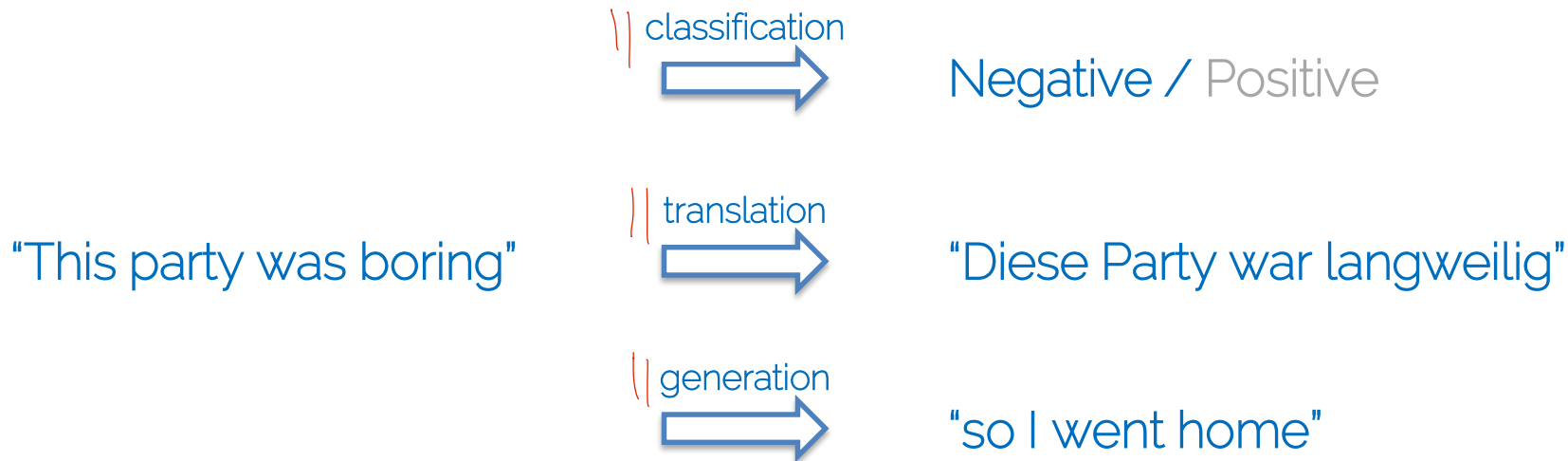


Sequence Models

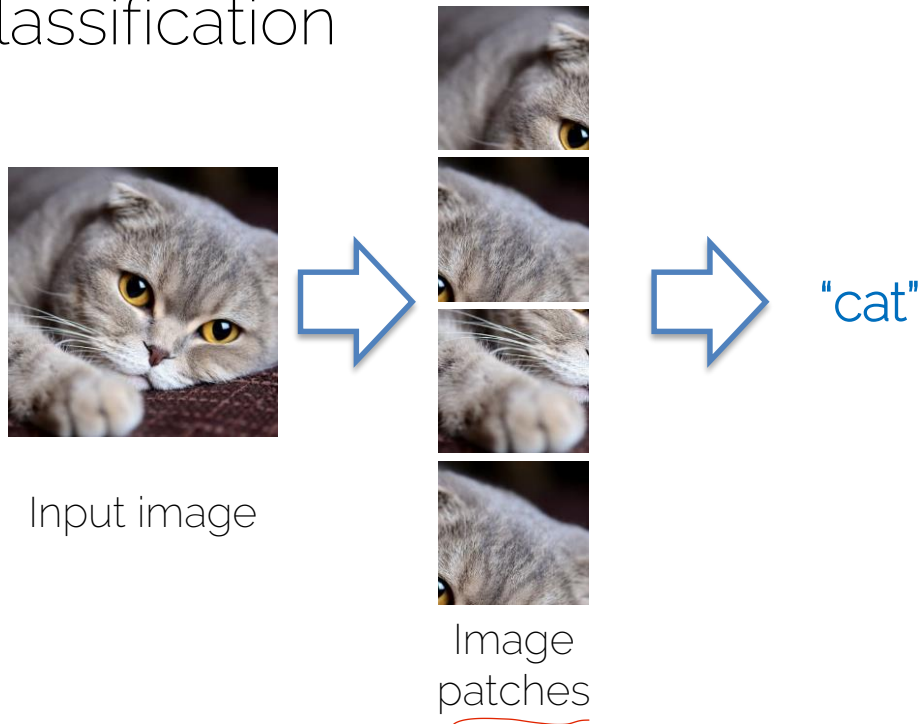
Sequence Modelling

- Texts as sequences
 - Sequences are natural representations for text data



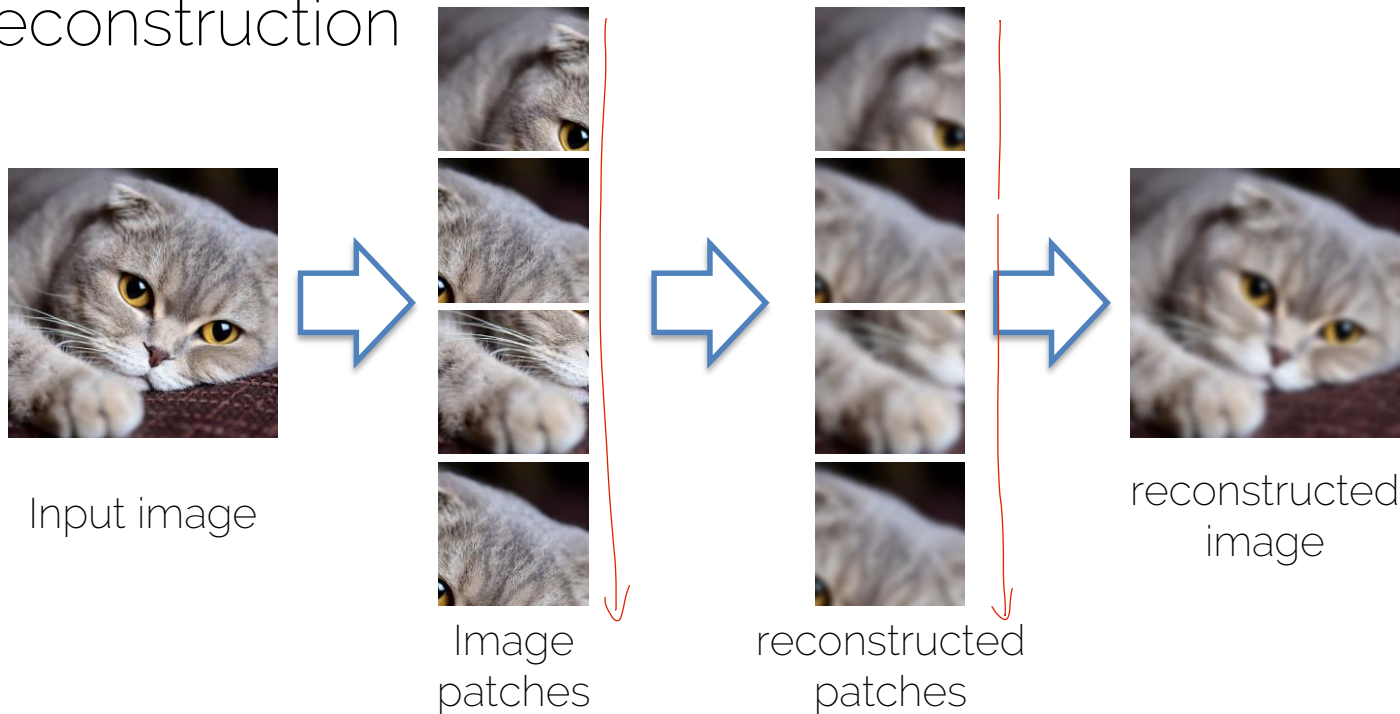
Sequence Modelling

- Images can also be represented as sequences!
 - Classification



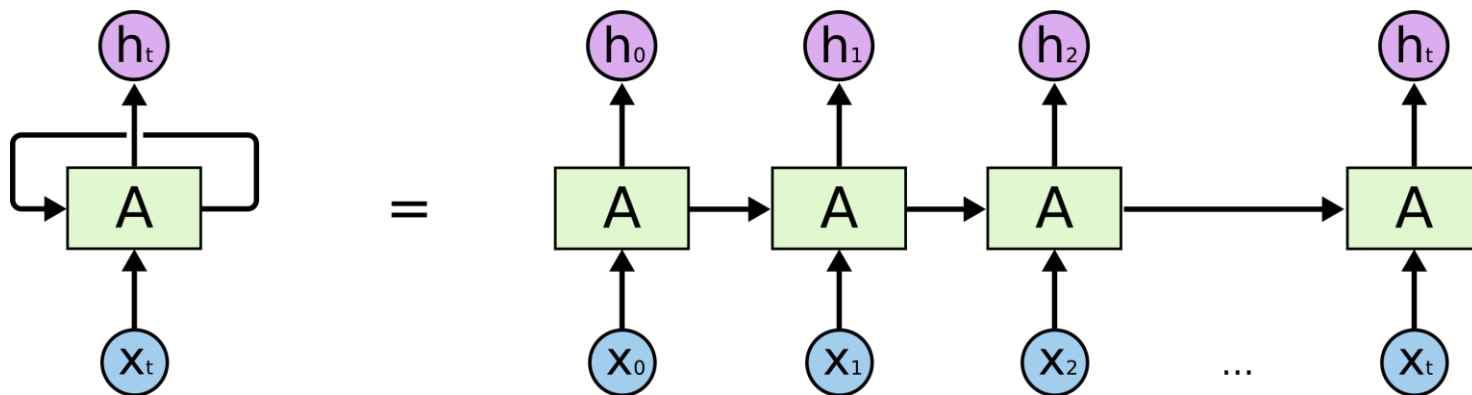
Sequence Modelling

- Images can also be represented as sequences!
 - Reconstruction



Traditional Sequence Models

- RNNs
 - Recurrent Neural Networks
 - Can be unrolled in time

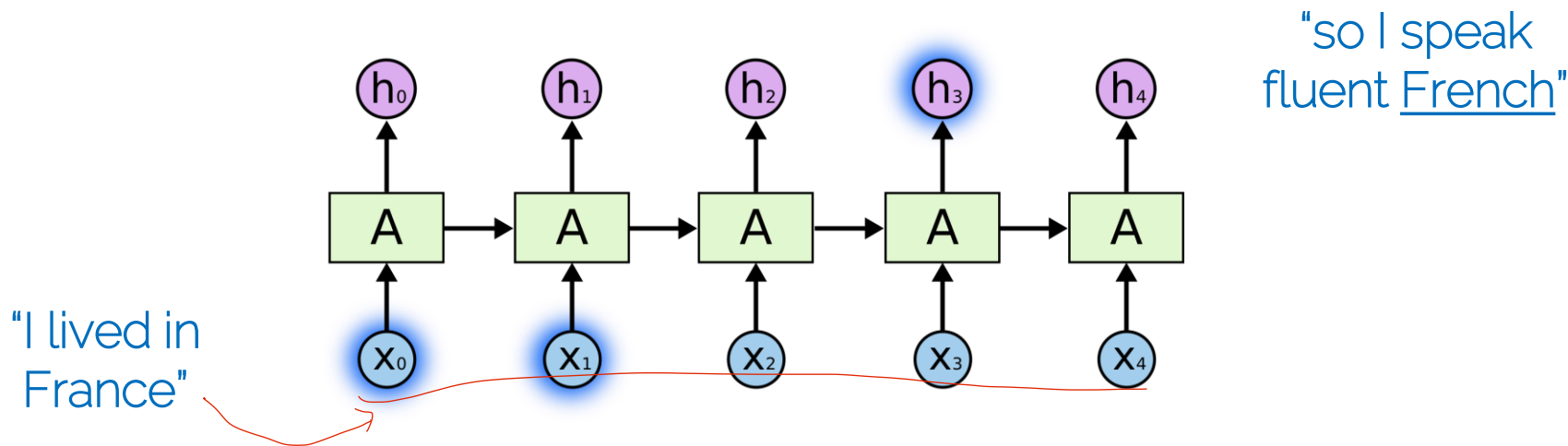


Traditional Sequence Models

- RNNs

- Good at handling short sequences

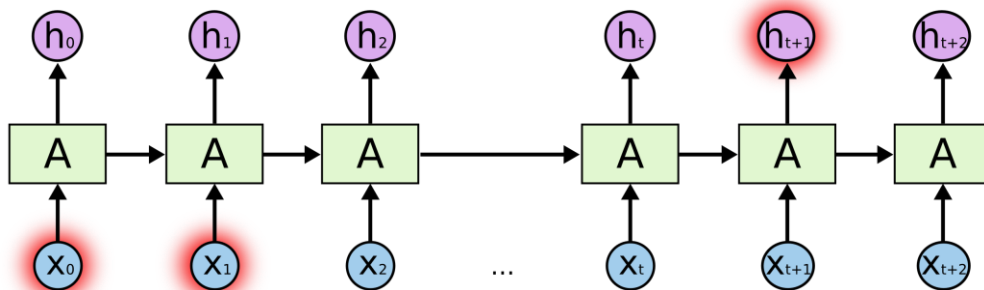
problem: long sequences



Traditional Sequence Models

- RNNs
 - Long sequences are difficult -> long-term dependency issue

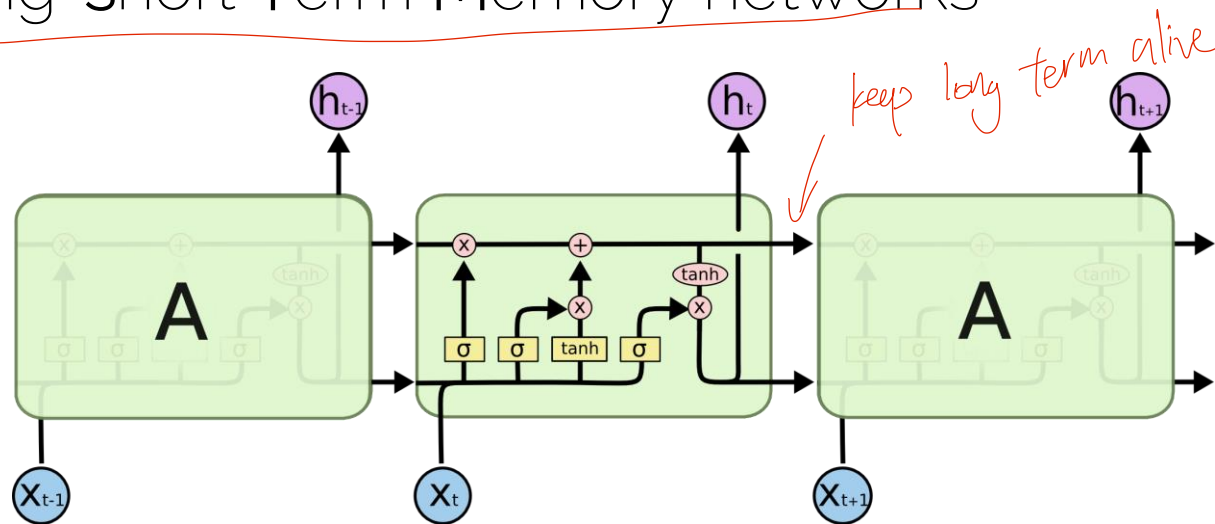
"I lived in
France"



"so I speak
fluent ___?"
(I forgot what I
said!)

Traditional Sequence Models

- LSTMs
 - Long-short Term Memory networks



Traditional Sequence Models

- LSTMs alleviates the long-term dependency issue. However, the issue still exists for extremely long sequences, e.g., documents!
- Not all words are born equal.

"I lived in France, so I speak fluent __?."

"France" is more important for predicting the word "French"



Attention
mechanism!

Attention Mechanism

- Deterministic vs. stochastic
 - Soft attention
 - Attend to each part of the input signal
 - Attention weights sum to 1
 - Deterministic and differentiable
 - E.g., when predicting the word “French”:

“I lived in France, so I speak fluent French.”

Attention Mechanism

- Deterministic vs. stochastic
 - Hard attention
 - Attend to **one** part of the input signal (one-hot)
 - Stochastic and non-differentiable
 - Need to use Monte Carlo estimator to approximate the gradients
 - E.g., when predicting the word “French”:

“I lived in France, so I speak fluent French.”

Attention Mechanism

- Modality for attention
 - Self-attention
 - Attend to the input signal itself

"I lived in France, so I speak fluent French."



"I lived in France, so I speak fluent French."

Attention Mechanism

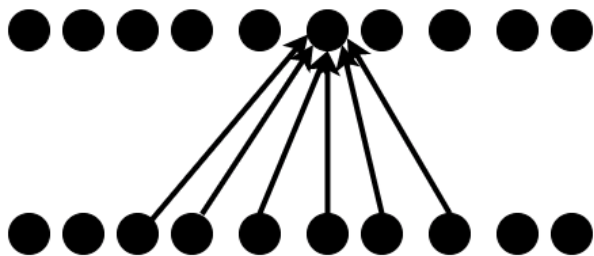
- Modality for attention
 - Cross-attention
 - Attend to another input signal as side information



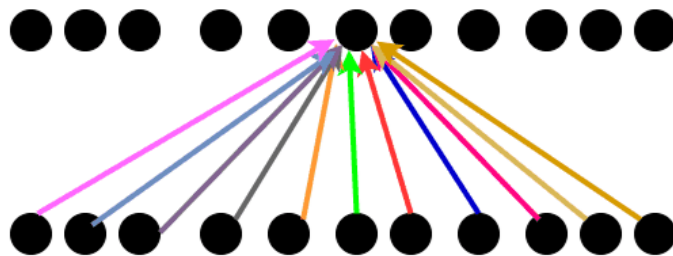
