Announcements

- Challenge released.
- Quiz-4 releasing today. 24 hours to work on it.
- No laptops during class.

Last time

Practical scenarios where RNN is used

RNN Gradients

Attention

Types of attention

Transformers for language

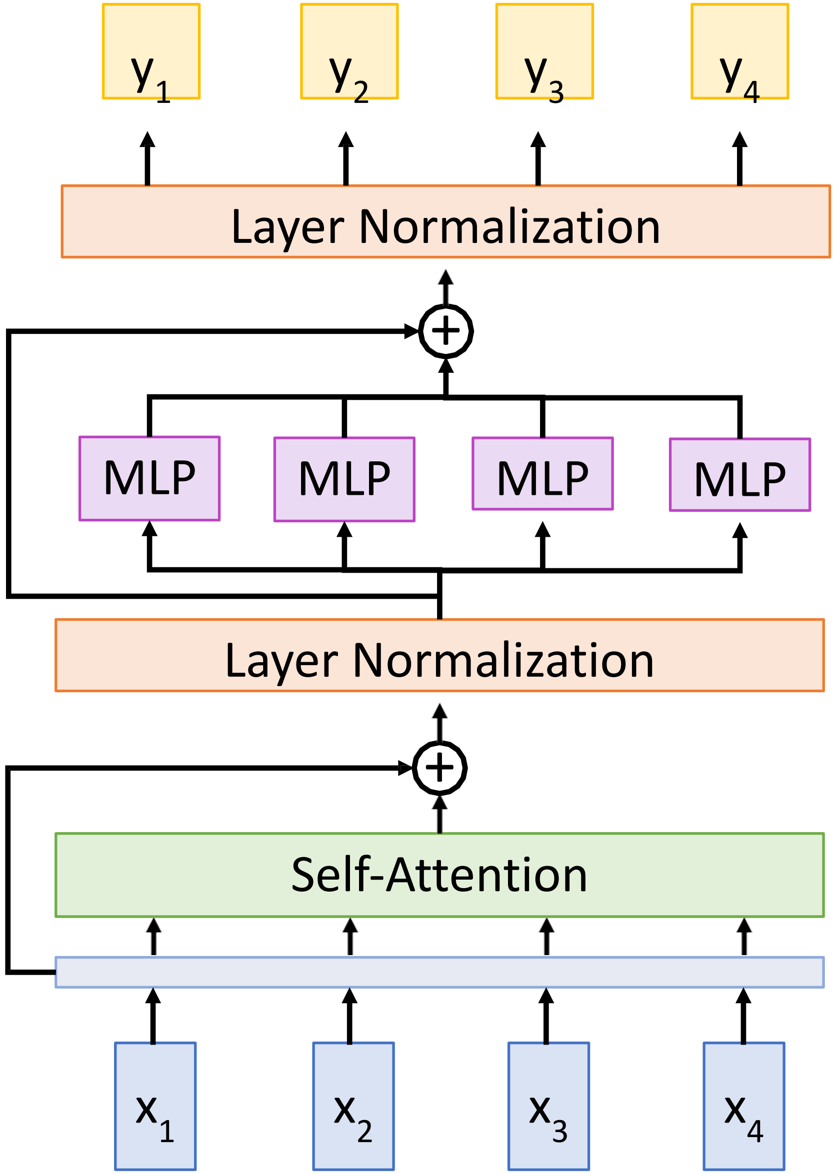
The Transformer

Residual connection

MLP independently
on each vector

Residual connection

All vectors interact
with each other



Vaswani et al, "Attention is all you need", NeurIPS 2017

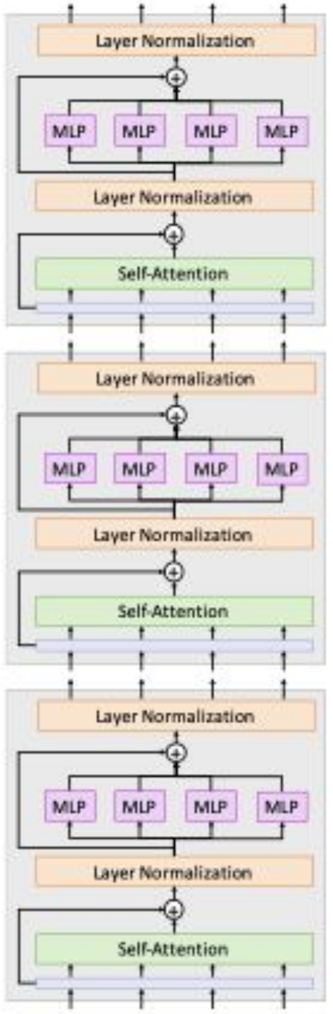
The Transformer: Transfer Learning

“ImageNet Moment for Natural Language Processing”

Pretraining:
Download a lot of text from the internet

Train a giant Transformer model for language modeling

Finetuning:
Fine-tune the Transformer on your own NLP task



Devlin et al, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", EMNLP 2018

Today

- A few questions from last class
- Scaling up of transformers
- Extending transformers to computer vision

Today

- **A few questions from last class**
- Scaling up of transformers
- Extending transformers to computer vision

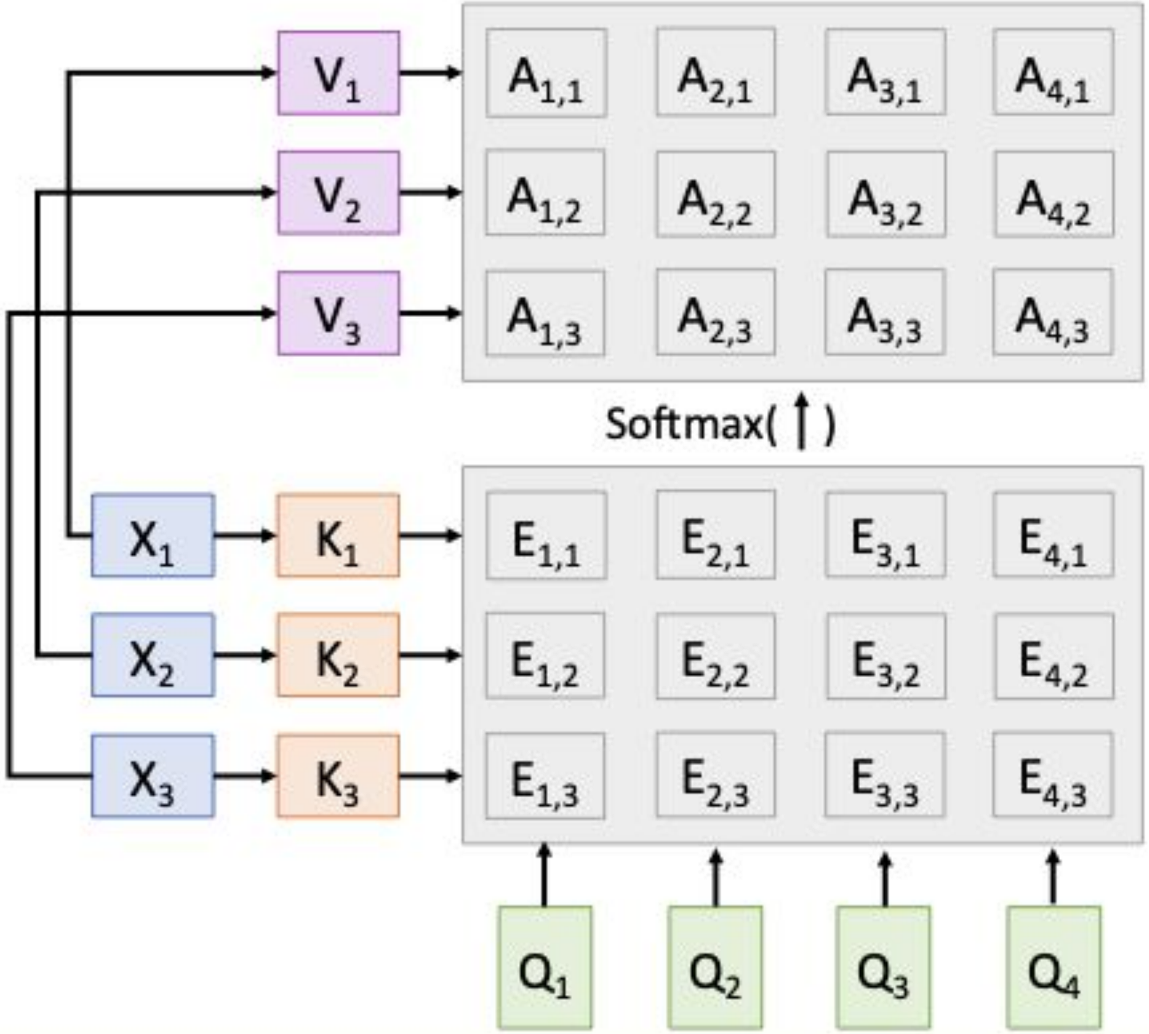
What is being learnt as Q, K, V?

Q = query
K = key
V = value

- **Q = query**
- **K = key**
- **V = value**

Attention Layer

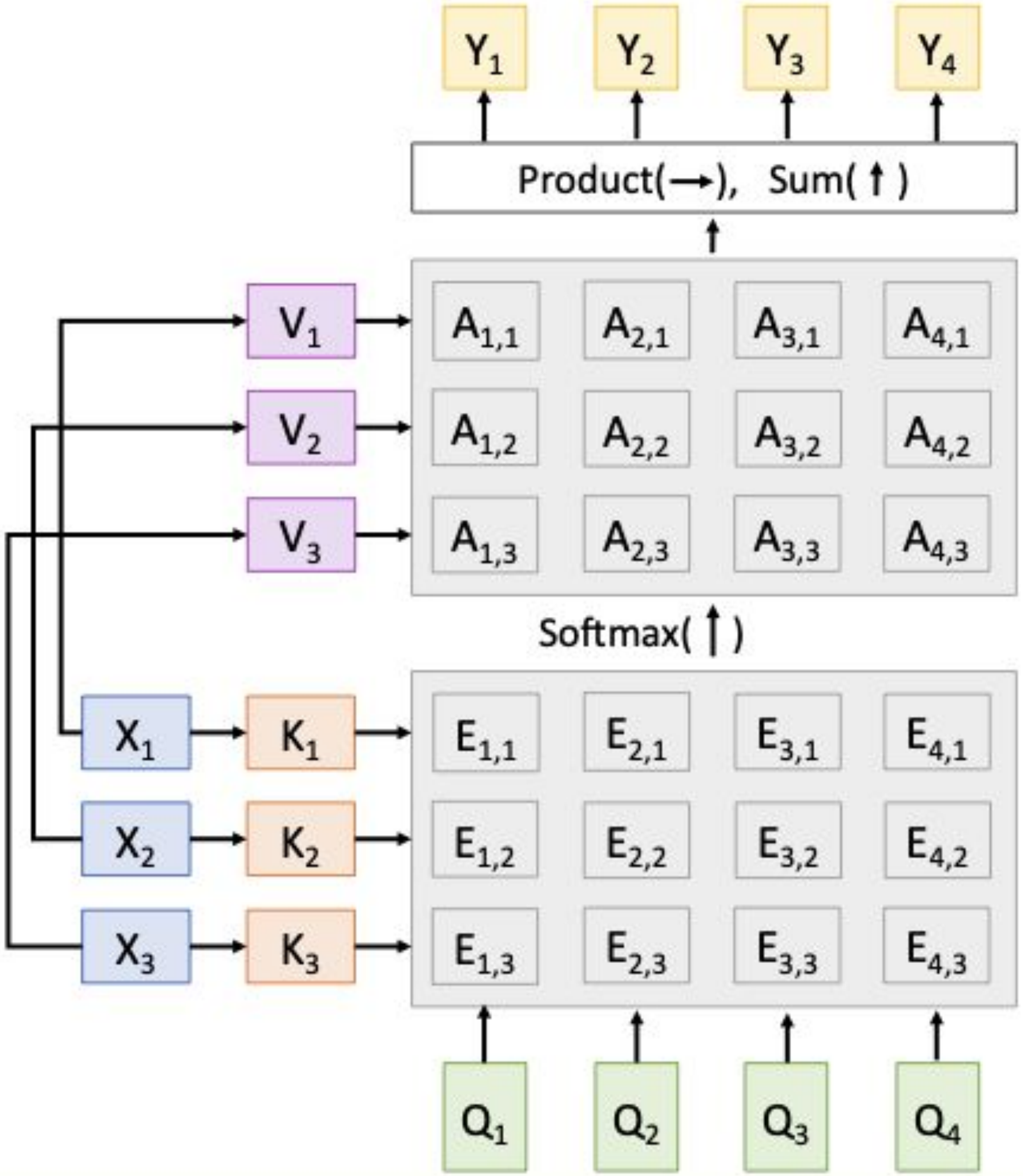
Inputs:
Query vectors: **Q** (Shape: N_Q x D_Q)
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: W_K (Shape: D_X x D_Q)
Value matrix: W_V (Shape: D_X x D_V)

