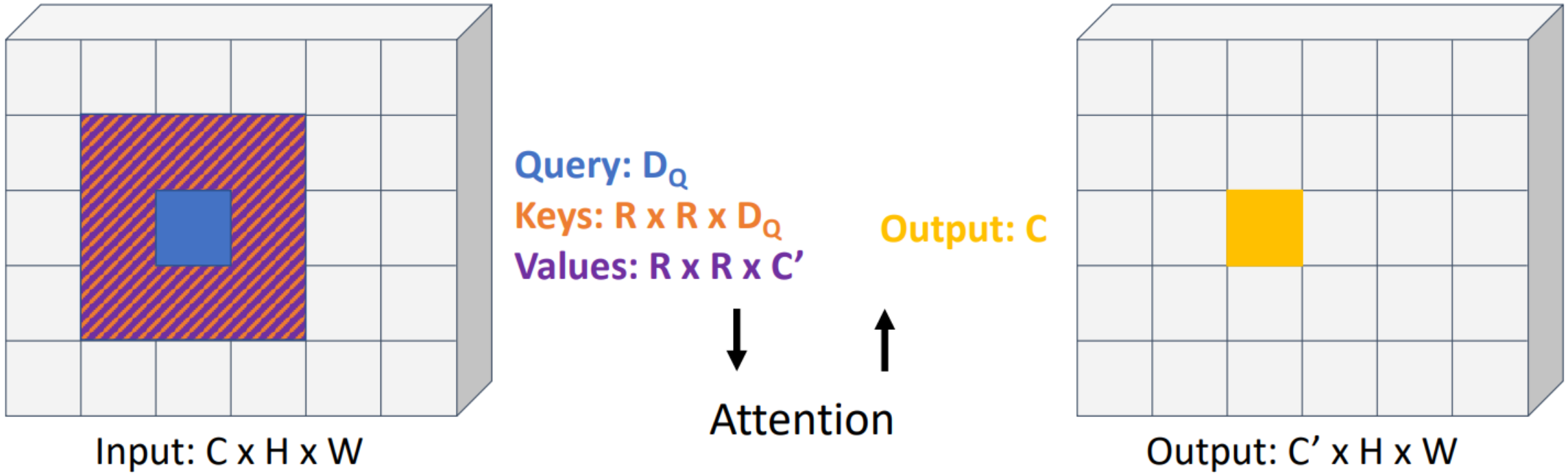
Output: C’ x H x W

Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #2: Replace Convolution with “Local Attention”

Map center of receptive field to **query**
Map each element in receptive field to **key** and **value**
Compute **output** using attention



Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #2: Replace Convolution with “Local Attention”

Map center of receptive field to **query**
Map each element in receptive field to **key** and **value**
Compute **output** using attention
Replace all conv in ResNet with local attention

LR = “Local Relation”

stage	output	ResNet-50	LR-Net-50 (7×7, m=8)
res1	112×112	7×7 conv, 64, stride 2	**1×1, 64** **7×7 LR, 64, stride 2**
res2	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
		[1×1, 64 3×3 conv, 64 1×1, 256] ×3	[1×1, 100 **7×7 LR, 100** 1×1, 256] ×3
res3	28×28	[1×1, 128 3×3 conv, 128 1×1, 512] ×4	[1×1, 200 **7×7 LR, 200** 1×1, 512] ×4
res4	14×14	[1×1, 256 3×3 conv, 256 1×1, 1024] ×6	[1×1, 400 **7×7 LR, 400** 1×1, 1024] ×6
res5	7×7	[1×1, 512 3×3 conv, 512 1×1, 2048] ×3	[1×1, 800 **7×7 LR, 800** 1×1, 2048] ×3
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params		**25.5**×10^6	**23.3**×10^6
FLOPs		**4.3**×10^9	**4.3**×10^9

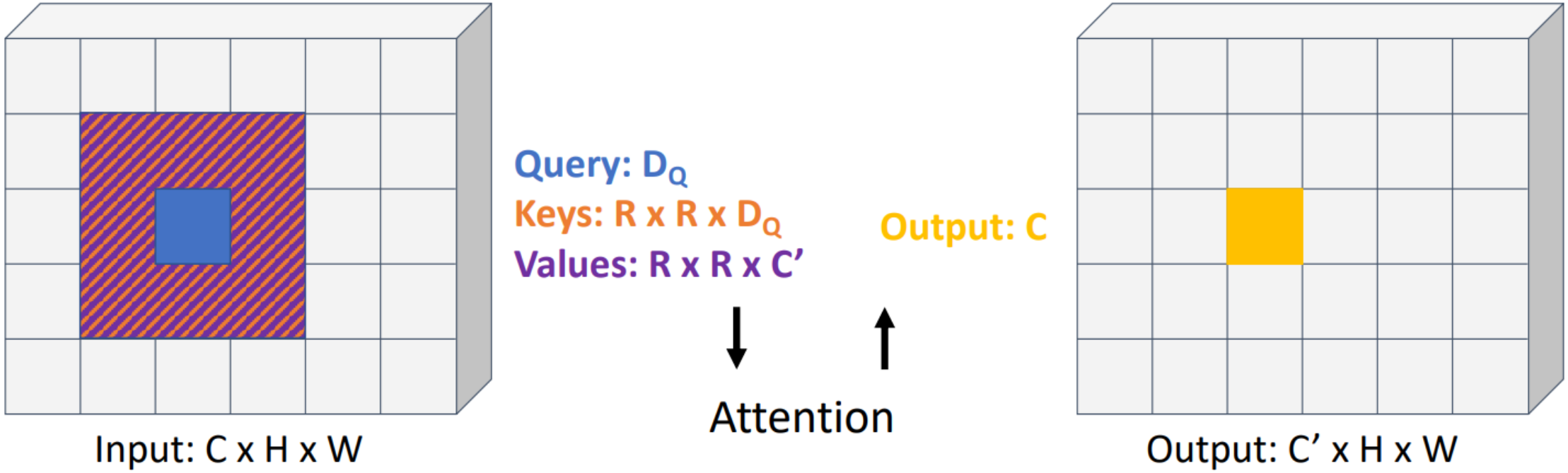
Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #2: Replace Convolution with “Local Attention”

Map center of receptive field to **query**
Map each element in receptive field to **key** and **value**
Compute **output** using attention
Replace all conv in ResNet with local attention

Lots of tricky details, hard to implement, only marginally better than ResNets

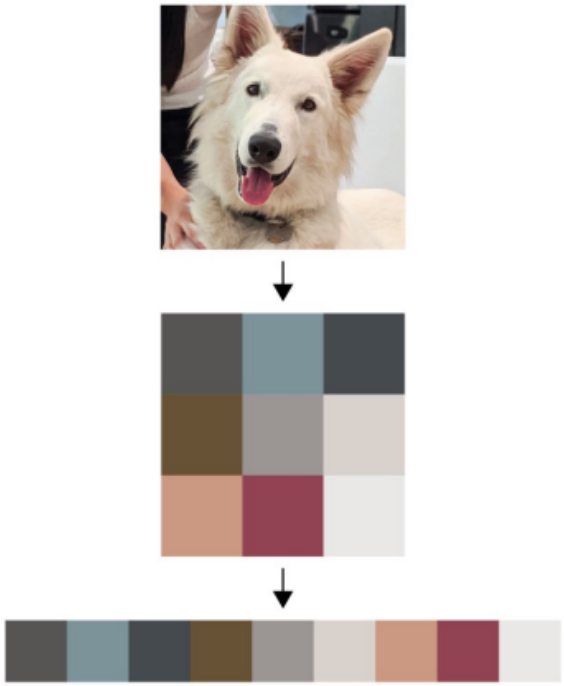


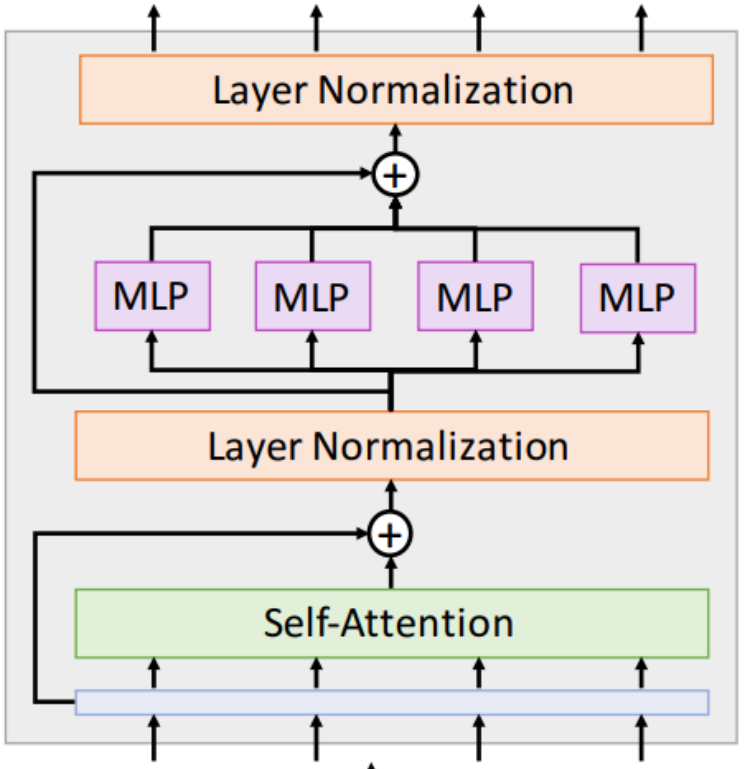
Hu et al, “Local Relation Networks for Image Recognition”, ICCV 2019; Ramachandran et al, “Stand-Alone Self-Attention in Vision Models”, NeurIPS 2019

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #3: Standard Transformer on Pixels

Treat an image as a set of pixel values





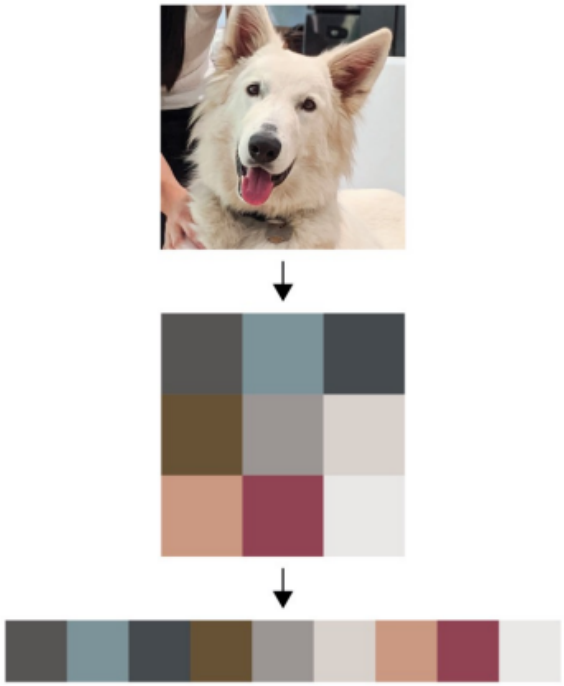
Feed as input to standard Transformer

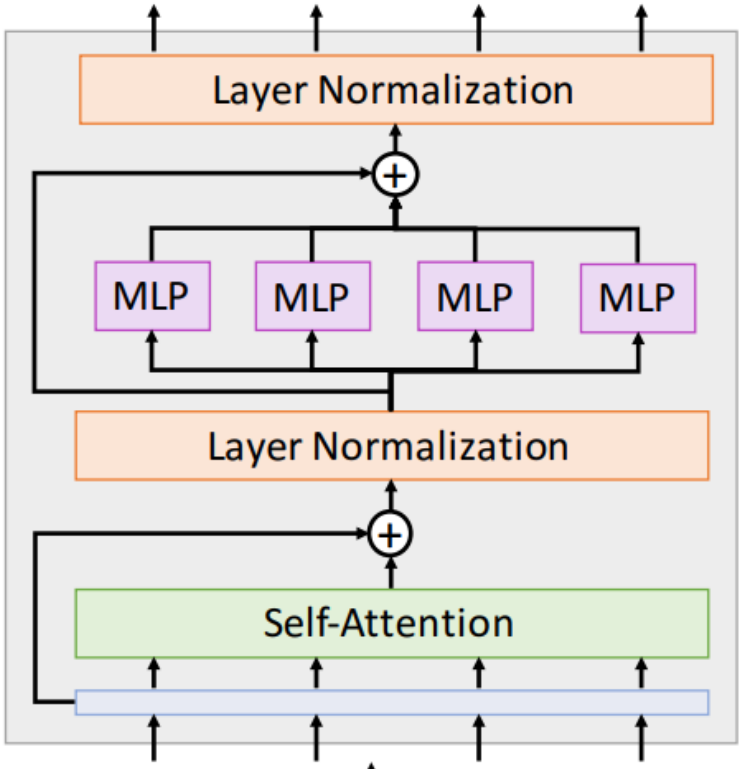
Chen et al, “Generative Pretraining from Pixels”, ICML 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #3: Standard Transformer on Pixels

Treat an image as a set of pixel values





Feed as input to standard Transformer

Problem: Memory use!

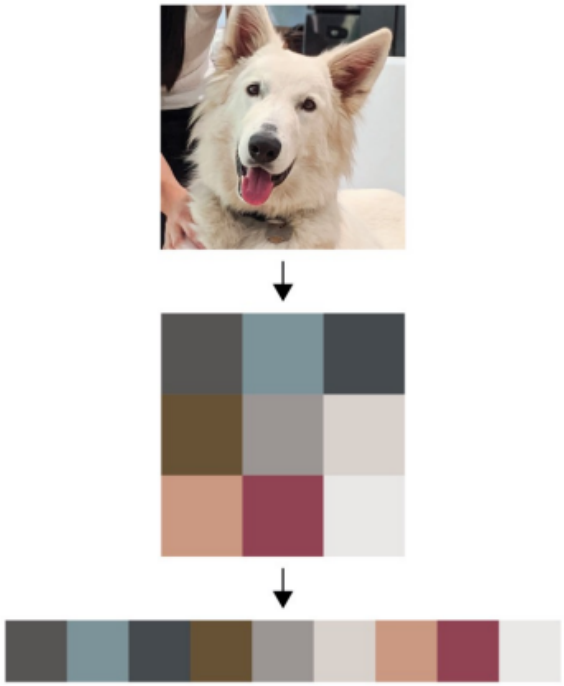
R x R image needs R^4 elements per attention matrix

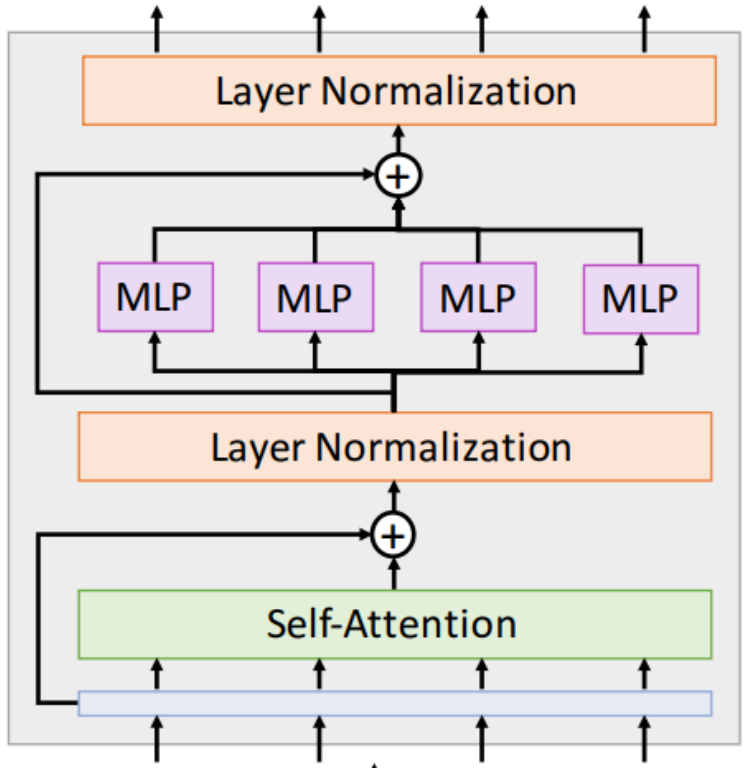
Chen et al, “Generative Pretraining from Pixels”, ICML 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #3: Standard Transformer on Pixels

Treat an image as a set of pixel values





Feed as input to standard Transformer

Problem: Memory use!

R x R image needs R^4 elements per attention matrix

R=128, 48 layers, 16 heads per layer takes 768GB of memory for attention matrices for a single example…

Chen et al, “Generative Pretraining from Pixels”, ICML 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

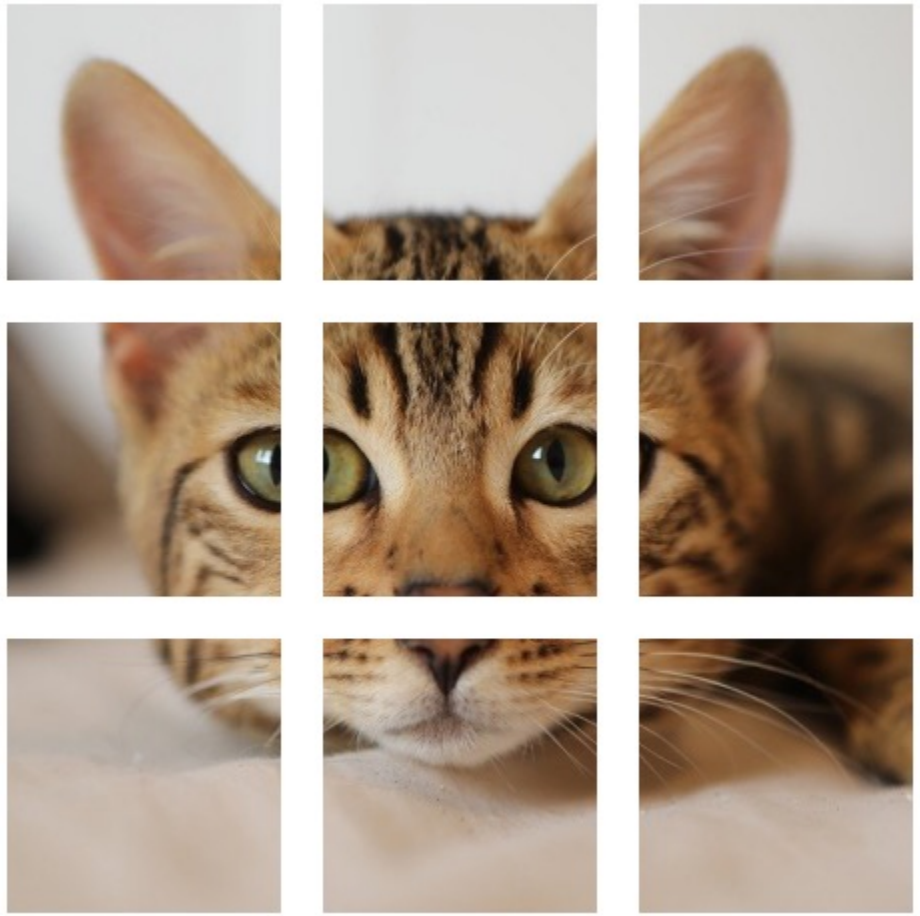
Idea #4: Standard Transformer on Patches

Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches



Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

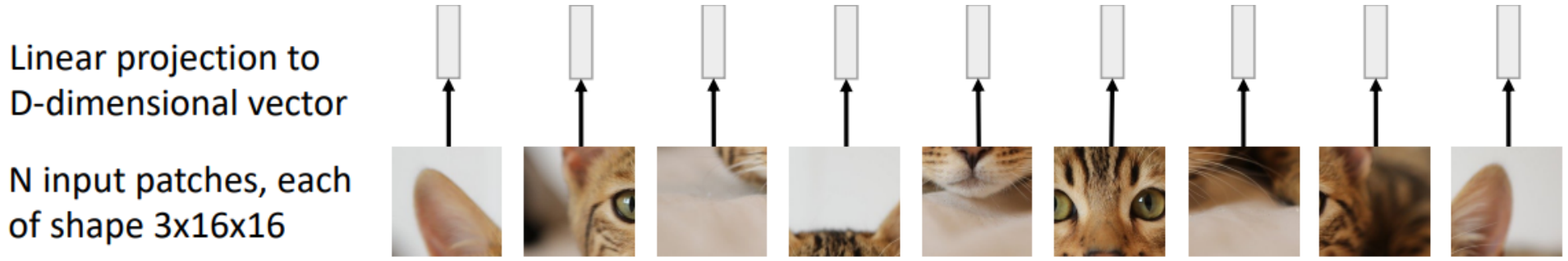
N input patches, each of shape 3x16x16

Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

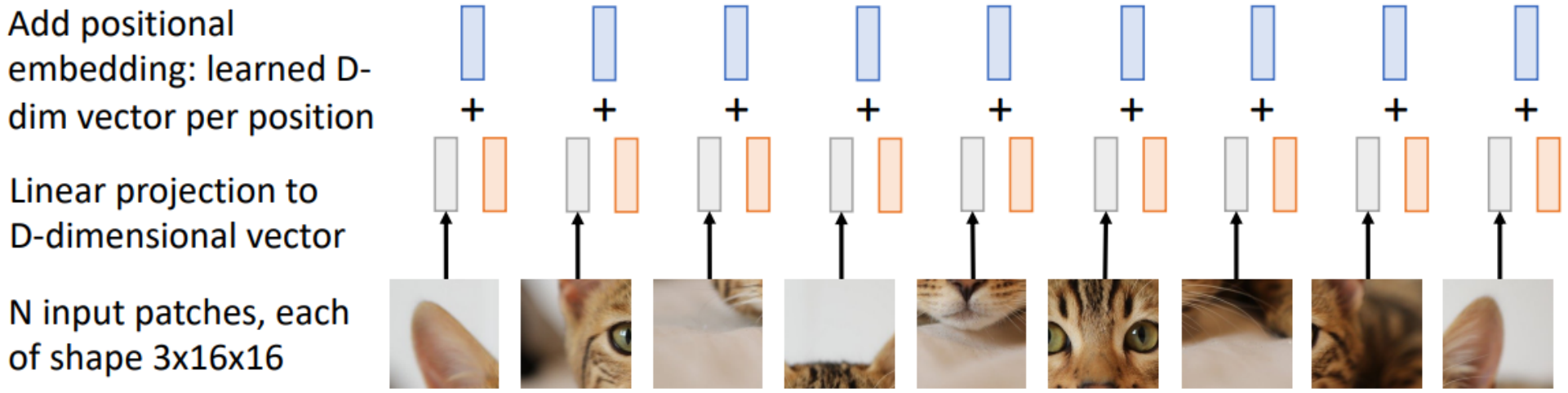


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

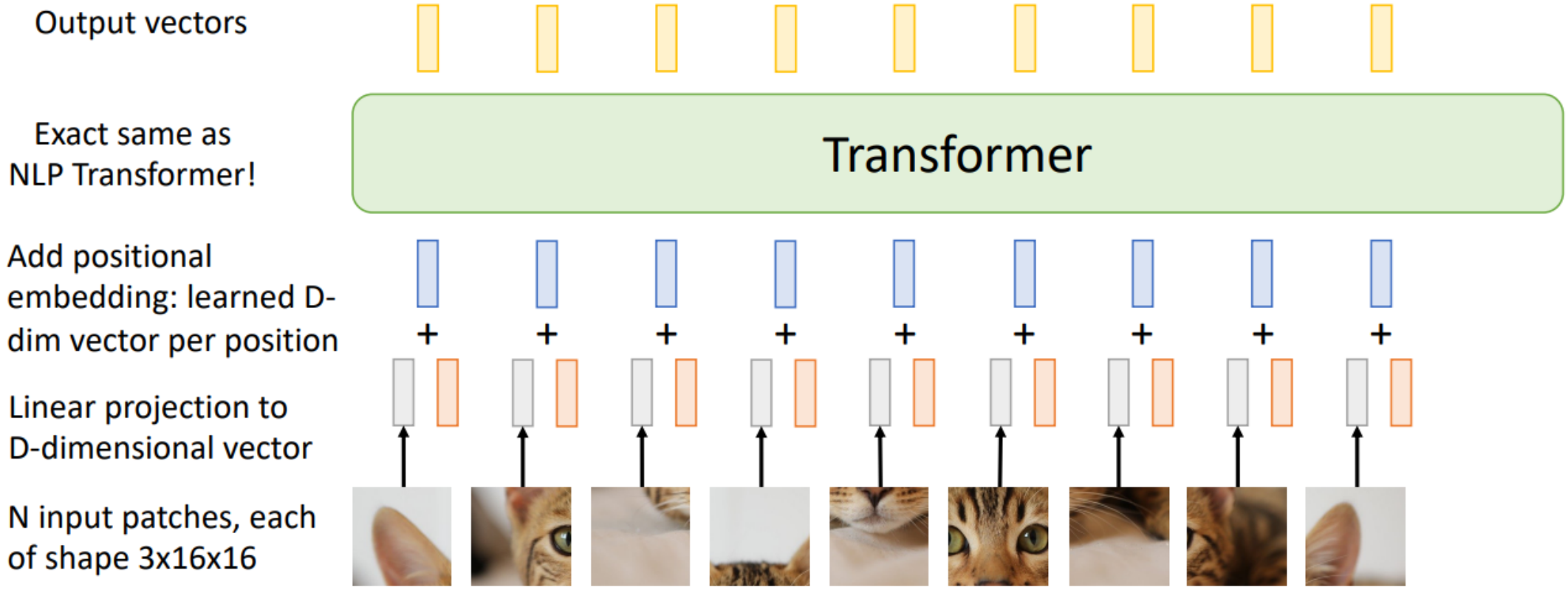


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

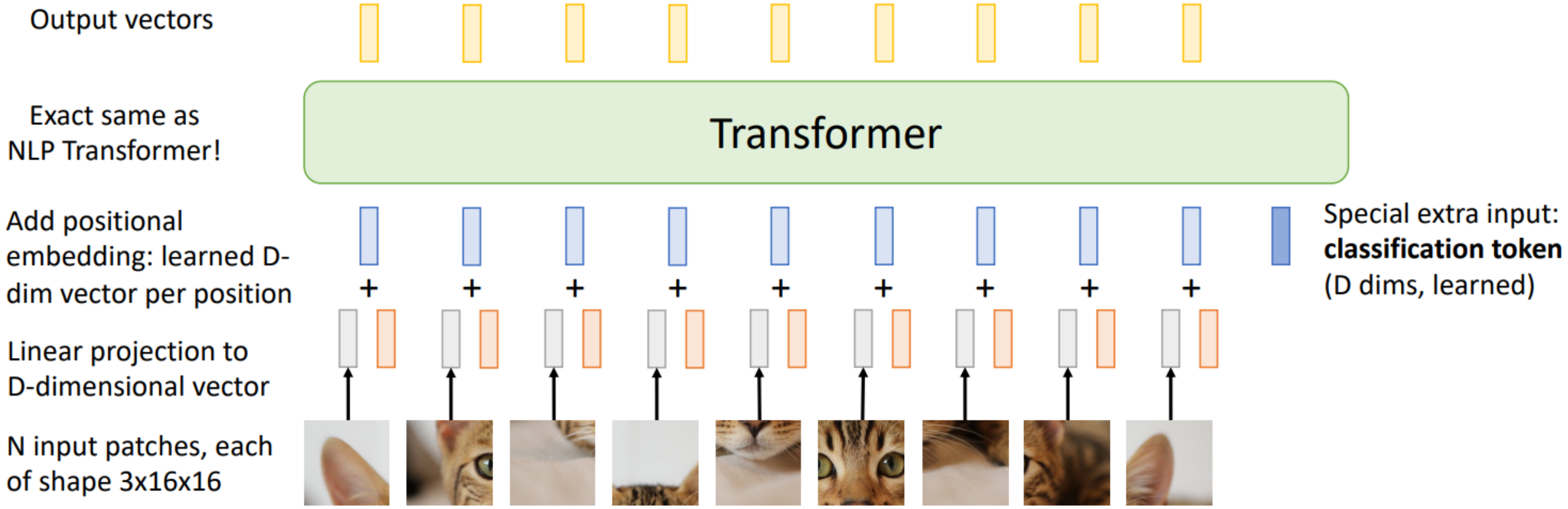


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

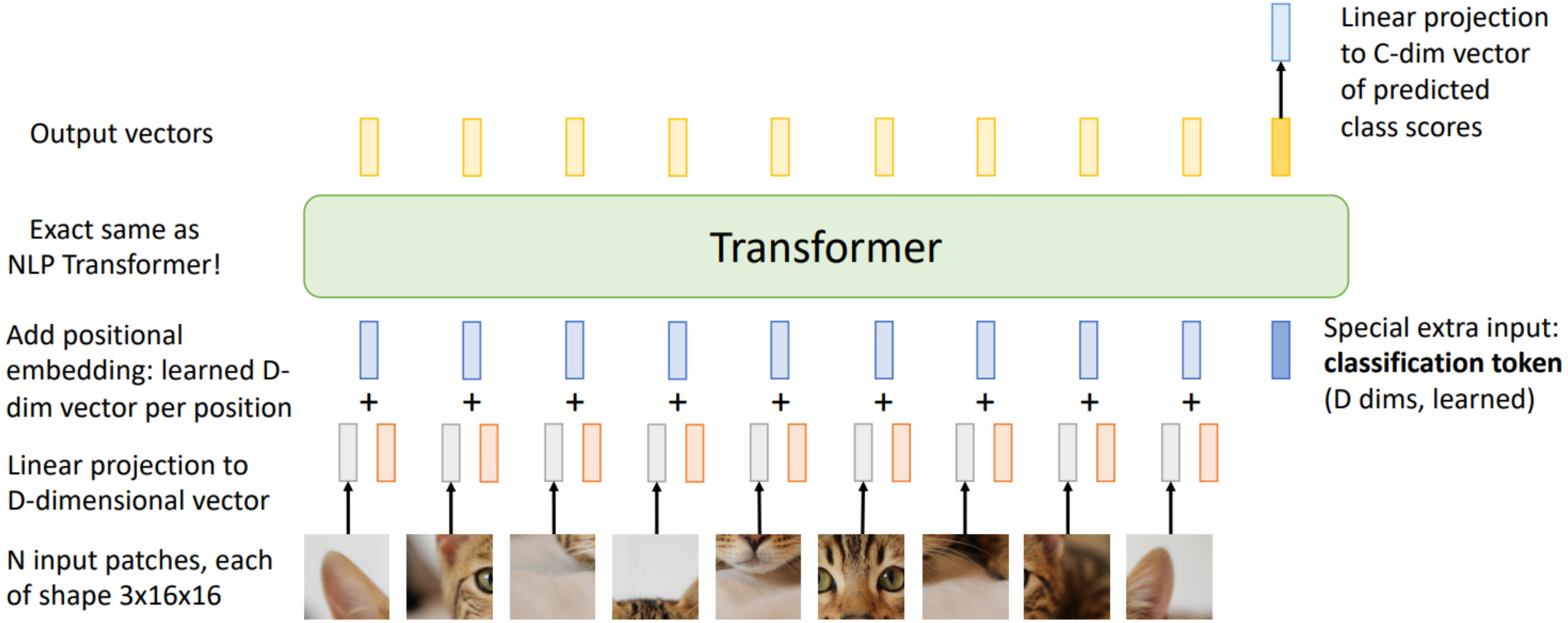


Dosovitskiy et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Idea #4: Standard Transformer on Patches