A person is standing on a beach
with a surfboard.

Attention vs Convolution

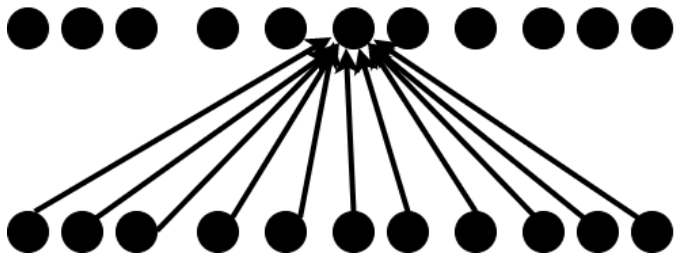
Convolution



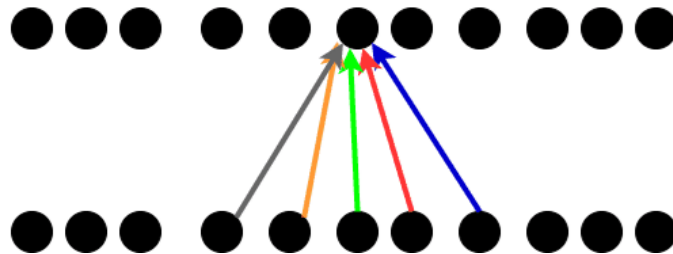
Global attention



Fully Connected layer



Local attention



Attention Mechanism

- Attention mechanism is a powerful tool to handle sequence data
- It used to be an additional plug-and-play module on top of the recurrent neural networks
- Can attention mechanism be used for handling sequence data **DIRECTLY**? -> Yes, **transformers**!

Transformers in Language

Attention is All You Need

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

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

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* †
illia.polosukhin@gmail.com

Attention is All You Need

- Scale up pure self-attention layers as the first transformer architecture
- Solve the long-term dependency issue in sequence modelling
- Extremely powerful at handling sequence data (texts)
- Bigger model, bigger capacity -> better performance when trained on large-scale datasets

Attention

Index the values
via a differentiable
operator

Multiply queries
with keys

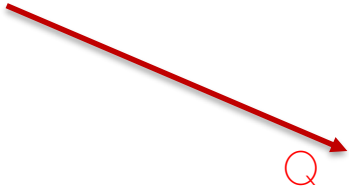
Get the values

tell me where I should look up
match
look up table
tell me how important the location is
similarity
sum to 1

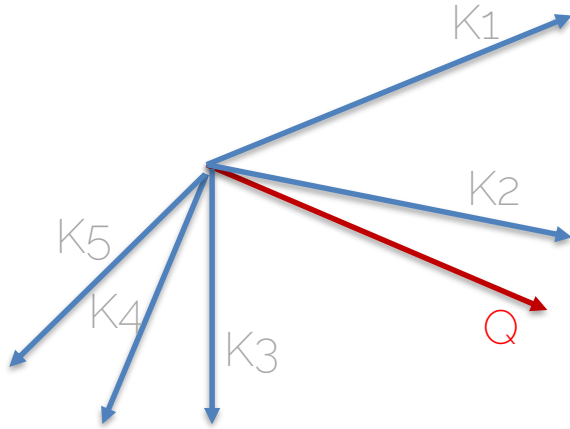
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

To train them well, divide by $\sqrt{d_k}$, "probably" because for large values of the key's dimension, the dot product grows large in magnitude, pushing the softmax function into regions where it has extremely small gradients.

Attention

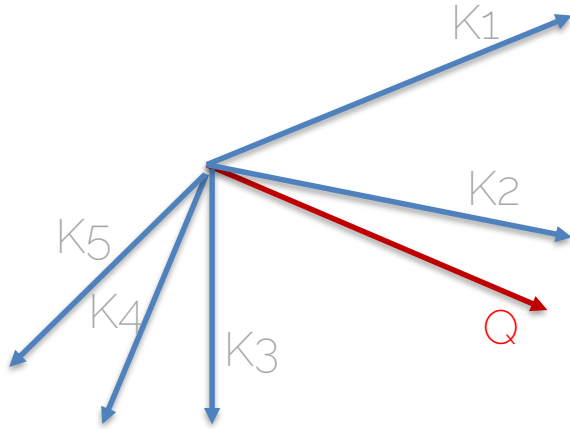


Attention



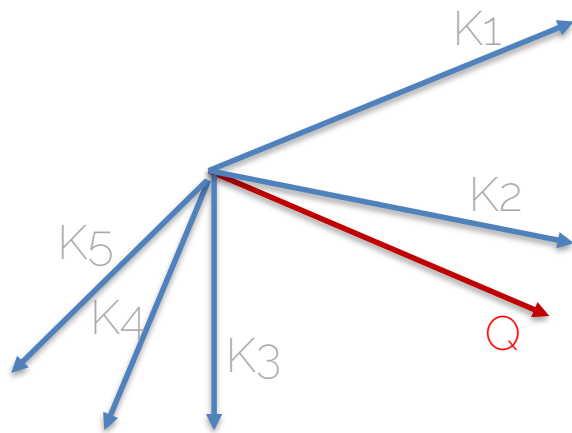
Attention

compare Q with all k
and $k \leftrightarrow v$



Values
V_1
V_2
V_3
V_4
V_5

Attention

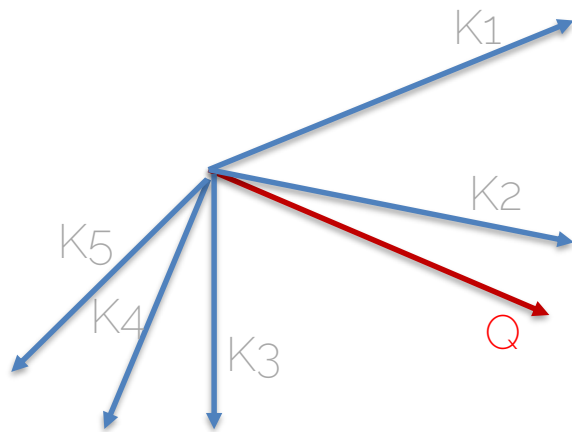


Values
V1
V2
V3
V4
V5

QK^T

Dot product between $\langle Q, K1 \rangle$, $\langle Q, K2 \rangle$, $\langle Q, K3 \rangle$, $\langle Q, K4 \rangle$, $\langle Q, K5 \rangle$.

Attention

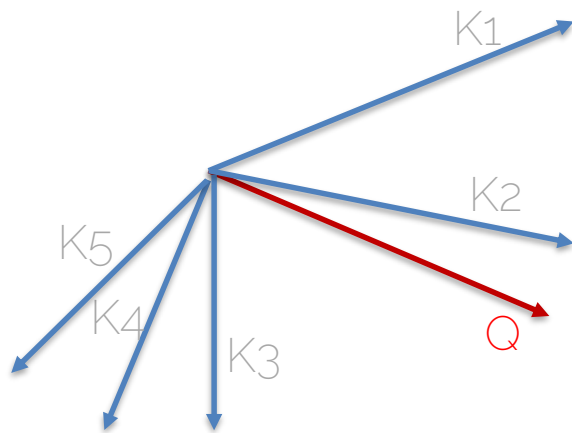


Values
V1
V2
V3
V4
V5

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

Is simply inducing a distribution over the values.
The larger a value is, the higher is its softmax value.
Can be interpreted as a differentiable soft indexing.