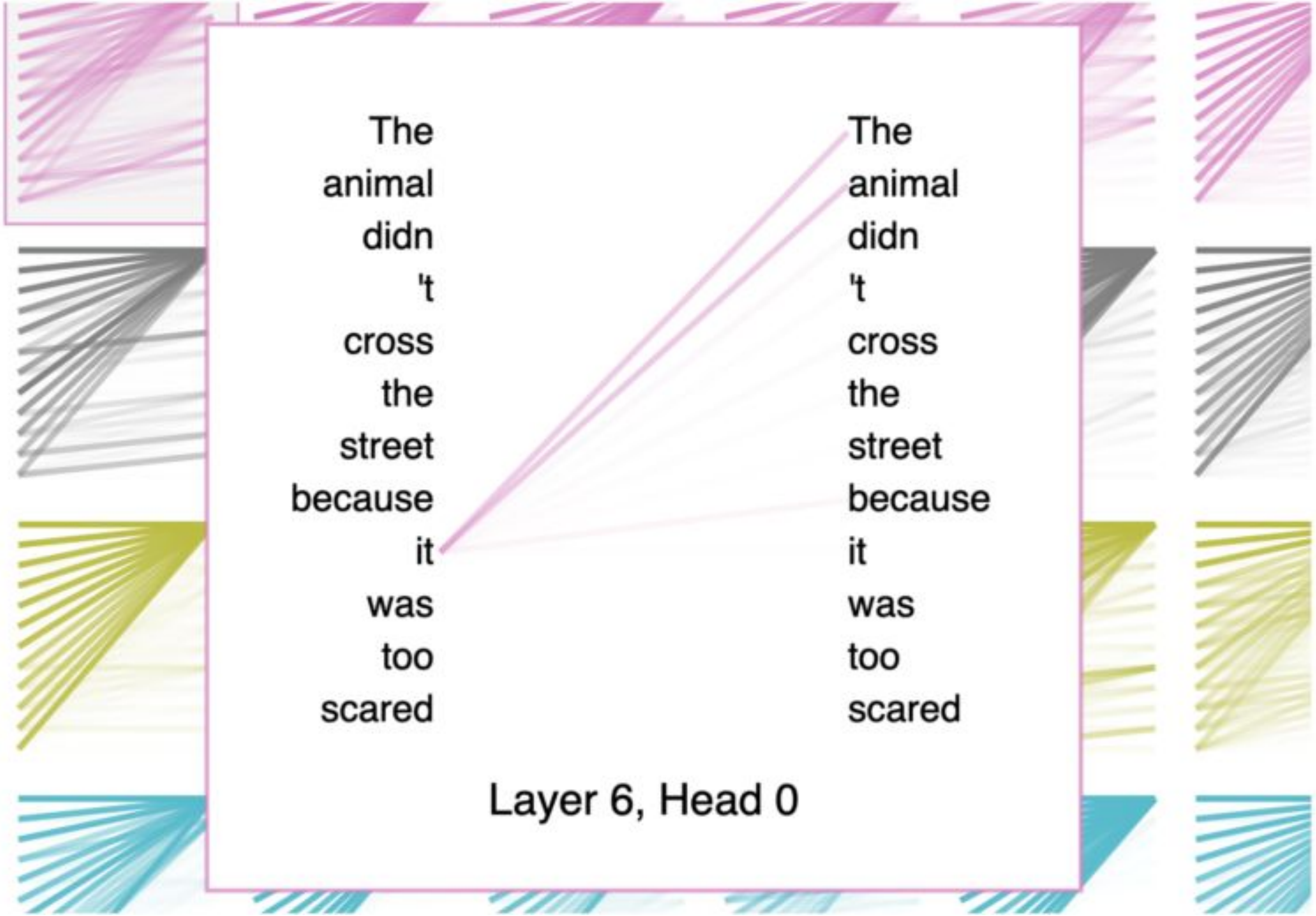


Attention Layer

Inputs:
Query vectors: **Q** (Shape: N_Q x D_Q)
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: **W_K** (Shape: D_X x D_Q)
Value matrix: **W_V** (Shape: D_X x D_V)

Computation:
Key vectors: **K** = **XW_K** (Shape: N_X x D_Q)
Value Vectors: **V** = **XW_V** (Shape: N_X x D_V)
Similarities: E = $QK^T / \sqrt{D_Q}$ (Shape: N_Q x N_X) $E_{i,j} = (Q_i \cdot K_j) / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) (Shape: N_Q x N_X)
Output vectors: Y = AV (Shape: N_Q x D_V) $Y_i = \sum_j A_{i,j} V_j$




The
animal
didn
't
cross
the
street
because
it
was
too
scared
The
animal
didn
't
cross
the
street
because
it
was
too
scared
Layer 6, Head 0

Parallelization in attention

Q = query
K = key
V = value

- Matrix multiplication is hardware friendlier than convolution
- For a fixed set of GFLOPs, transformers can be trained faster than Convolutional Networks

Parallelization in attention

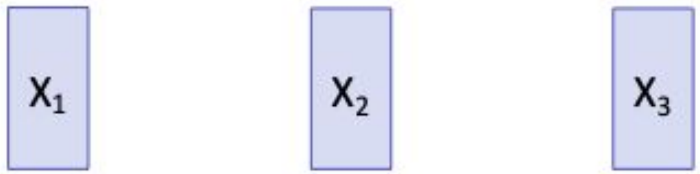
Q = query
K = key
V = value

Multihead Self-Attention

Inputs:
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: $\mathbf{W_K}$ (Shape: D_X x D_Q)
Value matrix: $\mathbf{W_V}$ (Shape: D_X x D_V)
Query matrix: $\mathbf{W_Q}$ (Shape: D_X x D_Q)

Computation:
Query vectors: **Q** = $\mathbf{XW_Q}$
Key vectors: **K** = $\mathbf{XW_K}$ (Shape: N_X x D_Q)
Value Vectors: **V** = $\mathbf{XW_V}$ (Shape: N_X x D_V)
Similarities: $E = \mathbf{QK}^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (\mathbf{Q_i} \cdot \mathbf{K_j}) / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) (Shape: N_X x N_X)
Output vectors: Y = A**V** (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j} \mathbf{V_j}$

Use H independent "Attention Heads" in parallel

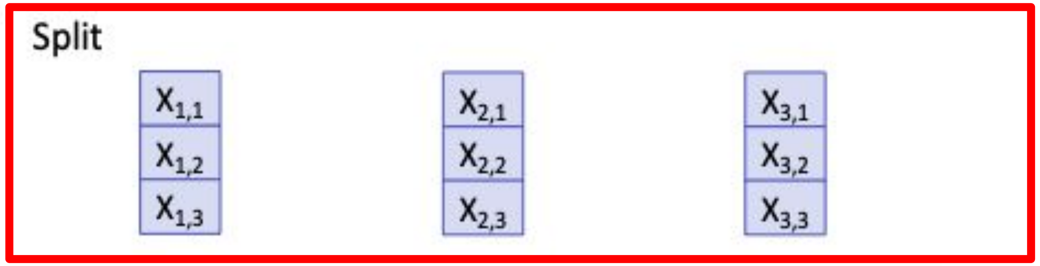


Multihead Self-Attention

Inputs:
Input vectors: $\mathbf{X}$ (Shape: N_X x D_X)
Key matrix: $\mathbf{W_K}$ (Shape: D_X x D_Q)
Value matrix: $\mathbf{W_V}$ (Shape: D_X x D_V)
Query matrix: $\mathbf{W_Q}$ (Shape: D_X x D_Q)

Use H independent "Attention Heads" in parallel

Computation:
Query vectors: $\mathbf{Q} = \mathbf{XW_Q}$
Key vectors: $\mathbf{K} = \mathbf{XW_K}$ (Shape: N_X x D_Q)
Value Vectors: $\mathbf{V} = \mathbf{XW_V}$ (Shape: N_X x D_V)
Similarities: $E = \mathbf{QK}^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (\mathbf{Q_i} \cdot \mathbf{K_j}) / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) (Shape: N_X x N_X)
Output vectors: $Y = A\mathbf{V}$ (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j}\mathbf{V_j}$

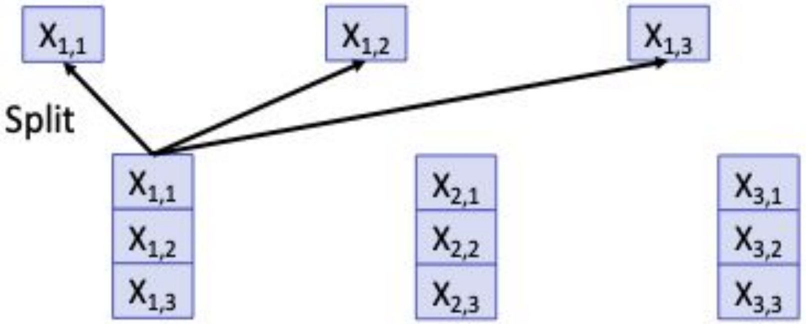


Multihead Self-Attention

Inputs:
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: $\mathbf{W_K}$ (Shape: D_X x D_Q)
Value matrix: $\mathbf{W_V}$ (Shape: D_X x D_V)
Query matrix: $\mathbf{W_Q}$ (Shape: D_X x D_Q)

Computation:
Query vectors: **Q** = $\mathbf{XW_Q}$
Key vectors: **K** = $\mathbf{XW_K}$ (Shape: N_X x D_Q)
Value Vectors: **V** = $\mathbf{XW_V}$ (Shape: N_X x D_V)
Similarities: E = $\mathbf{QK}^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (\mathbf{Q_i} \cdot \mathbf{K_j}) / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) (Shape: N_X x N_X)
Output vectors: Y = A**V** (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j} \mathbf{V_j}$

Use H independent "Attention Heads" in parallel

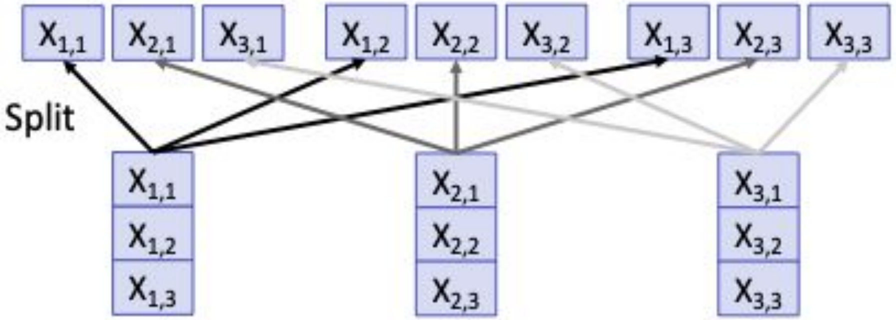


Multihead Self-Attention

Inputs:
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: W_K (Shape: D_X x D_Q)
Value matrix: W_V (Shape: D_X x D_V)
Query matrix: W_Q (Shape: D_X x D_Q)

Use H independent "Attention Heads" in parallel

Computation:
Query vectors: $Q = XW_Q$
Key vectors: $K = XW_K$ (Shape: N_X x D_Q)
Value Vectors: $V = XW_V$ (Shape: N_X x D_V)
Similarities: $E = QK^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (Q_i \cdot K_j) / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) (Shape: N_X x N_X)
Output vectors: $Y = AV$ (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j} V_j$



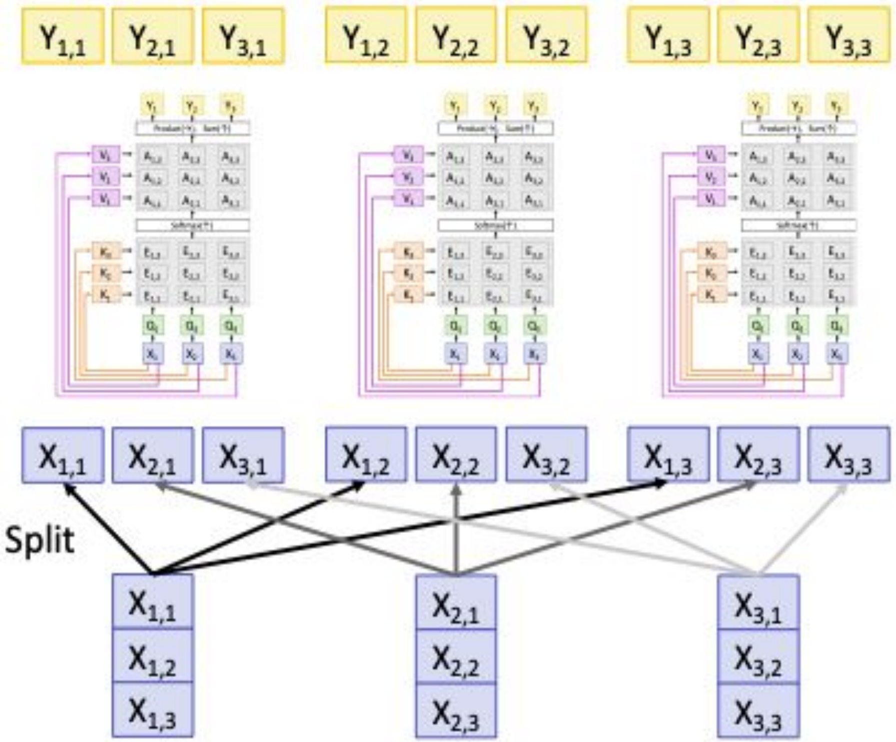
Multihead Self-Attention

Run self-attention in parallel on each set of input vectors (different weights per head)