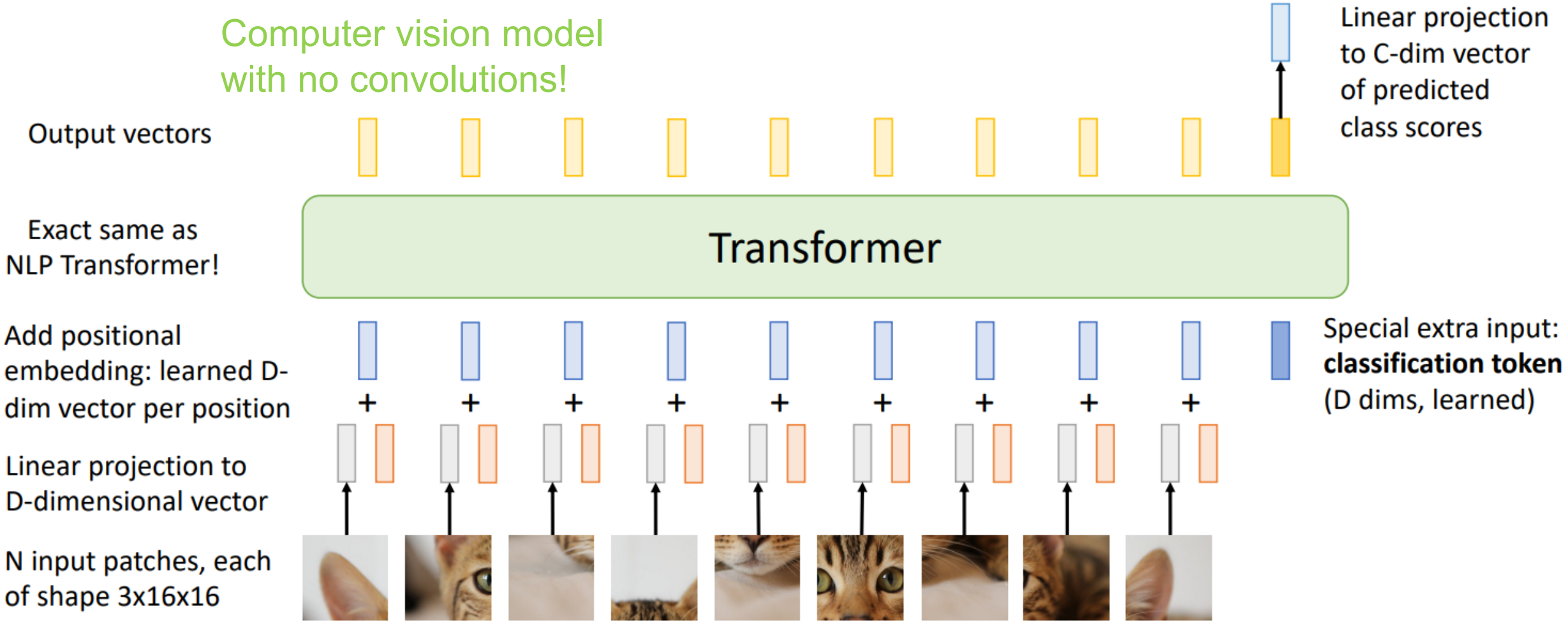


Dosovitskiy et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)



Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

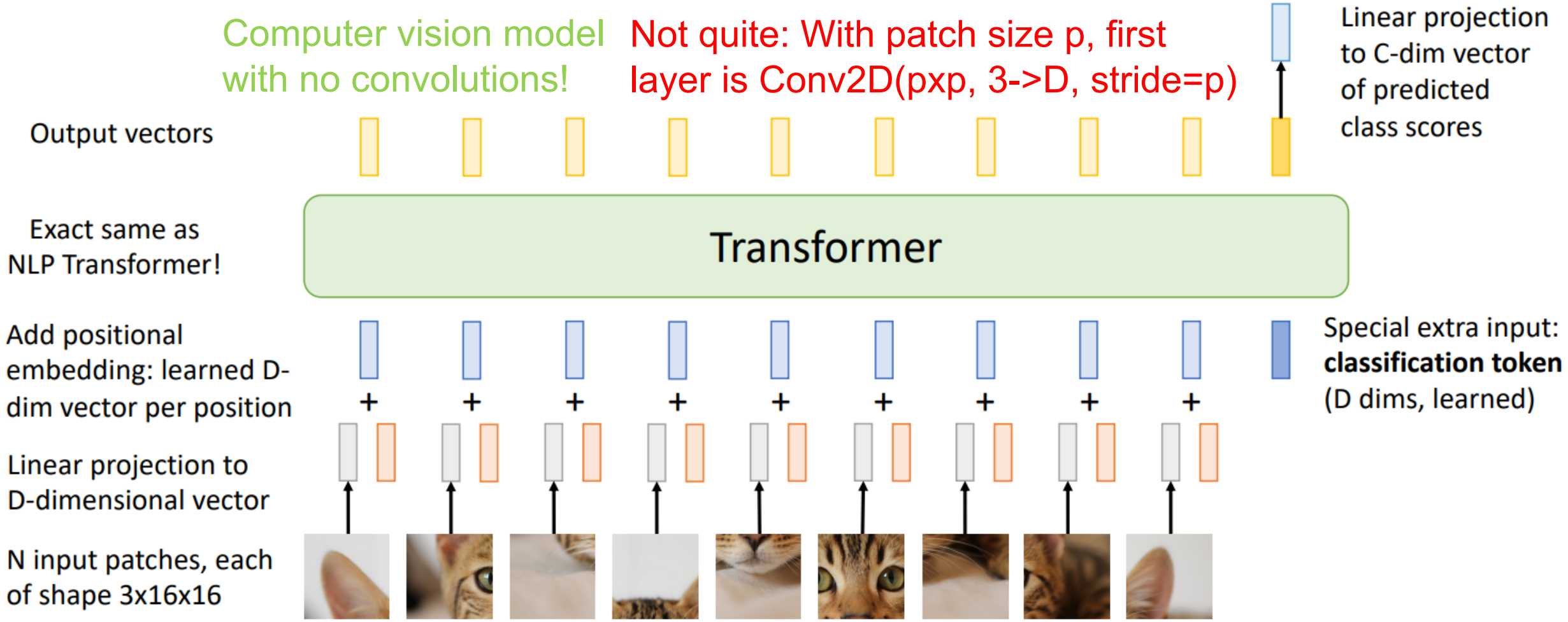
Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)

Computer vision model with no convolutions!

Not quite: With patch size p, first layer is Conv2D(pxp, 3->D, stride=p)



Dosovitskiy et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", ICLR 2021

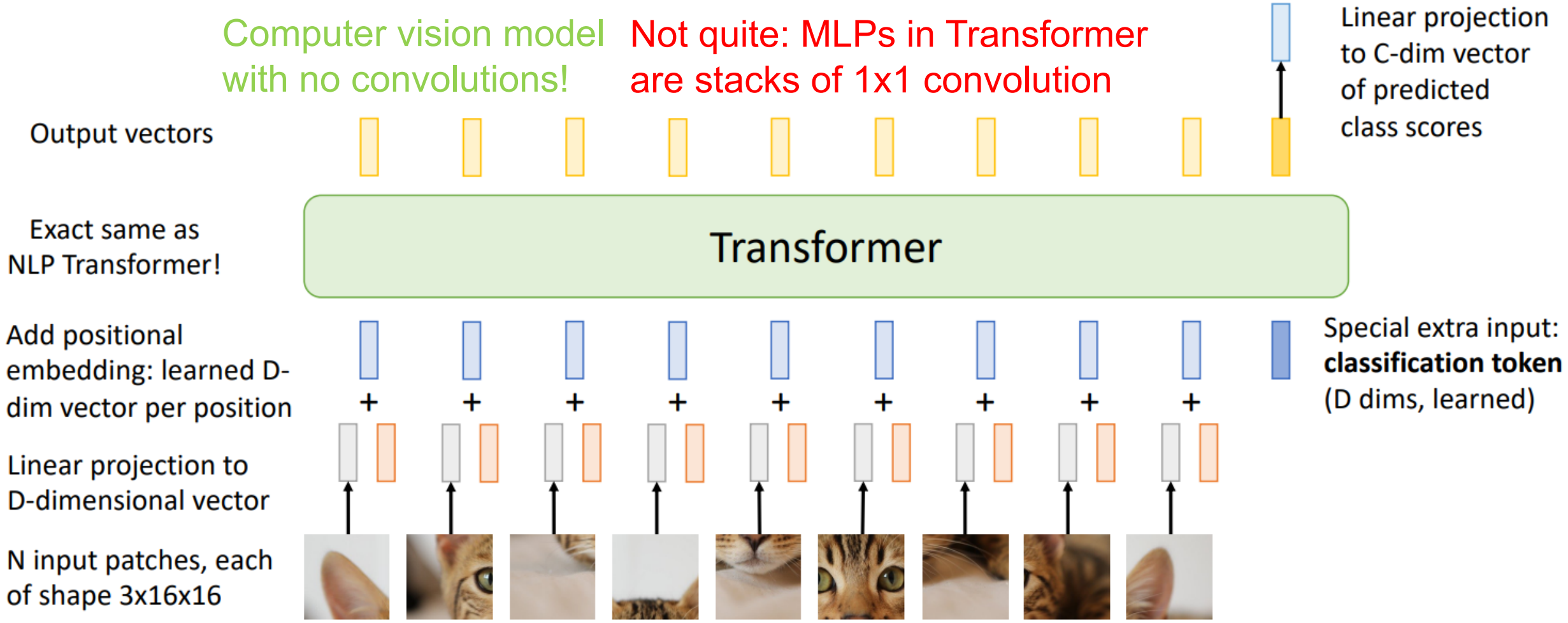
Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)

Computer vision model with no convolutions!

Not quite: MLPs in Transformer are stacks of 1x1 convolution



Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

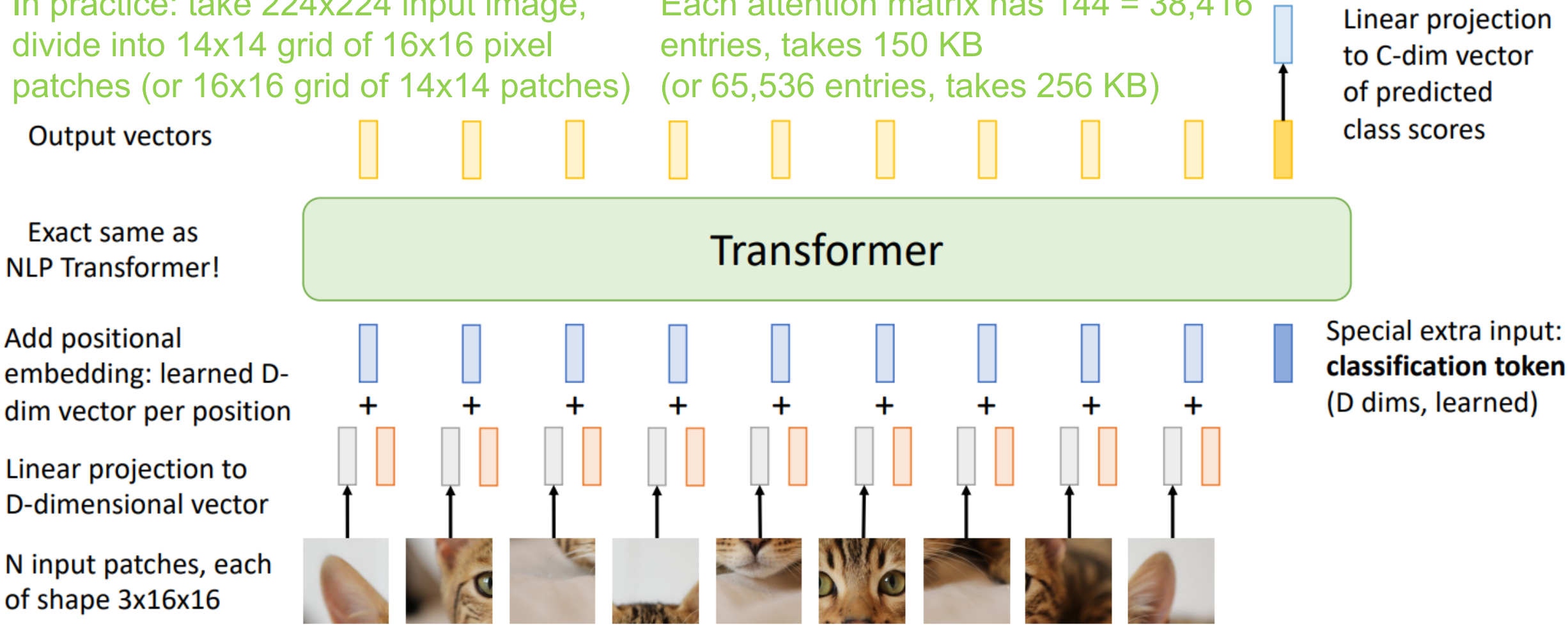
Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)

In practice: take 224x224 input image, divide into 14x14 grid of 16x16 pixel patches (or 16x16 grid of 14x14 patches)

Each attention matrix has 144 = 38,416 entries, takes 150 KB
(or 65,536 entries, takes 256 KB)



Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

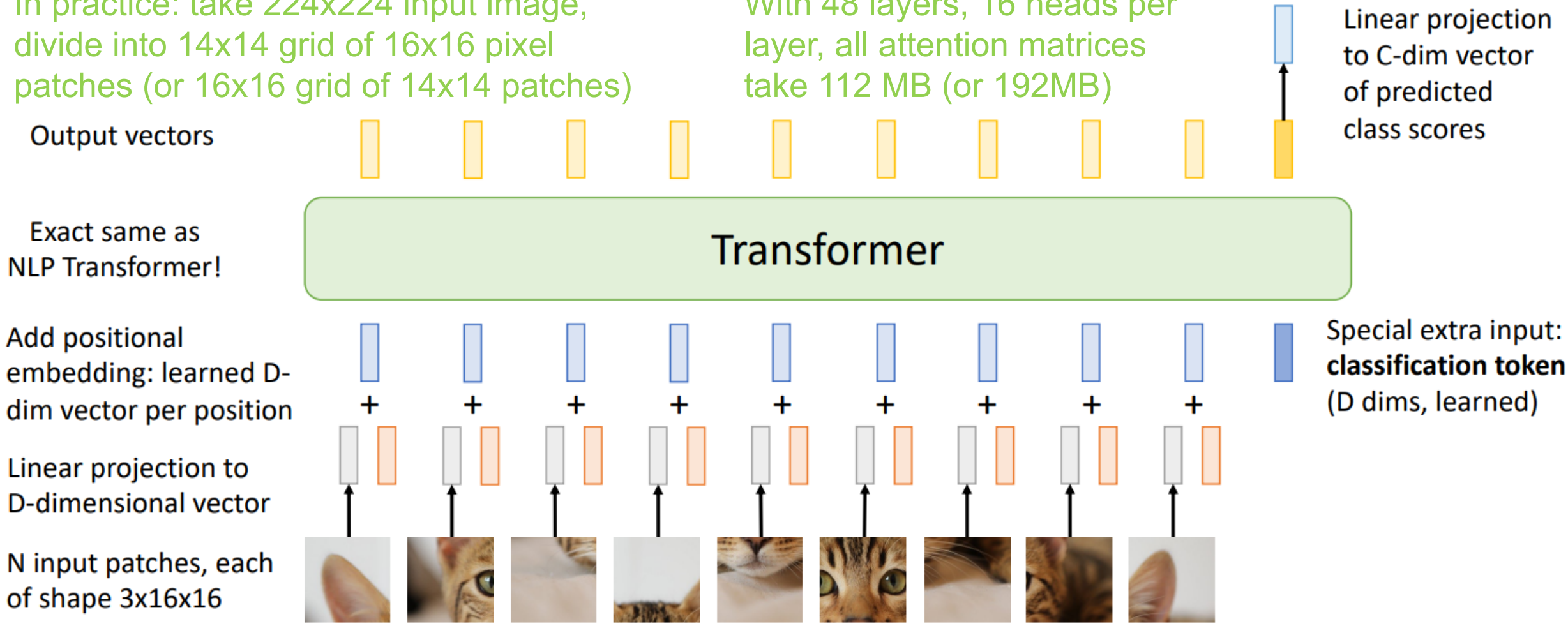
Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)

In practice: take 224x224 input image, divide into 14x14 grid of 16x16 pixel patches (or 16x16 grid of 14x14 patches)

With 48 layers, 16 heads per layer, all attention matrices take 112 MB (or 192MB)



Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

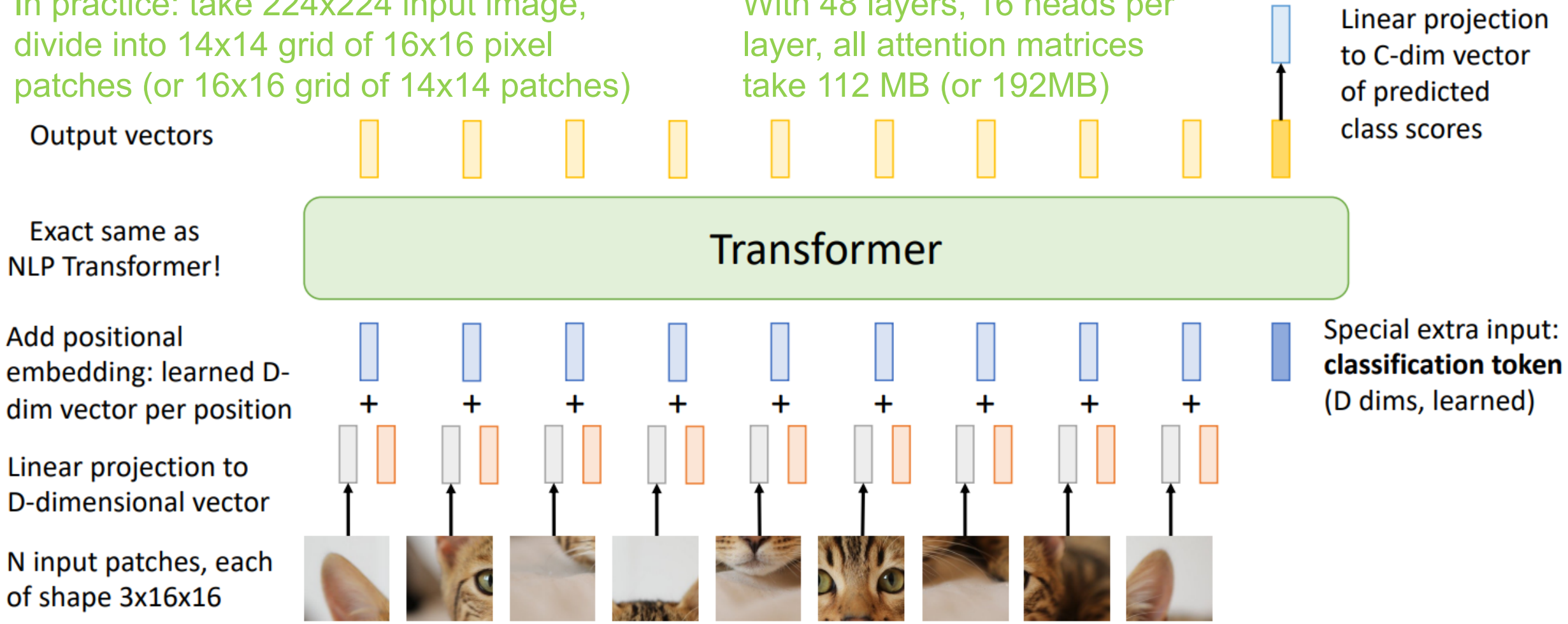
Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT)

In practice: take 224x224 input image, divide into 14x14 grid of 16x16 pixel patches (or 16x16 grid of 14x14 patches)

With 48 layers, 16 heads per layer, all attention matrices take 112 MB (or 192MB)

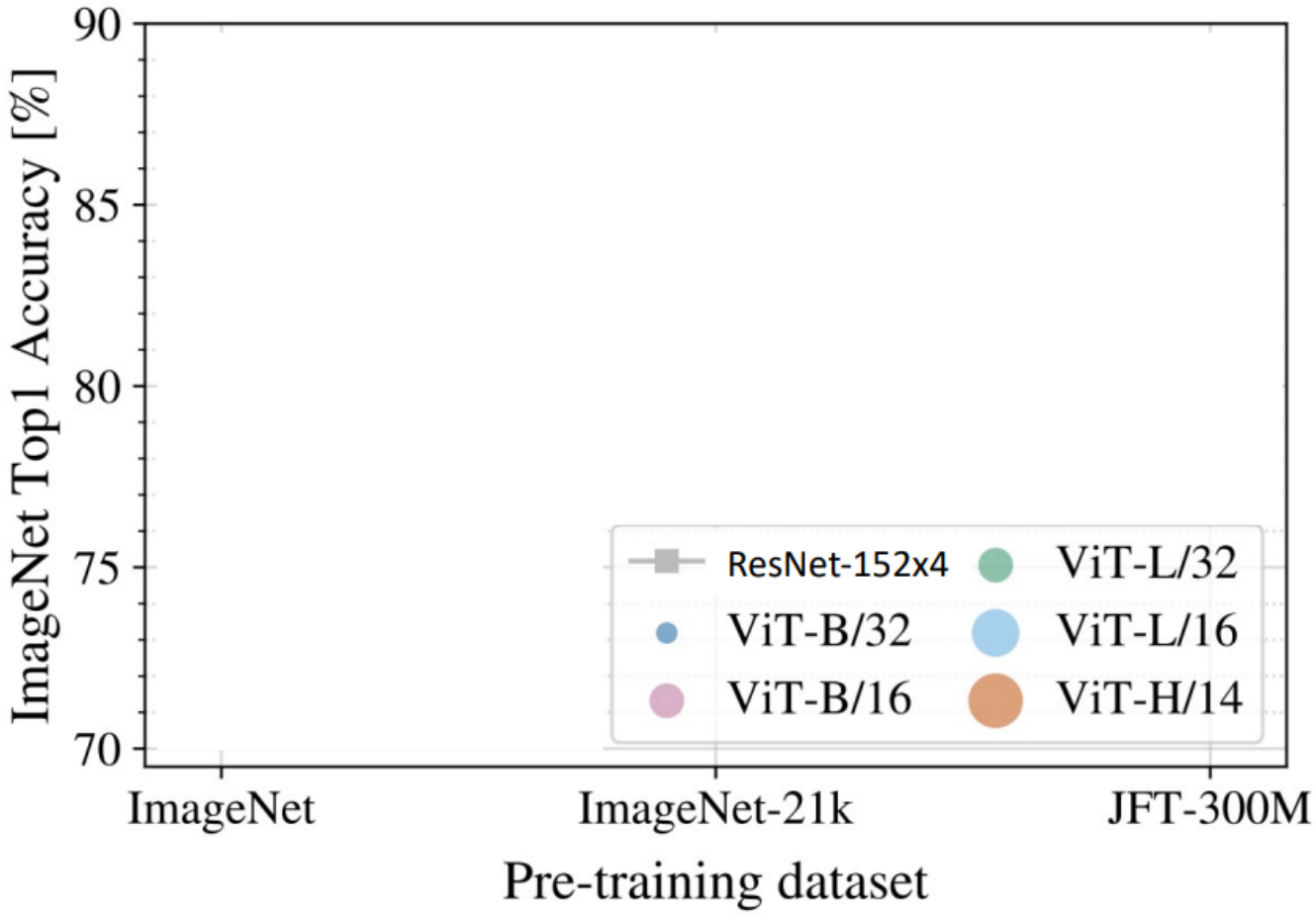


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Cat image is free for commercial use under a Pixabay license

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets



B = Base
L = Large
H = Huge

/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

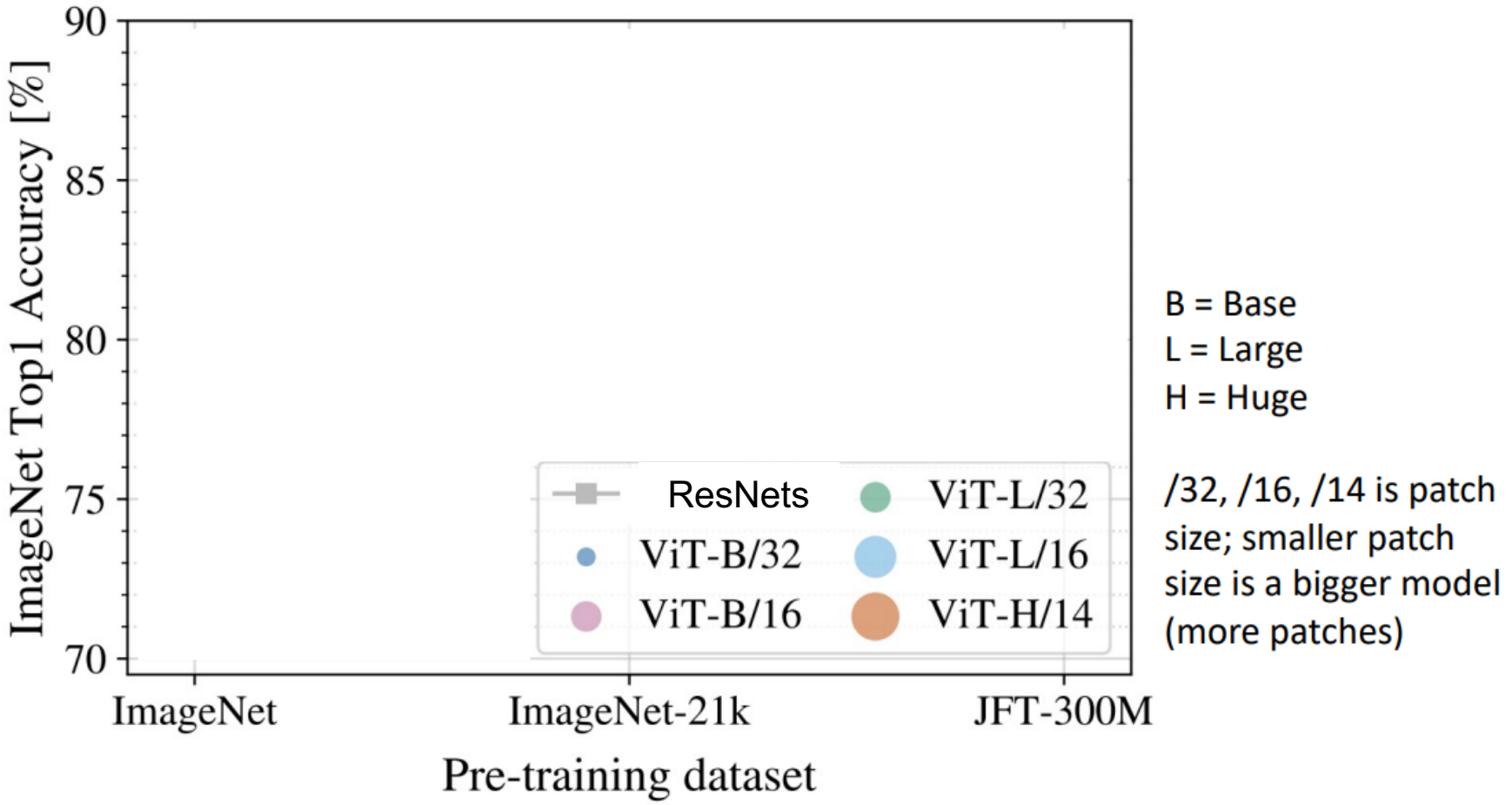
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

Recall: ImageNet dataset has 1k categories, 1.2M Images

When trained on ImageNet, ViT models perform worse than ResNets



B = Base
L = Large
H = Huge

/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

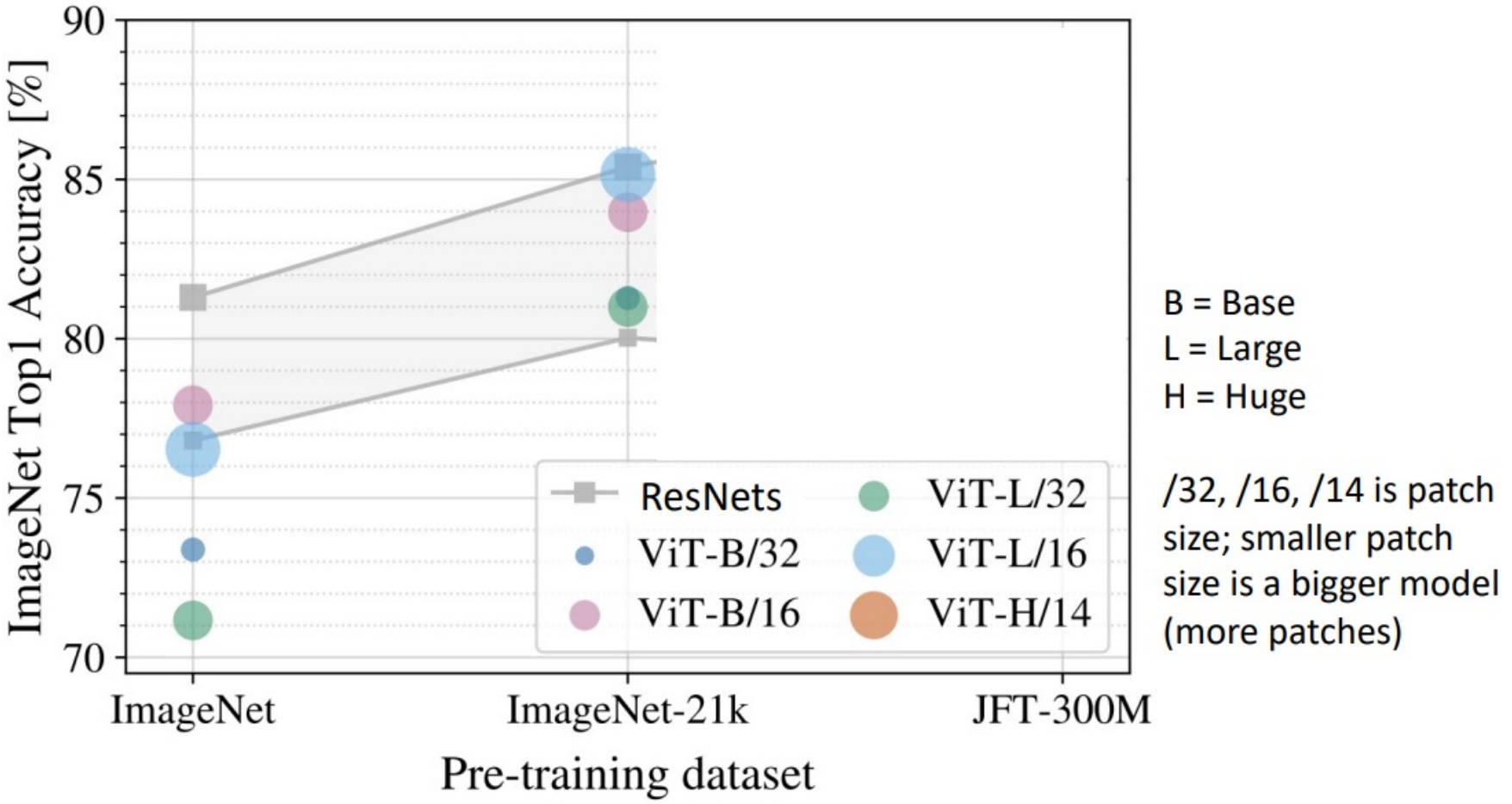
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

ImageNet-21k has 14M images with 21k categories

If you pretrain on ImageNet-21k and fine-tune on ImageNet, ViT does better: big ViTs match
big ResNets