Attention

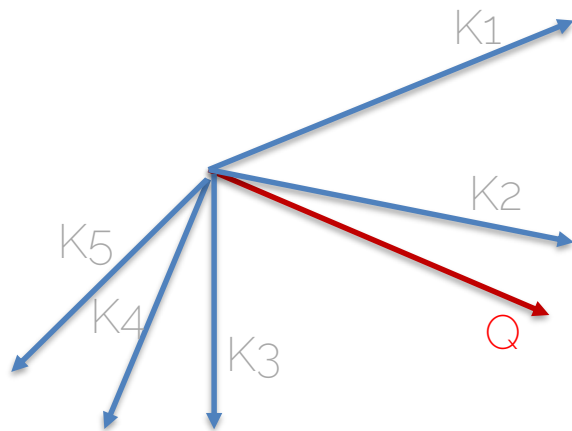


Values
V1
V2
V3
V4
V5

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

Is simply inducing a distribution over the values.
The larger a value is, the higher is its softmax value.
Can be interpreted as a differentiable soft indexing.

Attention



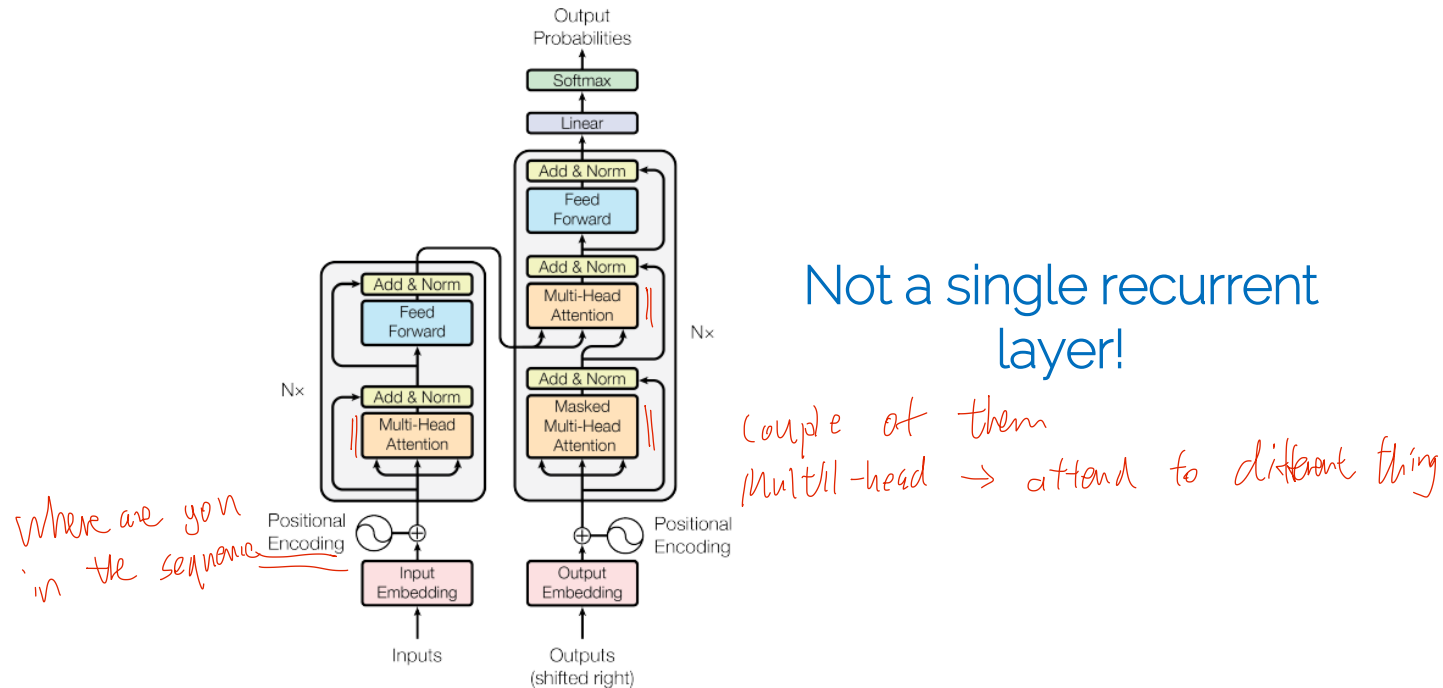
Values
V1
V2
V3
V4
V5

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

Selecting the value V where the network needs to attend..

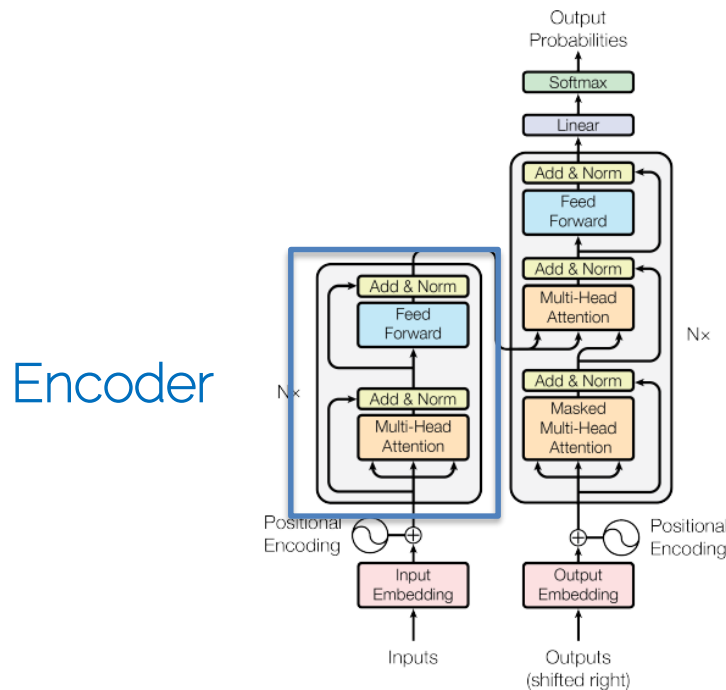
Attention is All You Need

- Transformers under the hood



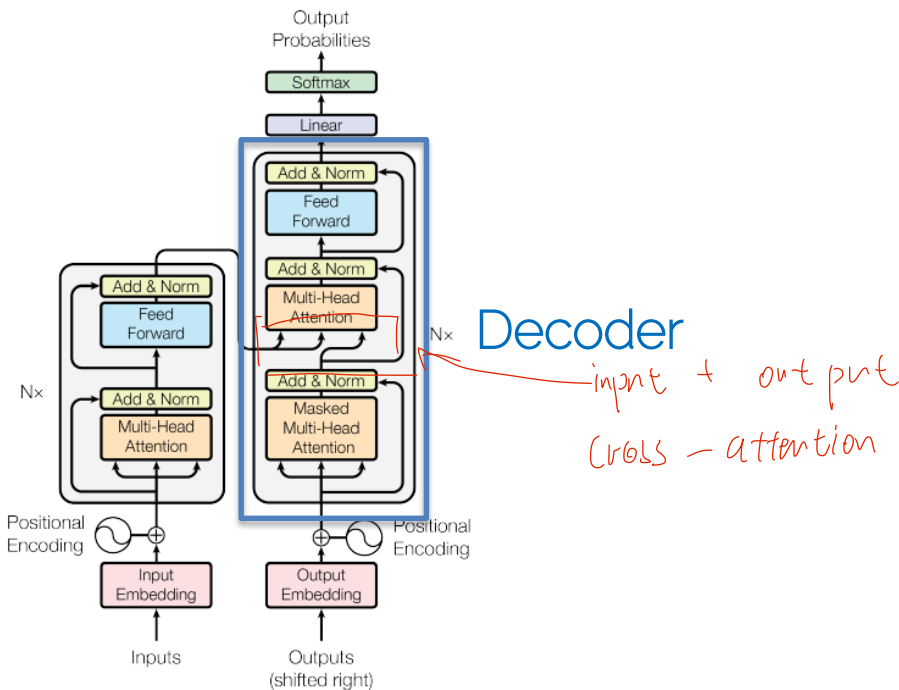
Attention is All You Need

- Transformers under the hood



Attention is All You Need

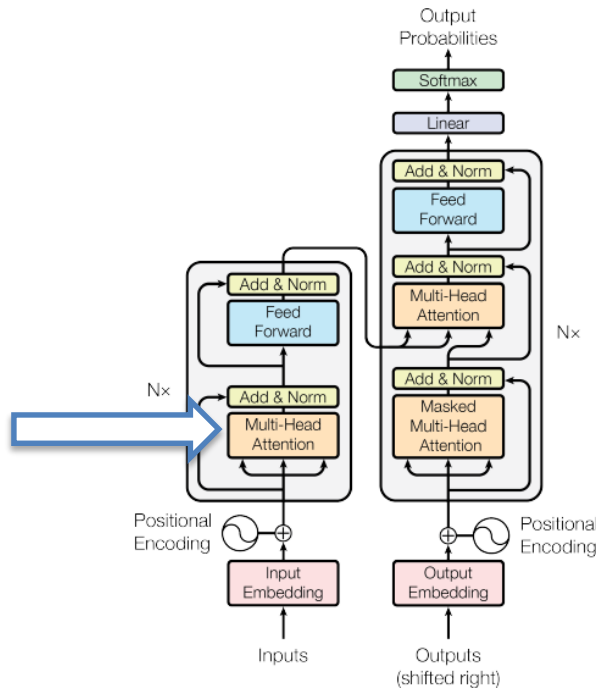
- Transformers under the hood



Attention is All You Need

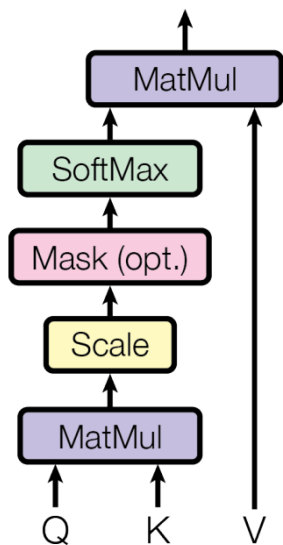
- Transformers under the hood

Core unit: scaled
dot-product
attention



Attention is All You Need

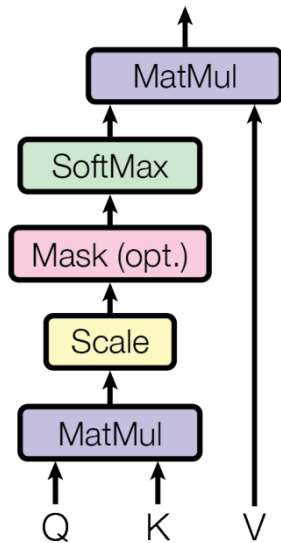
- Scaled dot-product attention



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

Attention is All You Need

- Scaled dot-product attention

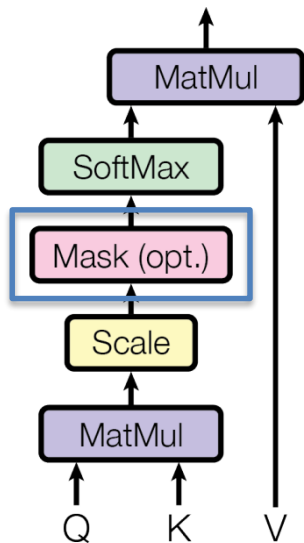


$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

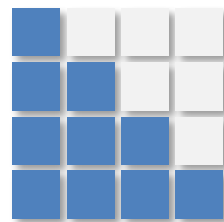
Self-attention masks

Attention is All You Need

- Scaled dot-product attention



Triangular masking for unidirectional
(left-to-right) modelling



■ attend
■ do not attend

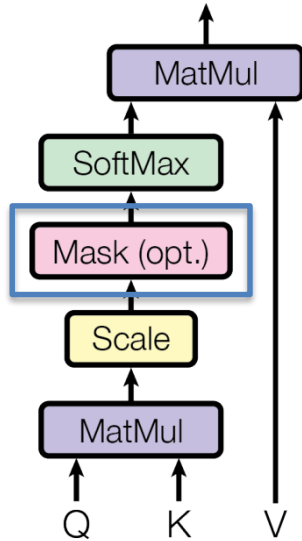
can only attend to the past

"German people speak German"

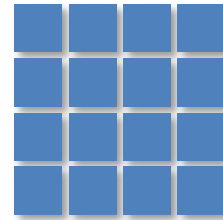
Only attend to the words before it and itself