Inputs:
Input vectors: **X** (Shape: N_X x D_X)
Key matrix: W_K (Shape: D_X x D_Q)
Value matrix: W_V (Shape: D_X x D_V)
Query matrix: W_Q (Shape: D_X x D_Q)

Computation:
Query vectors: $Q = XW_Q$
Key vectors: $K = XW_K$ (Shape: N_X x D_Q)
Value Vectors: $V = XW_V$ (Shape: N_X x D_V)
Similarities: $E = QK^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (Q_i \cdot K_j) / \sqrt{D_Q}$
Attention weights: $A = \text{softmax}(E, \text{dim}=1)$ (Shape: N_X x N_X)
Output vectors: $Y = AV$ (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j} V_j$

Use H independent "Attention Heads" in parallel

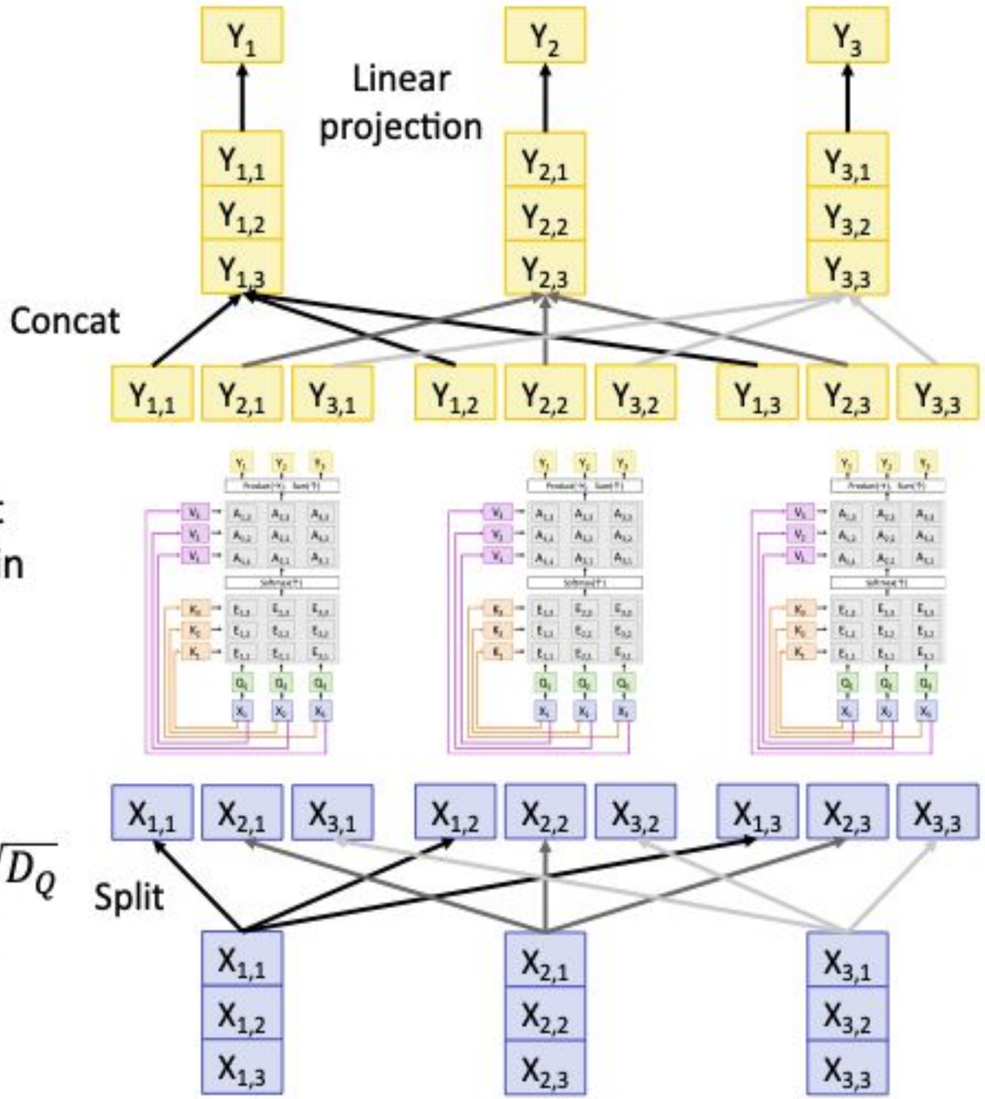


Multihead Self-Attention

Inputs:
Input vectors: $\mathbf{X}$ (Shape: N_X x D_X)
Key matrix: $\mathbf{W_K}$ (Shape: D_X x D_Q)
Value matrix: $\mathbf{W_V}$ (Shape: D_X x D_V)
Query matrix: $\mathbf{W_Q}$ (Shape: D_X x D_Q)

Computation:
Query vectors: $\mathbf{Q} = \mathbf{XW_Q}$
Key vectors: $\mathbf{K} = \mathbf{XW_K}$ (Shape: N_X x D_Q)
Value Vectors: $\mathbf{V} = \mathbf{XW_V}$ (Shape: N_X x D_V)
Similarities: $E = \mathbf{QK}^T / \sqrt{D_Q}$ (Shape: N_X x N_X) $E_{i,j} = (\mathbf{Q_i} \cdot \mathbf{K_j}) / \sqrt{D_Q}$
Attention weights: $A = \text{softmax}(E, \text{dim}=1)$ (Shape: N_X x N_X)
Output vectors: $Y = A\mathbf{V}$ (Shape: N_X x D_V) $Y_i = \sum_j A_{i,j}\mathbf{V_j}$

Use H independent "Attention Heads" in parallel



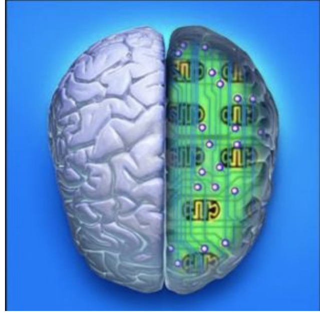
Summary so far

- GPUs favor matrix multiplications over convolutions
- Multi-head attention allows parallel processing of tokens

Today

- A few questions from last class
- **Scaling up of transformers**
- Extending transformers to computer vision

What helped us get here?



✅
- CNNs
- Transformers

✅
- Corpus of webdata
- Weak and self supervision

Slide credit: Justin Johnson

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer-Base	12	512	8	65M		8x P100 (12 hours)
Transformer-Large	12	1024	16	213M		8x P100 (3.5 days)
BERT-Base	12	768	12	110M	13 GB	
BERT-Large	24	1024	16	340M	13 GB	
XLNet-Large	24	1024	16	~340M	126 GB	512x TPU-v3 (2.5 days)
RoBERTa	24	1024	16	355M	160 GB	1024x V100 GPU (1 day)

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer-Base	12	512	8	65M		8x P100 (12 hours)
Transformer-Large	12	1024	16	213M		8x P100 (3.5 days)
BERT-Base	12	768	12	110M	13 GB	
BERT-Large	24	1024	16	340M	13 GB	
XLNet-Large	24	1024	16	~340M	126 GB	512x TPU-v3 (2.5 days)
RoBERTa	24	1024	16	355M	160 GB	1024x V100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40 GB	
Megatron-LM	72	3072	32	8.3B	174 GB	512x V100 GPU (9 days)
Turing-NLG	78	4256	28	17B	?	256x V100 GPU
GPT-3	96	12,288	96	175B	694GB	?
Gopher	80	16,384	128	280B	10.55 TB	4096x TPUv3 (38 days)

Rae et al, "Scaling Language Models: Methods, Analysis, & Insights from Training Gopher", arXiv 2021

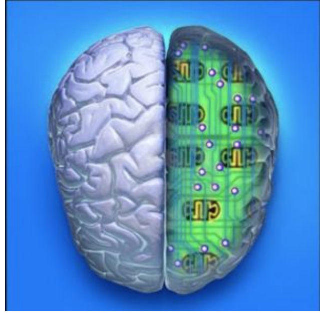
Scaling up Transformers

$3,768,320 on Google Cloud (eval price)

Model	Layers	Width	Heads	Params	Data	Training
Transformer-Base	12	512	8	65M		8x P100 (12 hours)
Transformer-Large	12	1024	16	213M		8x P100 (3.5 days)
BERT-Base	12	768	12	110M	13 GB	
BERT-Large	24	1024	16	340M	13 GB	
XLNet-Large	24	1024	16	~340M	126 GB	512x TPU-v3 (2.5 days)
RoBERTa	24	1024	16	355M	160 GB	1024x V100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40 GB	
Megatron-LM	72	3072	32	8.3B	174 GB	512x V100 GPU (9 days)
Turing-NLG	78	4256	28	17B	?	256x V100 GPU
GPT-3	96	12,288	96	175B	694GB	?
Gopher	80	16,384	128	280B	10.55 TB	4096x TPUv3 (38 days)

Rae et al, "Scaling Language Models: Methods, Analysis, & Insights from Training Gopher", arXiv 2021

Summary: what helped us get here?



✅
- CNNs
- Transformers

✅
- Corpus of webdata
- Weak and self supervision

✅
- A big reason for AI advancements

Slide credit: Justin Johnson

Examples of how generic these models are

GPT-3: Programming by prompt

Input / Output examples
Test example
Italics: Completion by GPT-3

Poor English input: I eated the purple berries.
Good English output: I ate the purple berries.
Poor English input: Thank you for picking me as your designer. I'd appreciate it.
Good English output: Thank you for choosing me as your designer. I appreciate it.
Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
Good English output: The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.
Poor English input: I'd be more than happy to work with you in another project.
Good English output:

Brown et al, "Language Models are Few-Shot Learners", arXiv 2020

GPT-3: Programming by prompt

Input / Output examples
Test example
Italics: Completion by GPT-3

Poor English input: I eated the purple berries.
Good English output: I ate the purple berries.
Poor English input: Thank you for picking me as your designer. I'd appreciate it.
Good English output: Thank you for choosing me as your designer. I appreciate it.
Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
Good English output: The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.
Poor English input: I'd be more than happy to work with you in another project.
Good English output: *I'd be more than happy to work with you on another project.*

Brown et al, "Language Models are Few-Shot Learners", arXiv 2020

https://www.gwern.net/GPT-3#successes

GPT-3: Literary Style Transfer

Topic: humor: literary parodies and homages, pastiches, style parodies.
Parodies of the fantasy novel series *Harry Potter* in the style of various famous authors:
By Ernest Hemingway:

https://www.gwern.net/GPT-3#successes

GPT-3: Literary Style Transfer

Topic: humor: literary parodies and homages, pastiches, style parodies.
Parodies of the fantasy novel series *Harry Potter* in the style of various famous authors:
By Ernest Hemingway:
"It was a cold day on Privet Drive. A child cried. Harry felt nothing. He was dryer than dust. He had been silent too long. He had not felt love. He had scarcely felt hate. Yet the Dementor's Kiss killed nothing. Death didn't leave him less dead than he had been a second before. It wasn't about living or dying really. It wasn't about death. It was about the Dark Lord, his creator, his engenderer. He was not resurrected. He was created again. He was whole again in the courtyard of Malfoy Manor."

https://www.gwern.net/GPT-3#successes

GPT-3: Literary Style Transfer

Topic: humor: literary parodies and homages, pastiches, style parodies.
Parodies of the fantasy novel series *Harry Potter* in the style of various famous authors:
By Ernest Hemingway:
"It was a cold day on Privet Drive. A child cried. Harry felt nothing. He was dryer than dust. He had been silent too long. He had not felt love. He had scarcely felt hate. Yet the Dementor's Kiss killed nothing. Death didn't leave him less dead than he had been a second before. It wasn't about living or dying really. It wasn't about death. It was about the Dark Lord, his creator, his engenderer. He was not resurrected. He was created again. He was whole again in the courtyard of Malfoy Manor."