B = Base
L = Large
H = Huge

/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

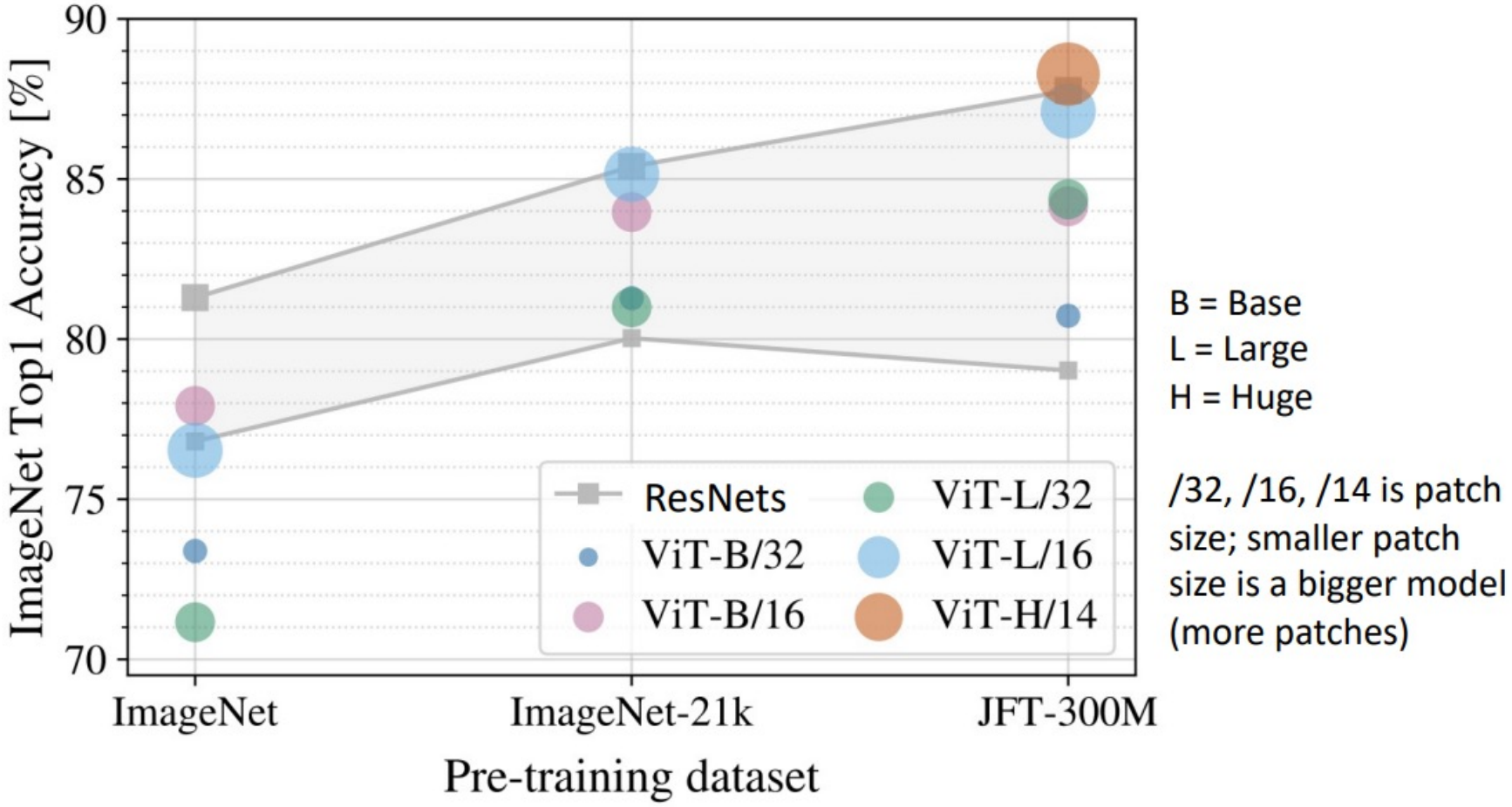
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

JFT-300M is an internal Google dataset with 300M labeled images

If you pretrain on JFT and finetune on ImageNet, large ViTs outperform large ResNets



B = Base
L = Large
H = Huge

/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

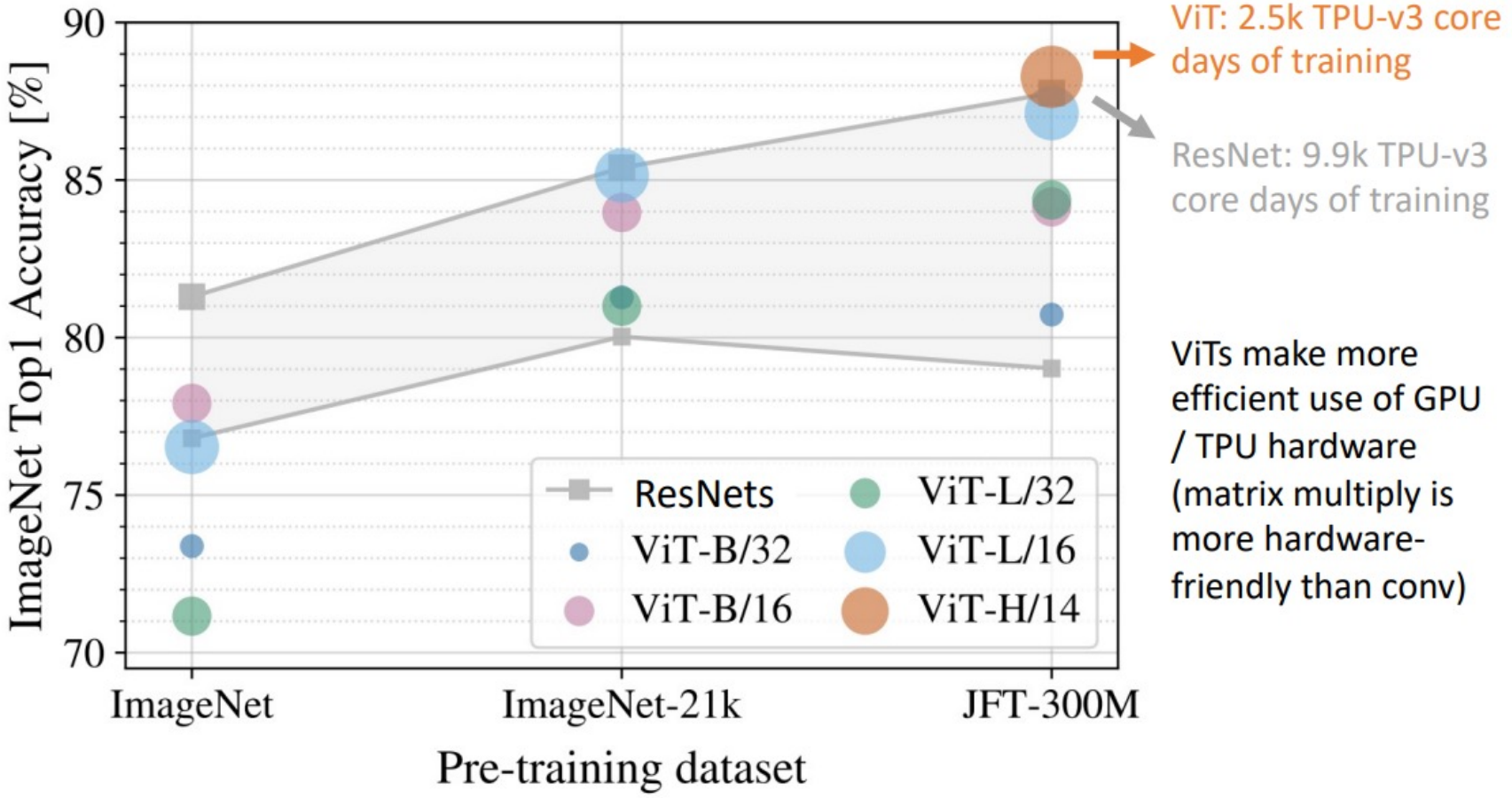
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

JFT-300M is an internal Google dataset with 300M labeled images

If you pretrain on JFT and finetune on ImageNet, large ViTs outperform large ResNets



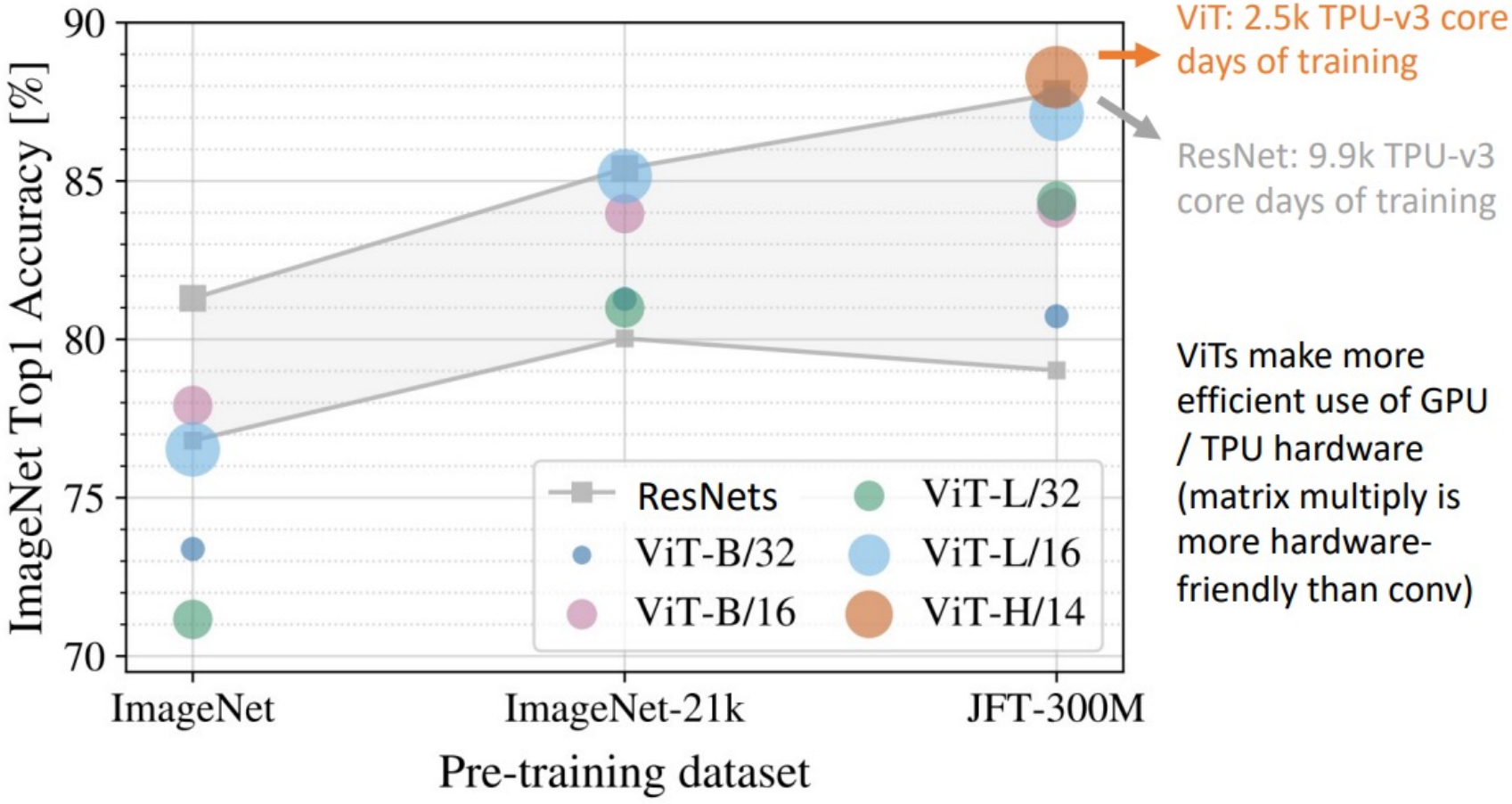
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

Claim: ViT models have
“less inductive bias” than ResNets, so need
more pretraining data to learn good features

(Not sure I buy this explanation: “inductive
bias” is not a welldefined concept we
can measure!)

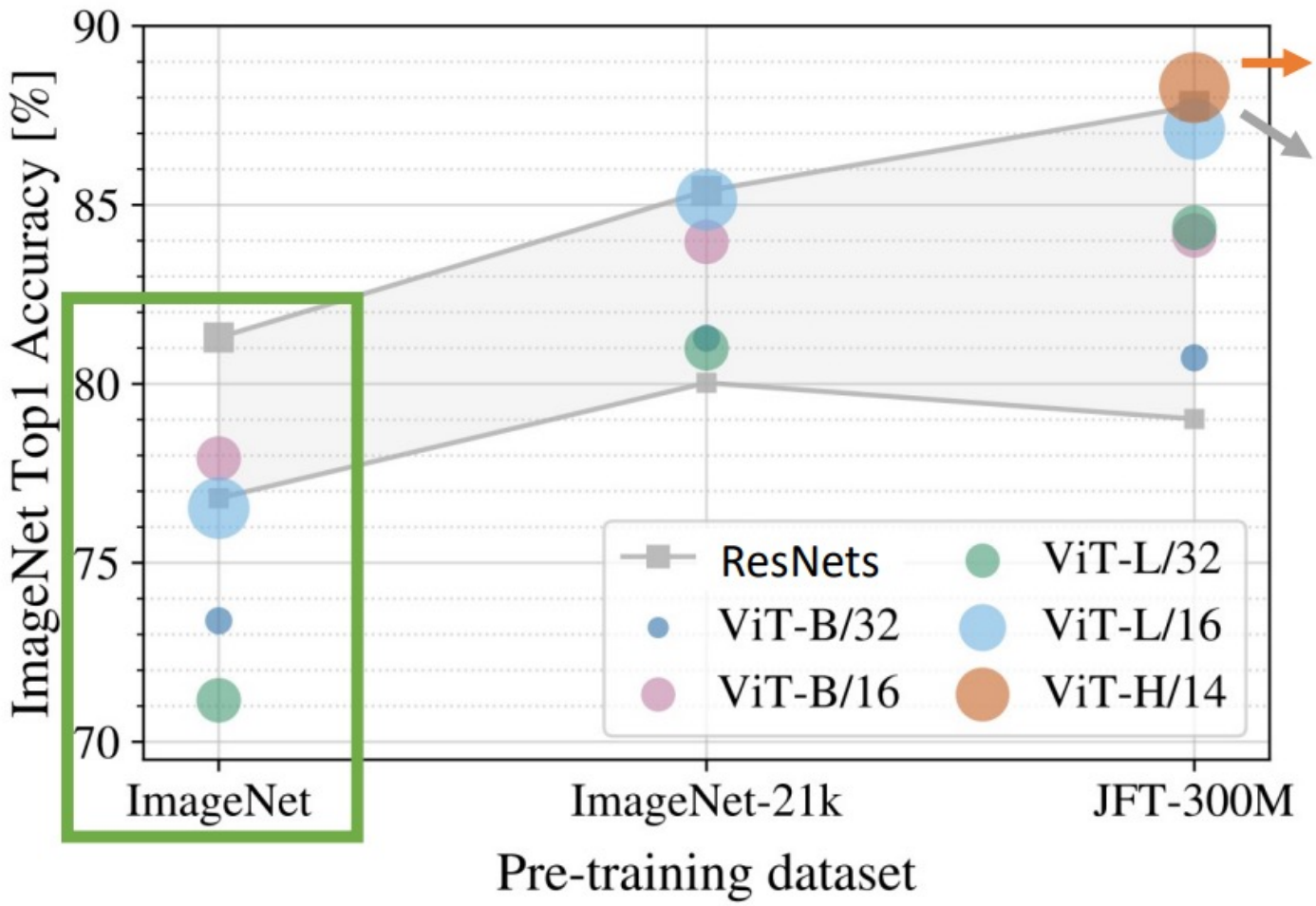


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Vision Transformer (ViT) vs ResNets

How can we improve the performance of ViT models on ImageNet?



ViT: 2.5k TPU-v3 core days of training

ResNet: 9.9k TPU-v3 core days of training

ViTs make more efficient use of GPU / TPU hardware (matrix multiply is more hardware-friendly than conv)

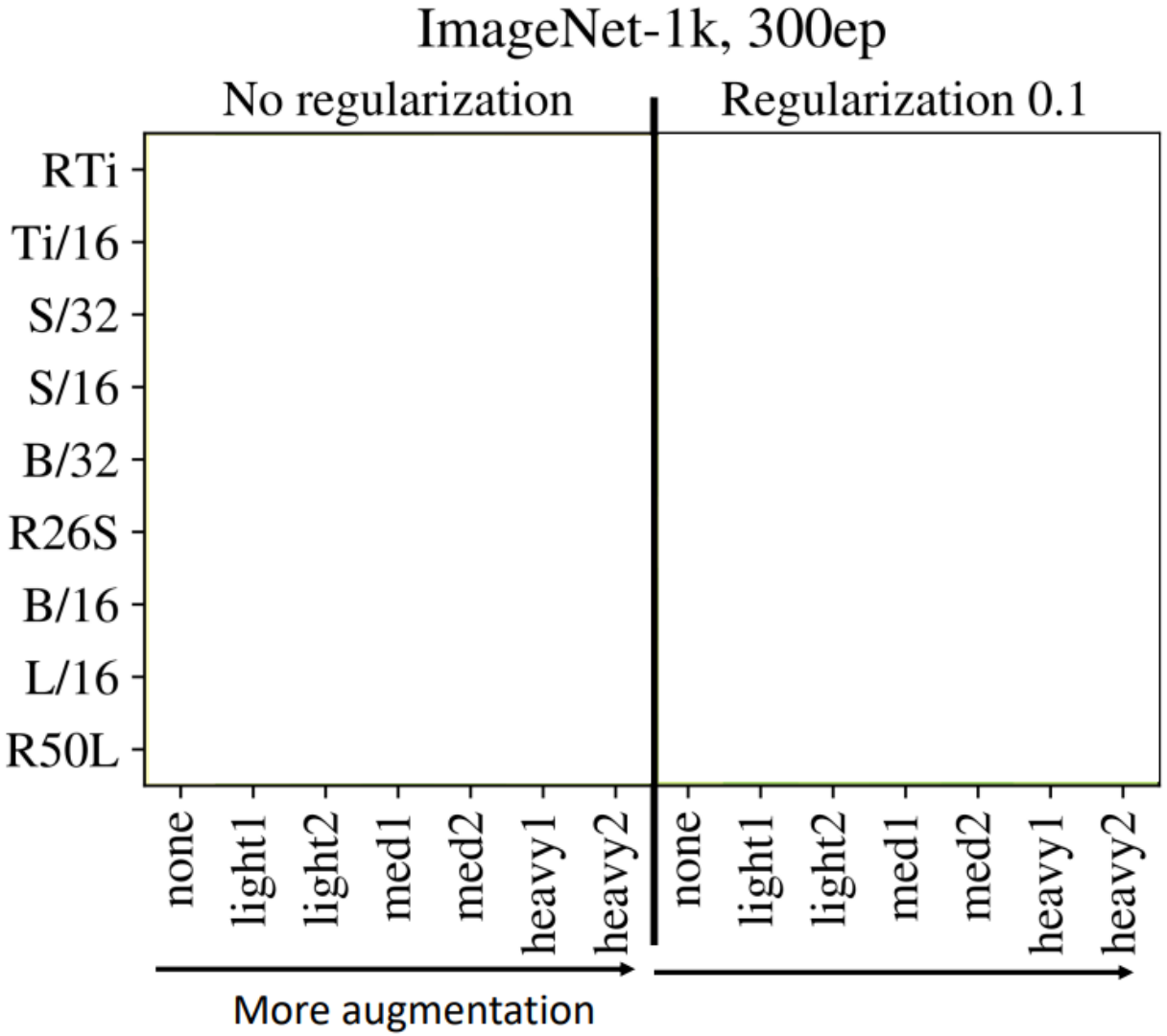
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment



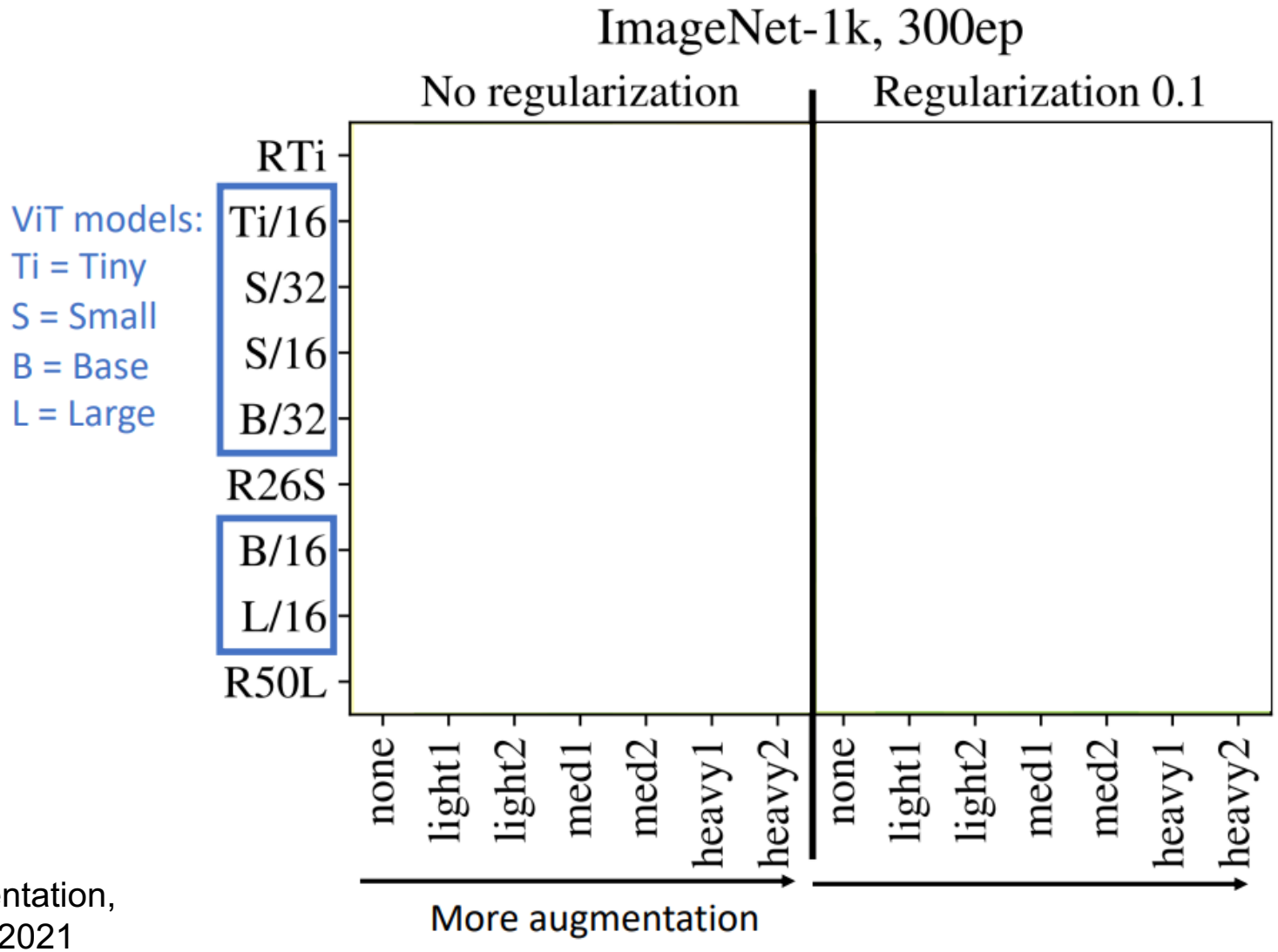
Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment



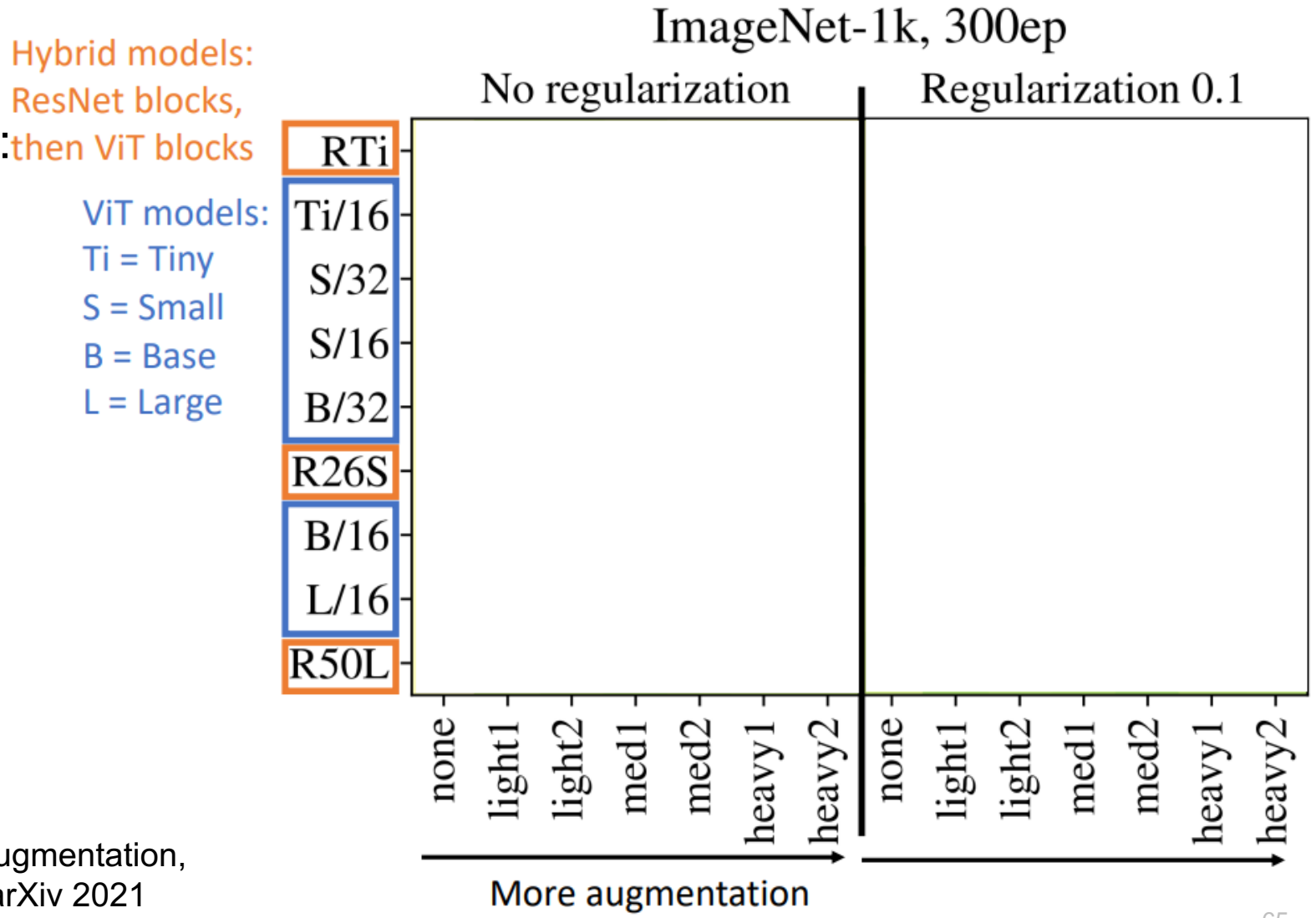
Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment



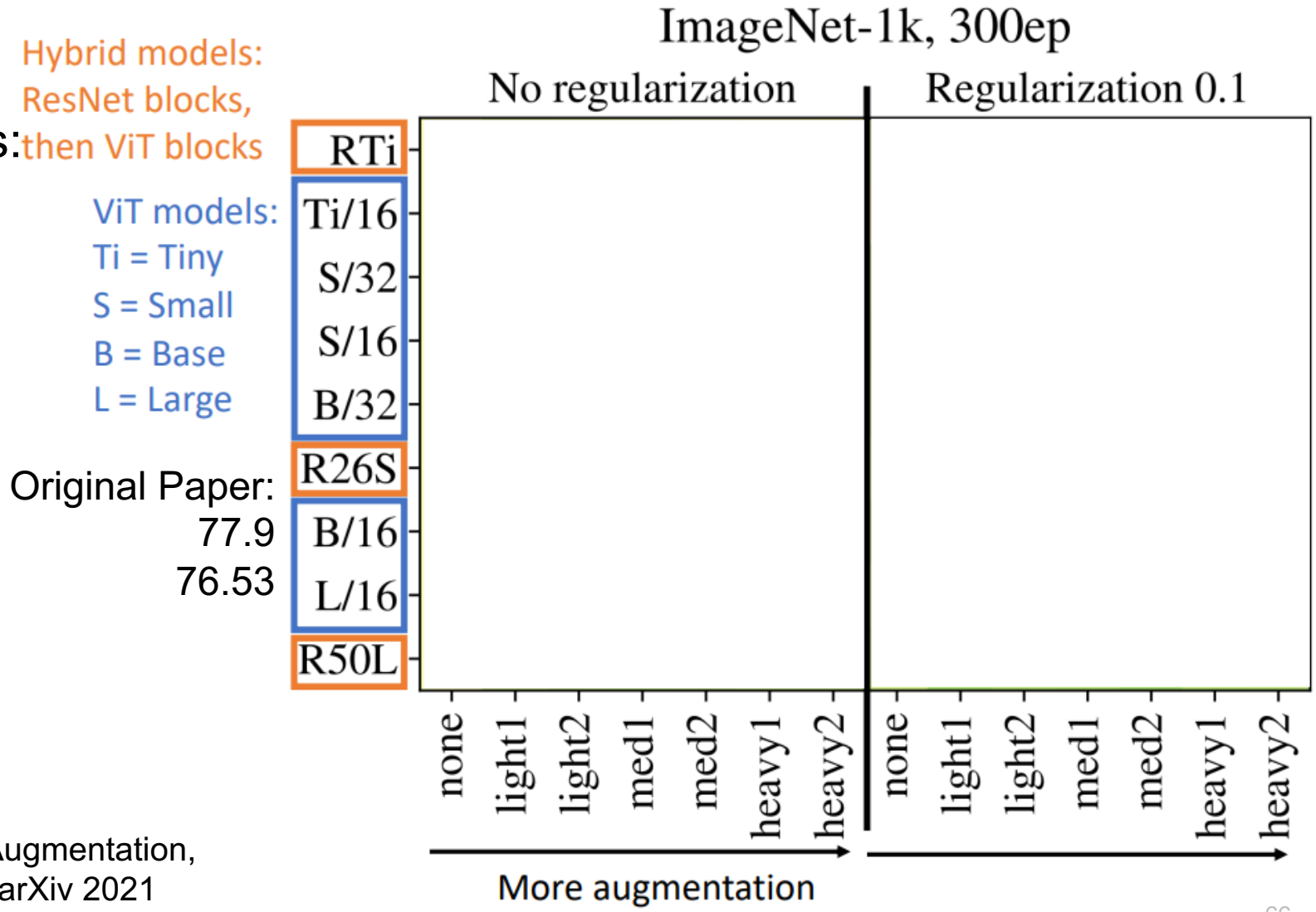
Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment



Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

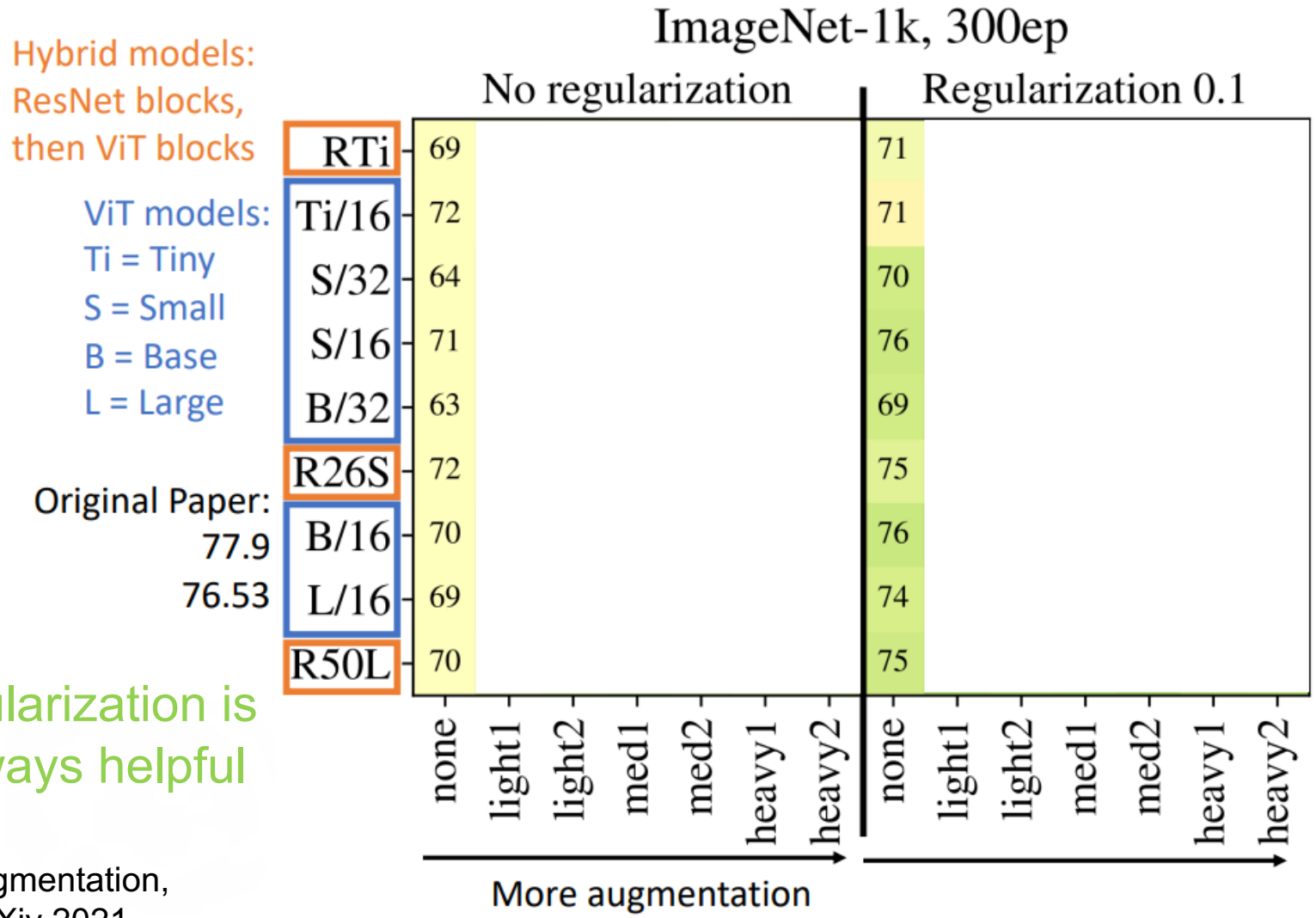
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment

Adding regularization is (almost) always helpful



Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

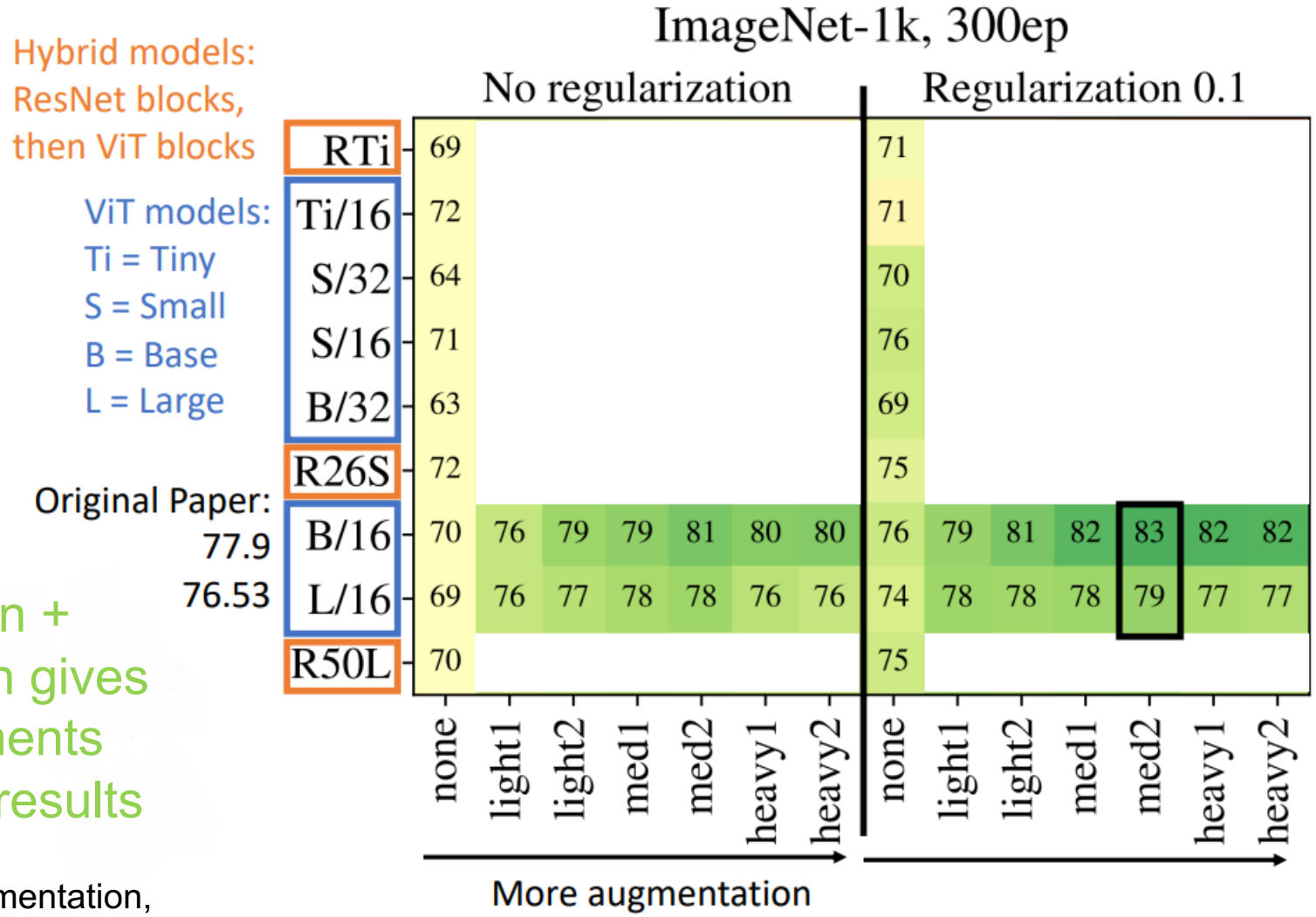
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

Data Augmentation for ViT models:
- MixUp
- RandAugment

Regularization + Augmentation gives big improvements over original results



Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Augmentation and Regularization

Regularization for ViT models:
- Weight Decay
- Stochastic Depth
- Dropout (in FFN layers of Transformer)

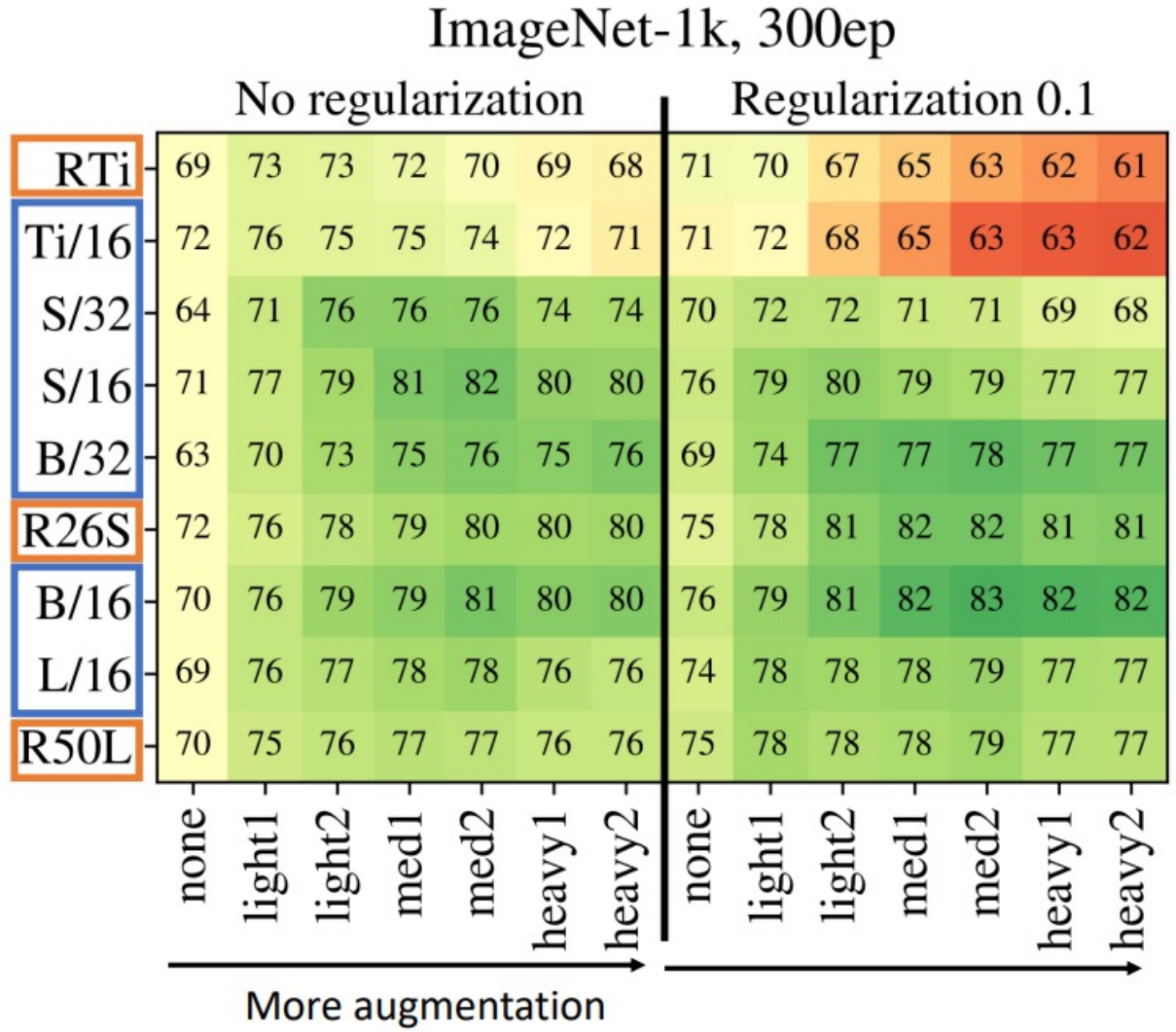
Data Augmentation for ViT models:
- MixUp
- RandAugment

Hybrid models: ResNet blocks, then ViT blocks

ViT models:
Ti = Tiny
S = Small
B = Base
L = Large

Original Paper:
77.9
76.53

Lots of other patterns in full results

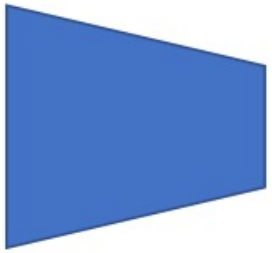


Steiner et al, “How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers”, arXiv 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation

Step 1: Train a **teacher model** on images and ground-truth labels



P(cat) = 0.9
P(dog) = 0.1

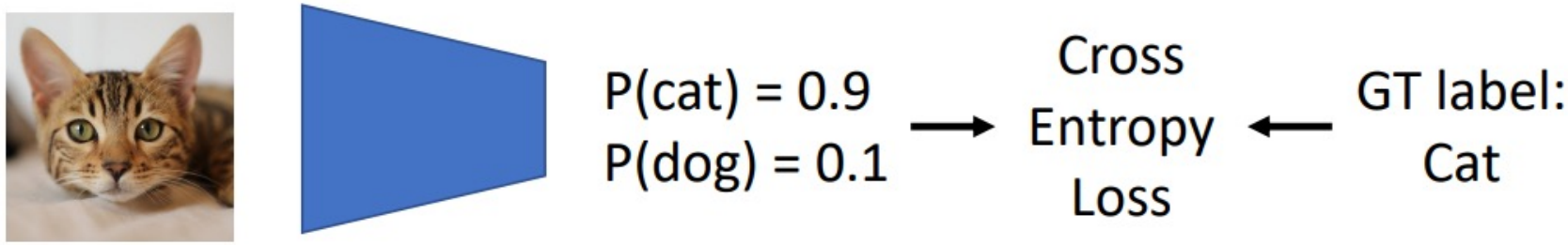
→ Cross Entropy Loss ← GT label: Cat

Hinton et al, “Distilling the knowledge in a neural network”, NeurIPS Deep Learning and Representation Learning Workshop, 2015

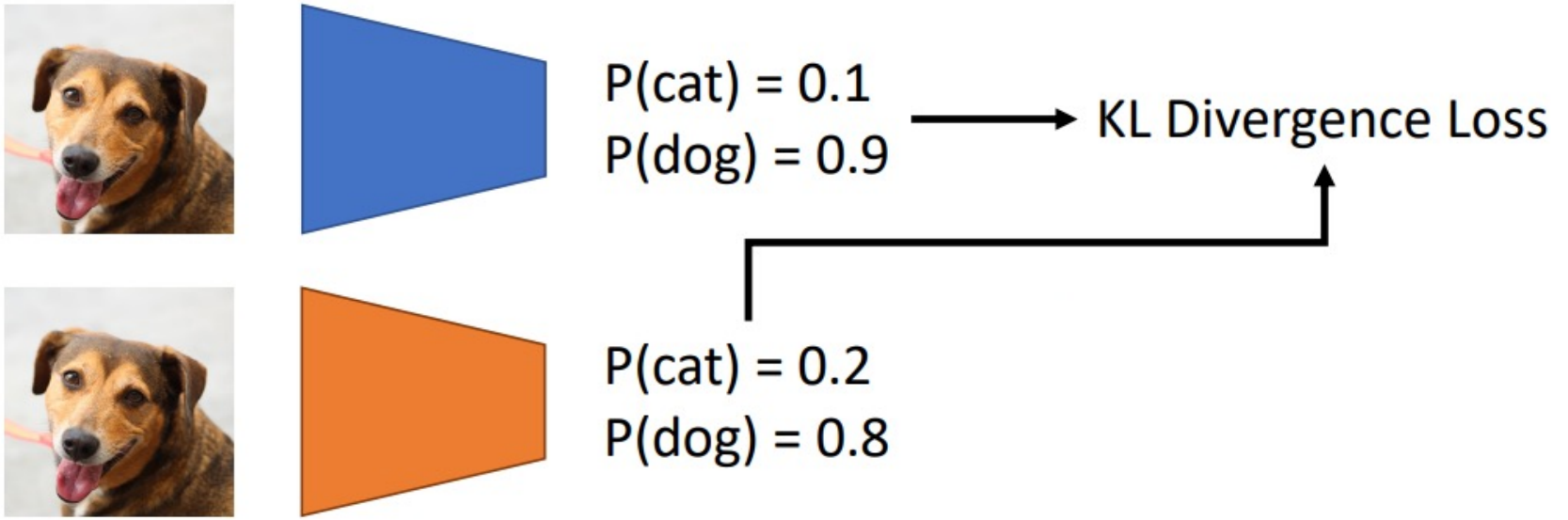
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation

Step 1: Train a **teacher model** on images and ground-truth labels



Step 2: Train a **student model** to match predictions from the **teacher**

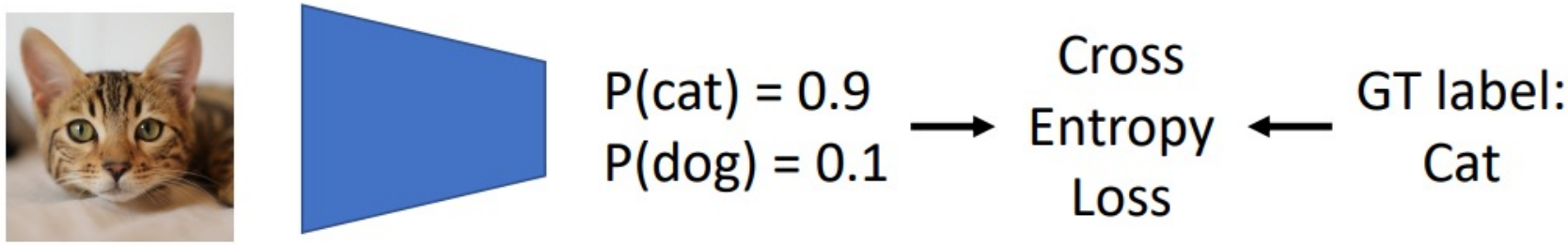


Hinton et al, “Distilling the knowledge in a neural network”, NeurIPS Deep Learning and Representation Learning Workshop, 2015

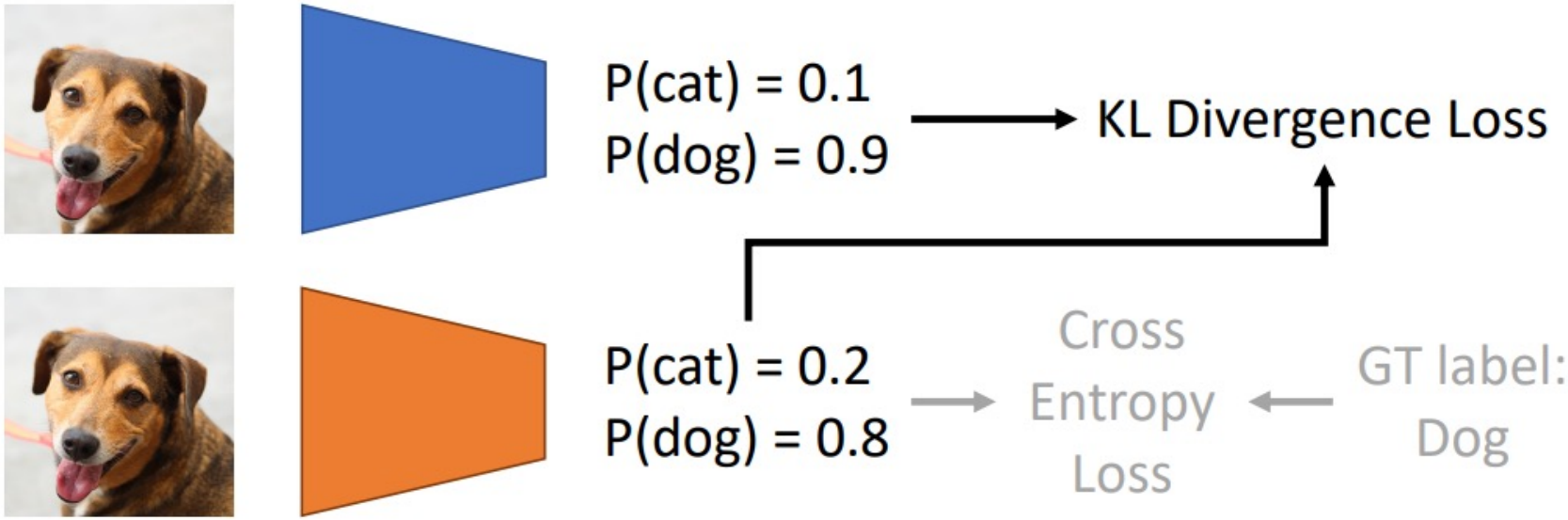
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation

Step 1: Train a **teacher model** on images and ground-truth labels



Step 2: Train a **student model** to match predictions from the **teacher** (sometimes also to match GT labels)



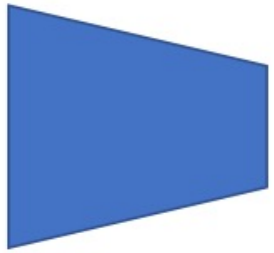
Hinton et al, “Distilling the knowledge in a neural network”, NeurIPS Deep Learning and Representation Learning Workshop, 2015

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation

Can also train student on **unlabeled** data! (Semi-supervised learning)

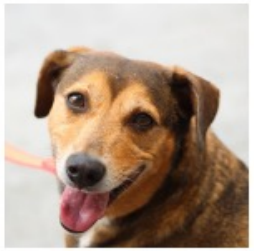
Step 1: Train a **teacher model** on images and ground-truth labels

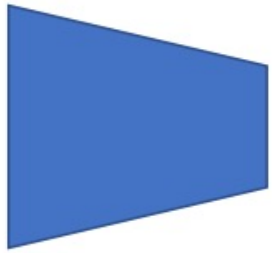


P(cat) = 0.9
P(dog) = 0.1

→ Cross Entropy Loss ← GT label: Cat

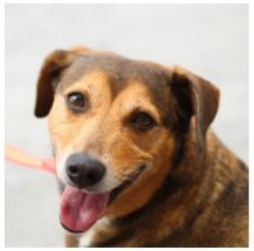
Step 2: Train a **student model** to match predictions from the **teacher** (sometimes also to match GT labels)

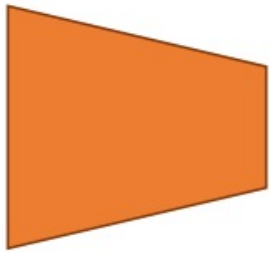




P(cat) = 0.1
P(dog) = 0.9

→ KL Divergence Loss





P(cat) = 0.2
P(dog) = 0.8

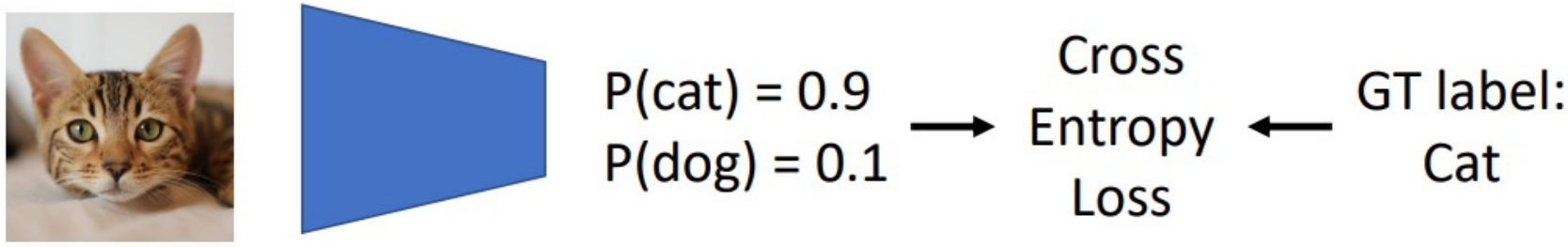
→ Cross Entropy Loss ← GT label: Dog

Hinton et al, “Distilling the knowledge in a neural network”, NeurIPS Deep Learning and Representation Learning Workshop, 2015

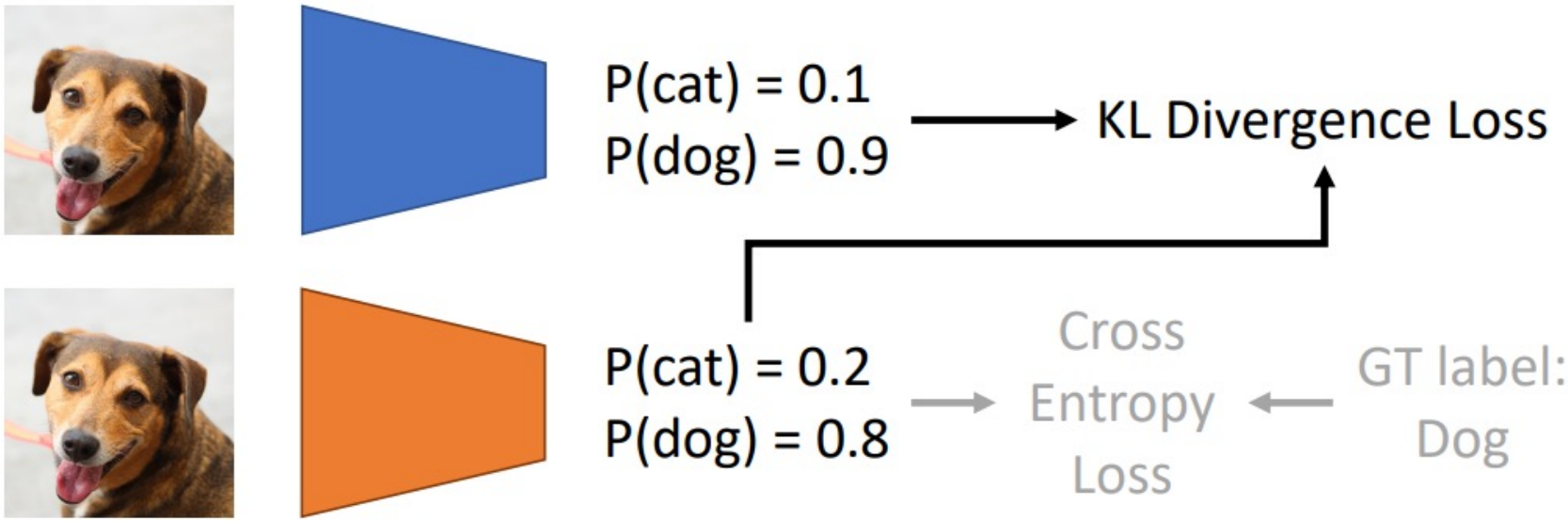
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation

Step 1: Train a teacher CNN on ImageNet



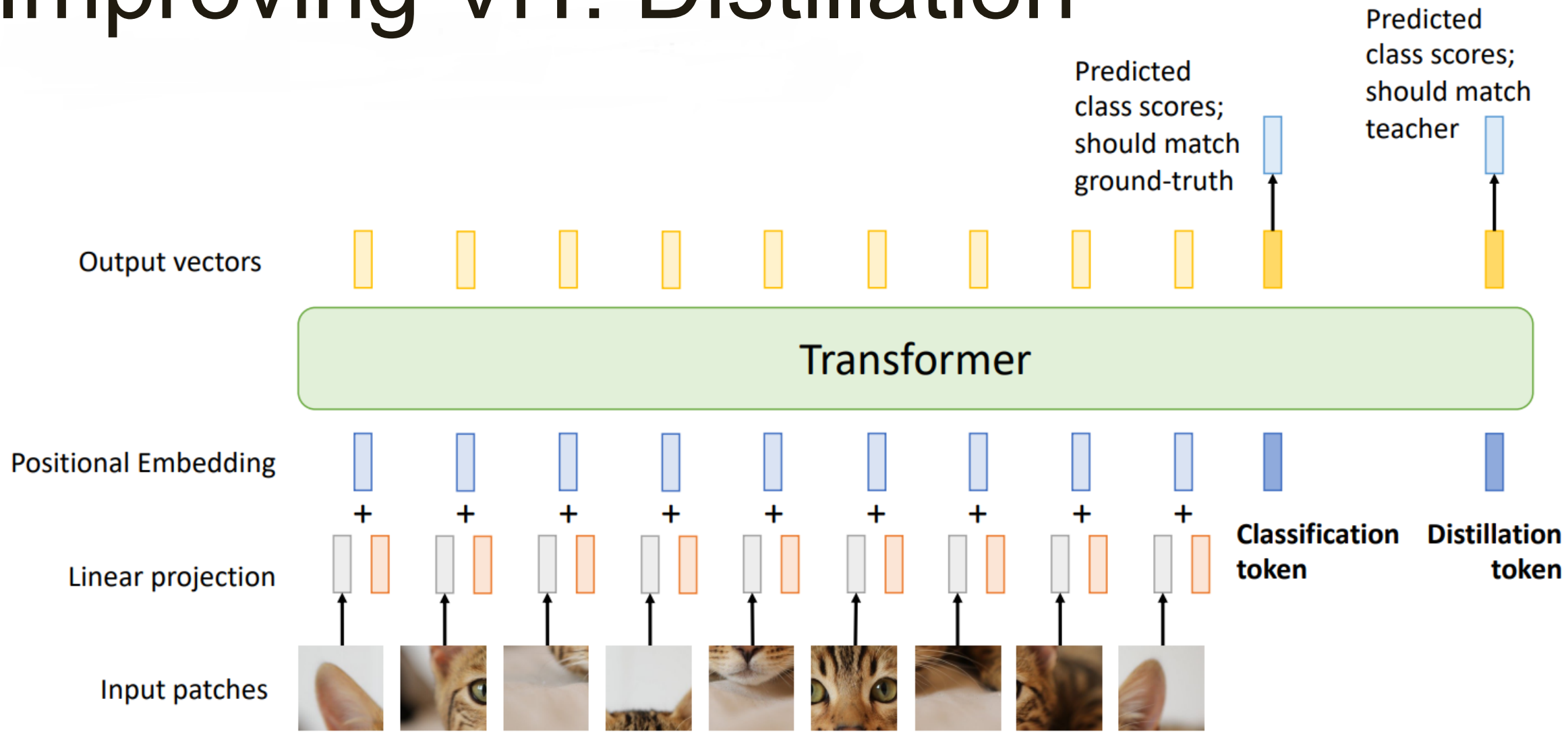
Step 2: Train a student ViT to match ImageNet predictions from the teacher CNN (and match GT labels)



Touvrom et al, “Training data-efficient image transformers & distillation through attention”, ICML 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

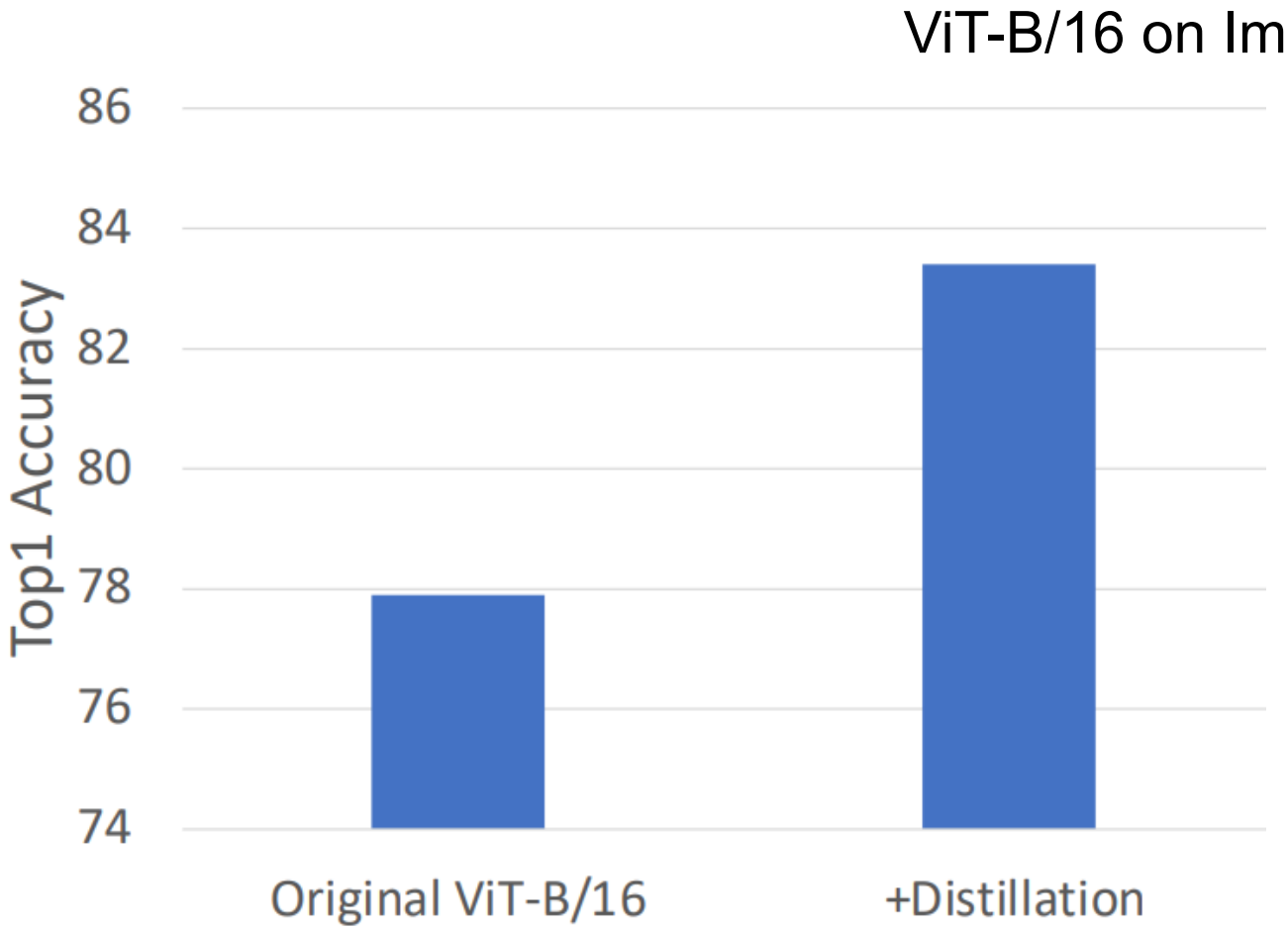
Improving ViT: Distillation



Touvrom et al, “Training data-efficient image transformers & distillation through attention”, ICML 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