Attention is All You Need

- Scaled dot-product attention



Full masking for bidirectional modelling



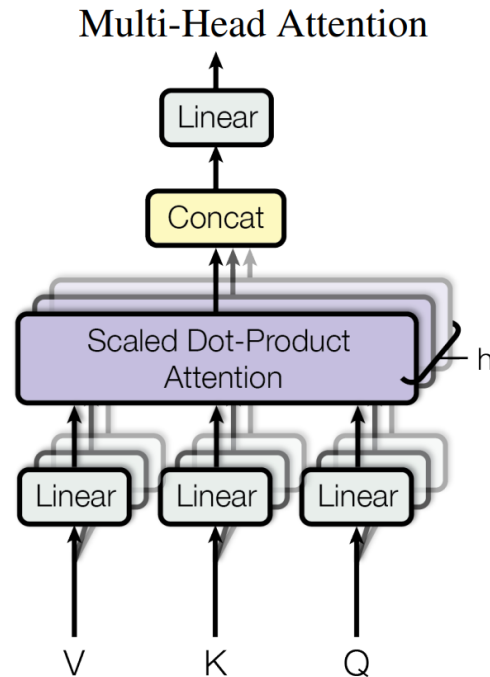
"German people speak German"

Attend to the words before and after it, and itself

Attention is All You Need

- Multi-head Attention

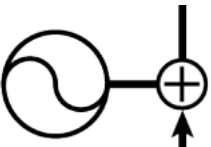
Stack up the scaled dot-product attention module as attention heads



Different heads attend to different parts of the input signals

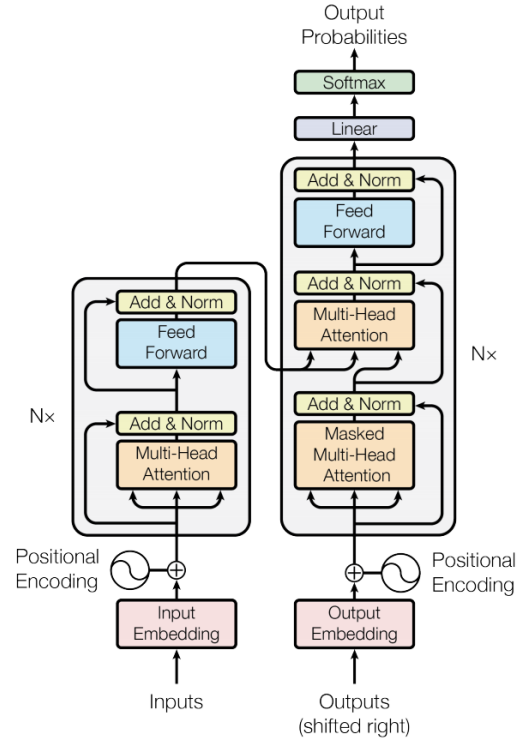
Positional Encoding

Uses fixed positional encoding based on trigonometric series, in order for the model to make use of the order of the sequence

Positional Encoding 

$$PE_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$
$$PE_{(pos, 2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

dimension *integer value*

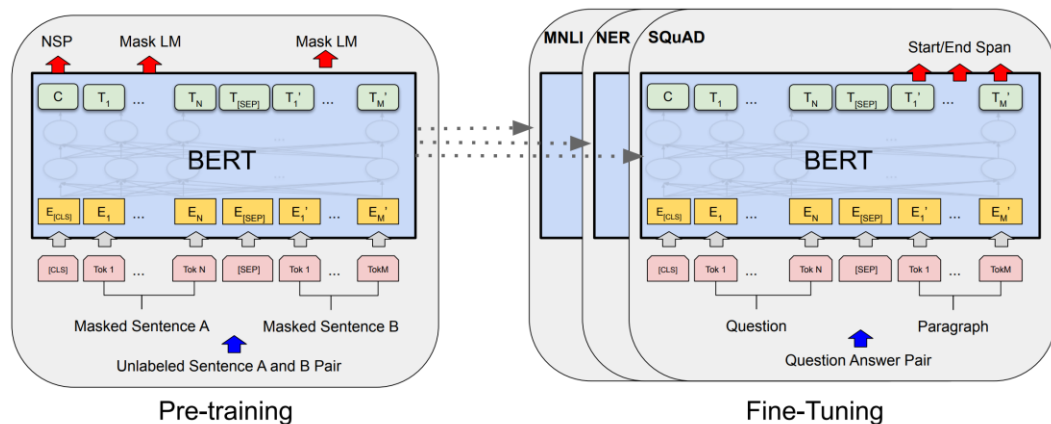


Attention is All You Need

- Significantly improved SOTA in Machine Translation
- Launched a transformer revolution in the NLP field
- Foundation of large NLP models like BERT (Google) and GPT-3, ChatGPT (OpenAI)!
- Transformers finally made its way to compute vision (will talk about it later!)

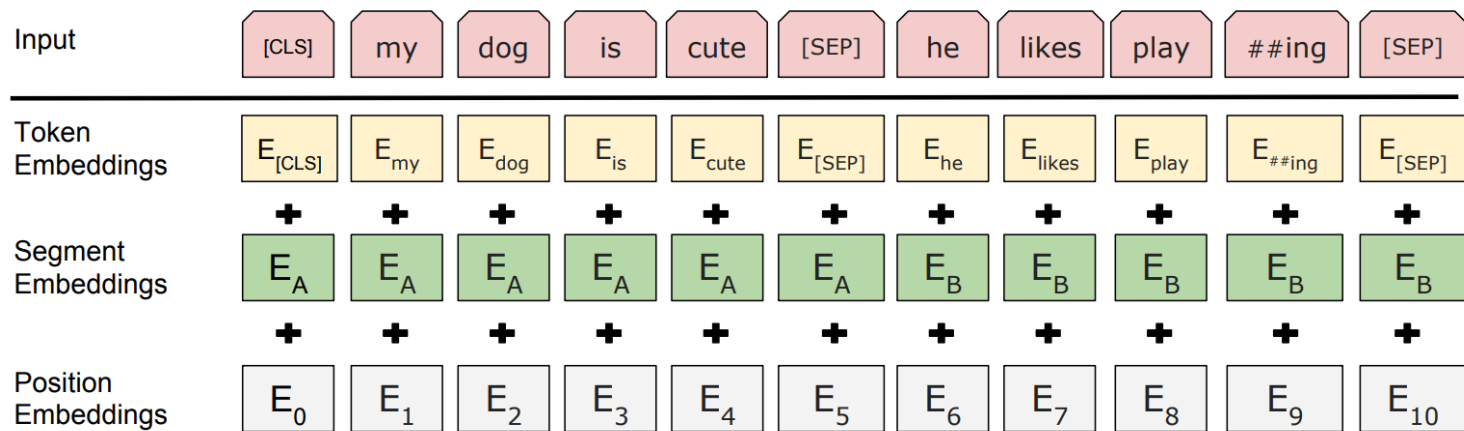
BERT

- Bidirectional Encoder Representations from Transformers
 - A big transformer as a text encoder