By Jane Austen:
"It is a truth universally acknowledged, that a broken Harry is in want of a book. This he knows to his cost; pressing that wretched nose against the window of a bookstore in Diagon Alley the last thing before he goes into hiding for several years, where upon, he goes straight to Azkaban."

https://www.gwern.net/GPT-3#successes

GPT-3: Literary Style Transfer

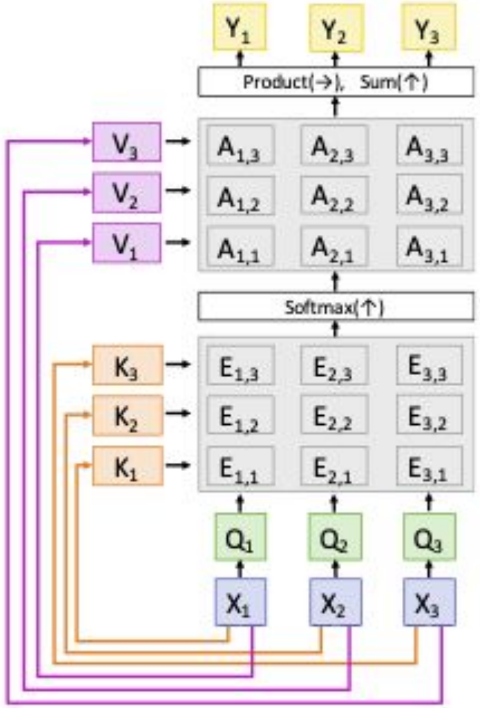
Topic: humor: literary parodies and homages, pastiches, style parodies.
Parodies of the fantasy novel series *Harry Potter* in the style of various famous authors:
By Ernest Hemingway:
"It was a cold day on Privet Drive. A child cried. Harry felt nothing. He was dryer than dust. He had been silent too long. He had not felt love. He had scarcely felt hate. Yet the Dementor's Kiss killed nothing. Death didn't leave him less dead than he had been a second before. It wasn't about living or dying really. It wasn't about death. It was about the Dark Lord, his creator, his engenderer. He was not resurrected. He was created again. He was whole again in the courtyard of Malfoy Manor."
By Jane Austen:
"It is a truth universally acknowledged, that a broken Harry is in want of a book. This he knows to his cost; pressing that wretched nose against the window of a bookstore in Diagon Alley the last thing before he goes into hiding for several years, where upon, he goes straight to Azkaban."
By Arthur Conan Doyle:
"Harry pushed at the swinging doors of the bookshop hard, and nearly knocked himself unconscious. He staggered in with his ungainly package, his cheeks scarlet with cold and the shame of having chosen the wrong month to go Christmas shopping. The proprietor of the store, however, didn't cast even a cursory glance at him, being far more interested in an enormous hunk of wood lying in the middle of the floor, which certainly looked like a gallows. Yes, the proprietor said to a reedy old man wearing a bowler hat and a forlorn expression that made Harry want to kick him, I can rent you such a gallows for a small fee."

Summary

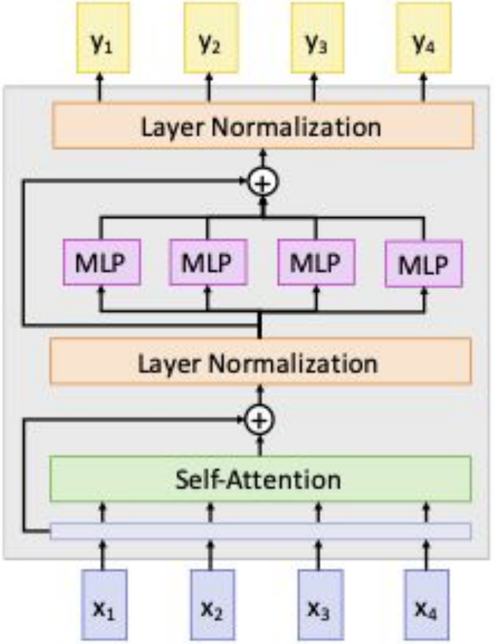
Adding **Attention** to RNN models lets them look at different parts of the input at each timestep

A dog is standing on a hardwood floor.

Generalized **Self-Attention** is new, powerful neural network primitive



Transformers are a new neural network model that only uses attention



Xu et al, "Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

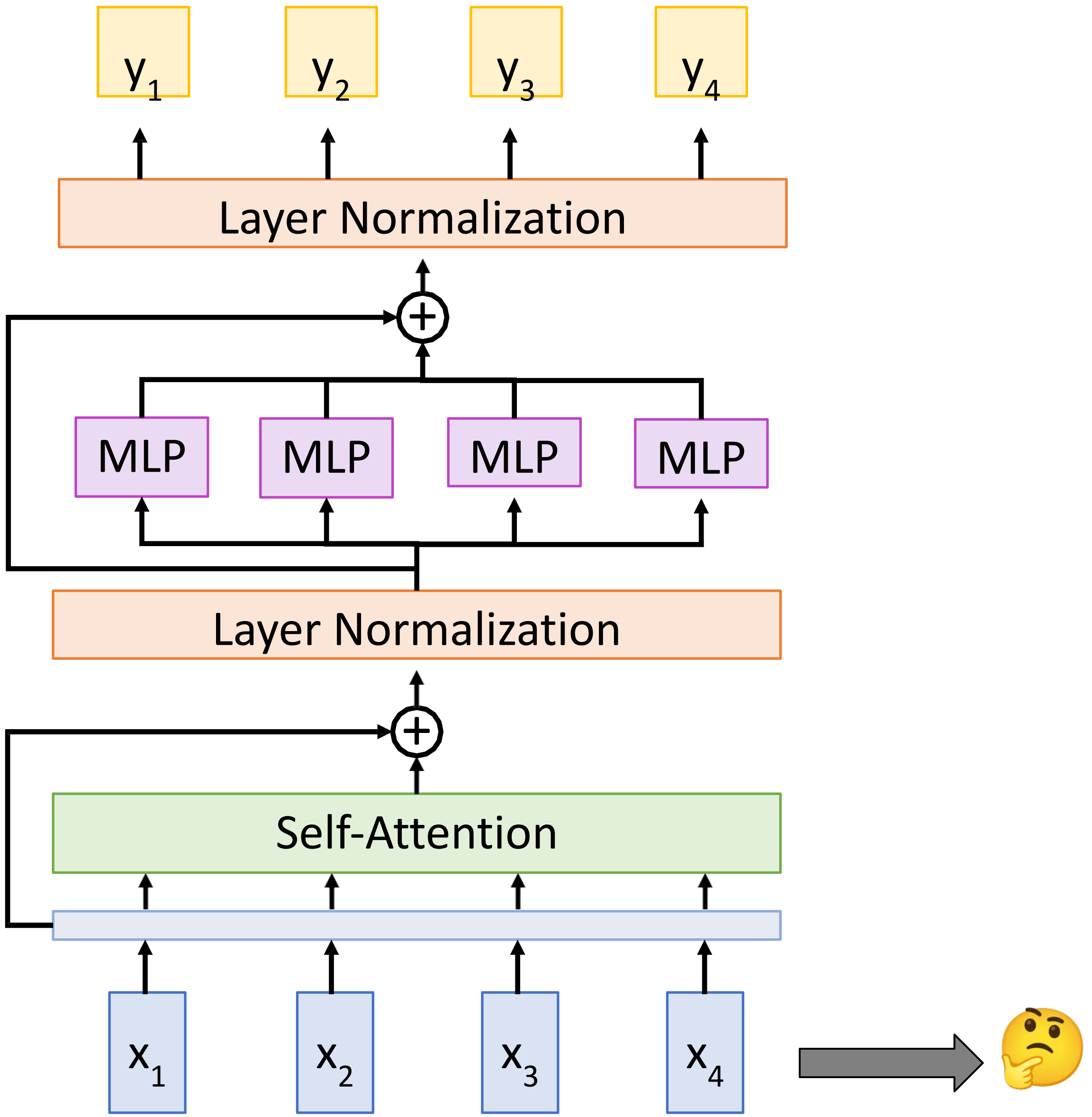
Words or characters as input

Residual connection

MLP independently on each vector

Residual connection

All vectors interact with each other

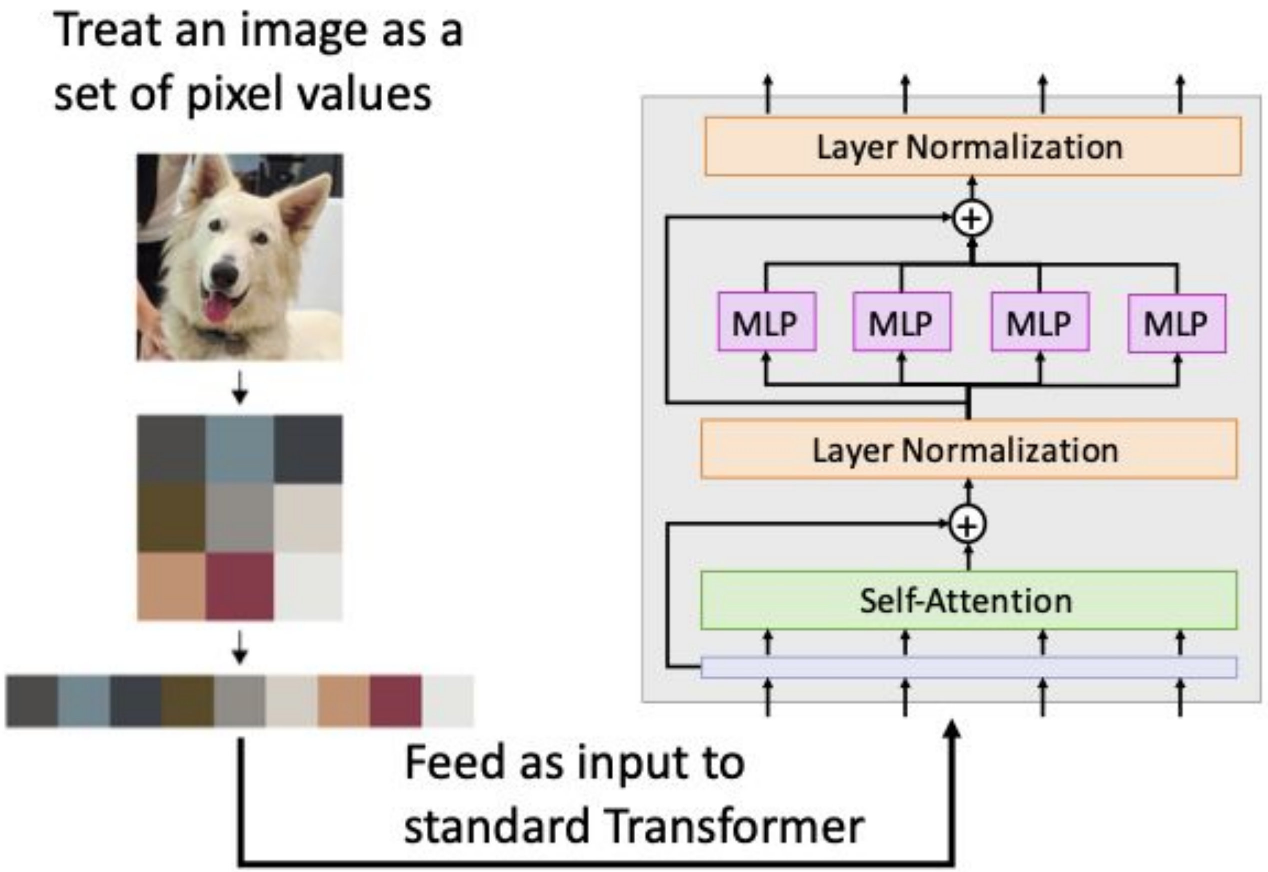


Today

- A few questions from last class
- Scaling up of transformers
- **Extending transformers to computer vision**

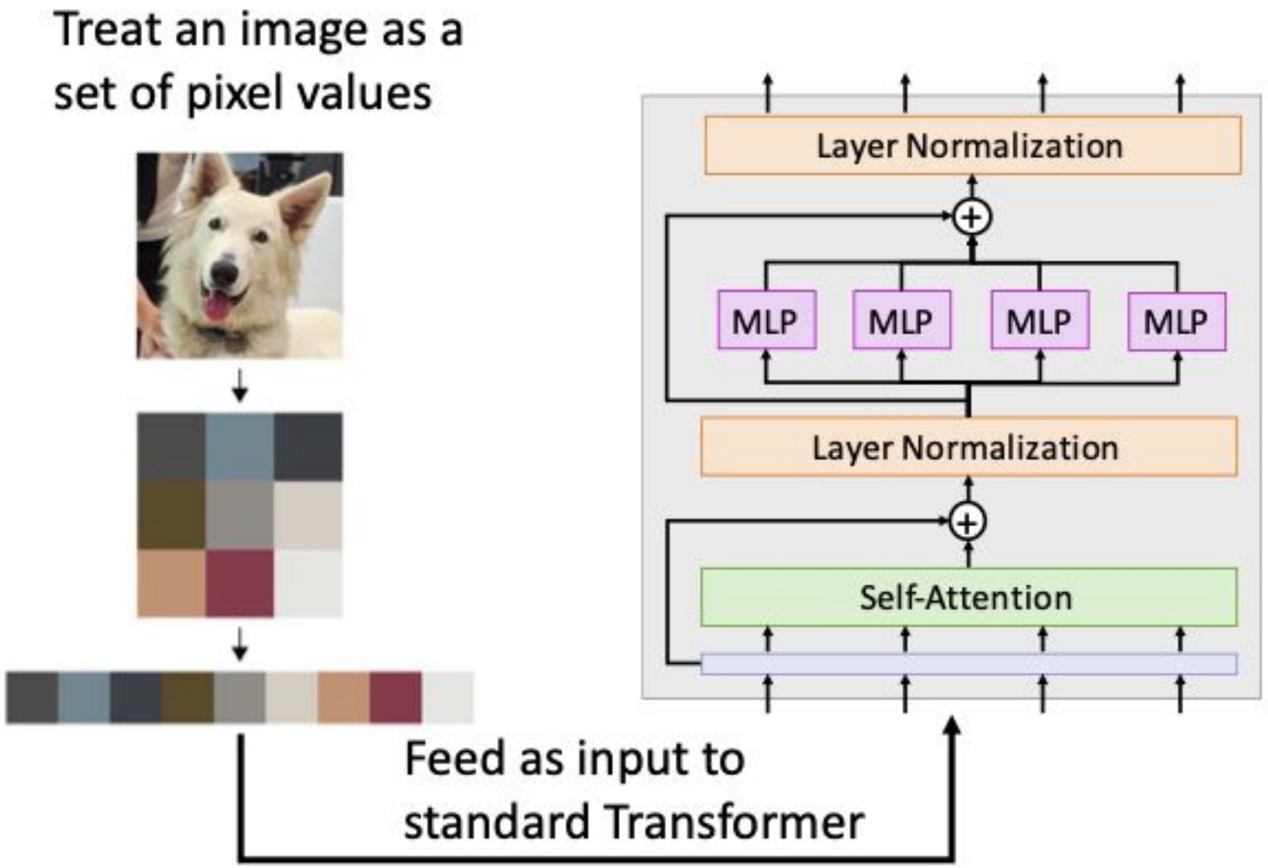
How do we extend transformers to images?