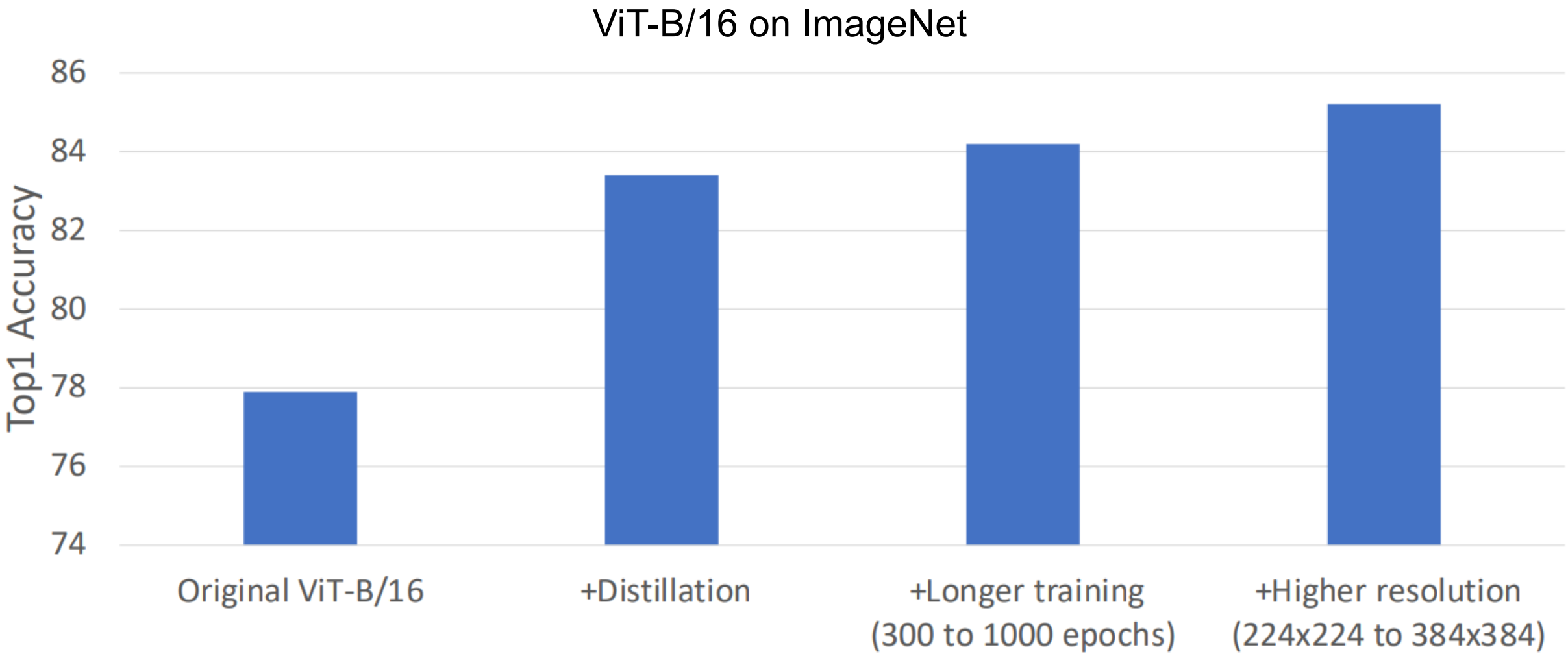
Improving ViT: Distillation



Touvrom et al, “Training data-efficient image transformers & distillation through attention”, ICML 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Improving ViT: Distillation



Touvrom et al, “Training data-efficient image transformers & distillation through attention”, ICML 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

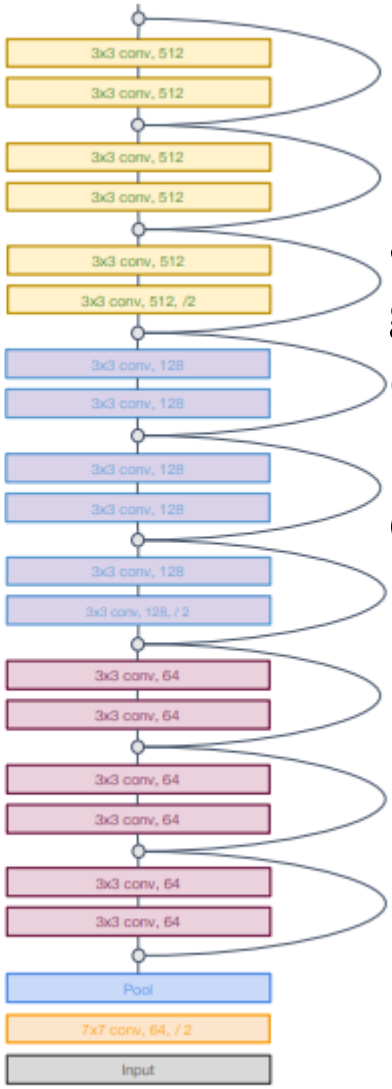
ViT vs CNN

Stage 3:
256 x 14 x 14

Stage 2:
128 x 28 x 28

Stage 1:
64 x 56 x 56

Input:
3 x 224 x 224



In most CNNs (including ResNets), **decrease** resolution and **increase** channels as you go deeper in the network (Hierarchical architecture)

Useful since objects in images can occur at various scales

Slides original source: [Deep Learning for Computer Vision 2022 Lecture 18]

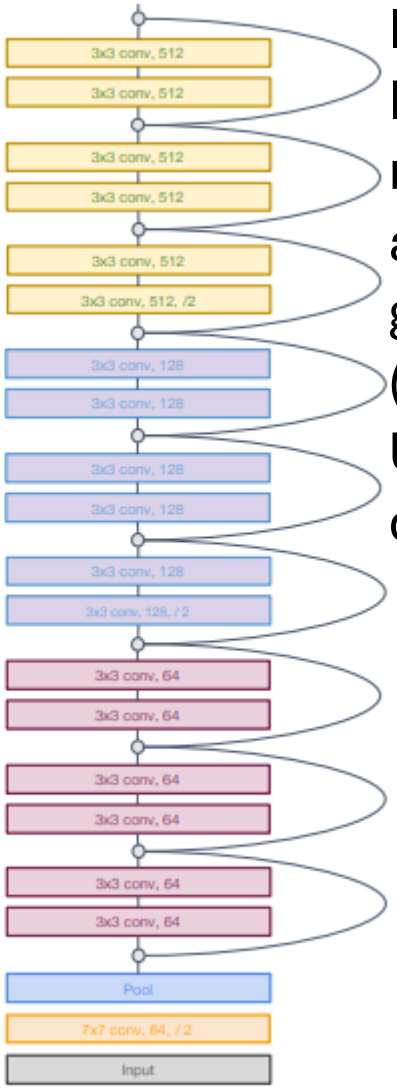
ViT vs CNN

Stage 3:
256 x 14 x 14

Stage 2:
128 x 28 x 28

Stage 1:
64 x 56 x 56

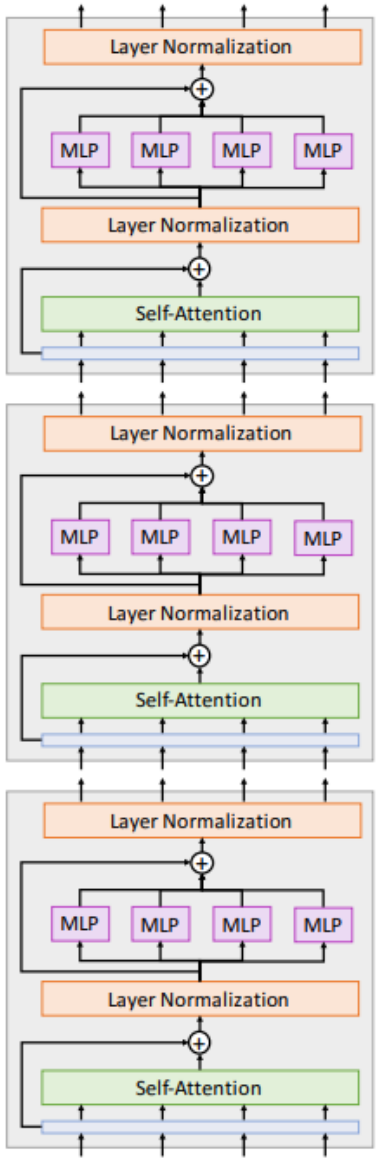
Input:
3 x 224 x 224



In most CNNs (including ResNets), **decrease** resolution and **increase** channels as you go deeper in the network (Hierarchical architecture)

Useful since objects in images can occur at various scales

In a ViT, all blocks have same resolution and number of channels (Isotropic architecture)



3rd block:
768 x 14 x 14

2nd block:
768 x 14 x 14

1st block:
768 x 14 x 14

Input:
3 x 224 x 224

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

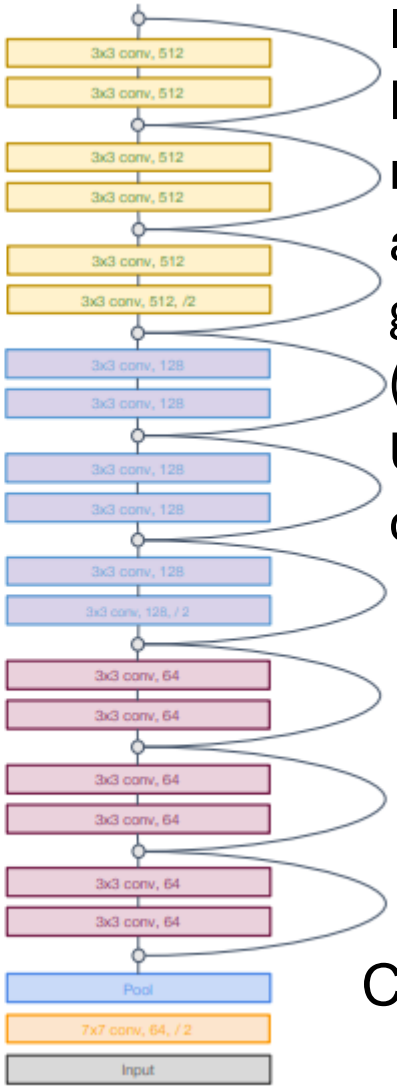
ViT vs CNN

Stage 3:
256 x 14 x 14

Stage 2:
128 x 28 x 28

Stage 1:
64 x 56 x 56

Input:
3 x 224 x 224

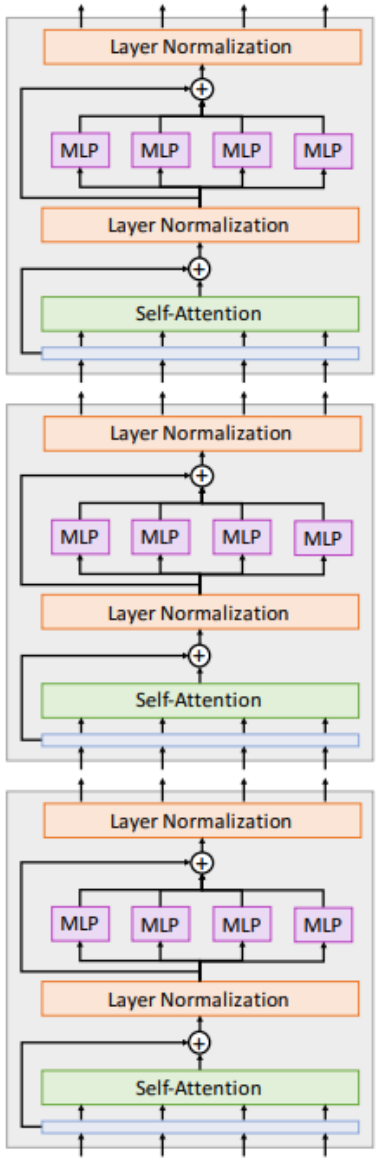


In most CNNs (including ResNets), **decrease** resolution and **increase** channels as you go deeper in the network (Hierarchical architecture)

Useful since objects in images can occur at various scales

In a ViT, all blocks have same resolution and number of channels (Isotropic architecture)

Can we build a **hierarchical** ViT model?