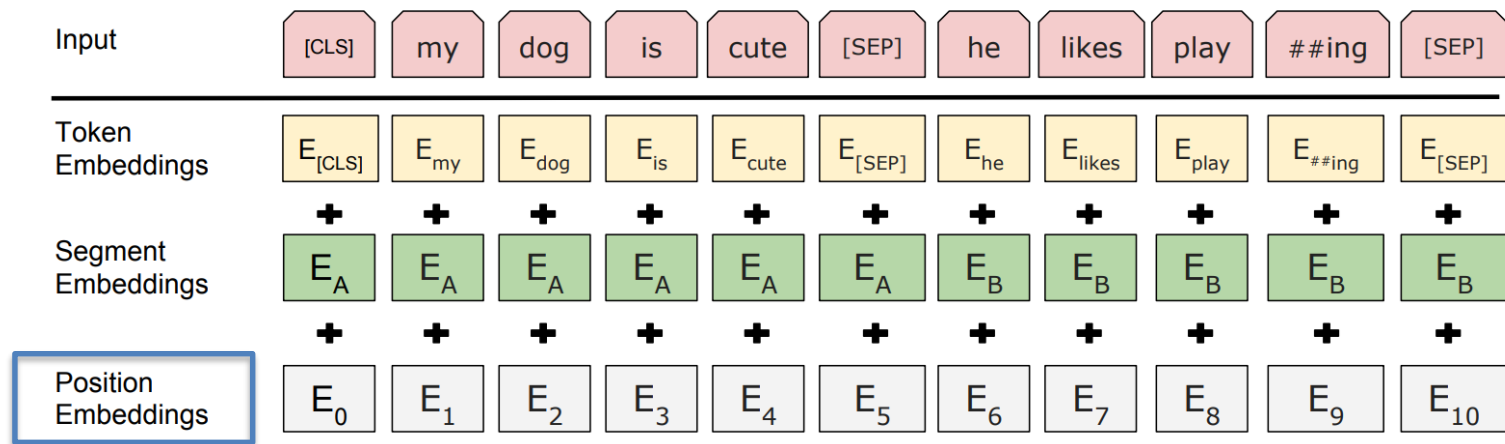
BERT

- BERT Input Representations – Three embeddings



BERT

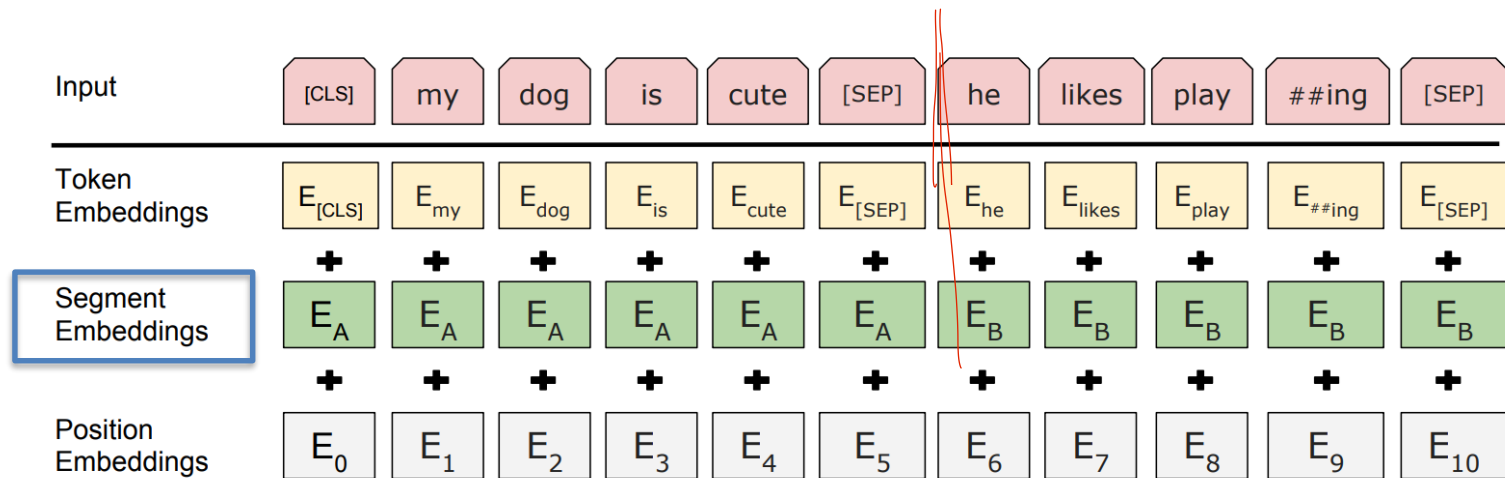
- BERT Input Representations



To indicate the positions in the sequence

BERT

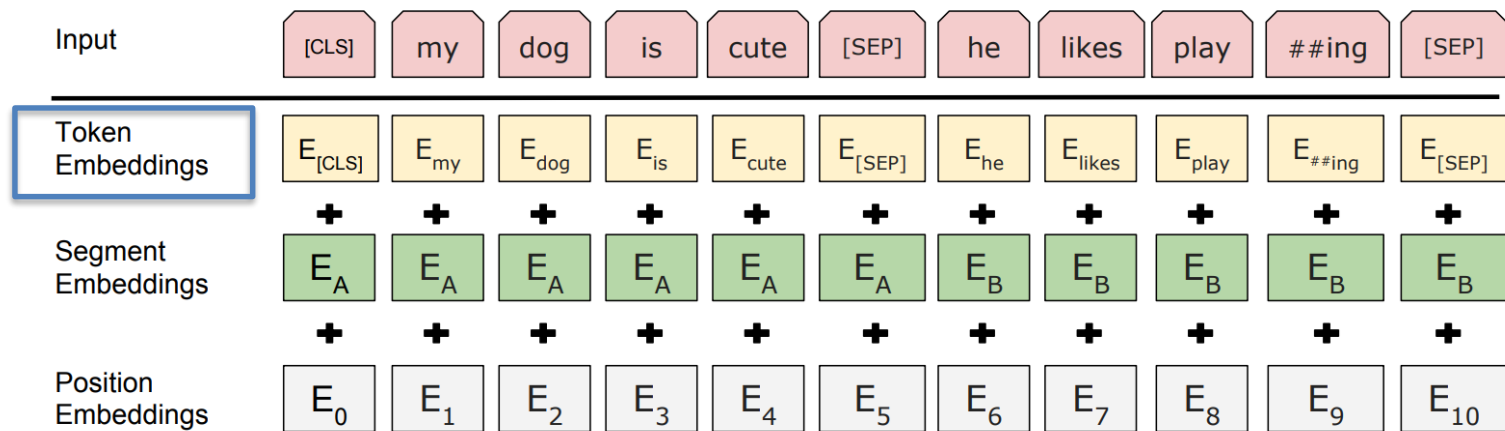
- BERT Input Representations



To indicate the sentence A or B

BERT

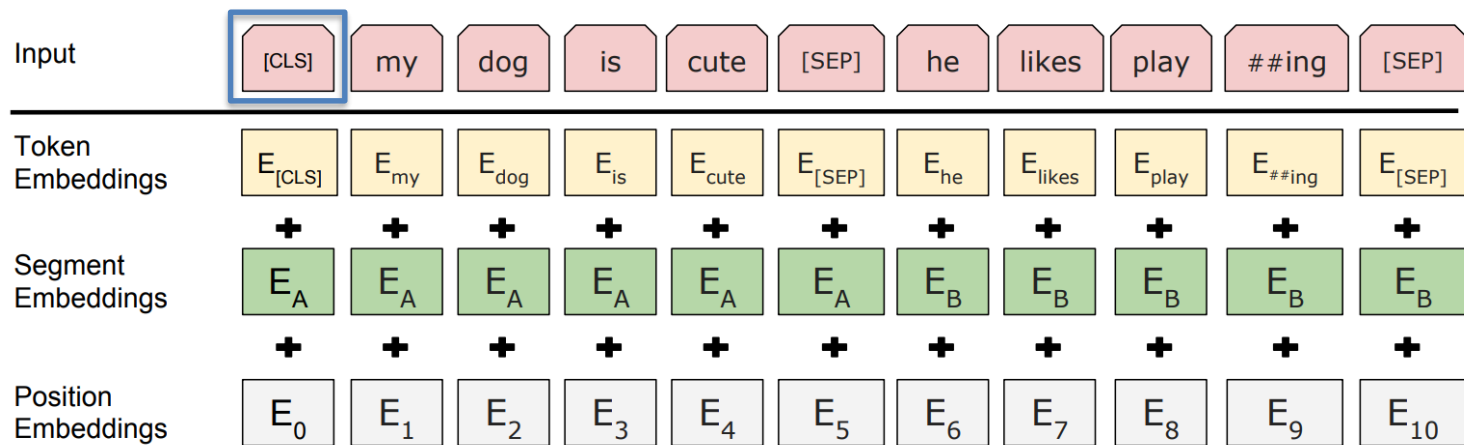
- BERT Input Representations



Word embeddings

BERT

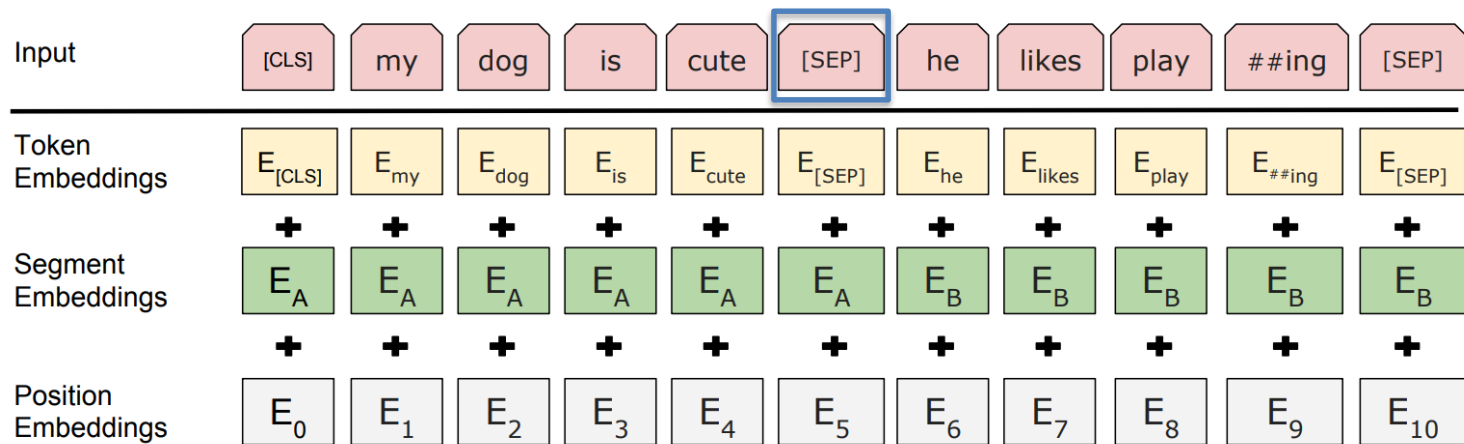
- BERT Input Representations



Learnable token for Next Sentence Prediction

BERT

- BERT Input Representations



Indicate the end of the sentence(s)

BERT

- Pre-training Objectives
 - Two unsupervised tasks
 - Masked Language Modelling (MLM)
 - Next Sentence Prediction (NSP)
 - No human annotation needed!

BERT

- Masked Language Modelling (MLM)
 - Key idea:
 - Randomly mask out some words from the input
 - Predict the masked words with the context from the input itself
 - Enforce the network to learn the word-level context

BERT

- Masked Language Modelling (MLM)

Input texts

"In Germany, people speak German"

BERT

- Masked Language Modelling (MLM)

Masked input

"In Germany, people [MASK] [MASK]"

Input texts

"In Germany, people speak German"



Random Masking

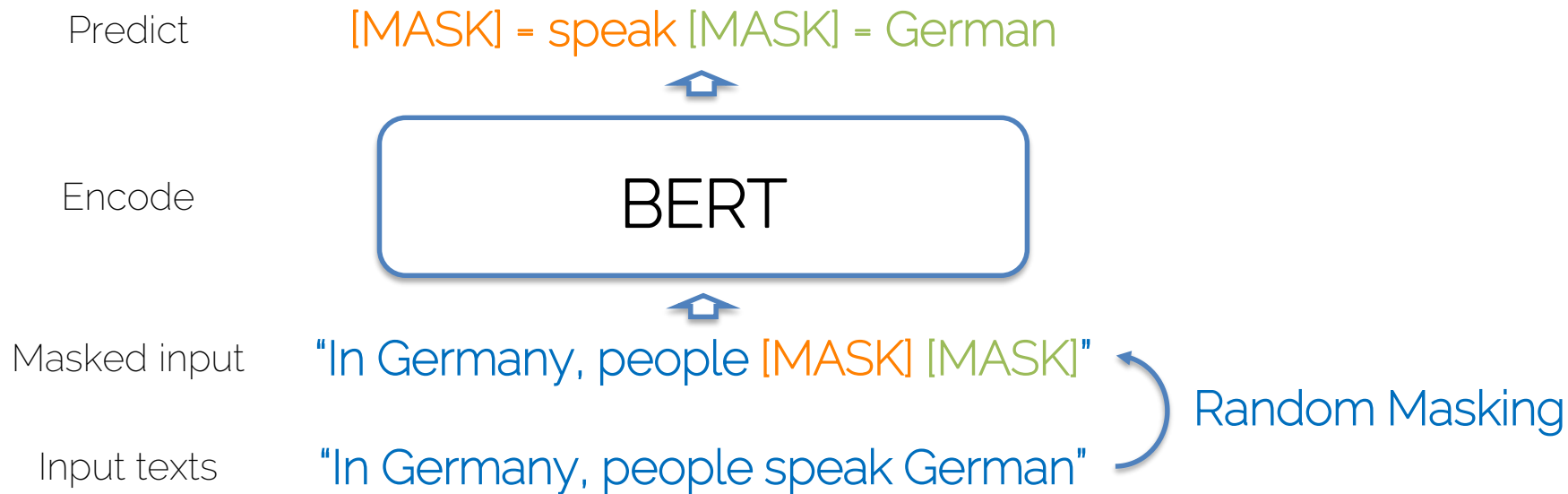
BERT

- Masked Language Modelling (MLM)



BERT

- Masked Language Modelling (MLM)



BERT

- Next Sentence Prediction (NSP)
 - Key idea:
 - Take two sentence A and B from the dataset
 - 50% of the time B is the actual next sentence of A, another 50% of the time B is randomly sampled from the dataset
 - Predict if B is the next sentence of A
 - Enforce the network to learn the sentence-level context

BERT

- Next Sentence Prediction (NSP)

A sample
from the
dataset

"In Germany, people speak German.
So German is the native language of the
German people."



Break up

Sentence A

"In Germany, people speak German. "

Sentence B

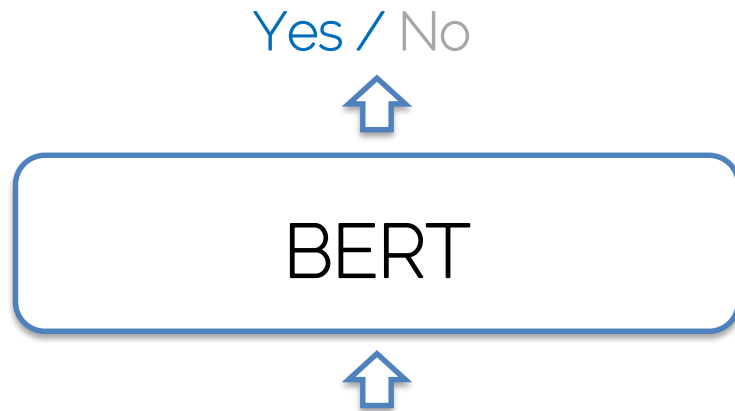
"So German is the native language of the
German people."

"Today is Wednesday."

Sentence B'
(another sentence from
the dataset)

BERT

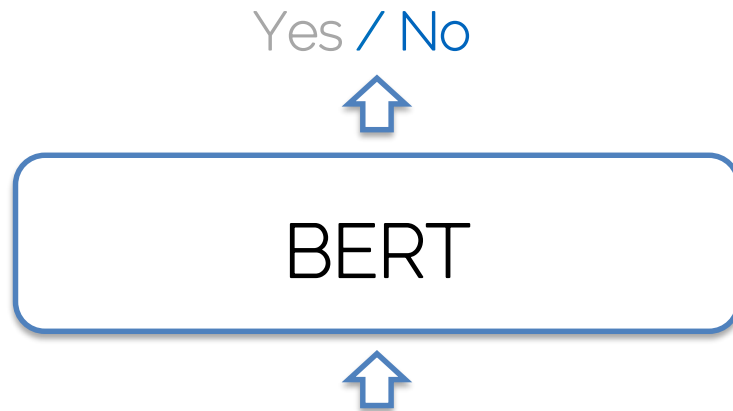
- Next Sentence Prediction (NSP)



"In Germany, people speak German. " "So German is the native language of the German people."