One option: Pixels as tokens



Chen et al, "Generative Pretraining from Pixels", ICML 2020

One option: Pixels as tokens



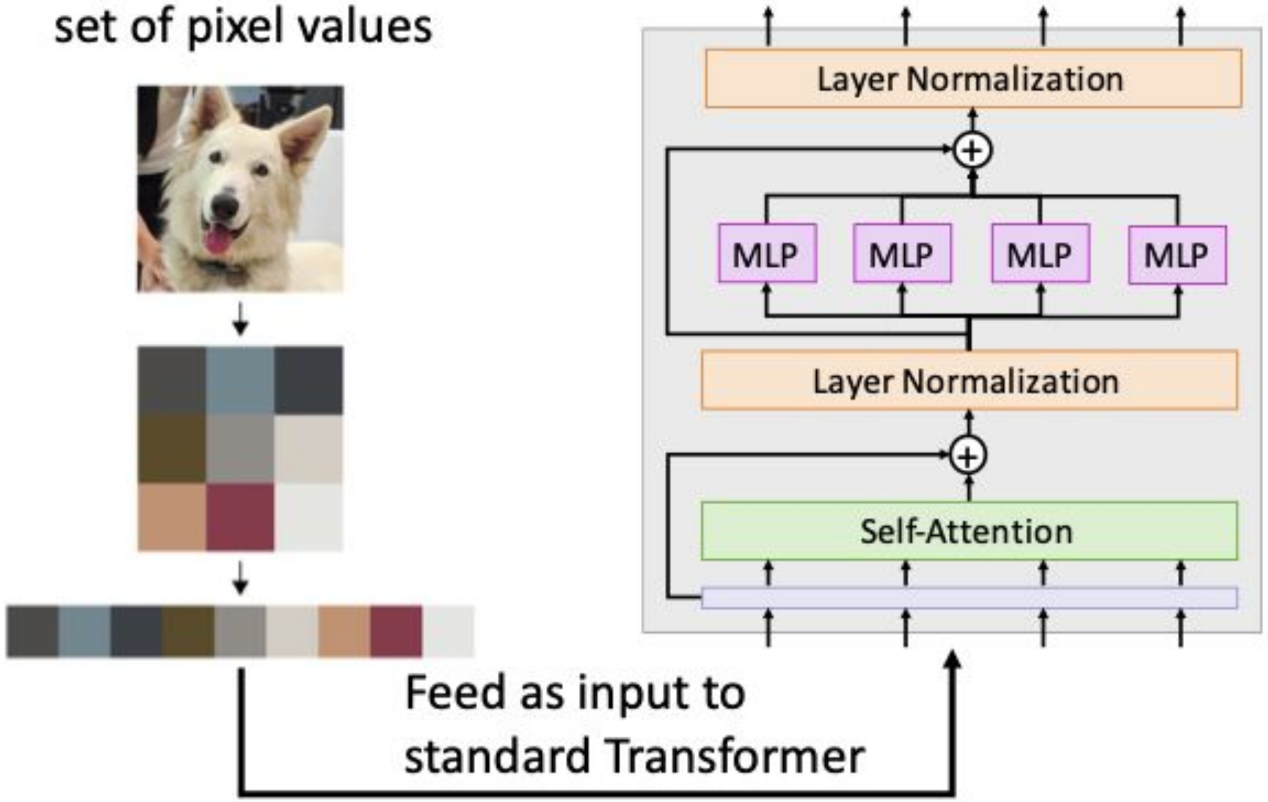
Problem: Memory use!

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

Chen et al, "Generative Pretraining from Pixels", ICML 2020

One option: Pixels as tokens

Treat an image as a set of pixel values



Feed as input to standard Transformer

Problem: Memory use!

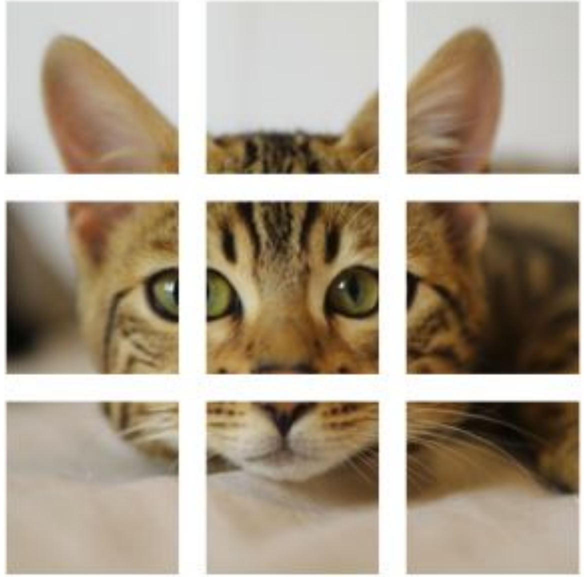
R x R image needs R^4 elements per attention matrix

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

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

Second option: Patches as tokens



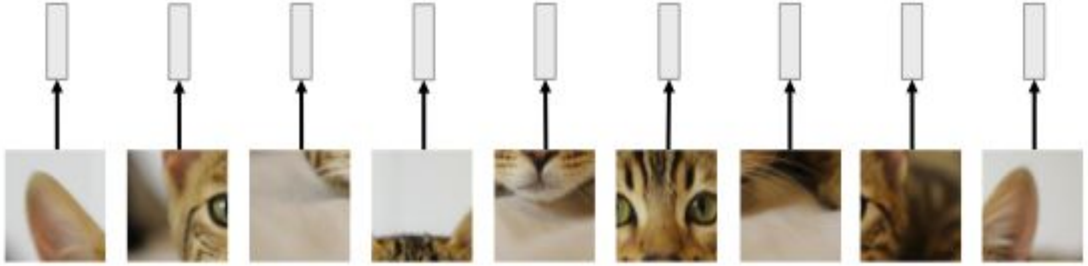
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

N input patches, each
of shape 3x16x16

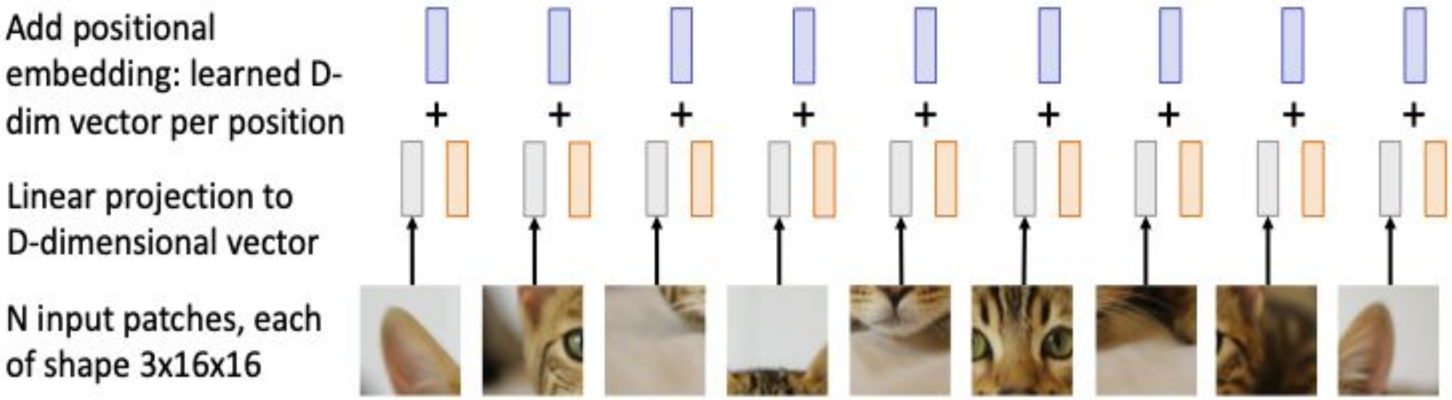
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Linear projection to
D-dimensional vector

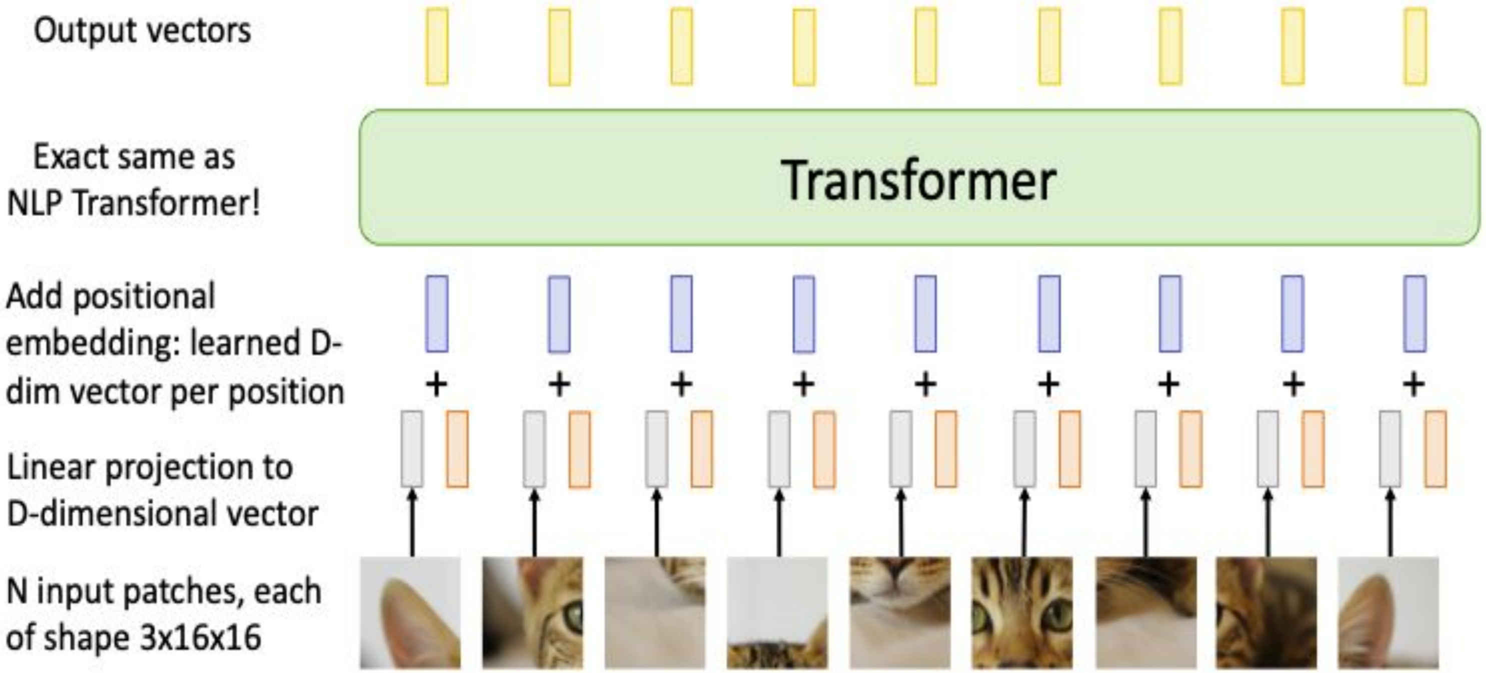
N input patches, each
of shape 3x16x16



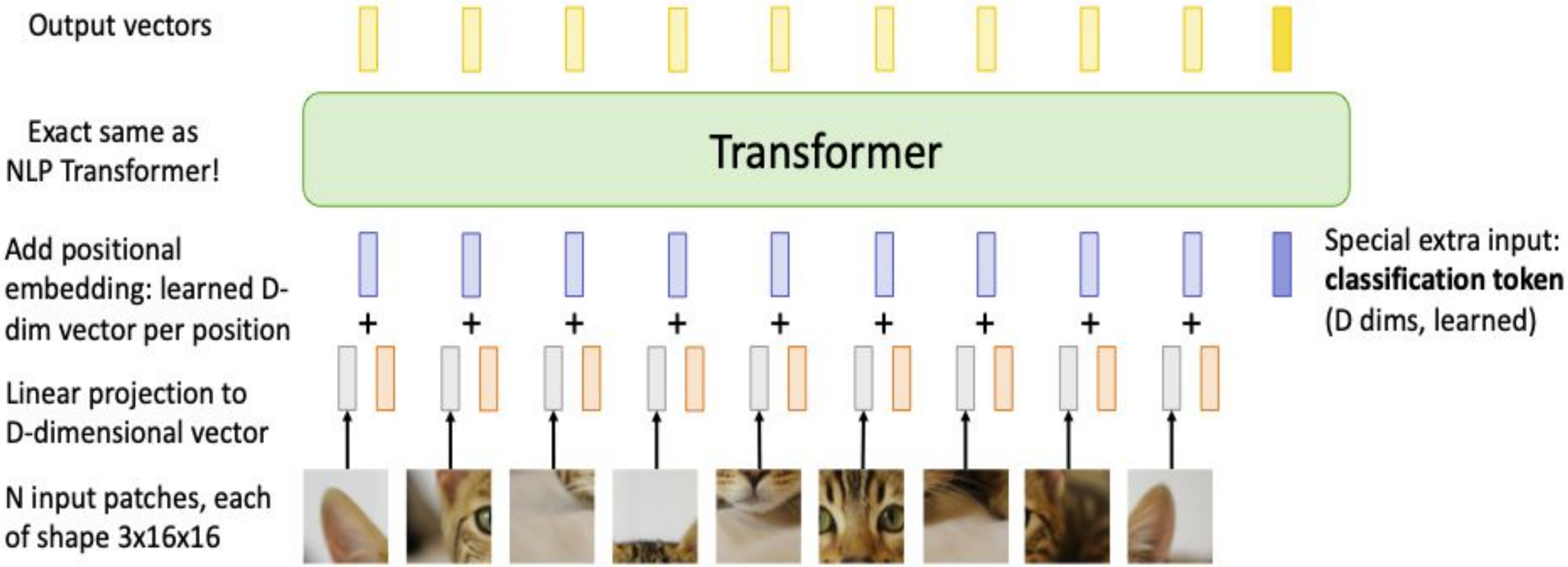
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021