3rd block:
768 x 14 x 14

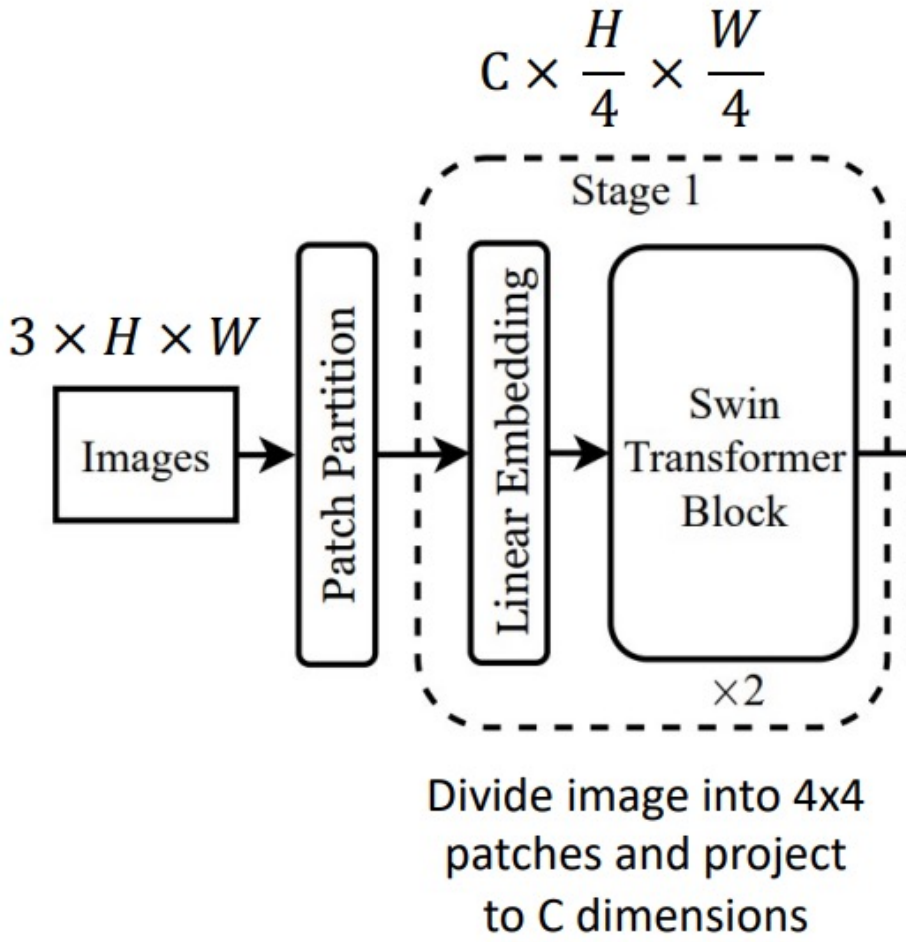
2nd block:
768 x 14 x 14

1st block:
768 x 14 x 14

Input:
3 x 224 x 224

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

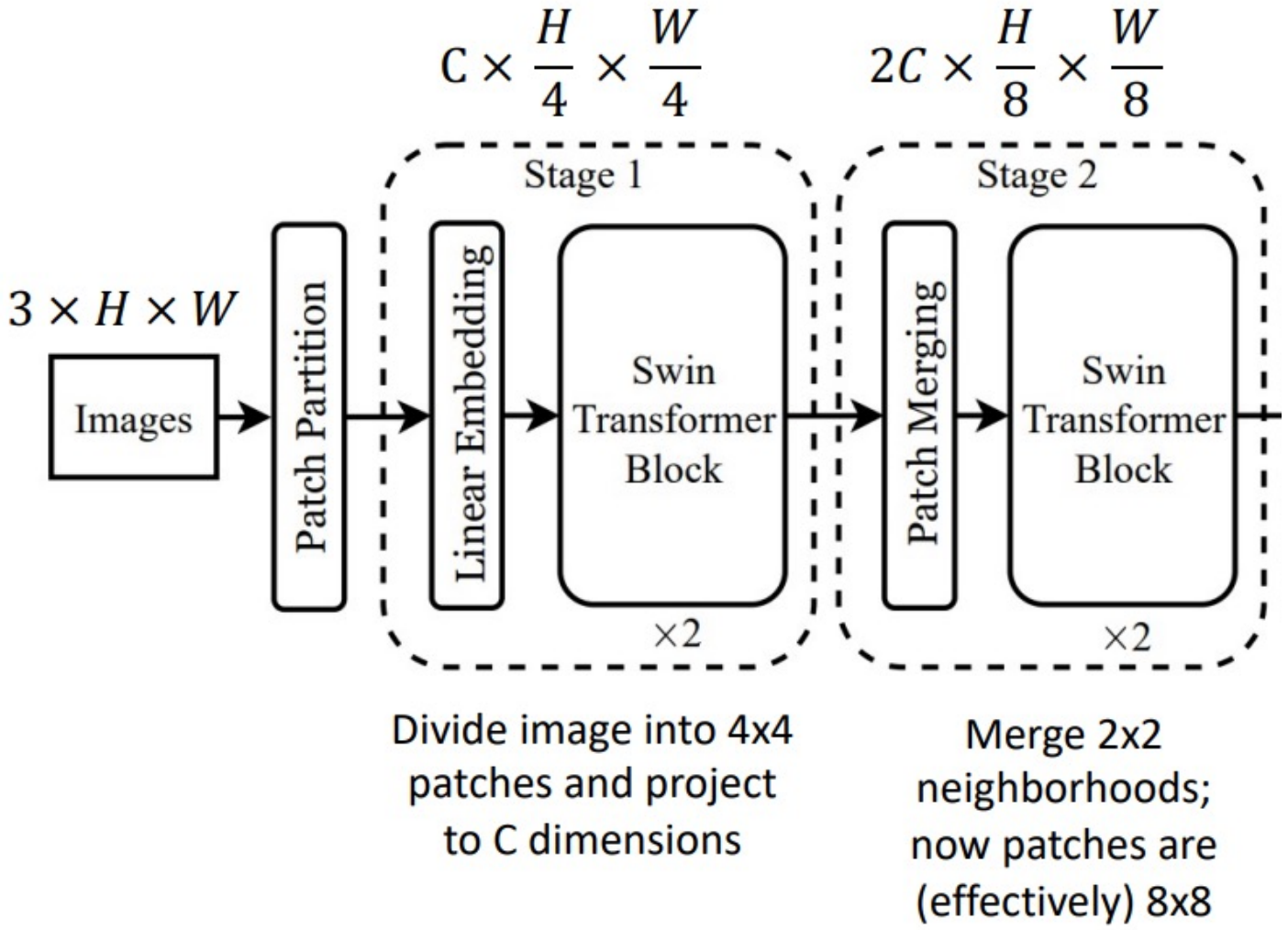
Hierarchical ViT: Swin Transformer



Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

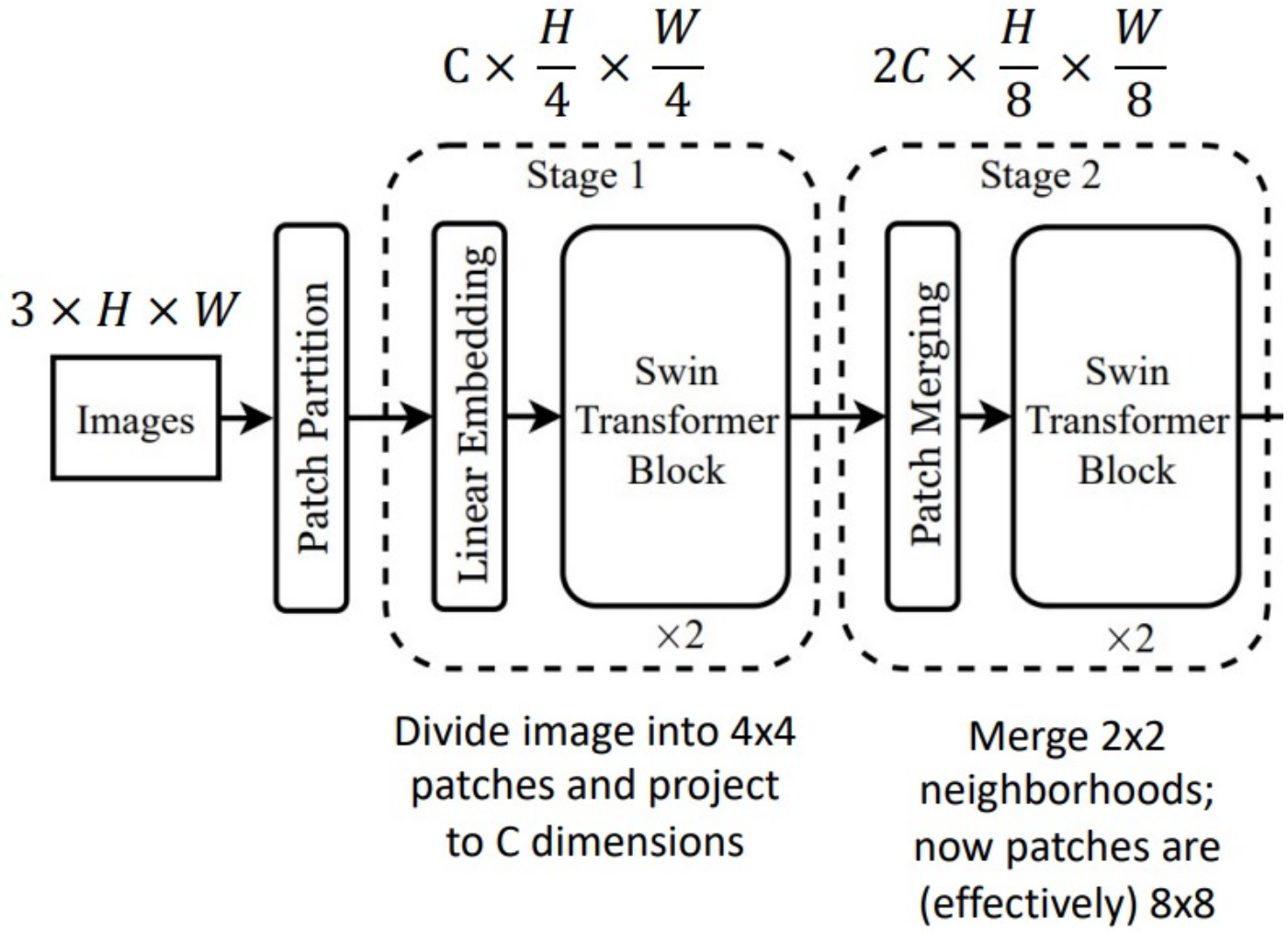
Hierarchical ViT: Swin Transformer

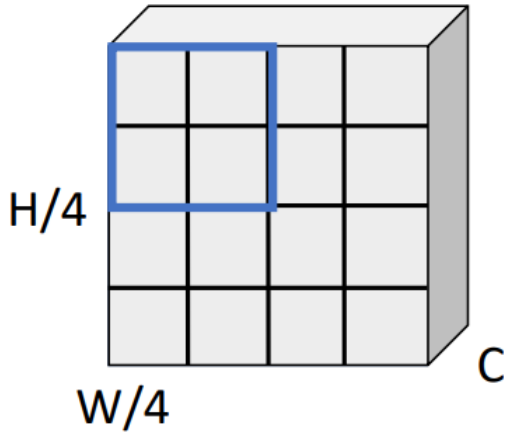


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

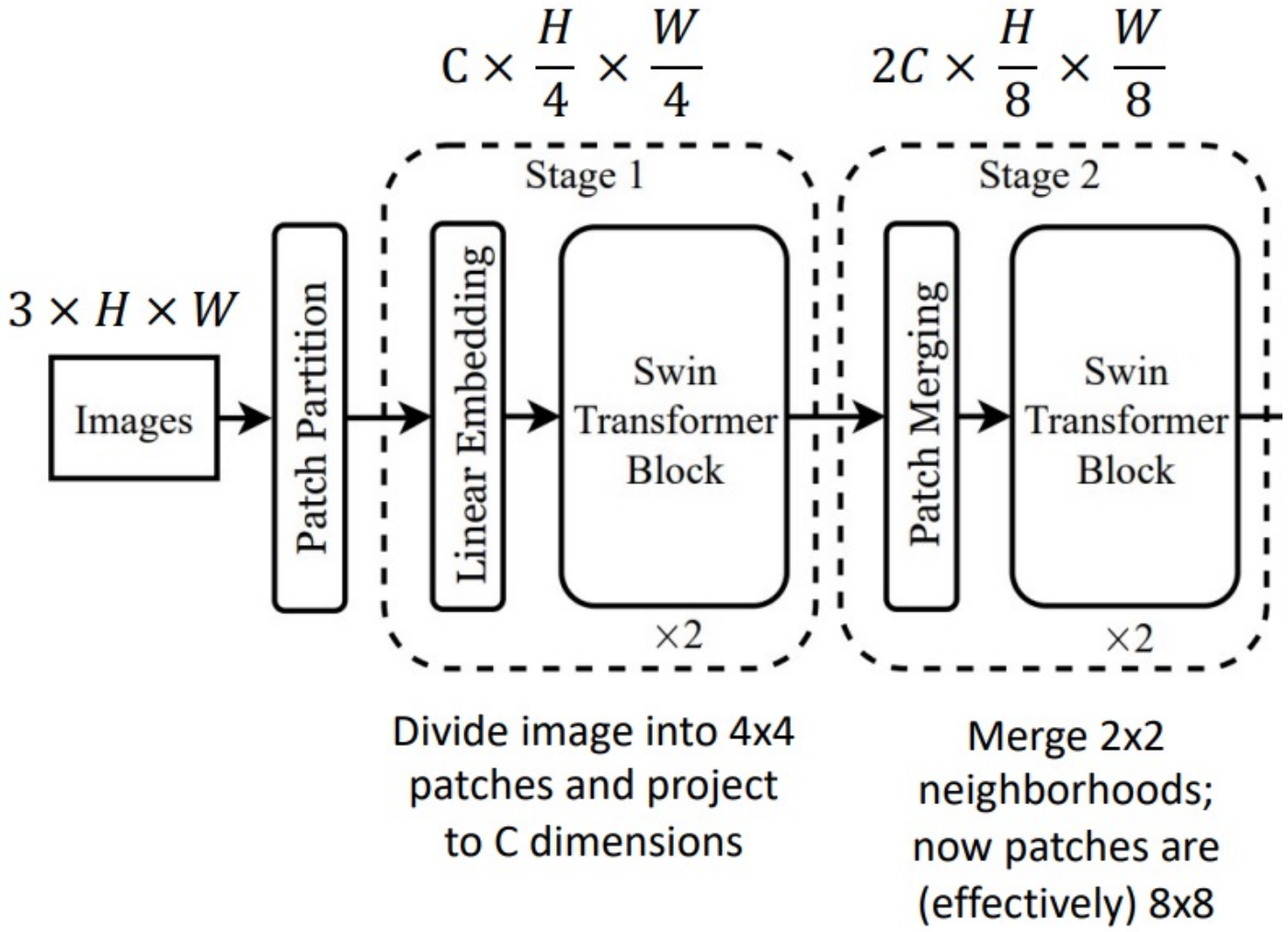


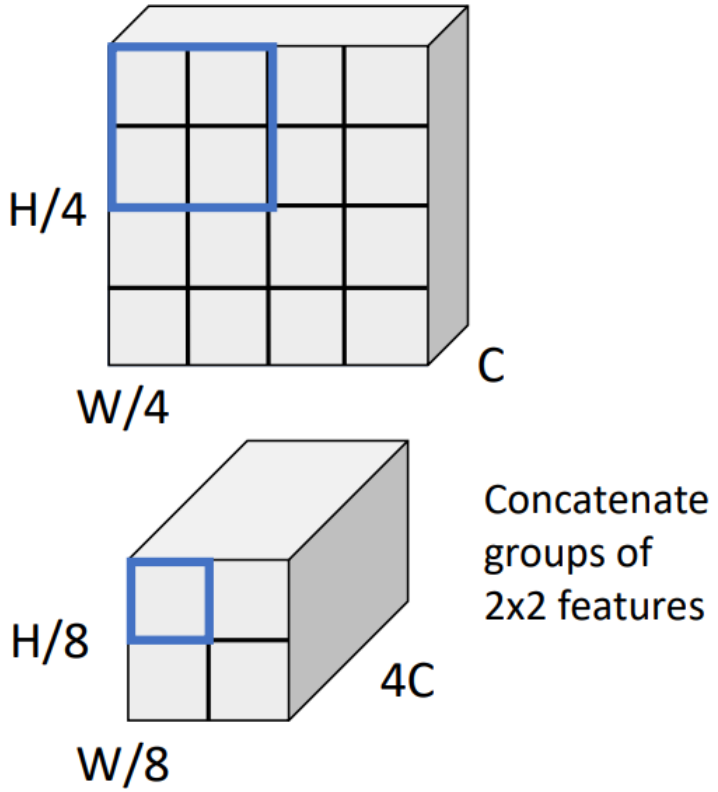


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

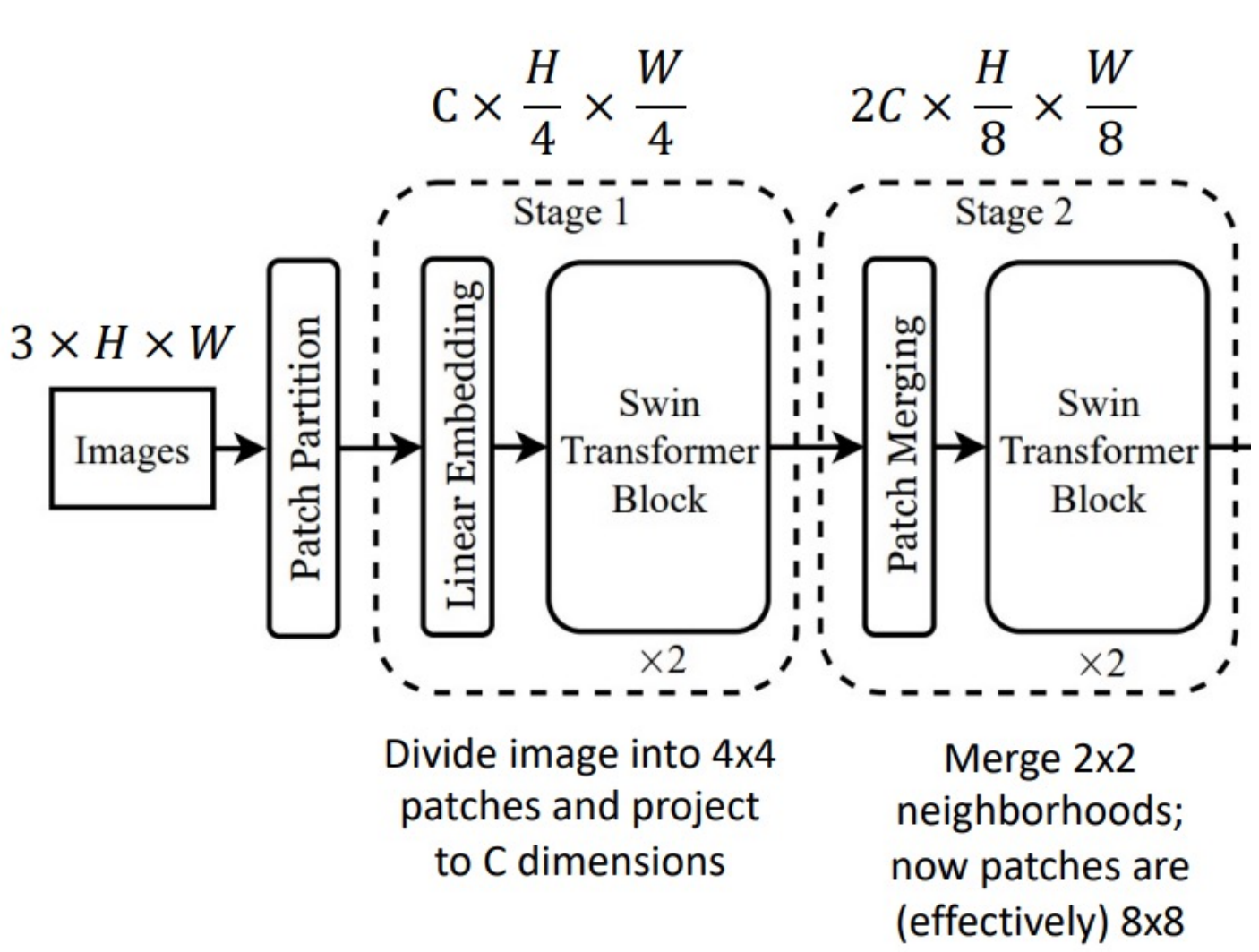


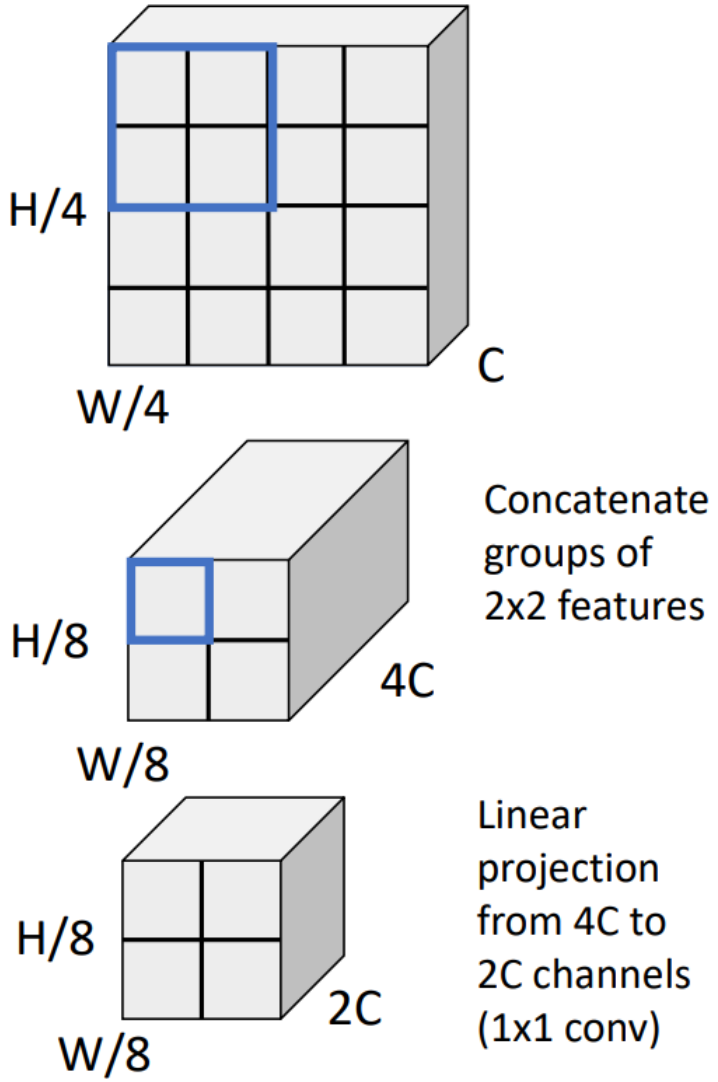


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

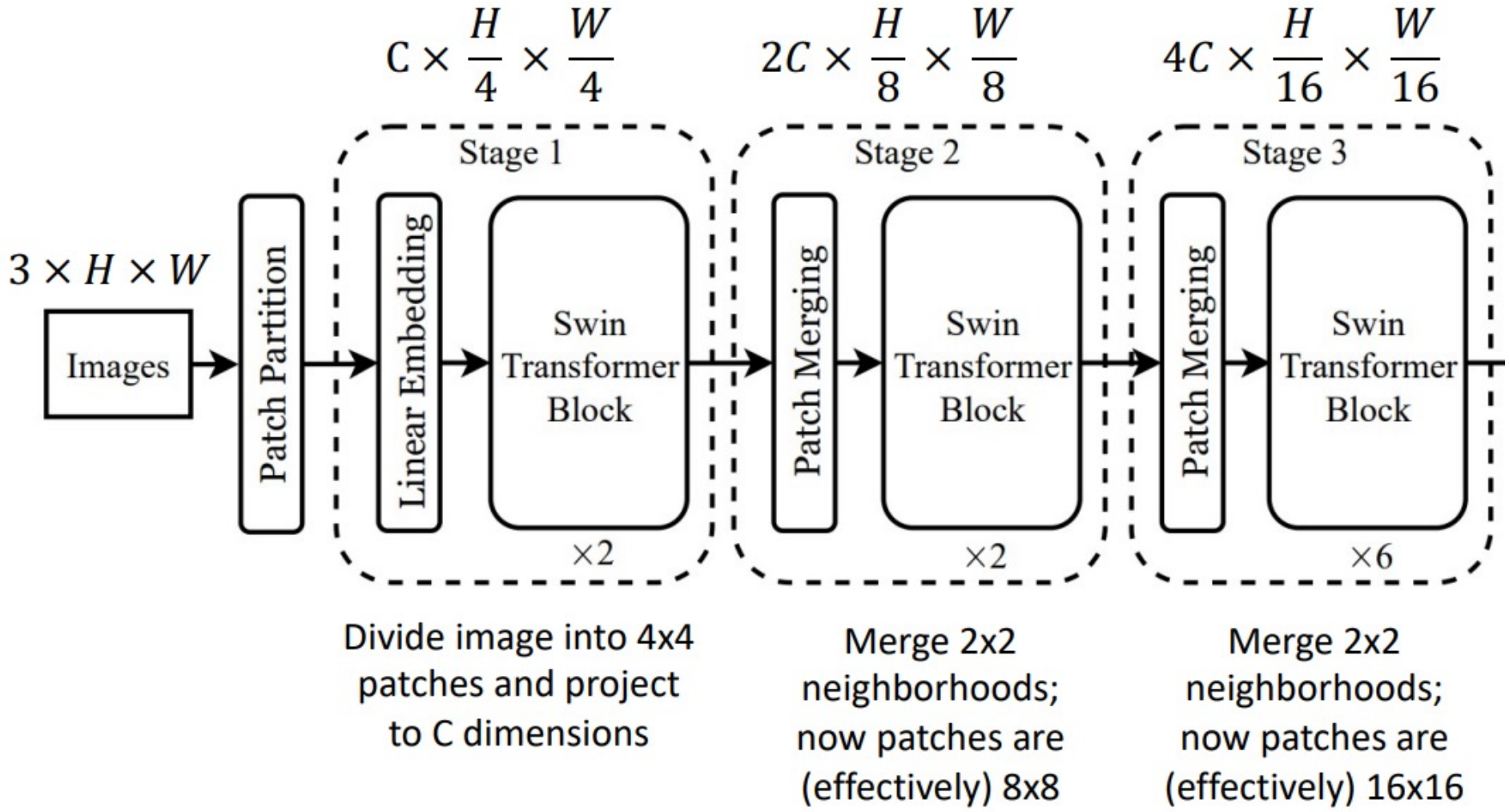




Liu et al, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows", CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

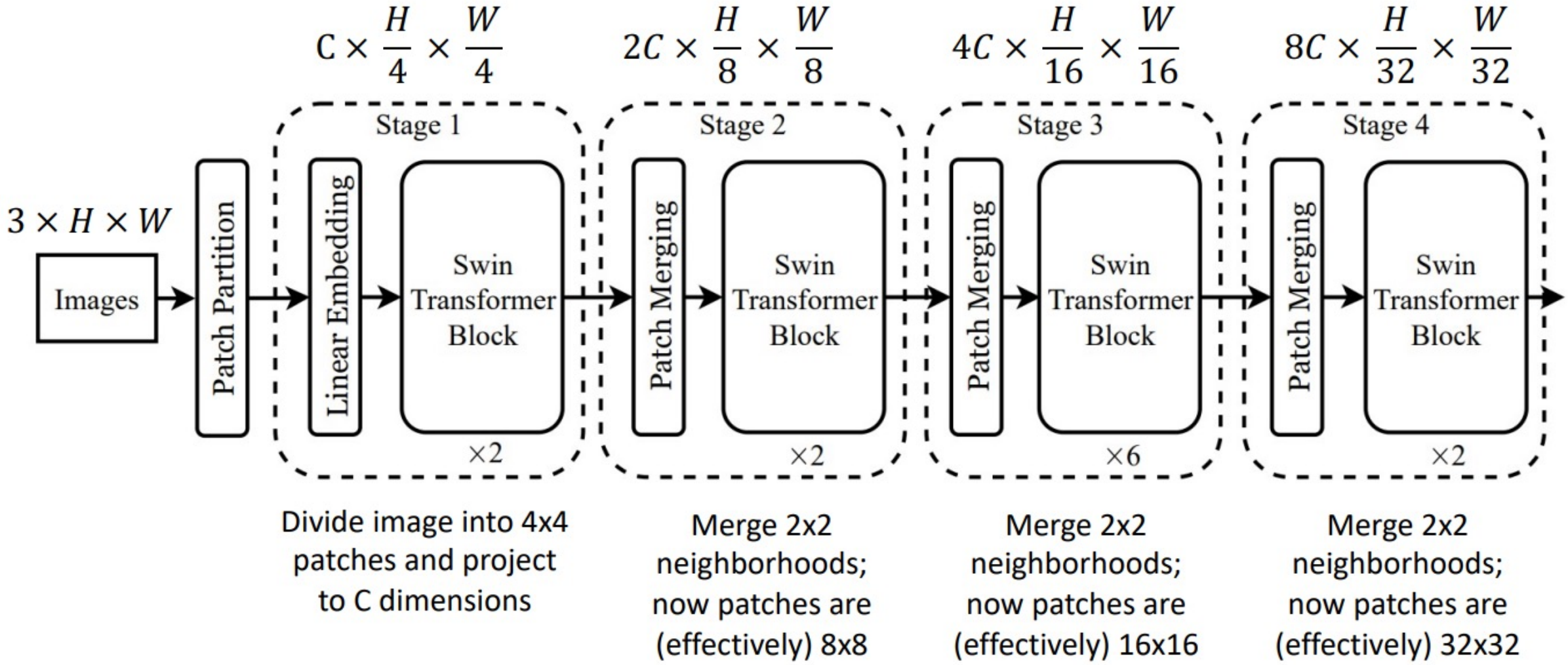
Hierarchical ViT: Swin Transformer



Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

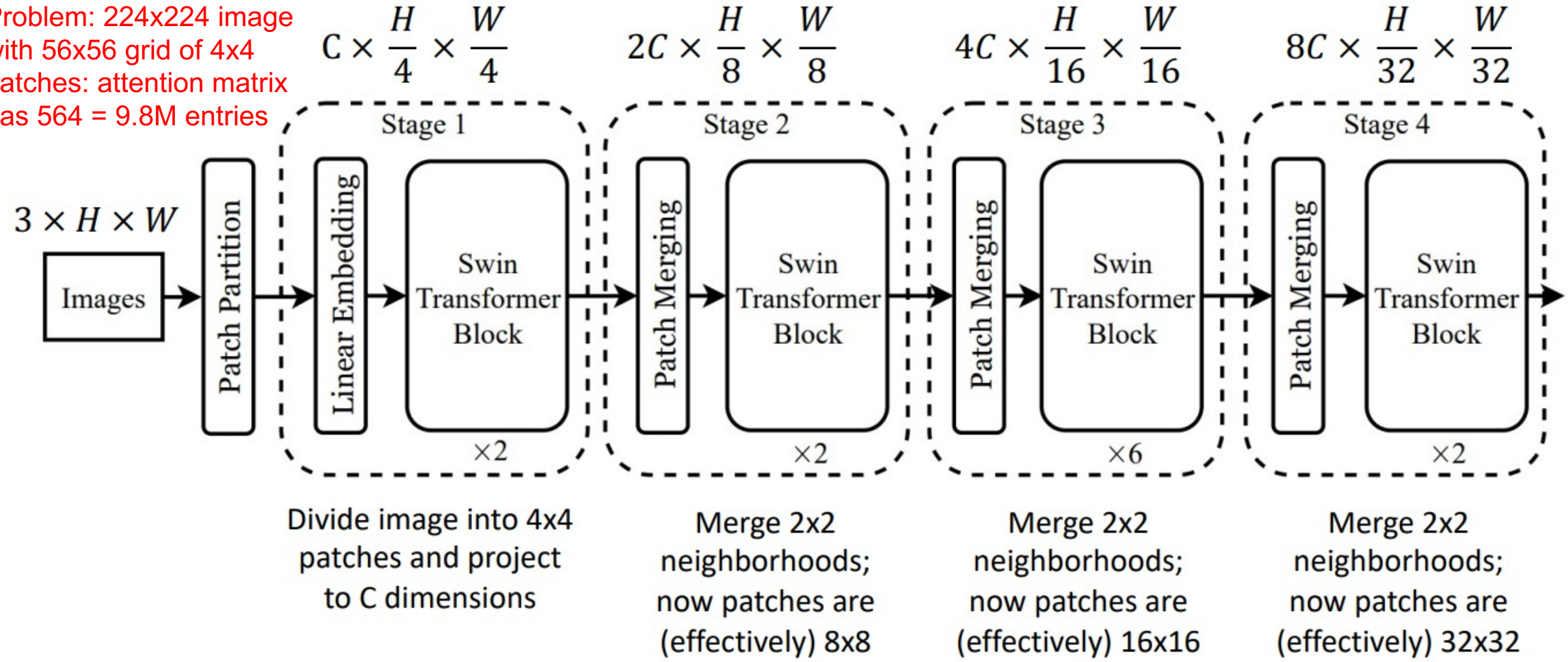


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

Problem: 224x224 image with 56x56 grid of 4x4 patches: attention matrix has 564 = 9.8M entries

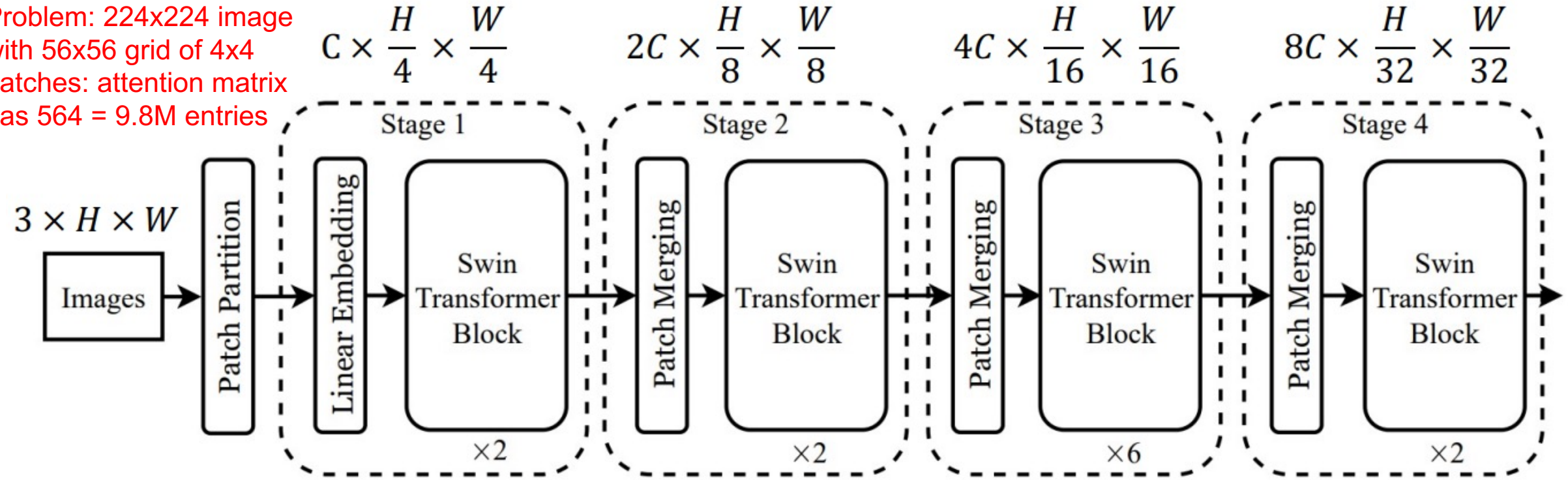


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Hierarchical ViT: Swin Transformer

Problem: 224x224 image with 56x56 grid of 4x4 patches: attention matrix has 564 = 9.8M entries



Solution: don't use full attention, instead use attention over patches

Divide image into 4x4 patches and project to C dimensions

Merge 2x2 neighborhoods; now patches are (effectively) 8x8

Merge 2x2 neighborhoods; now patches are (effectively) 16x16

Merge 2x2 neighborhoods; now patches are (effectively) 32x32

Liu et al, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows", CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Window Attention

With H x W grid of tokens, each attention matrix is H^2W^2 – **quadratic** in image size

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Window Attention

With H x W grid of tokens, each attention matrix is H^2W^2 – **quadratic** in image size

Rather than allowing each token to attend to all other tokens, instead divide into **windows** of M x M tokens (here M=4); only compute attention within each window

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Window Attention

With H x W grid of tokens, each attention matrix is H^2W^2 – **quadratic** in image size

Rather than allowing each token to attend to all other tokens, instead divide into **windows** of M x M tokens (here M=4); only compute attention within each window

Total size of all attention matrices is now: M^4(H/M)(W/M) = M^2HW

Linear in image size for fixed M! Swin uses M=7 throughout the network

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Window Attention

Problem: tokens only interact with other tokens within the same window; no communication across windows

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Shifted Window Attention

Solution: Alternate between normal windows and shifted windows in successive Transformer blocks

Ugly detail: Non-square windows at edges and corners

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Shifted Window Attention

Solution: Alternate between normal windows and shifted windows in successive Transformer blocks

Detail: Relative Positional Bias

ViT adds positional embedding to input tokens, encodes *absolute position* of each token in the image

Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Shifted Window Attention

Solution: Alternate between normal windows and shifted windows in successive Transformer blocks

Block L: Normal windows

Block L+1: Shifted Windows

Detail: Relative Positional Bias

ViT adds positional embedding to input tokens, encodes *absolute position* of each token in the image

Swin does not use positional embeddings, instead encodes *relative position* between patches when computing attention:

Standard Attention:

$$A = Softmax\left(\frac{QK^T}{\sqrt{D}}\right)V$$

$Q, K, V: M^2 \times D$ (Query, Key, Value)

Liu et al, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows", CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Shifted Window Attention

Solution: Alternate between normal windows and shifted windows in successive Transformer blocks

Block L: Normal windows

Block L+1: Shifted Windows

Detail: Relative Positional Bias

ViT adds positional embedding to input tokens, encodes *absolute position* of each token in the image

Swin does not use positional embeddings, instead encodes *relative position* between patches when computing attention:

Attention with relative bias:

$$A = Softmax\left(\frac{QK^T}{\sqrt{D}} + B\right)V$$

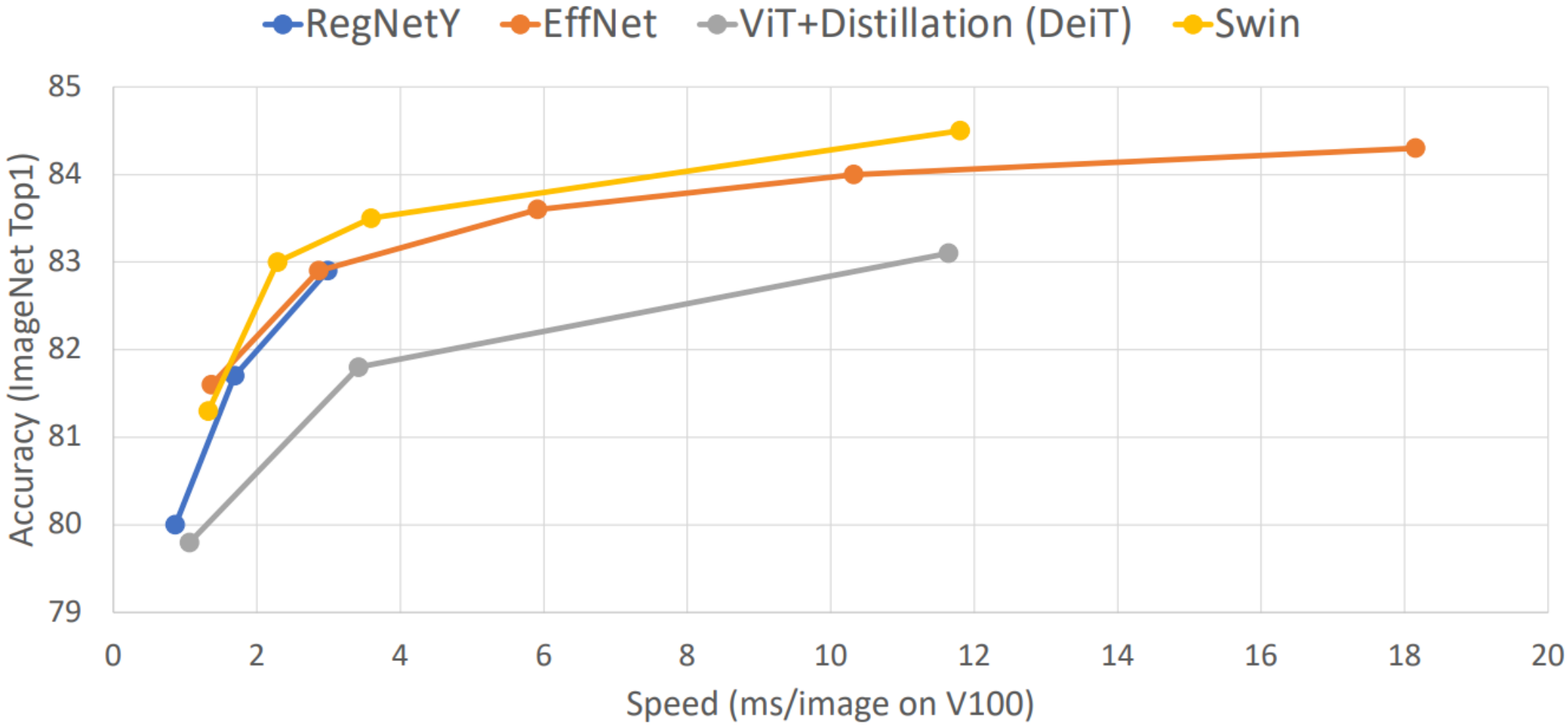
$Q, K, V: M^2 \times D$ (Query, Key, Value)

$B: M^2 \times M^2$ (learned biases)

Liu et al, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows", CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

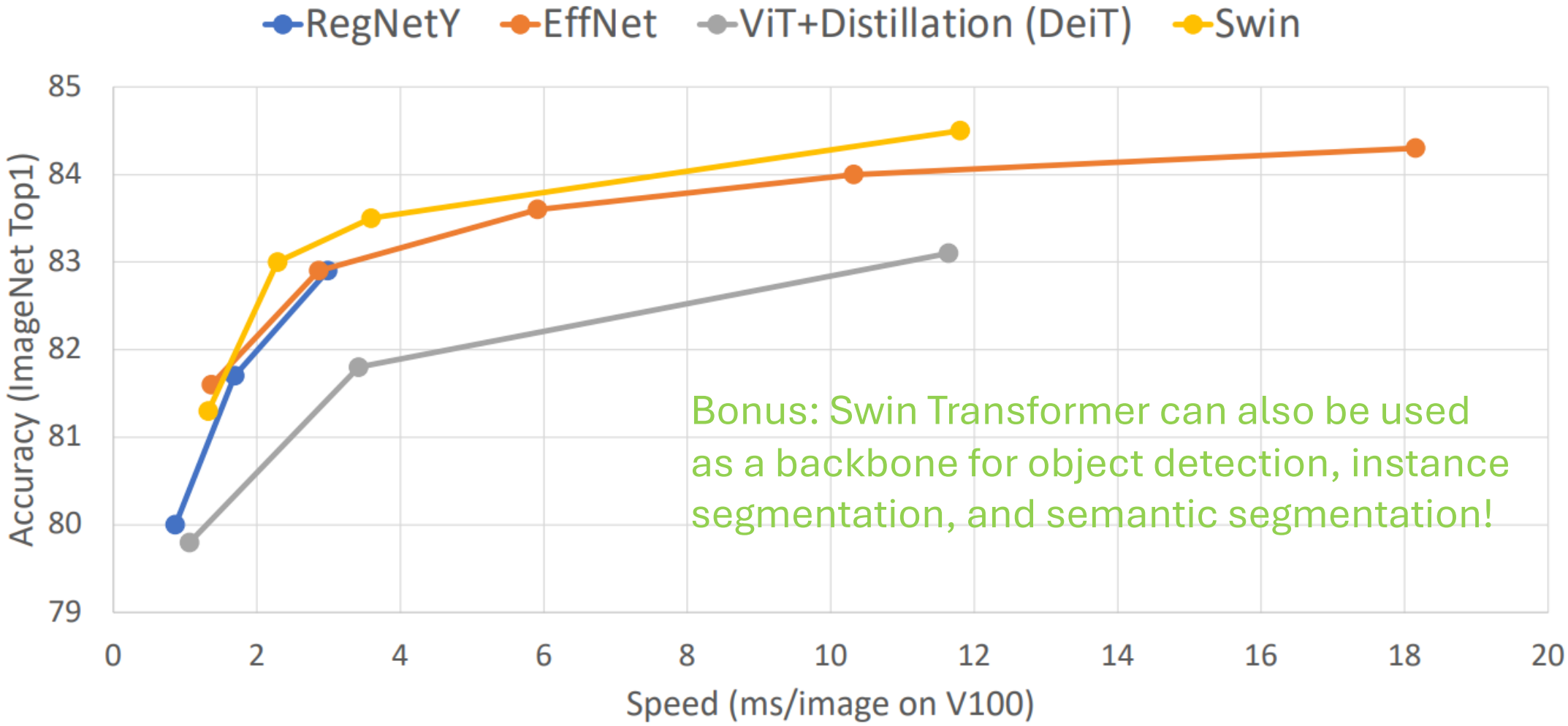
Swin Transformer: Speed vs Accuracy



Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Swin Transformer: Speed vs Accuracy

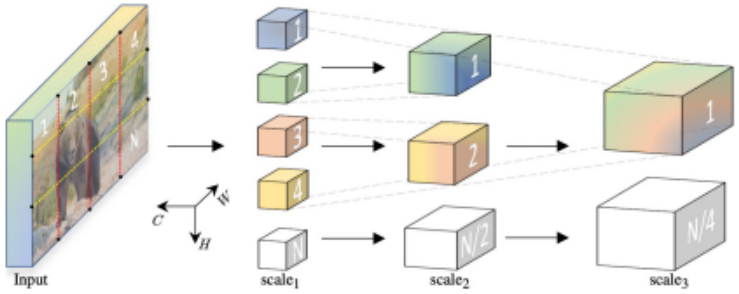


Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

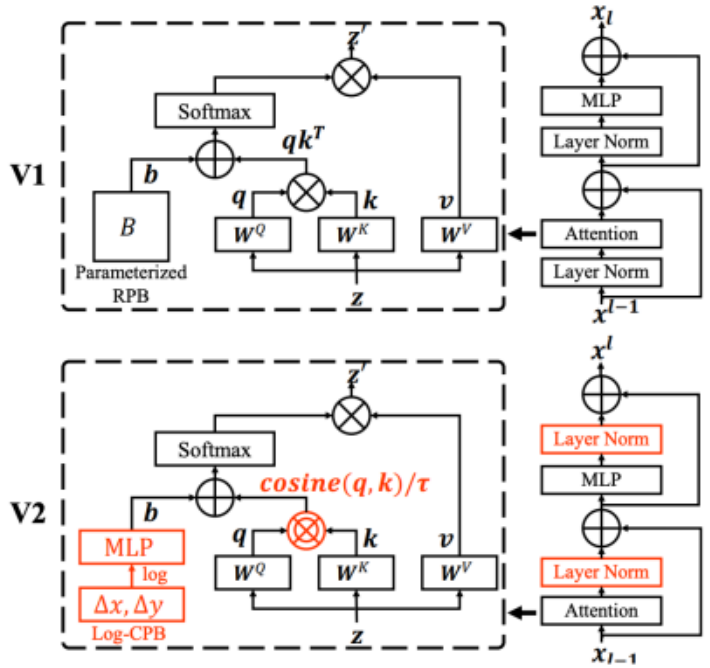
Other Hierarchical Vision Transformers

MViT



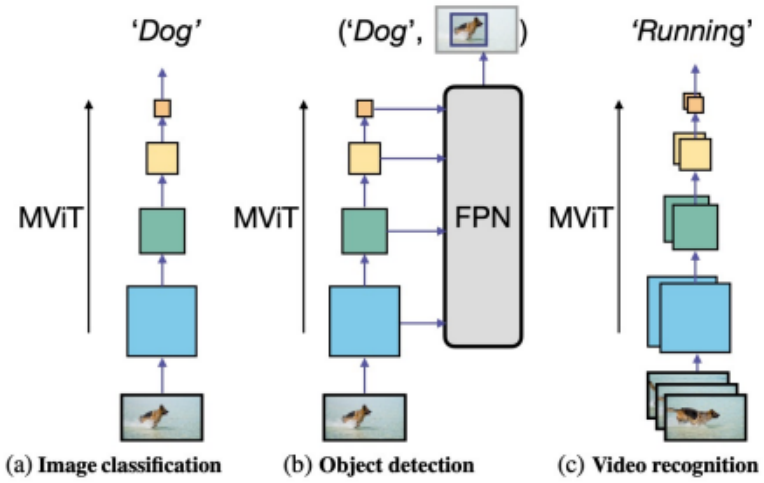
Fan et al, "Multiscale Vision Transformers", ICCV 2021

Swin-V2



Liu et al, "Swin Transformer V2: Scaling up Capacity and Resolution", CVPR 2022

Improved MViT



Li et al, "Improved Multiscale Vision Transformers for Classification and Detection", arXiv 2021

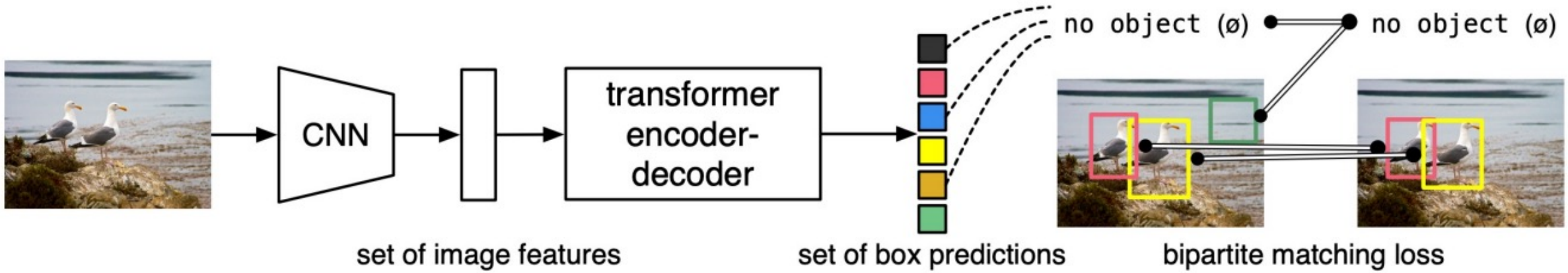
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Object Detection with Transformers: DETR

Simple object detection pipeline: directly output a set of boxes from a Transformer

No anchors, no regression of box transforms

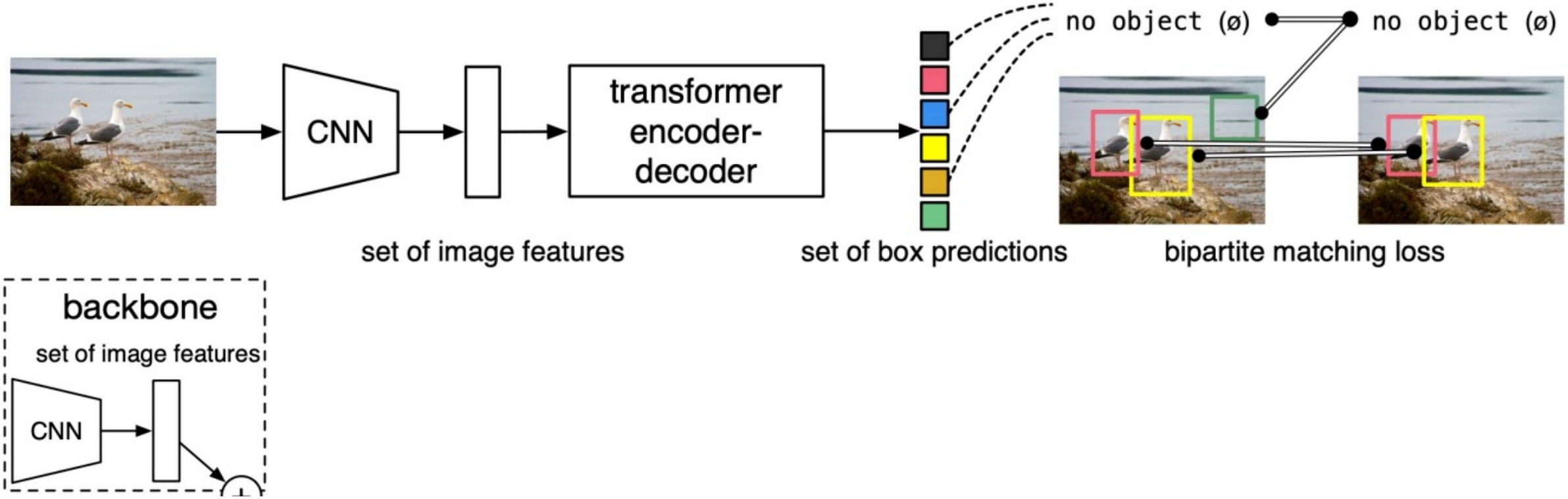
Match predicted boxes to GT boxes with bipartite matching; train to regress box coordinates



Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

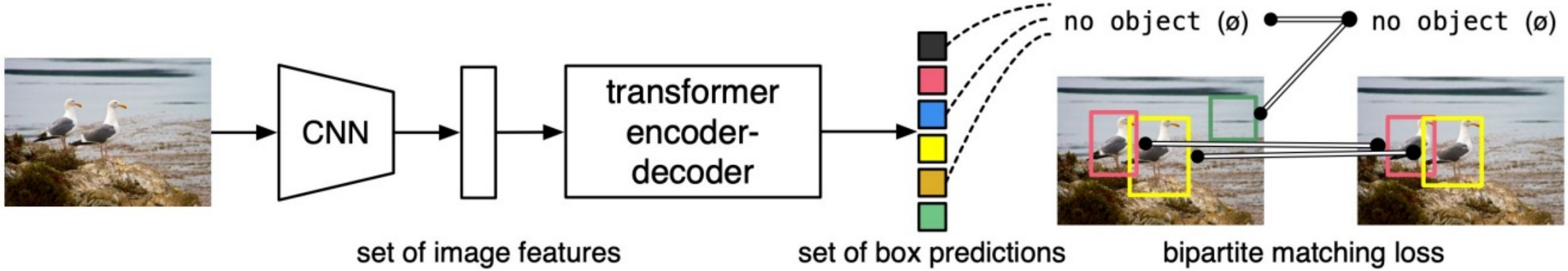
Object Detection with Transformers: DETR

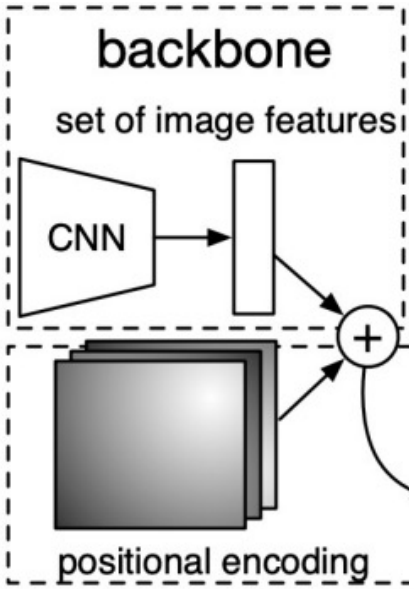


Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Object Detection with Transformers: DETR

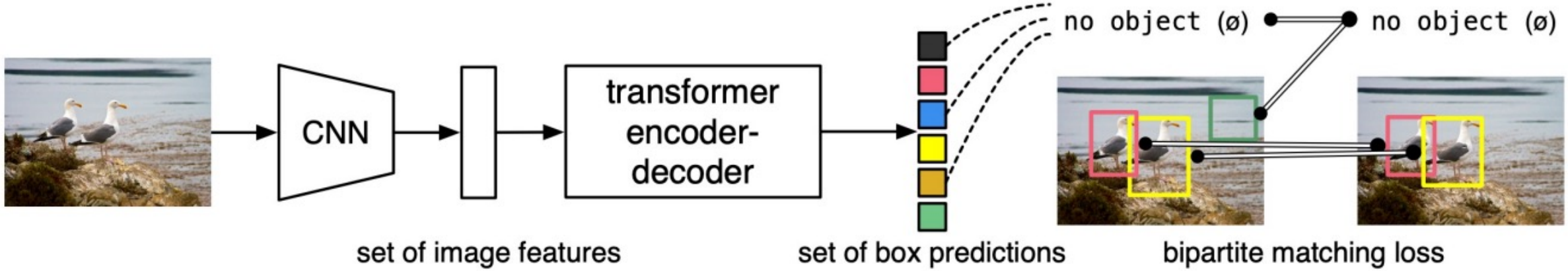


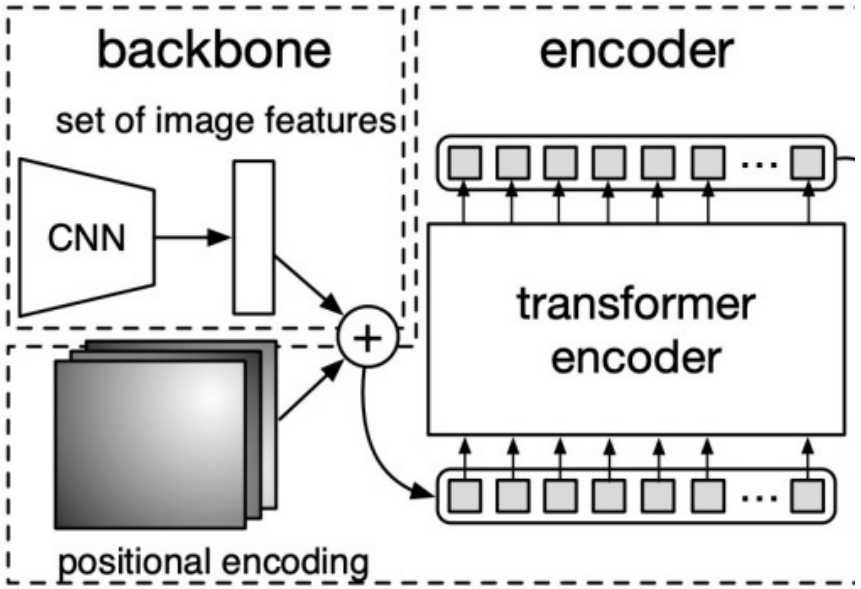


Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Object Detection with Transformers: DETR

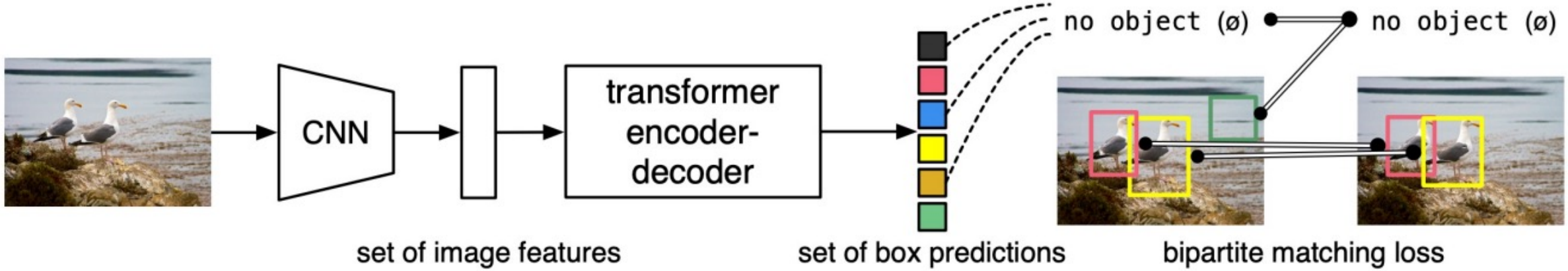


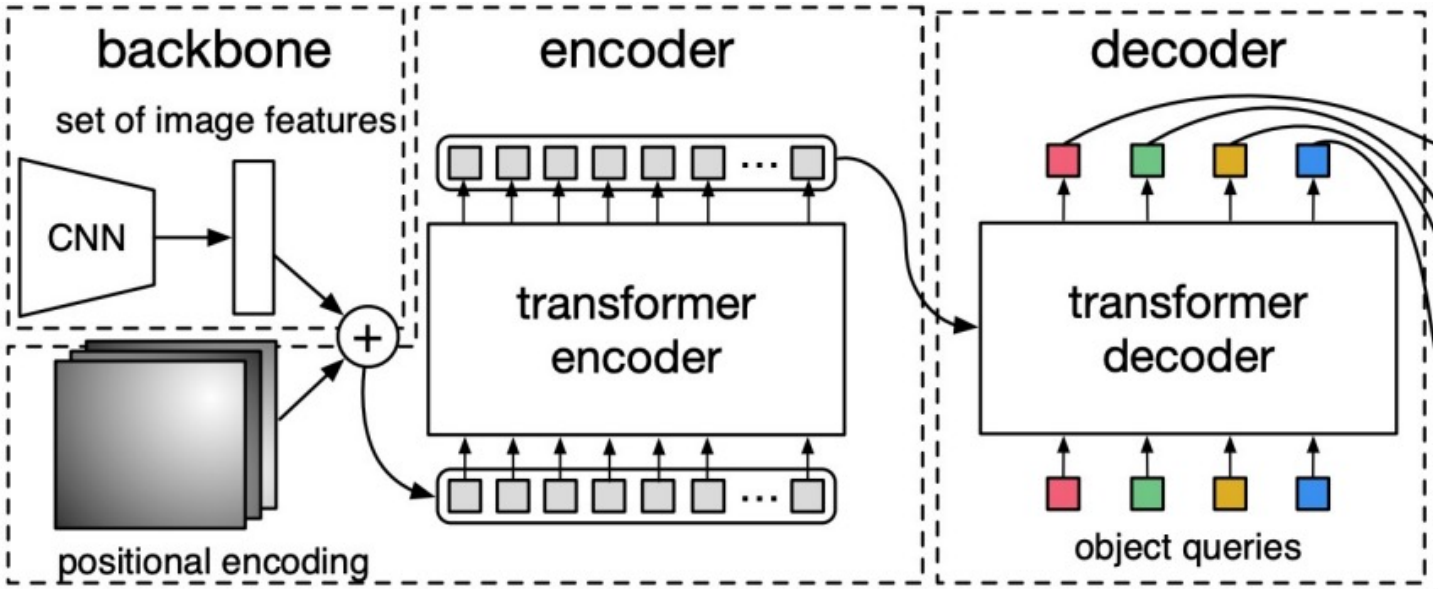


Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Object Detection with Transformers: DETR

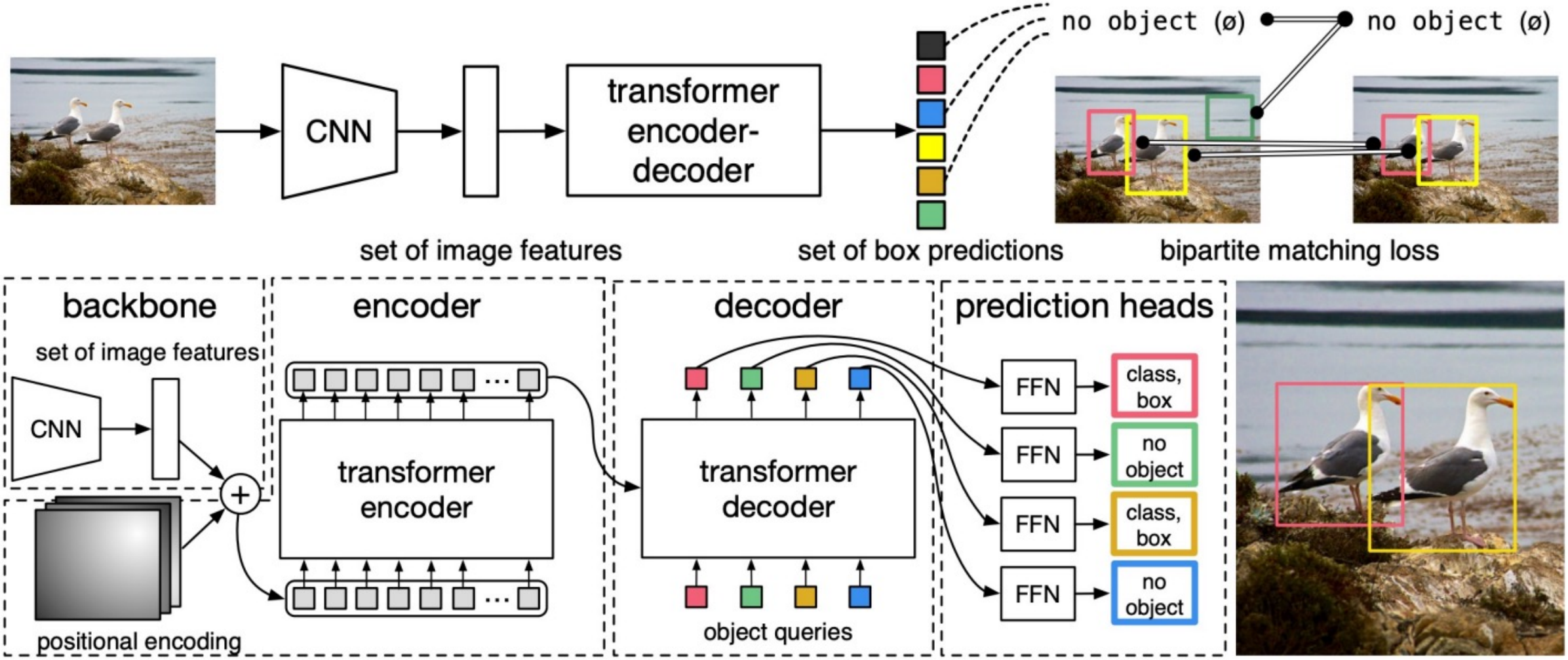




Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Object Detection with Transformers: DETR



Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Summary

Vision Transformers have been a super hot topic since introduction!

Very different architecture vs traditional CNNs

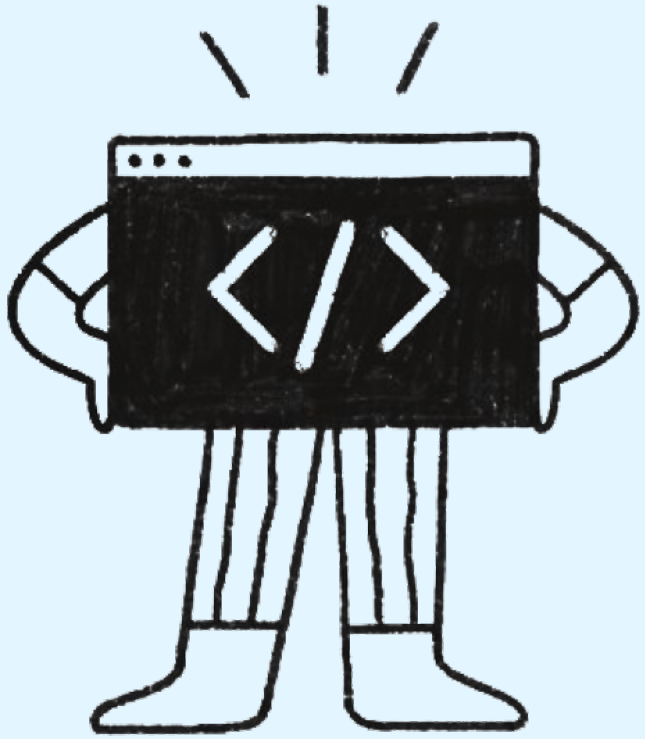
Applications to all tasks: classification, detection, segmentation, etc

My takeaway: Vison transformers are an evolution, not a revolution. We can still fundamentally solve the same problems as with CNNs.

Main benefit is probably speed: Matrix multiply is more hardwarefriendly than convolution, so ViTs with same FLOPs as CNNs can train and run much faster

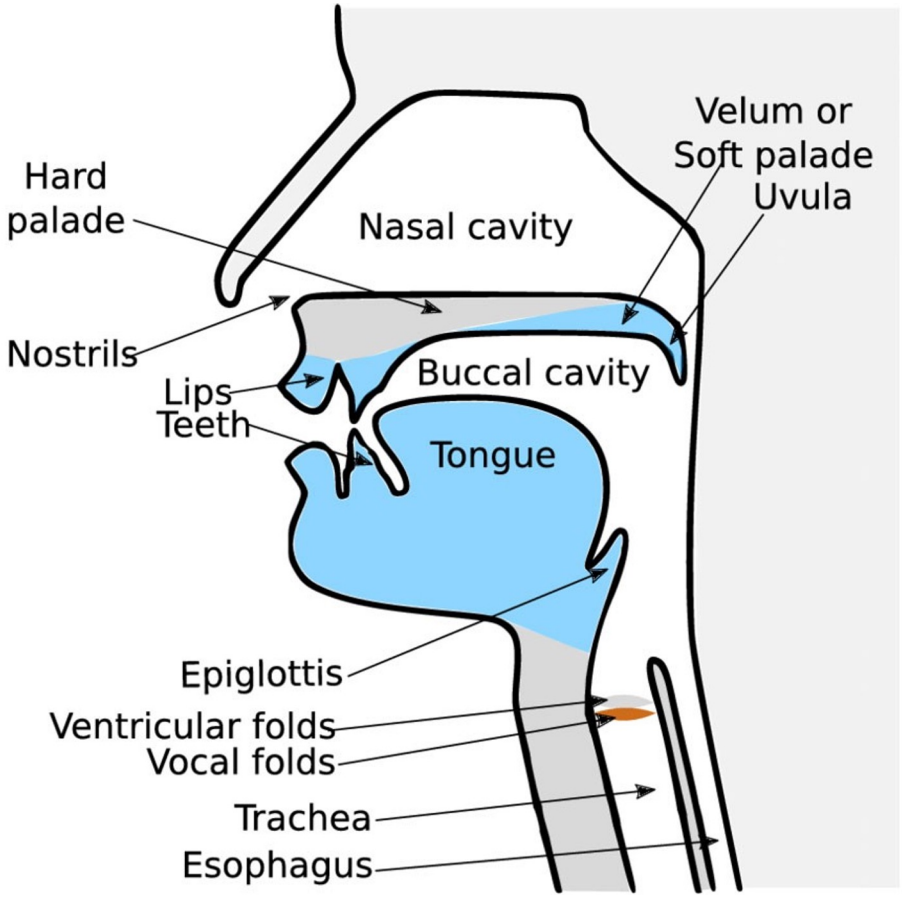
Slides original source: Deep Learning for Computer Vision 2022 Lecture 18

Бонус: обработка звука



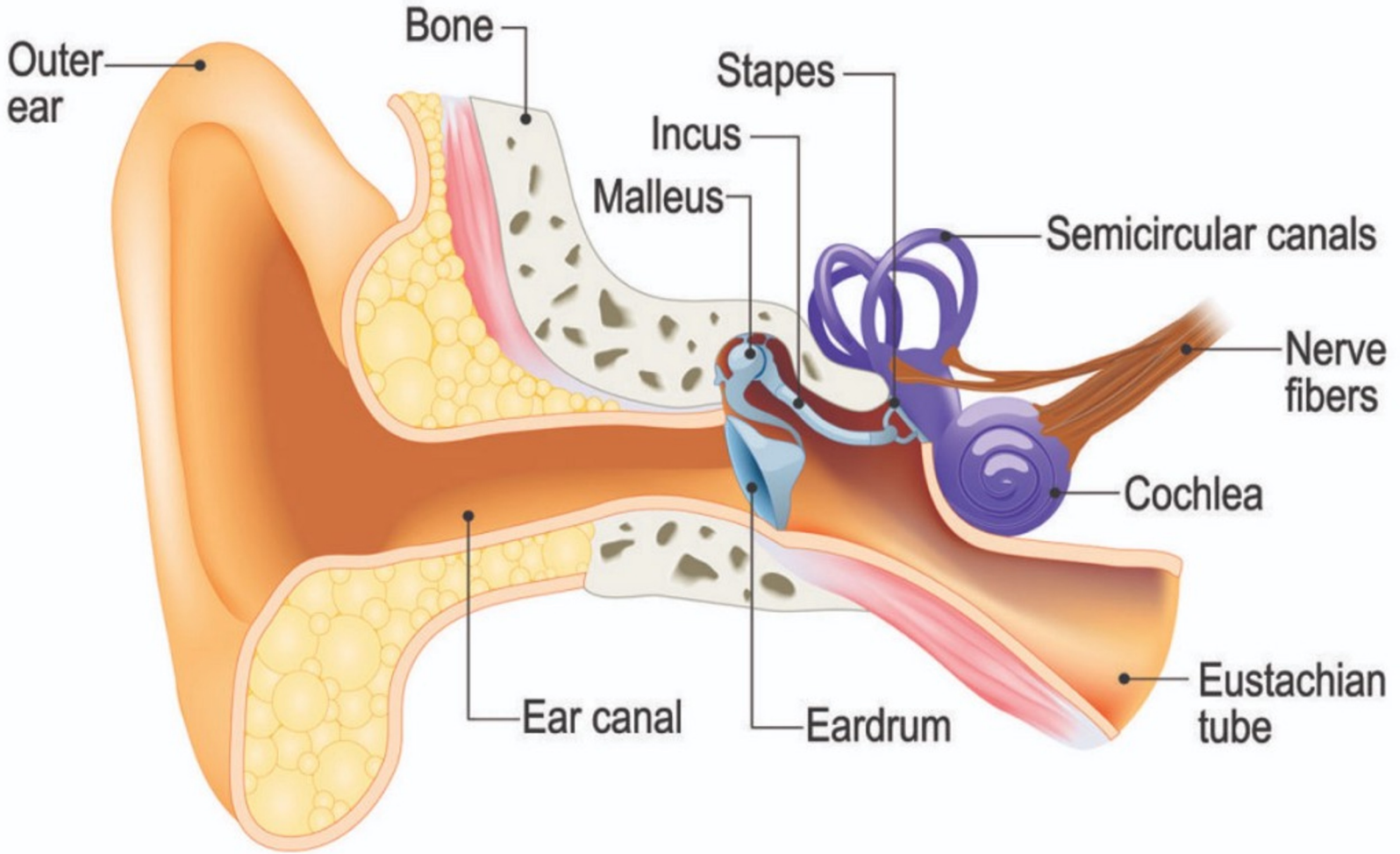
05

Foundations: vocal tract



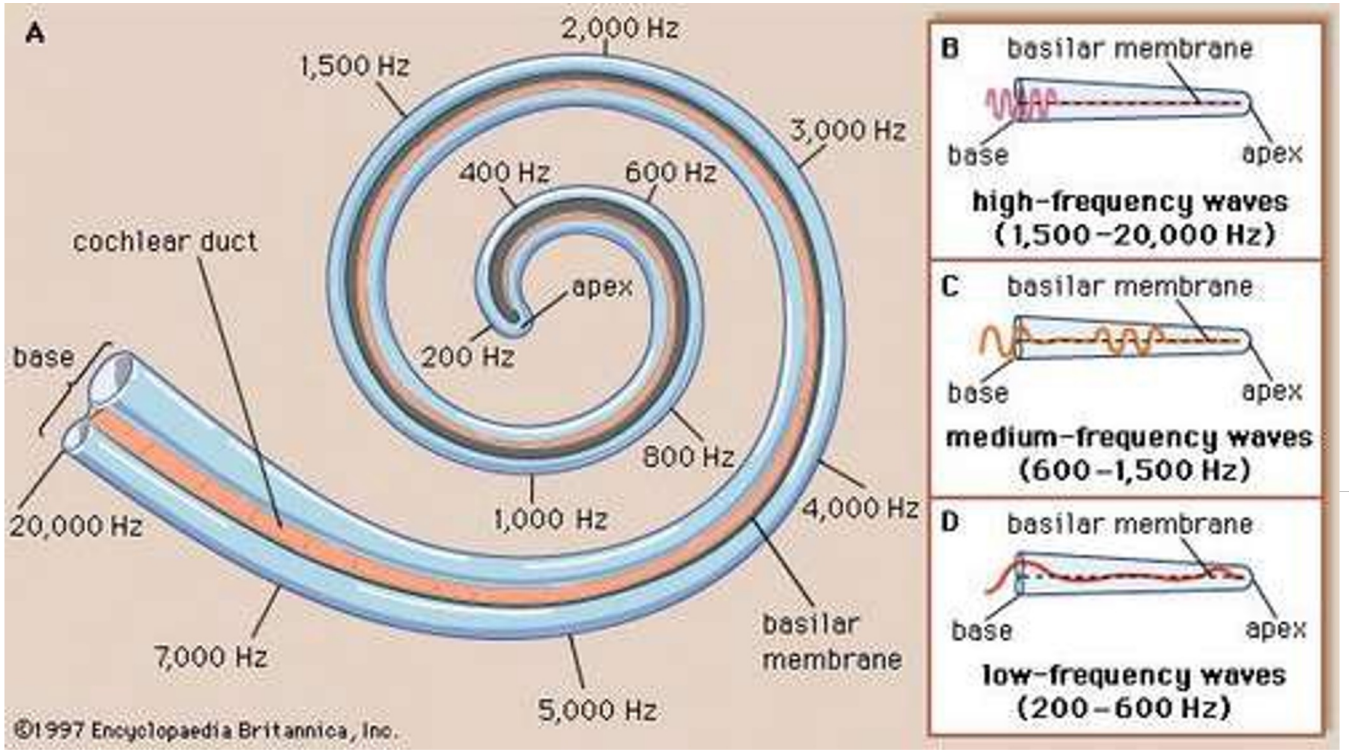
source: Medium post

Foundations: ear structure



source: Medium post

Foundations: ear structure



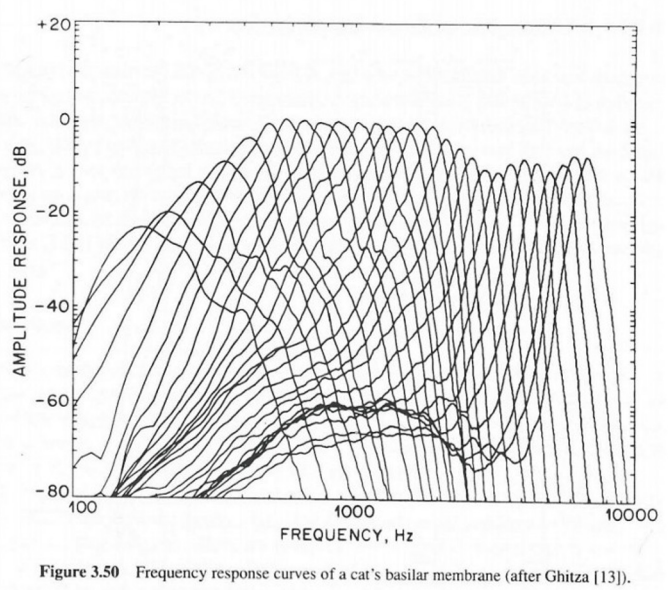


Figure 3.50 Frequency response curves of a cat's basilar membrane (after Ghitza [13]).

source: Medium post

Processing the sound

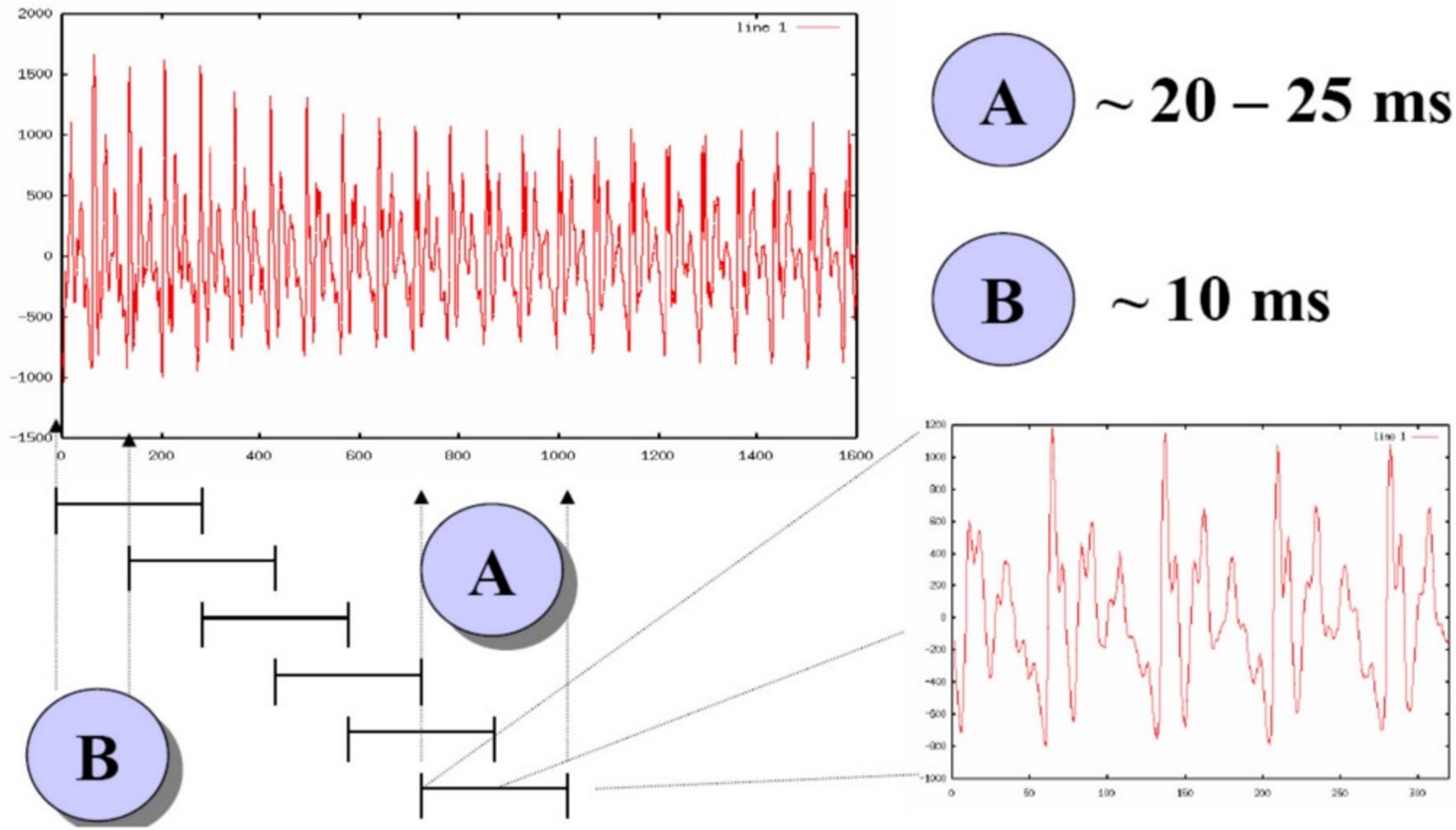


Image from Bryan Pellom

source: Medium post

Fourier Transform

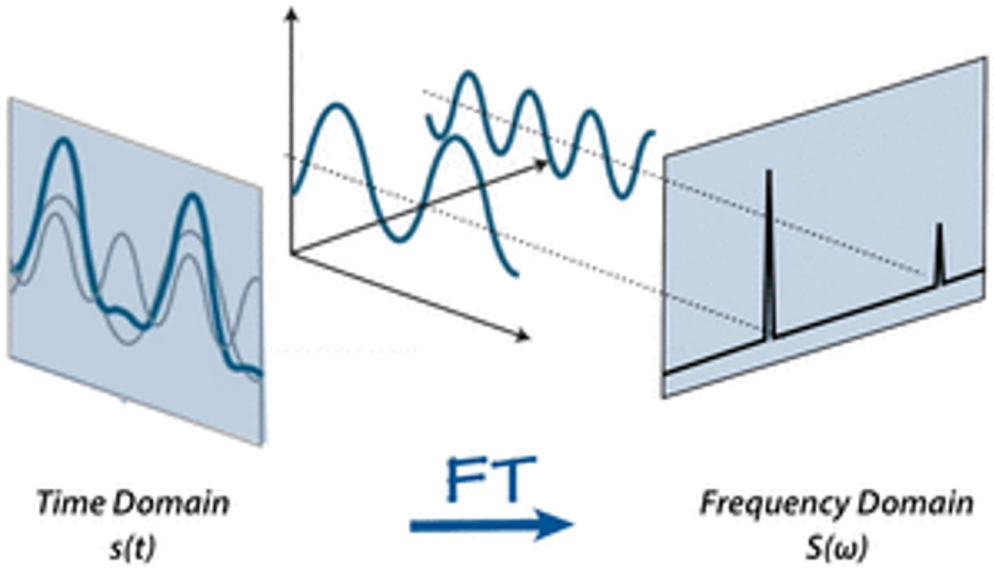
Definition [edit]

The *discrete Fourier transform* transforms a sequence of *N* complex numbers $\{\mathbf{x_n}\} := x_0, x_1, \ldots, x_{N-1}$ into another sequence of complex numbers, $\{\mathbf{X_k}\} := X_0, X_1, \ldots, X_{N-1}$, which is defined by

$$\begin{aligned} X_k &= \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N} kn} \\ &= \sum_{n=0}^{N-1} x_n \cdot \left[\cos\left(\frac{2\pi}{N} kn\right) - i \cdot \sin\left(\frac{2\pi}{N} kn\right)\right], \end{aligned} \qquad \text{(Eq.1)}$$

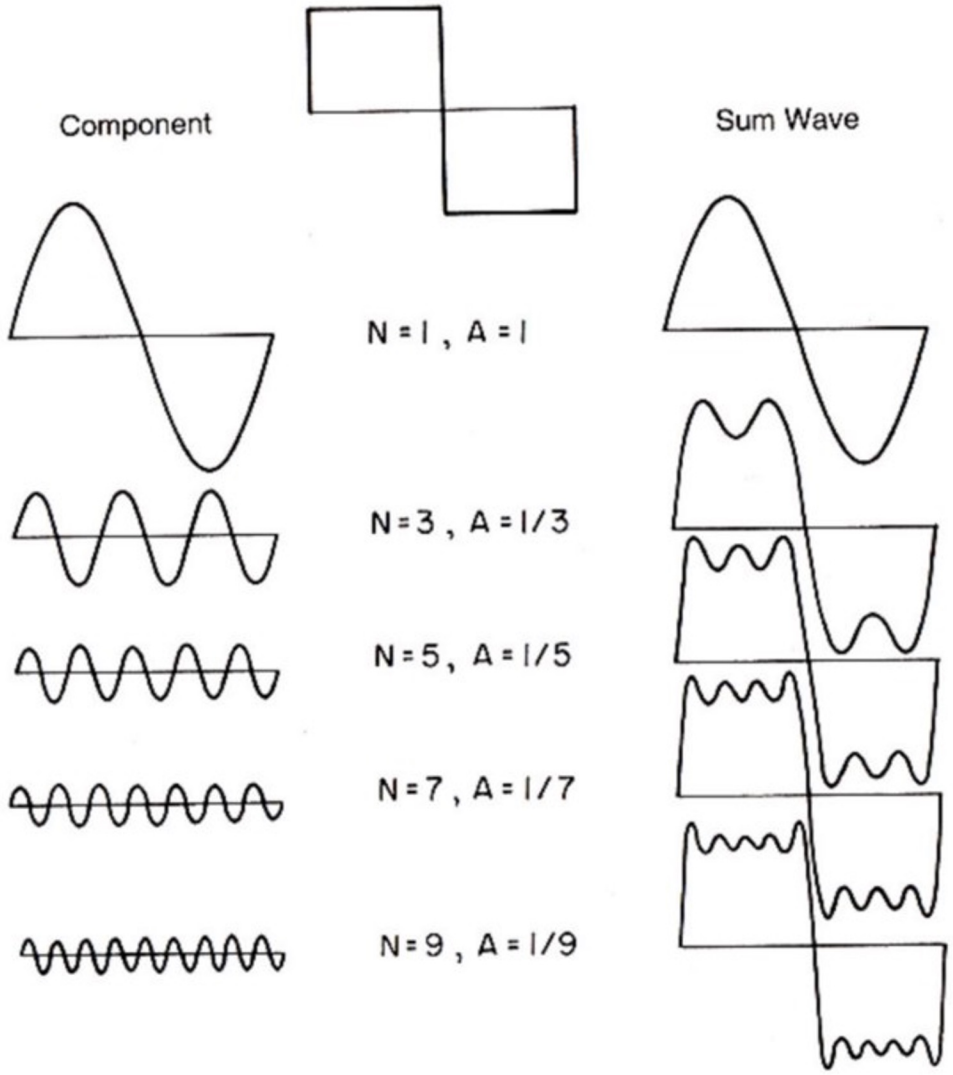
where the last expression follows from the first one by Euler's formula.