BERT

- Next Sentence Prediction (NSP)

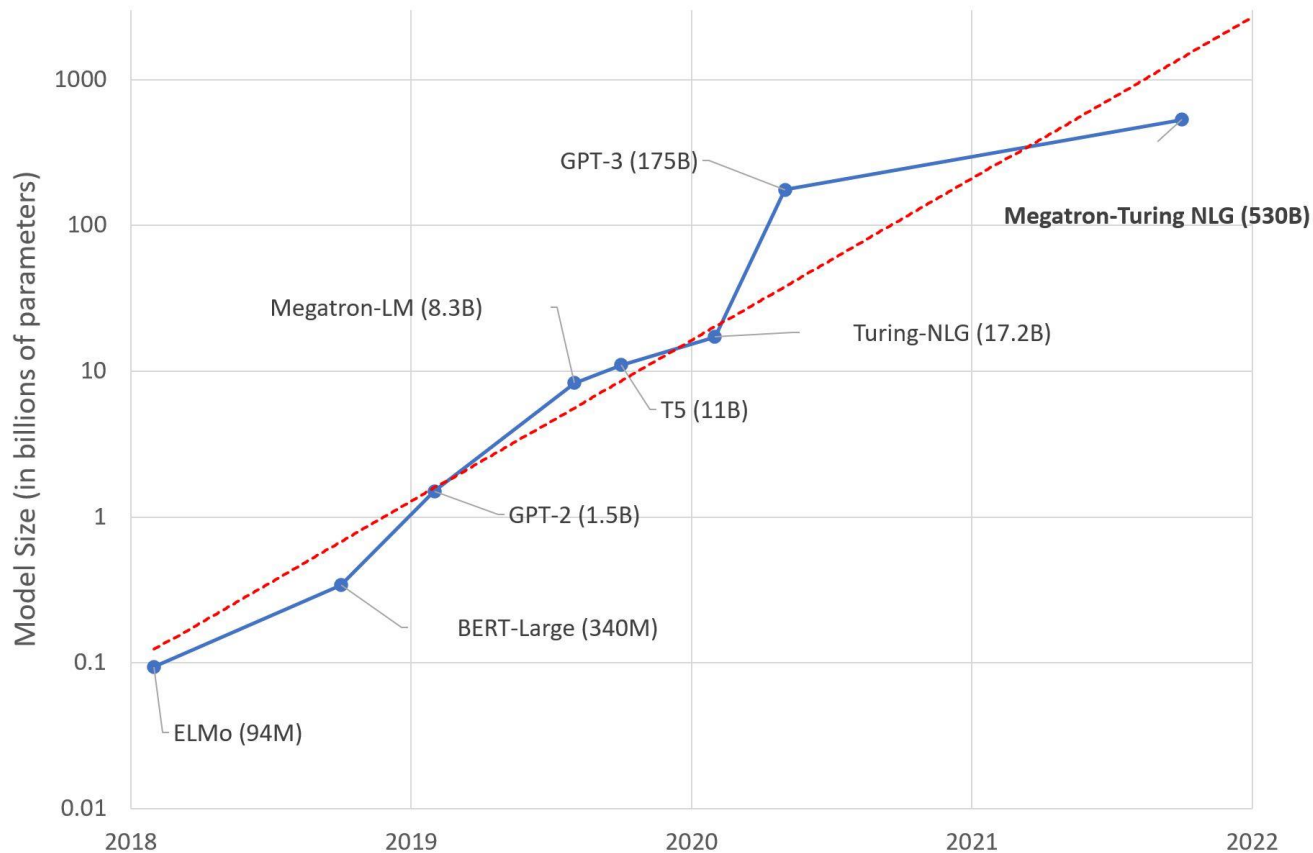


"In Germany, people speak German. " "Today is Wednesday."

BERT

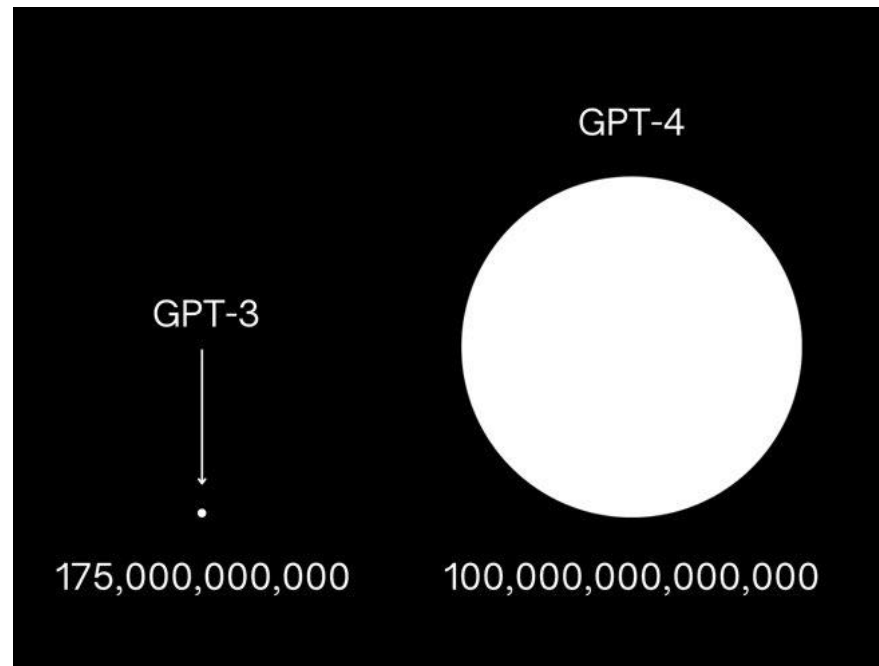
- Pre-training with two self-supervised objectives, then fine-tuned on downstream tasks
- No human annotations needed for pre-training! -> can be pre-trained on large-scale datasets, even those ones that have not been annotated, e.g., web documents.
- VERY BIG (back then in 2018)! – BERT_{large} has 340M parameters
- Masked Language Modelling has a huge impact for representation learning, even for **Computer Vision!**

Language Model Sizes



Language Model Sizes: Rumors...

- From social media
 - Take with caution!!!!
 - GPT-4 100 trillion params
- If you have not seen it:
<https://chat.openai.com/>
 - Based on GPT-3.5



Transformers in Computer Vision

Vision Transformers (ViTs)

- CNNs can be computationally demanding and require a great amount of design tricks and efforts
- Images can be modelled as sequences of patches
 - Vision Transformers (ViTs)



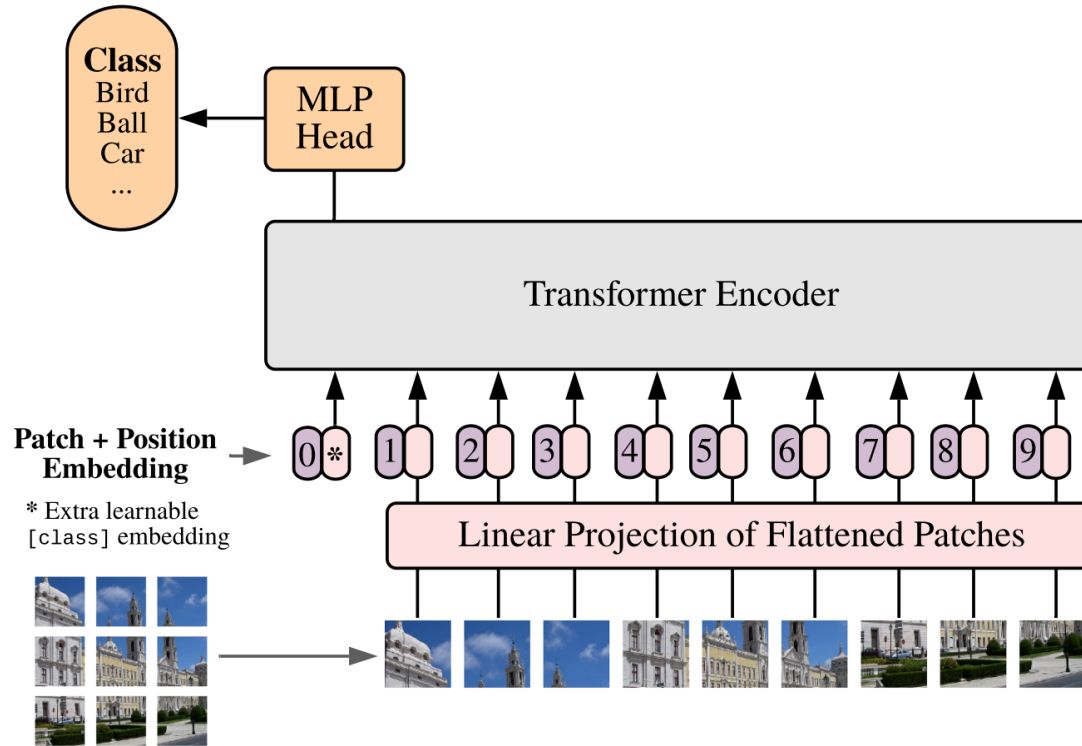
Input image

Image patches

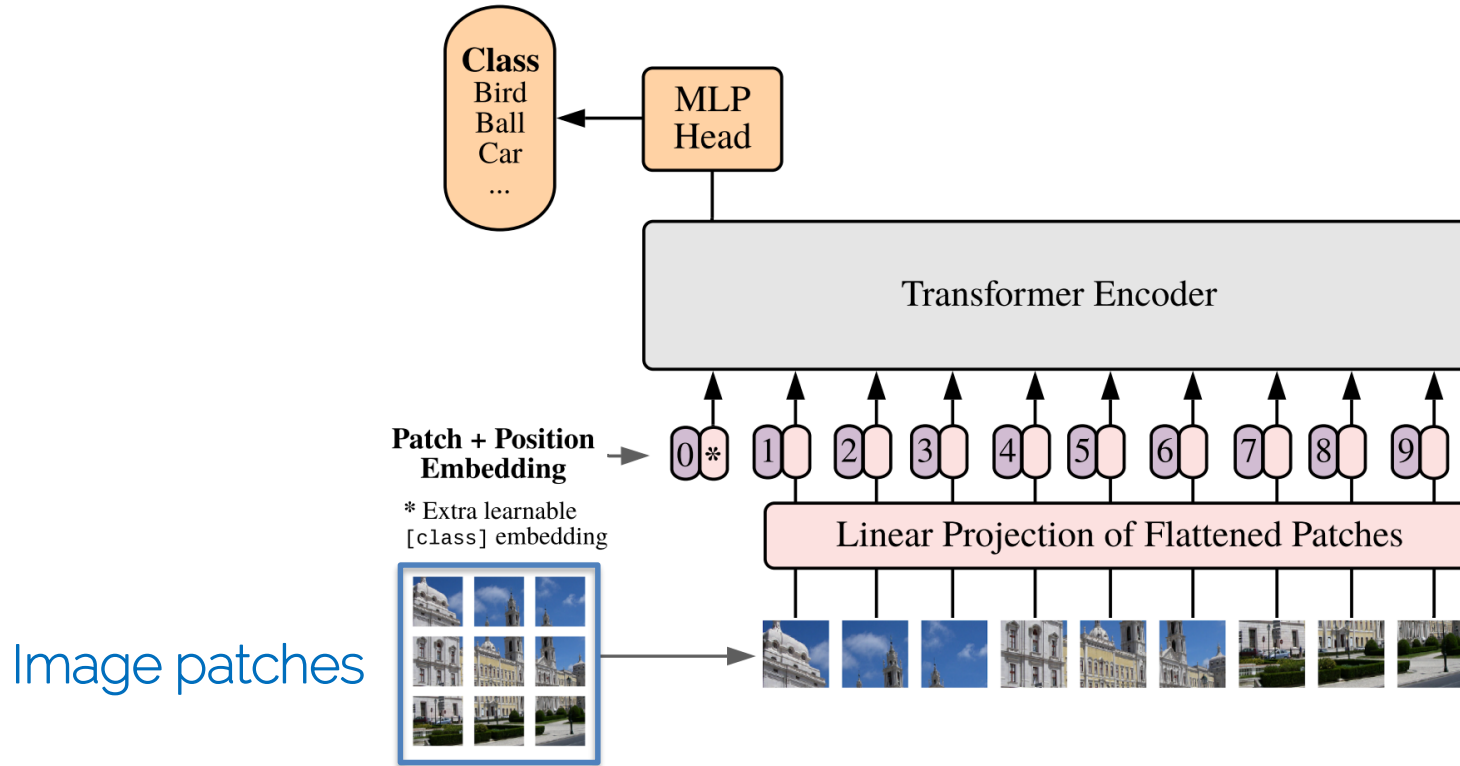
Vision Transformers (ViTs)