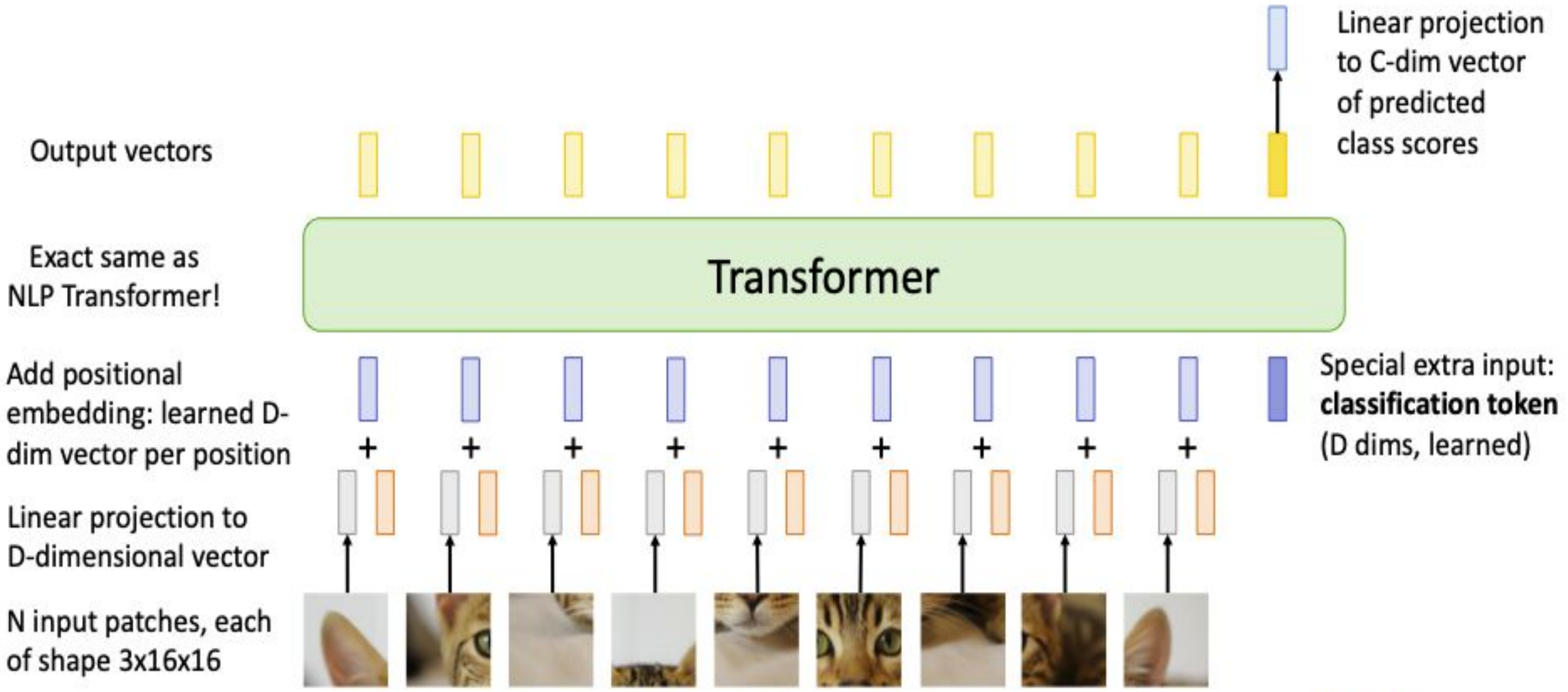
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021



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

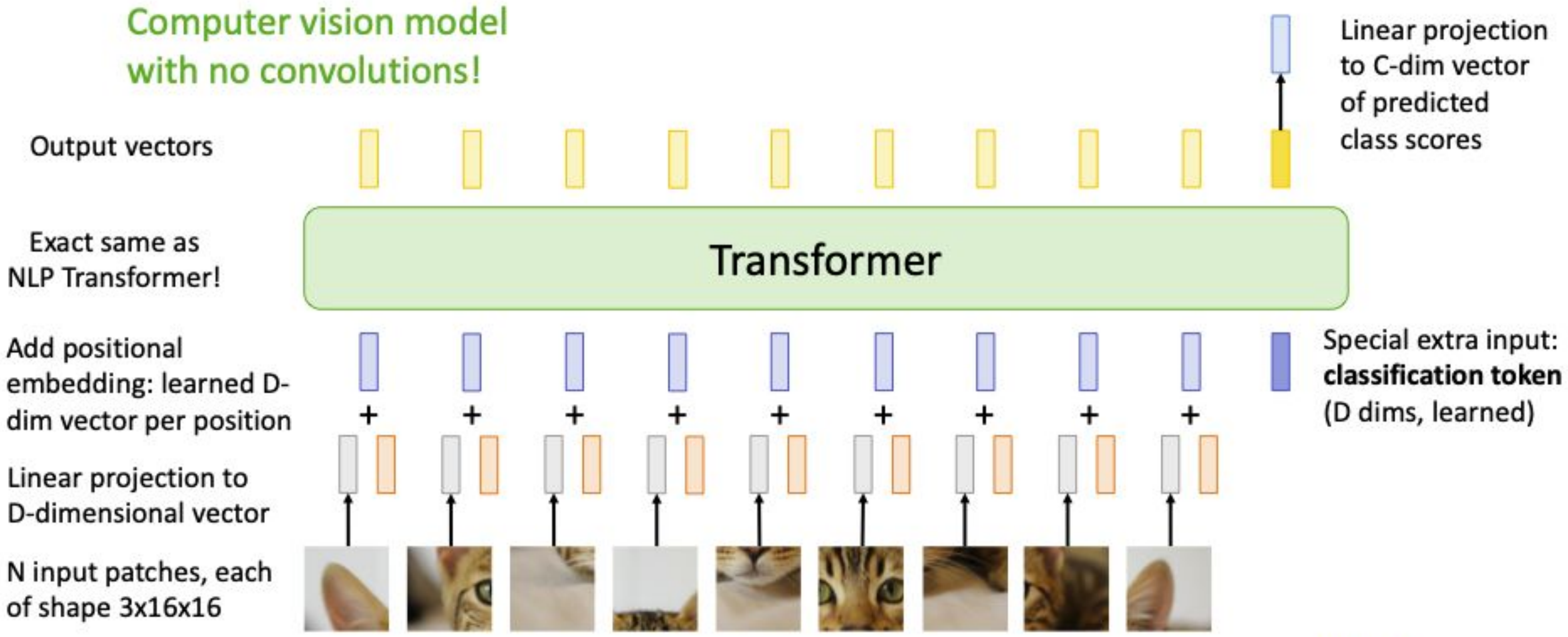


Dosovitskiy et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", ICLR 2021



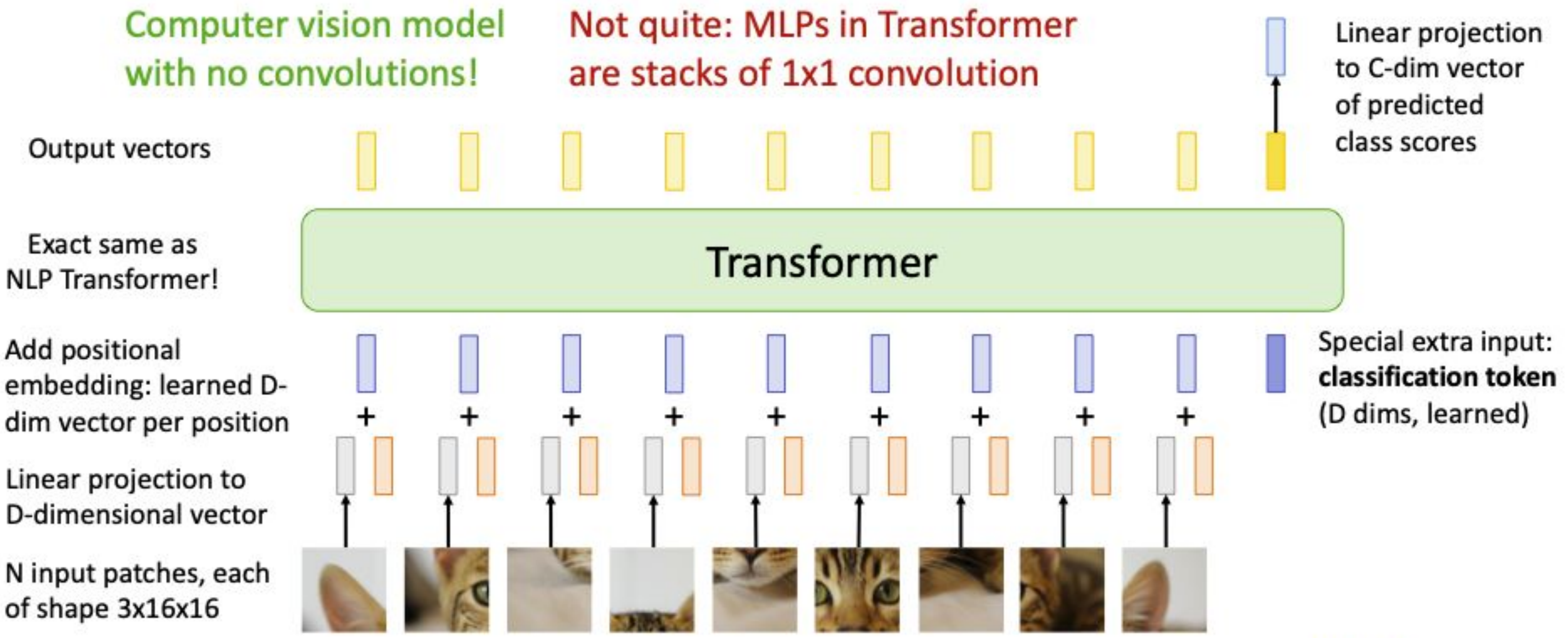
Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Vision Transformer (ViT)



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

Vision Transformer (ViT)

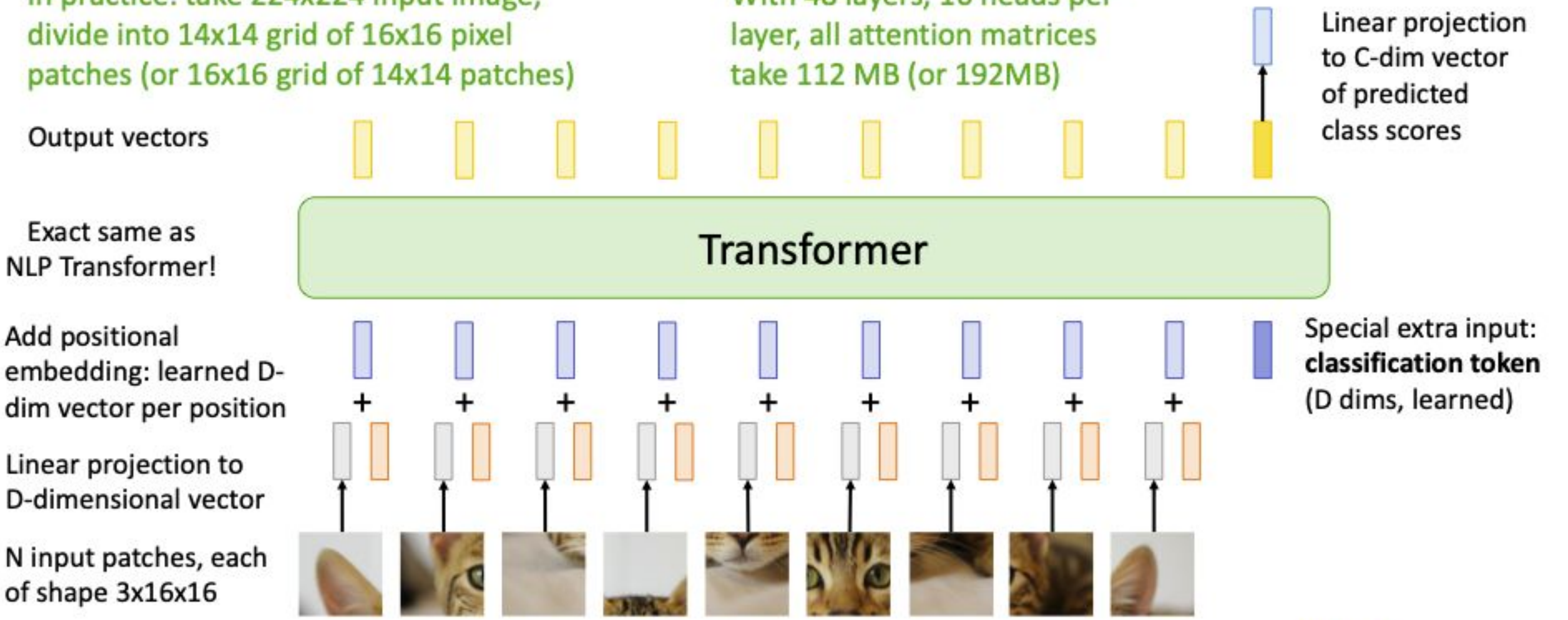


Dosovitskiy et al, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR 2021

Vision Transformer (ViT)