The transform is sometimes denoted by the symbol $\mathcal{F}$, as in $\mathbf{X} = \mathcal{F}\{\mathbf{x}\}$ or $\mathcal{F}(\mathbf{x})$ or $\mathcal{F}\mathbf{x}$.[A]



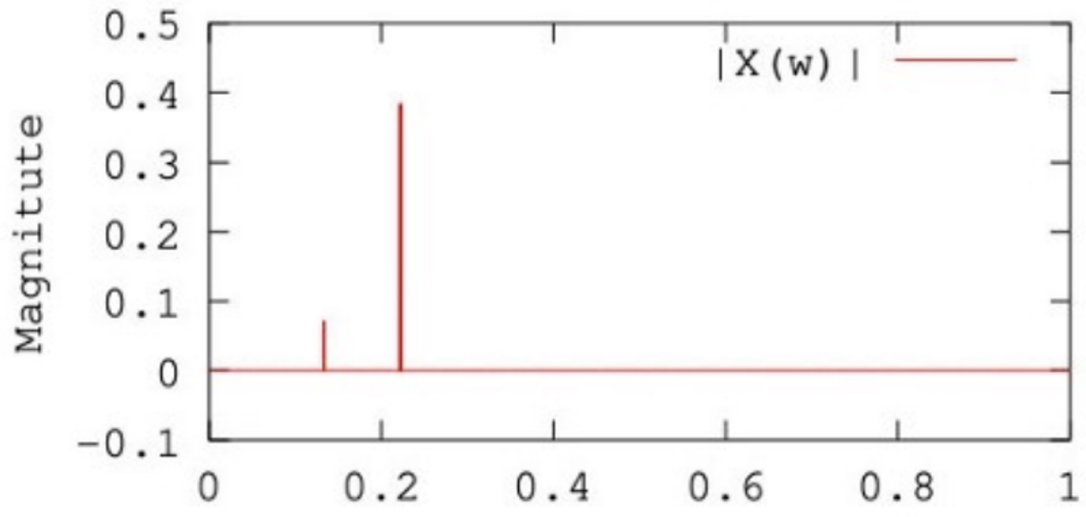
source: Wikipedia and Medium post

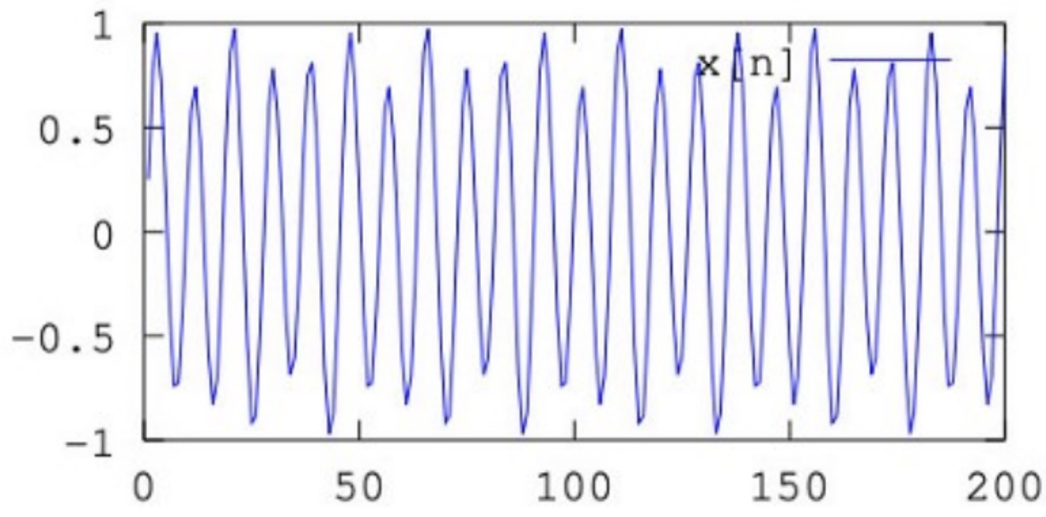
Fourier Transform

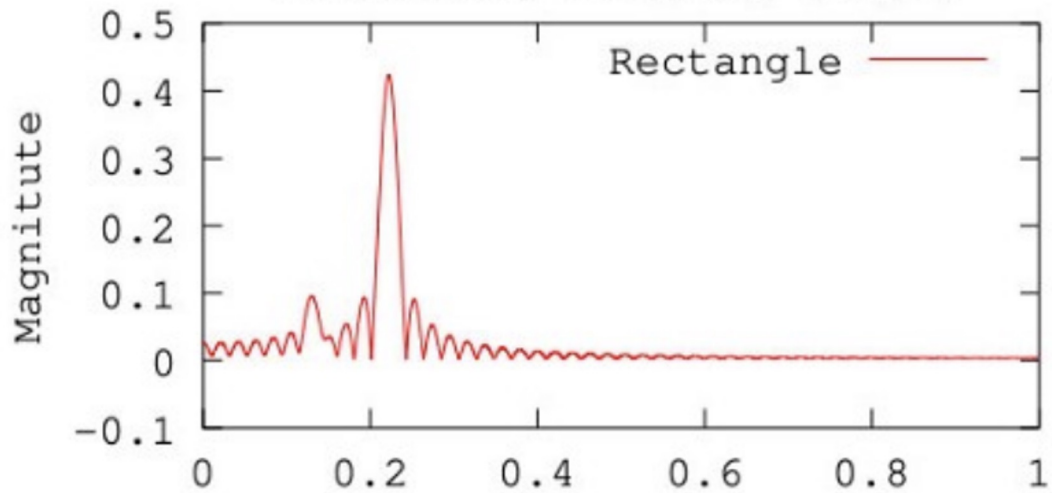


source: Wikipedia and Medium post

Fourier Transform: filtering

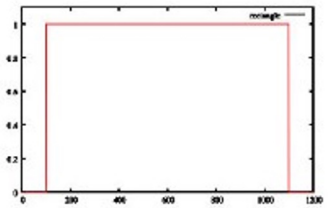




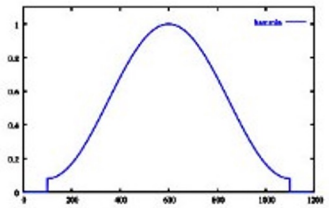


source: Medium post

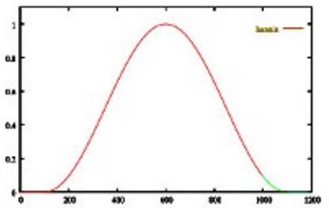
Fourier Transform: filtering



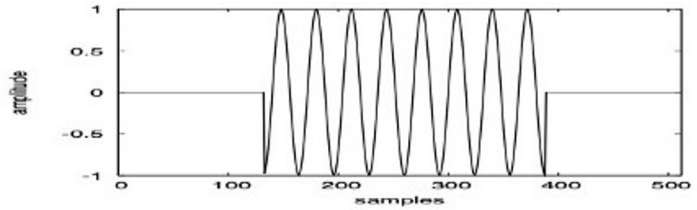
Rectangular



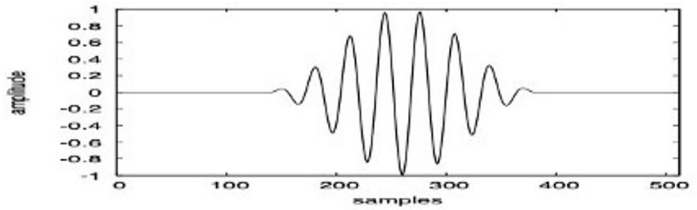
Hamming



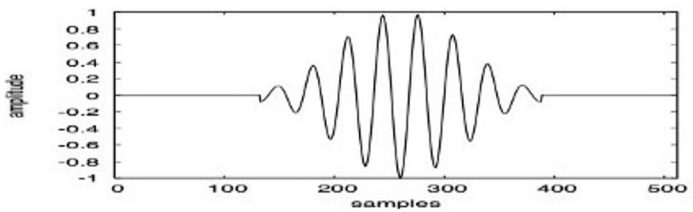
Hanning



(a) Rectangular window



(b) Hanning window

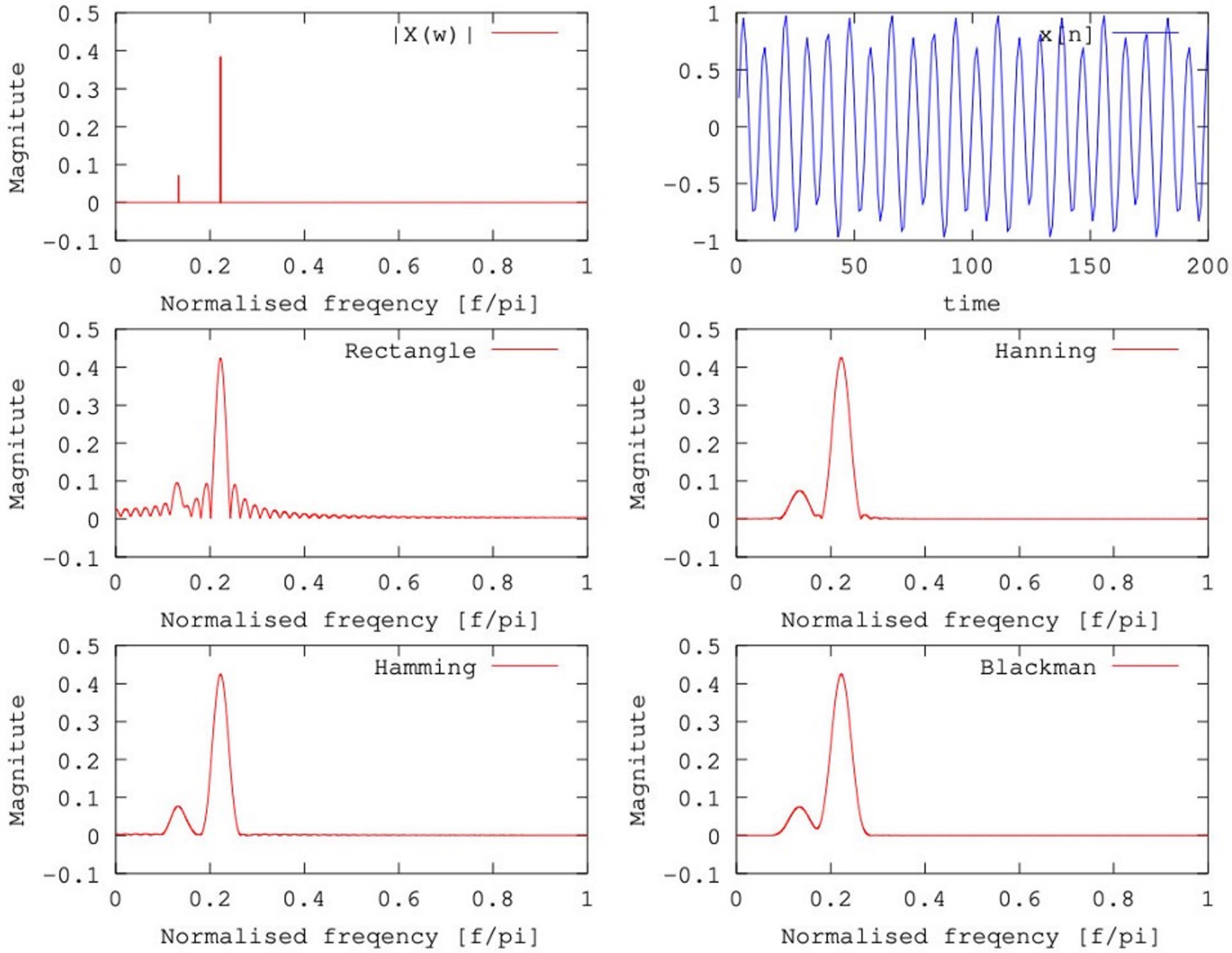


(c) Hamming window

(Taylor, fig 12.1)

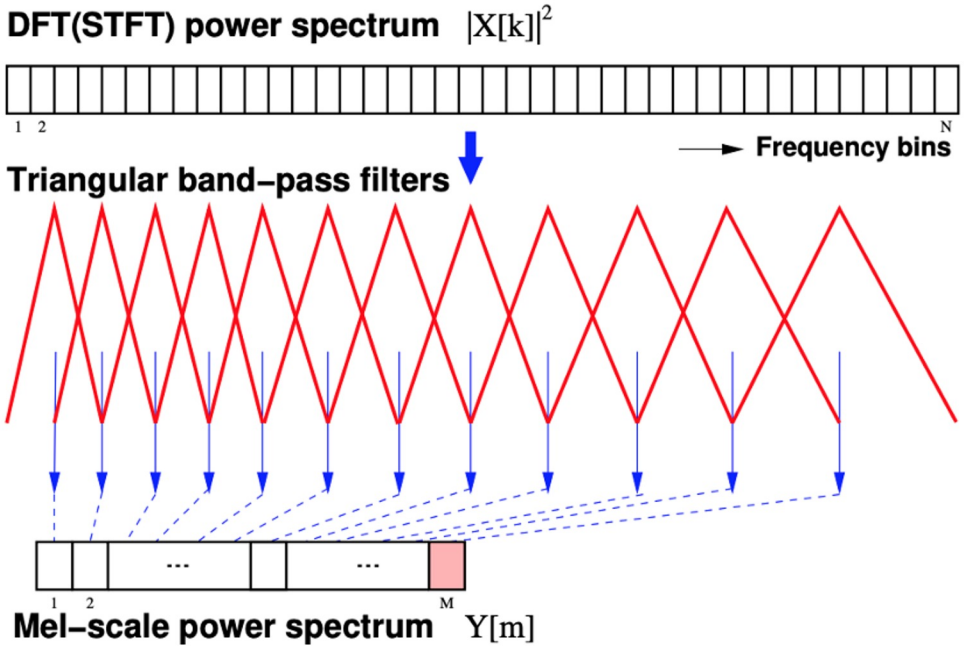
source: Medium post

Fourier Transform: filtering



source: Medium post

Features generation



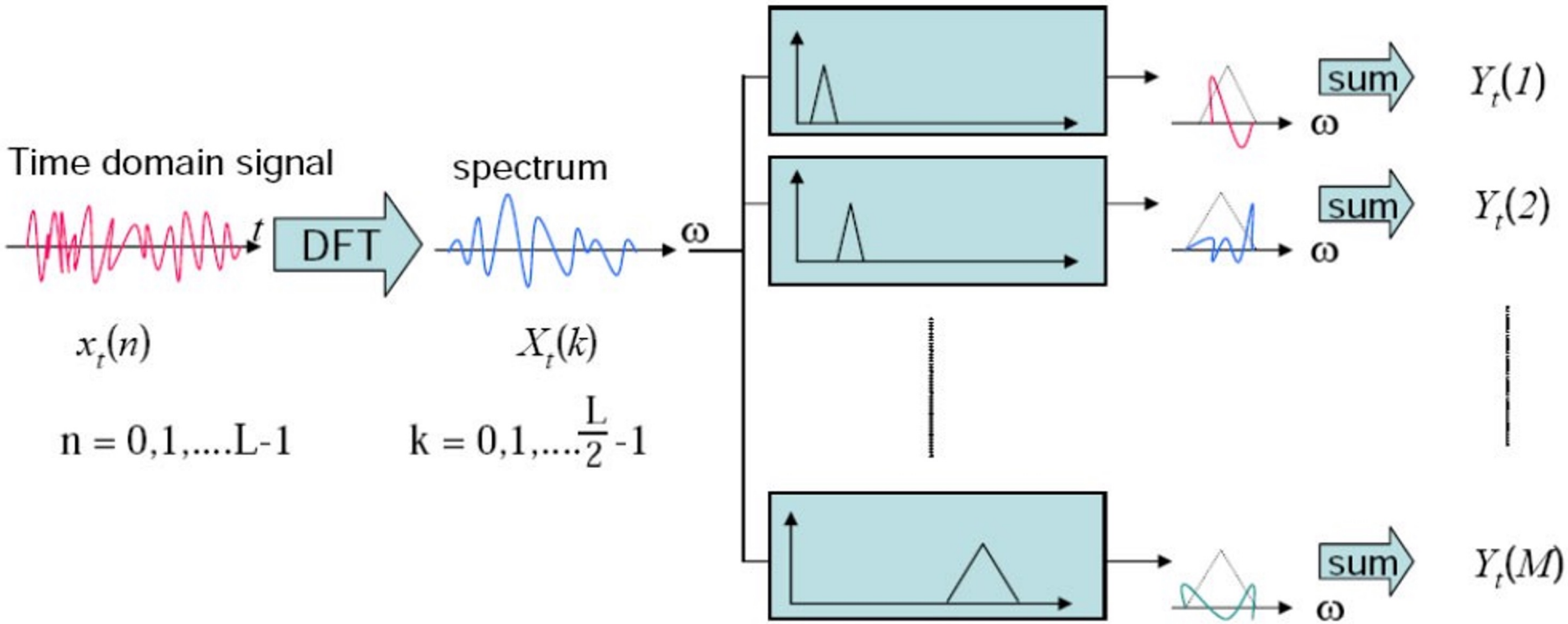
$$Y_t[m] = \sum_{k=1}^{N} W_m[k] \, |X_t[k]|^2$$

where k : DFT bin number $(1, \ldots, N)$

m : mel-filter bank number $(1, \ldots, M)$

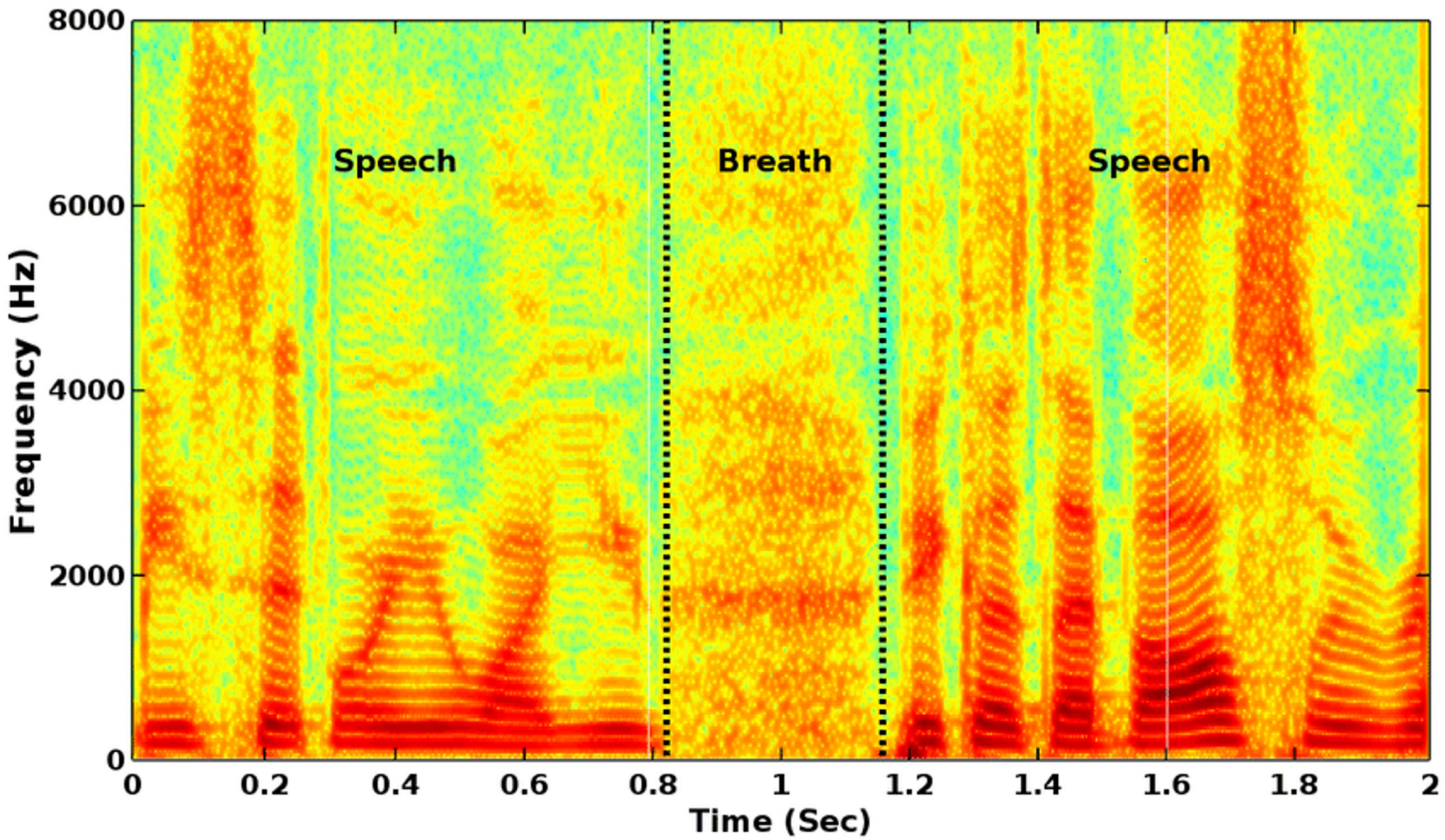
source: Medium post

Features generation



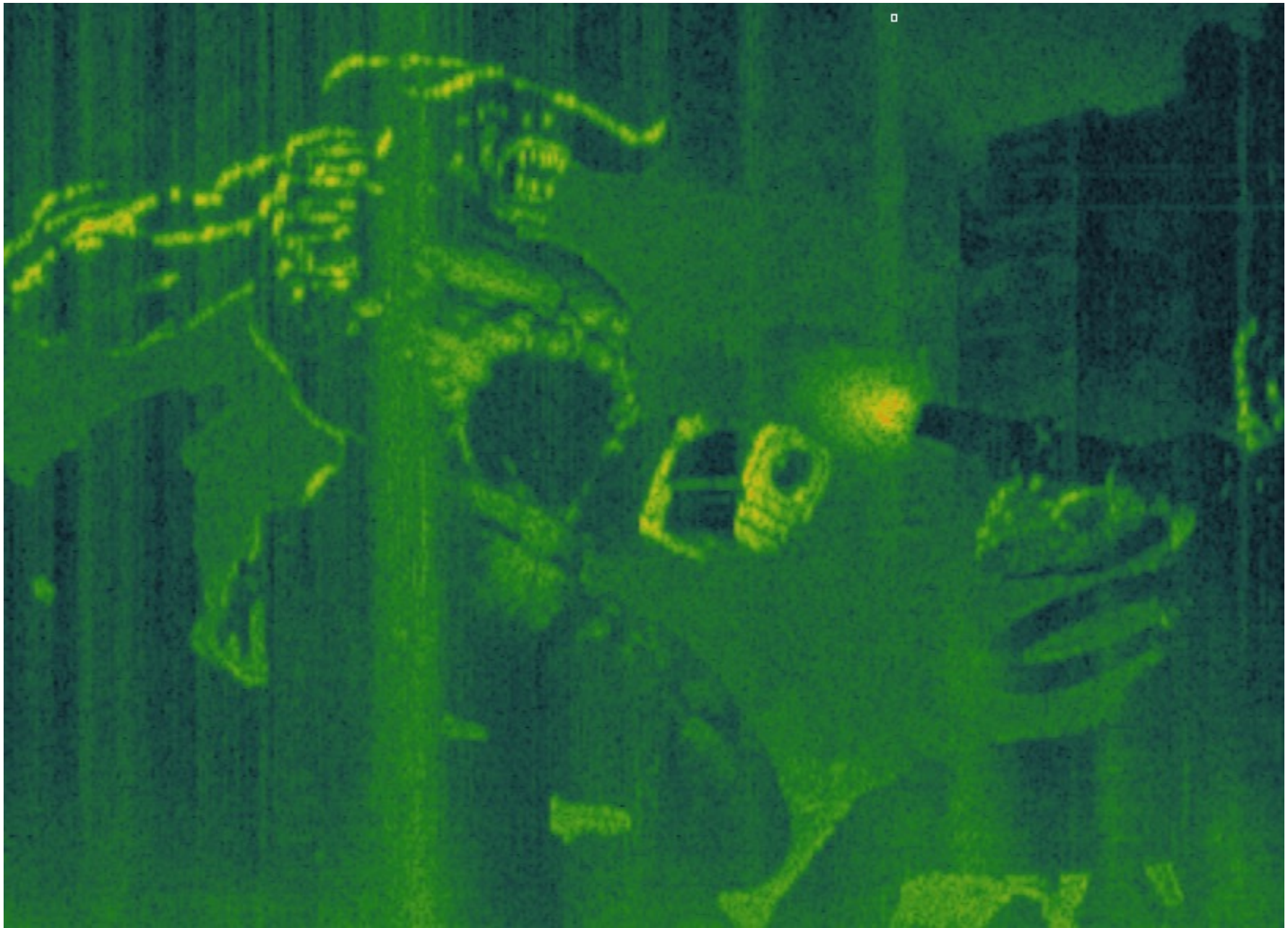
source: Medium post

Spectrogram



Source: An Algorithm for Detection of Breath Sounds in Spontaneous Speech with Application to Speaker Recognition

Spectrograms as Easter eggs. DOOM Eternal soundtrack



Source: reddit thread

Embeddings

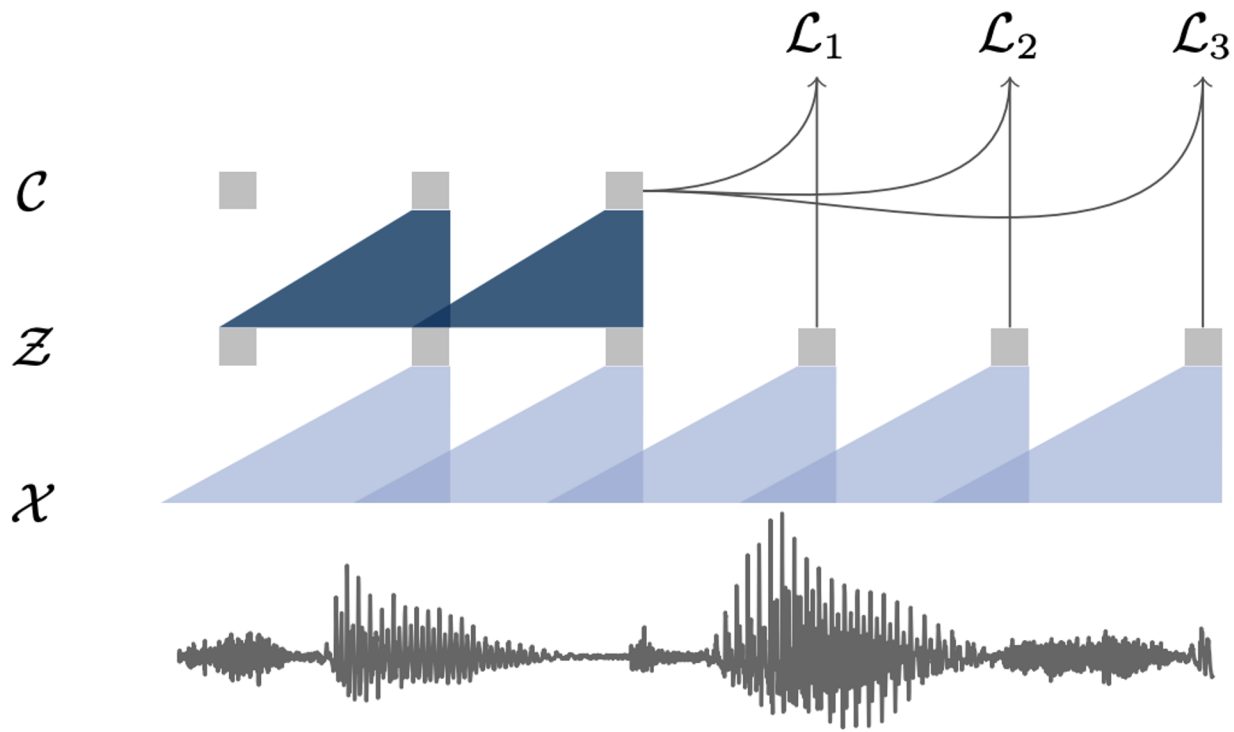


Figure 1: Illustration of pre-training from audio data $\mathcal{X}$ which is encoded with two convolutional neural networks that are stacked on top of each other. The model is optimized to solve a next time step prediction task.

Source: wav2vec: Unsupervised Pre-Training for Speech Recognition

Канал стажировок Яндекса

Спасибо за внимание

Доп. материалы

Y&Y