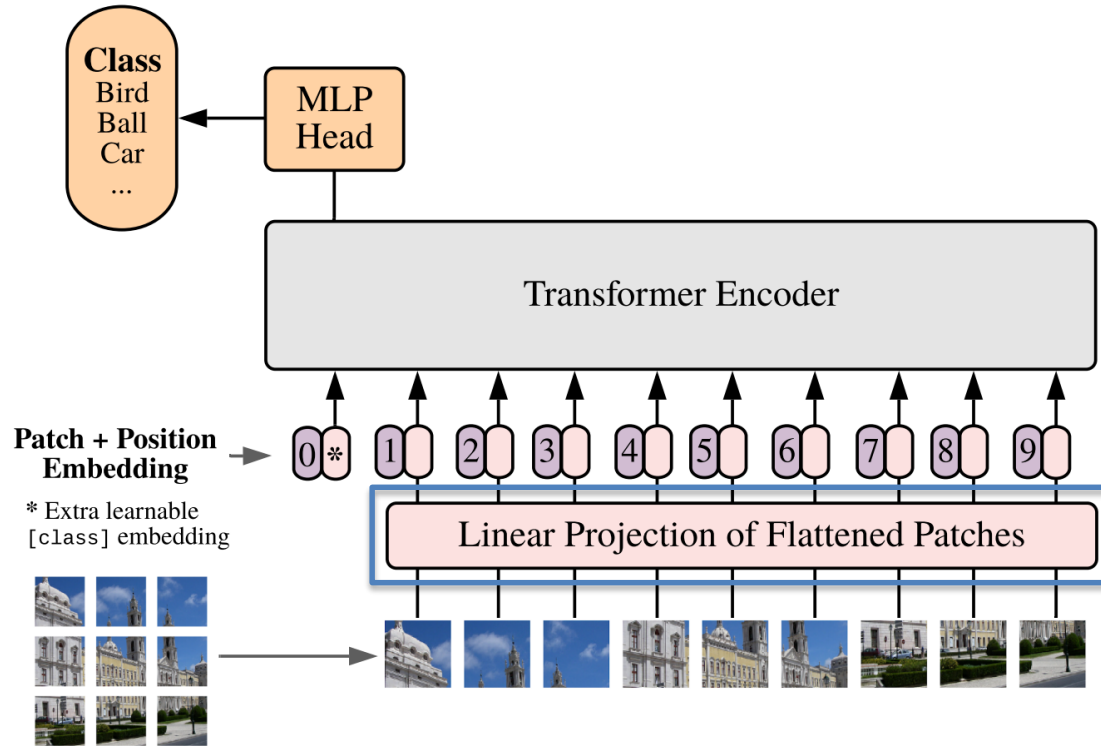
Vision Transformers (ViTs)



Vision Transformers (ViTs)

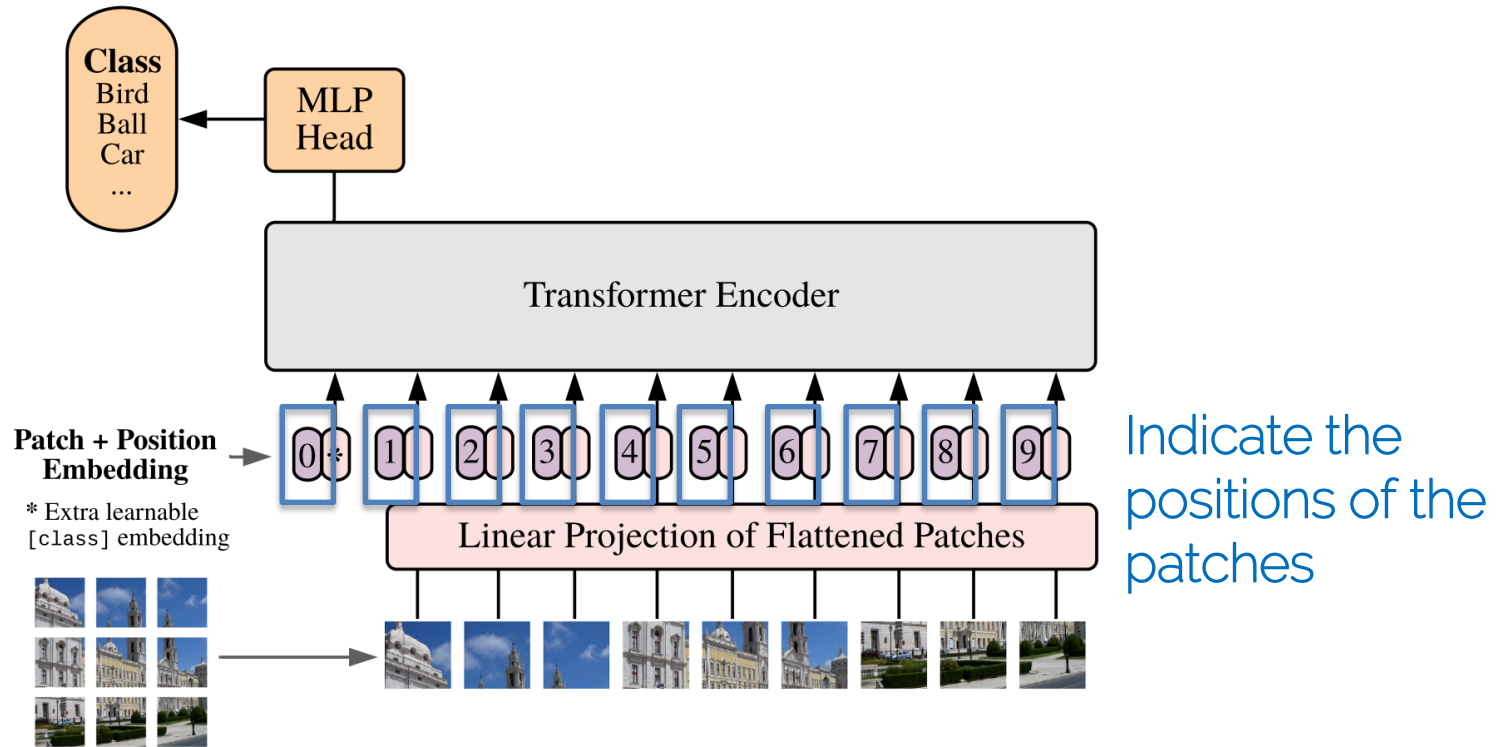


Vision Transformers (ViTs)



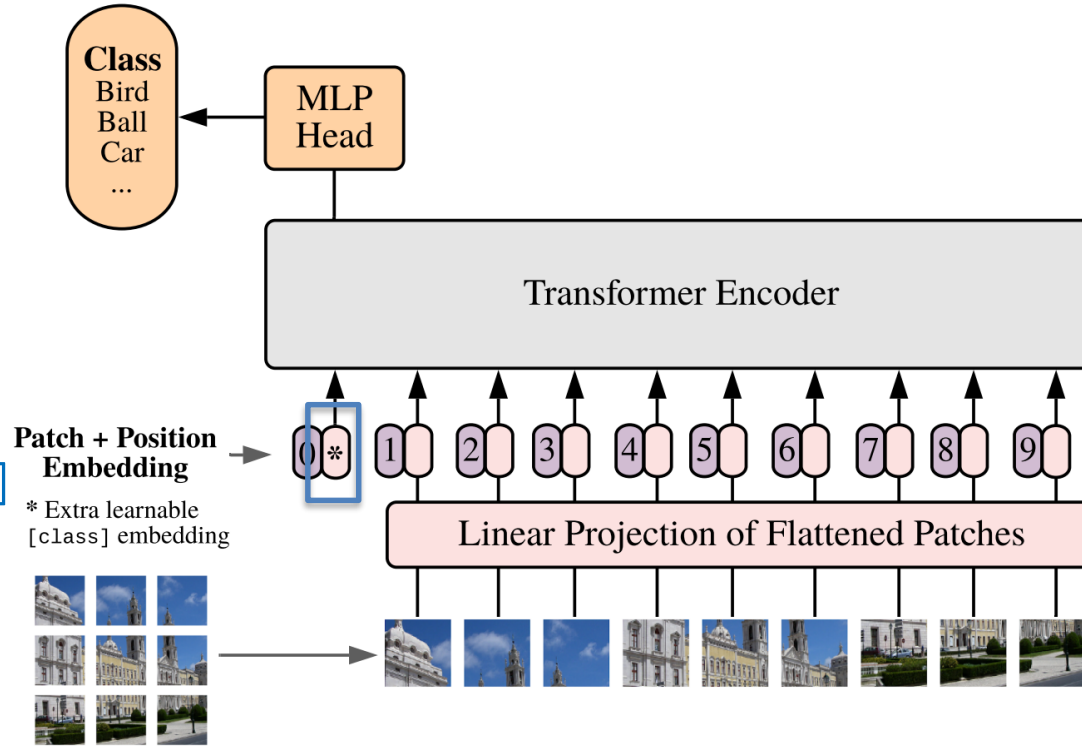
MLP (can be replaced by a tiny CNN)

Vision Transformers (ViTs)

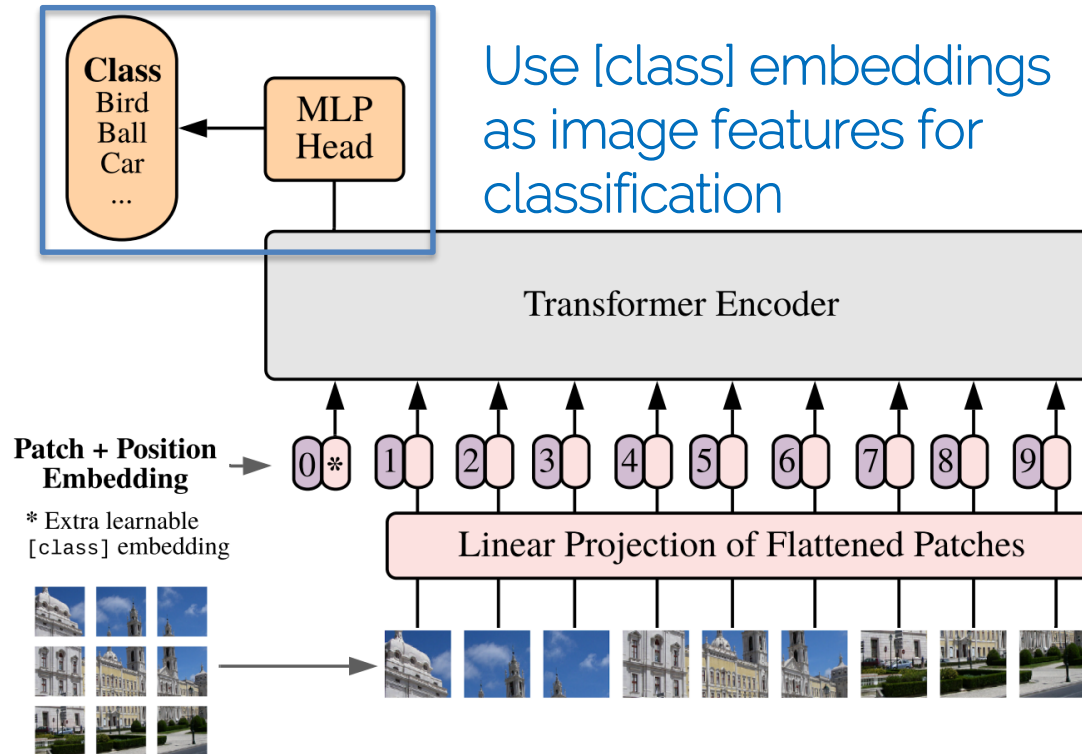


Vision Transformers (ViTs)

Learnable
special token
(similar to [CLS]
in BERT)



Vision Transformers (ViTs)



Vision Transformers (ViTs)

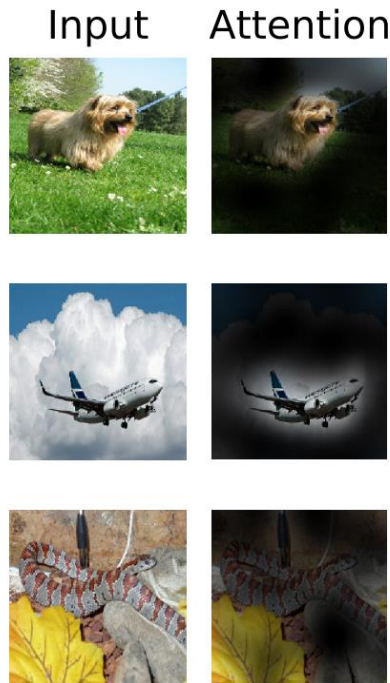
- Pre-trained on several big datasets
- Perform transfer learning on target datasets/benchmarks (freeze the pre-trained transformer backbone, fine-tune the classifier only)
- Outperform the ResNet baseline with substantially less computational resources for pre-training

Vision Transformers (ViTs)

- A closer look at ViTs
 - Transformers in language can give us the attention maps on the input words
 - Can ViTs provide the attention maps on the input image patches? -> YES!