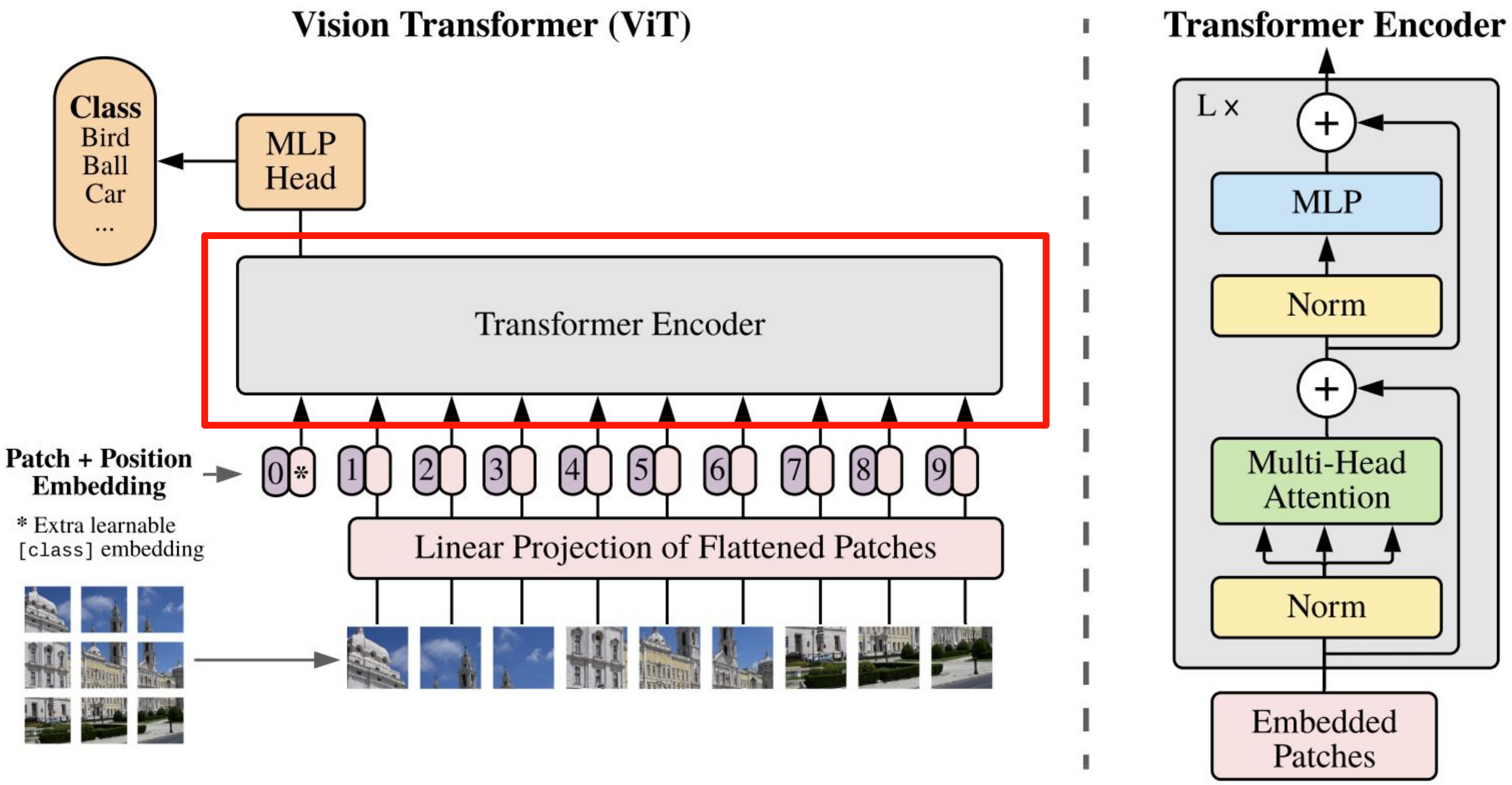
In practice: take 224x224 input image, divide into 14x14 grid of 16x16 pixel patches (or 16x16 grid of 14x14 patches)

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

Putting this all together…

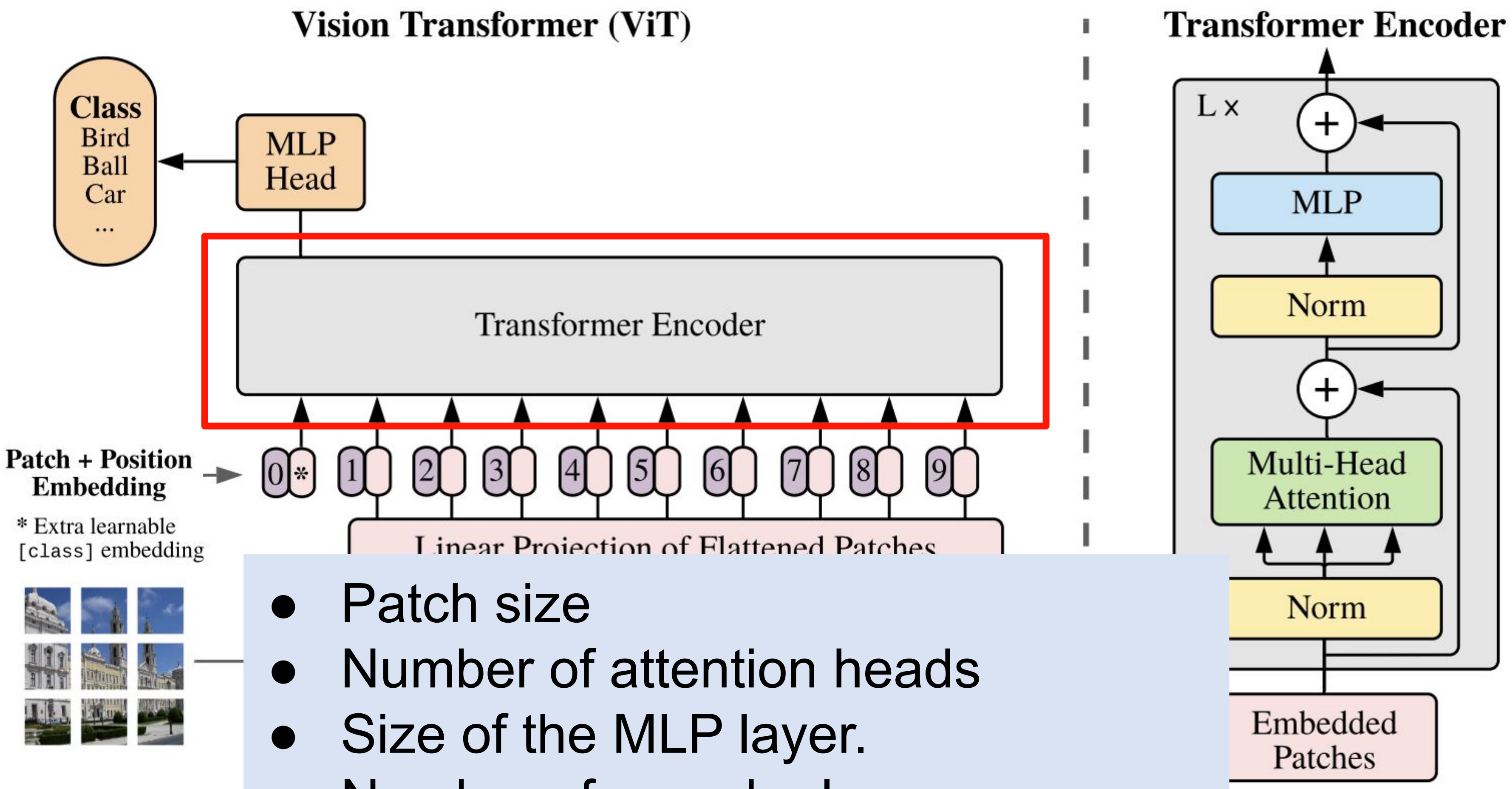


AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE, ICLR'21

Do not edit
How to change the design

What are some of the parameter choices involved in designing a ViT?

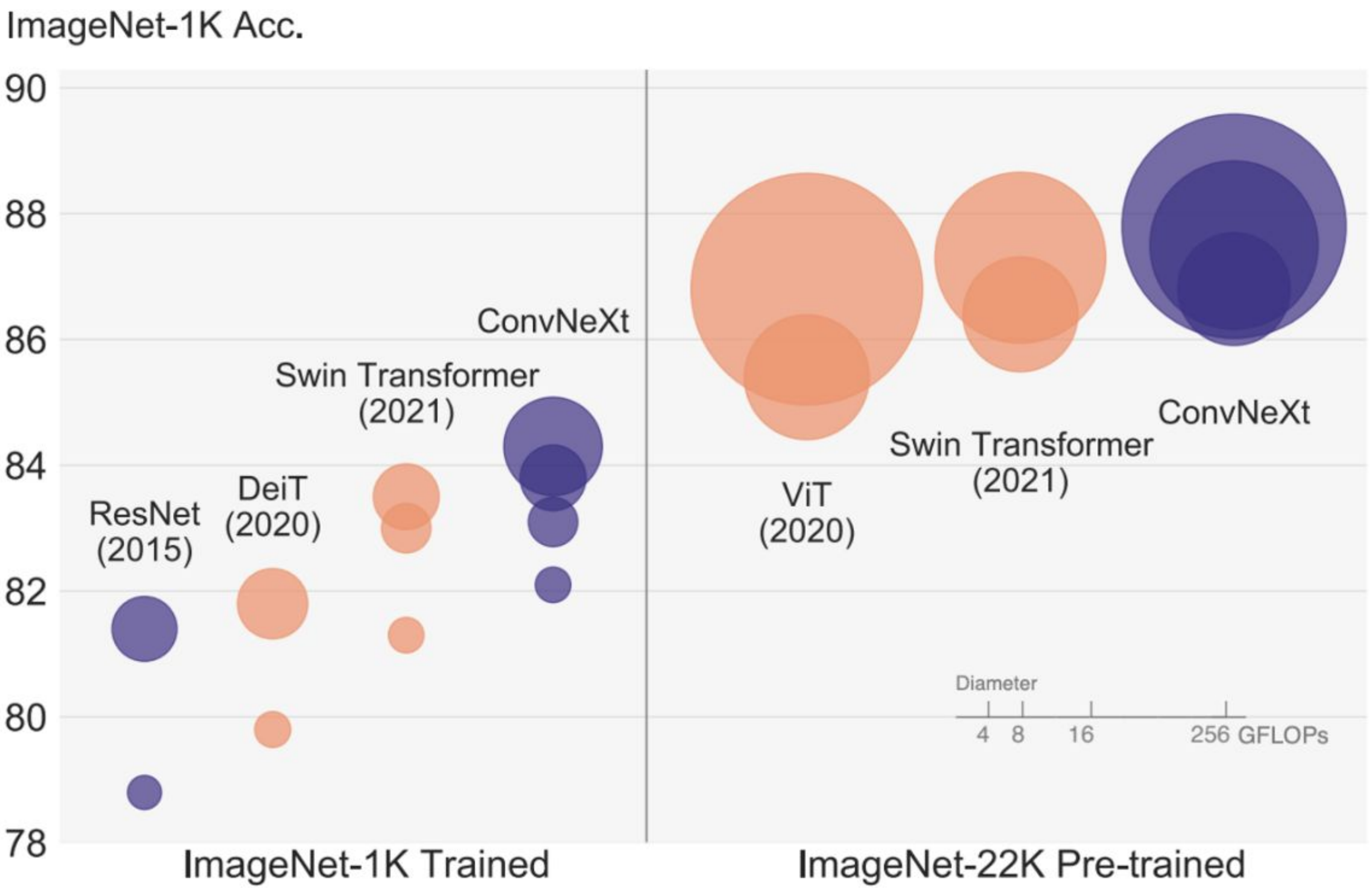
Putting this all together…



- Patch size
- Number of attention heads
- Size of the MLP layer.
- Number of encoder layers
- …

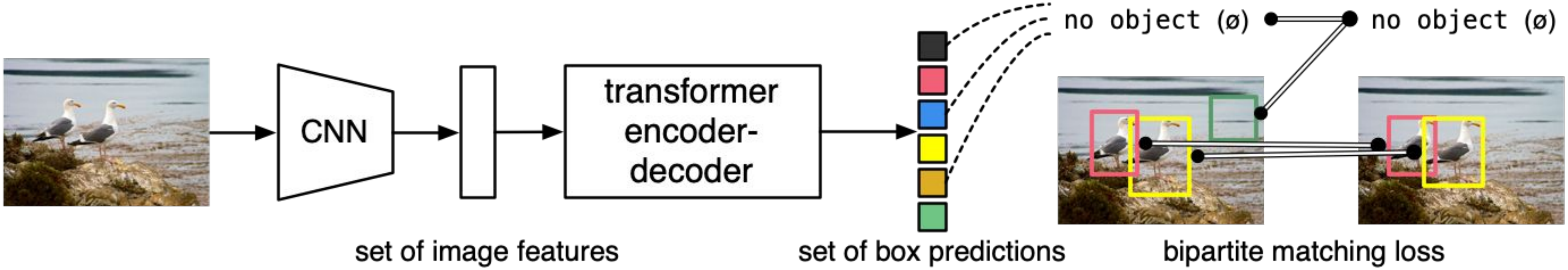
AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE, ICLR'21

Convolution ⚔️ Transformers



https://github.com/facebookresearch/ConvNeXt

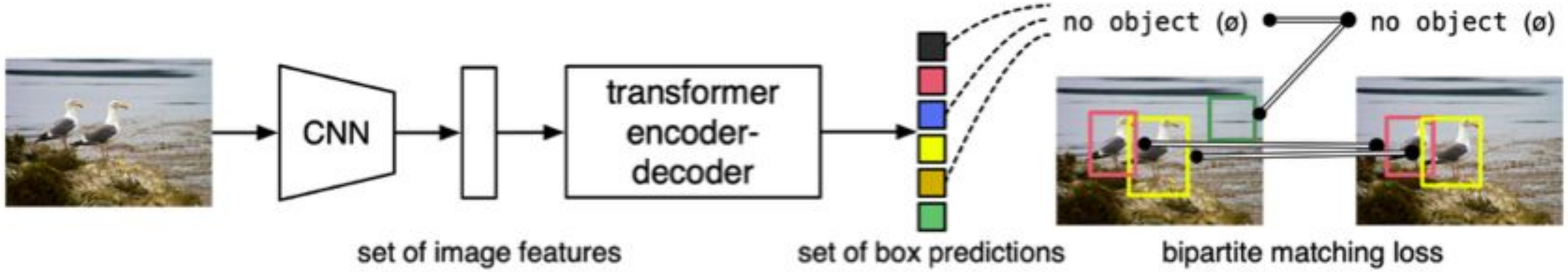
Convolution 🤝 Transformers

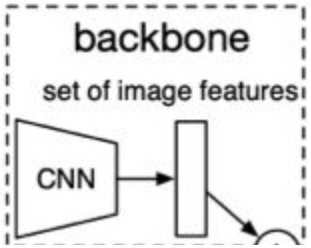


https://github.com/facebookresearch/detr

Convolution 🤝 Transformers

Object Detection with Transformers: DETR

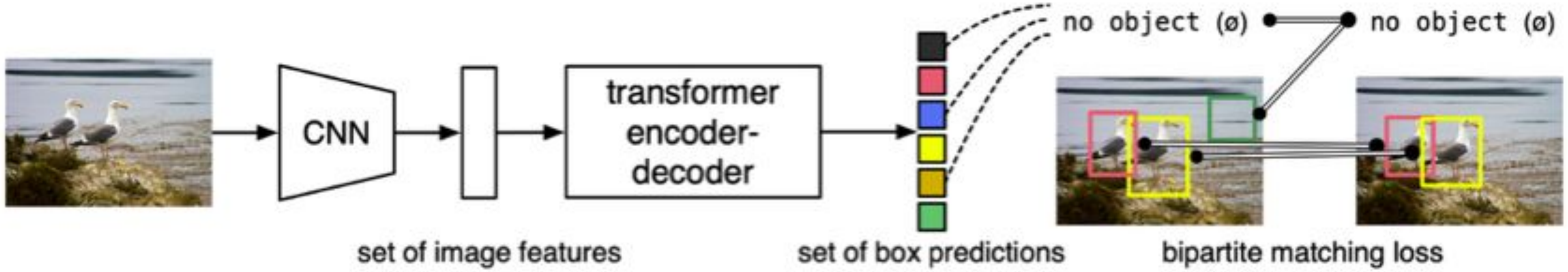


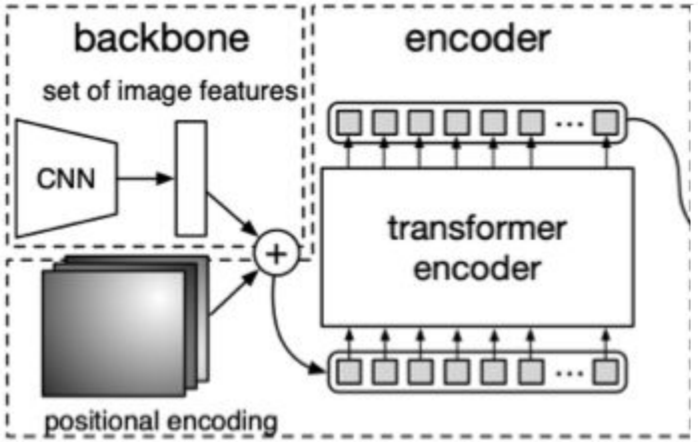


Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020

Convolution 🤝 Transformers

Object Detection with Transformers: DETR

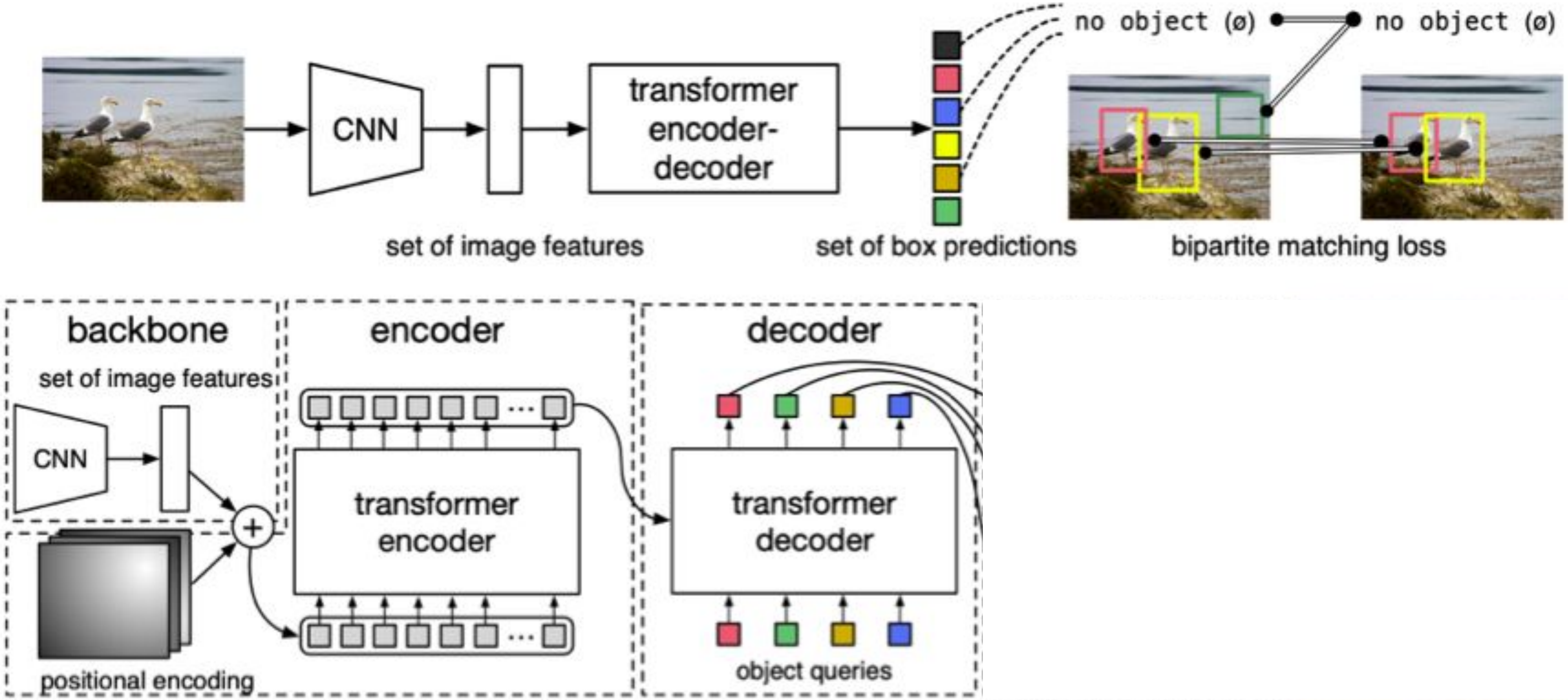




Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020

Convolution 🤝 Transformers

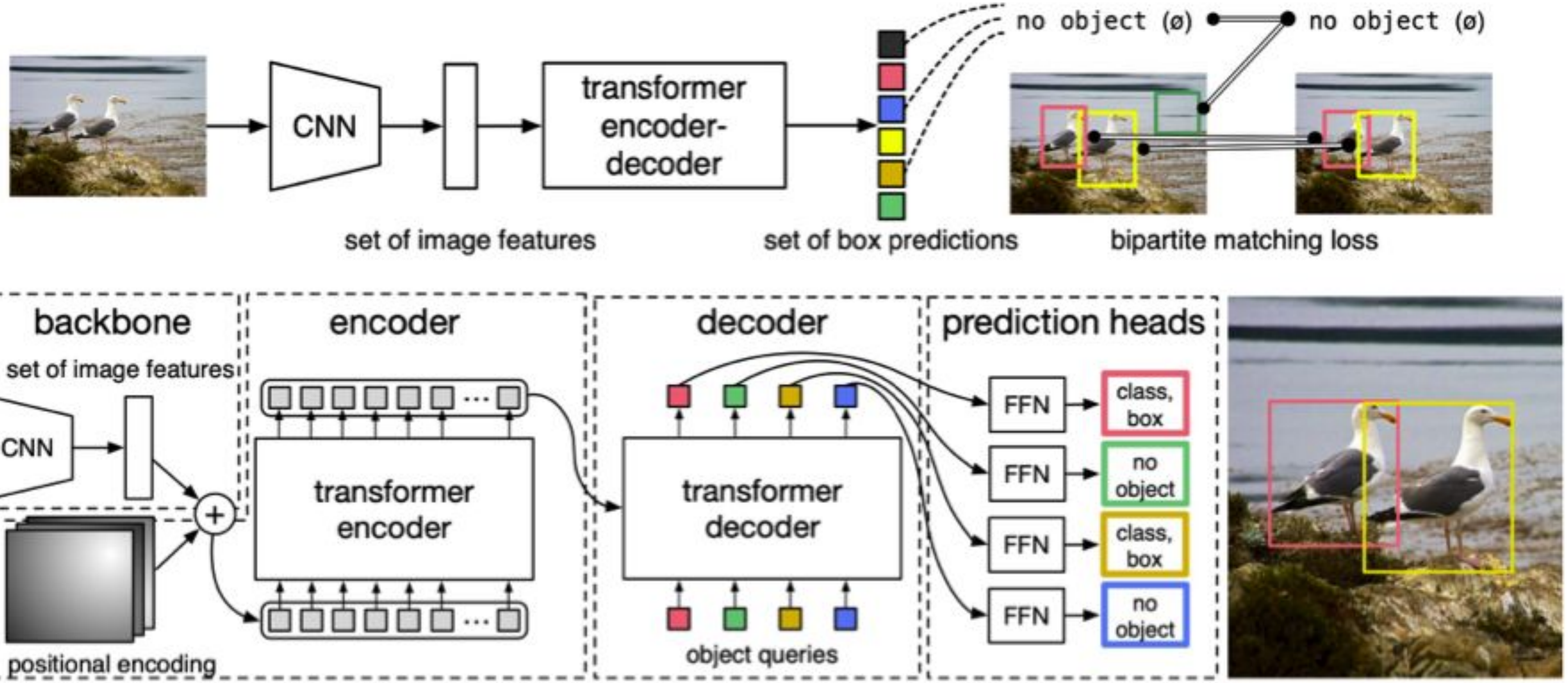
Object Detection with Transformers: DETR



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

Convolution 🤝 Transformers

Object Detection with Transformers: DETR



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

Transformer architectures are now the basic backbone architectures for most vision, audio, and language **generative** and **discriminative** models.

- Transformer architectures are now the basic backbone architectures for most vision, audio, and language **generative** and **discriminative** models.

- Allows interactions of multiple modalities through cross-attention.
    - Eg: In text to image generative models,
        - Cross-attention
    - Key = Value = image tokens
    - Query = text prompt (text tokens)