Vision Transformers (ViTs)

- Attention maps in ViTs



Attention maps on the input image while computing the attended [class] embedding for classification

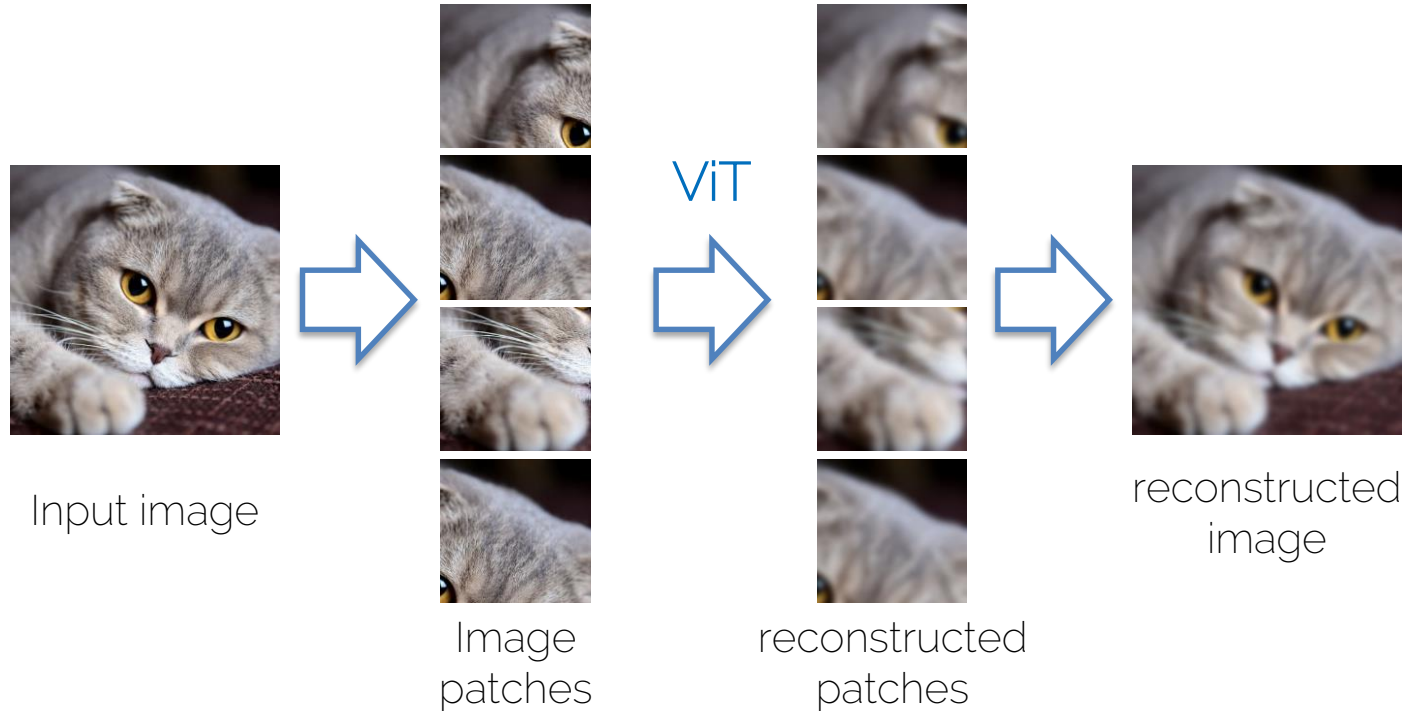
i.e., to classify an image, ViTs give us a hint for which part is most relevant to the predicted label.

Vision Transformers (ViTs)

- ViTs set a new form for image recognition
- ViTs are extremely powerful at representing image features
- ViTs are also applied in many other domains, such as representation learning (e.g., DINO, MoCo, CLIP, etc.), object detection, and multimodal learning.

Masked Auto-encoder (MAE)

- ViTs as auto-encoders

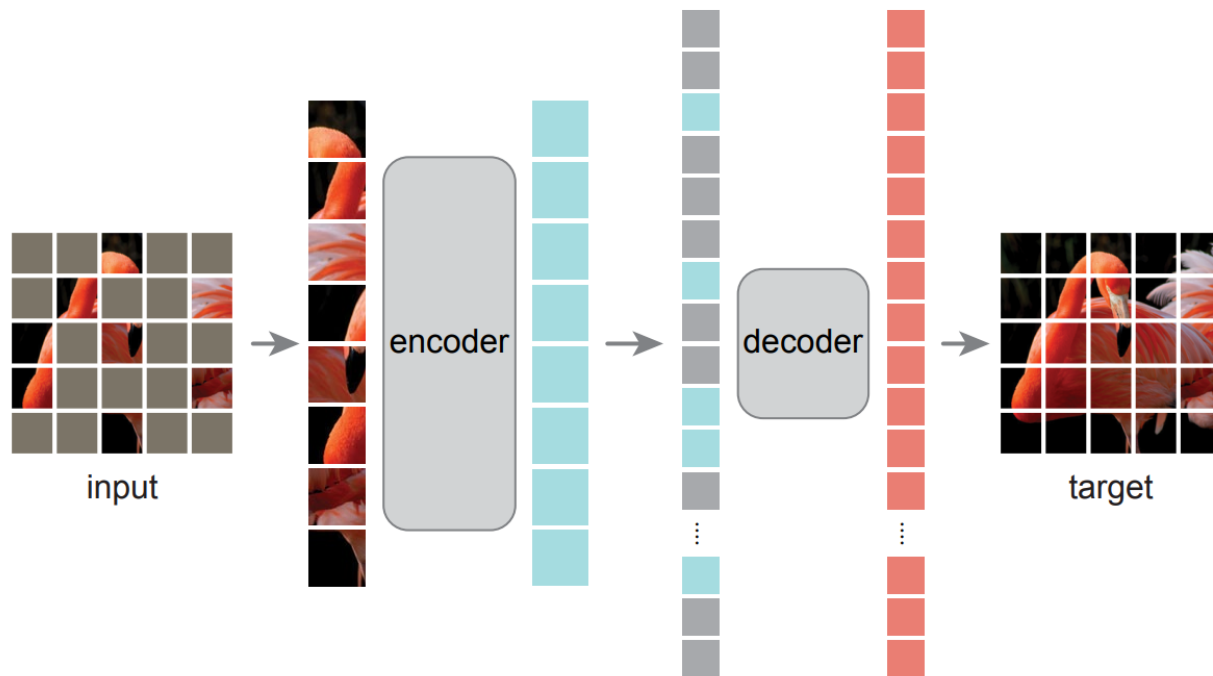


Masked Auto-encoder (MAE)

- What's new?
 - Mask out a lot of patches in the input image
 - Inputting the unmasked patches into the ViT only
 - -> Reduce the computational needs
 - Reconstruct the masked patches

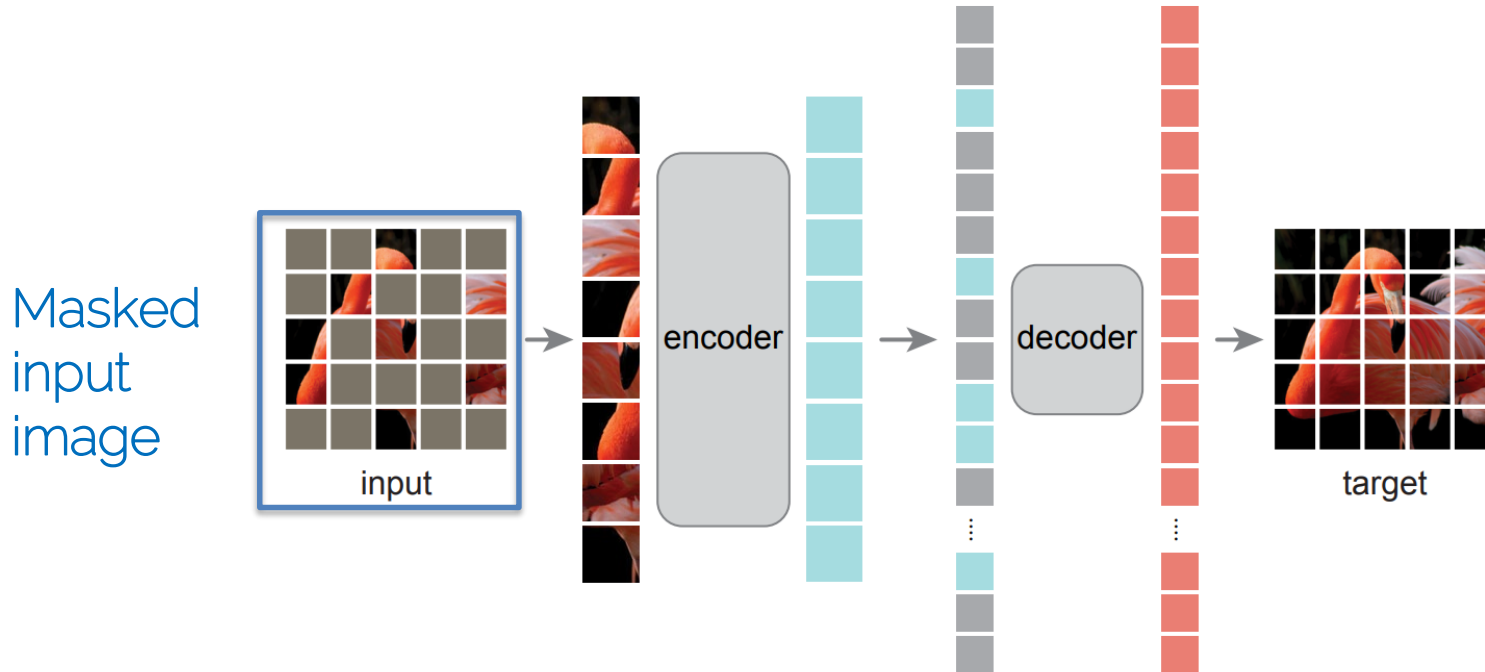
Masked Auto-encoder (MAE)

- Masked Image Modelling (MIM)



Masked Auto-encoder (MAE)

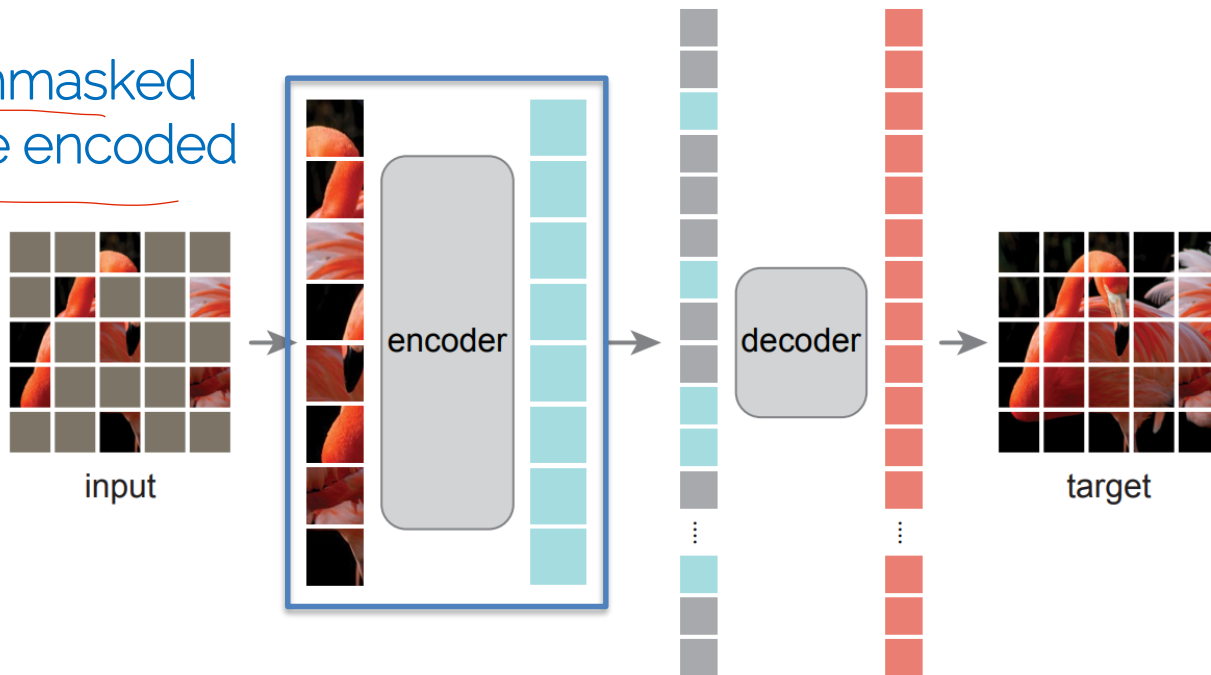
- Masked Image Modelling (MIM)



Masked Auto-encoder (MAE)

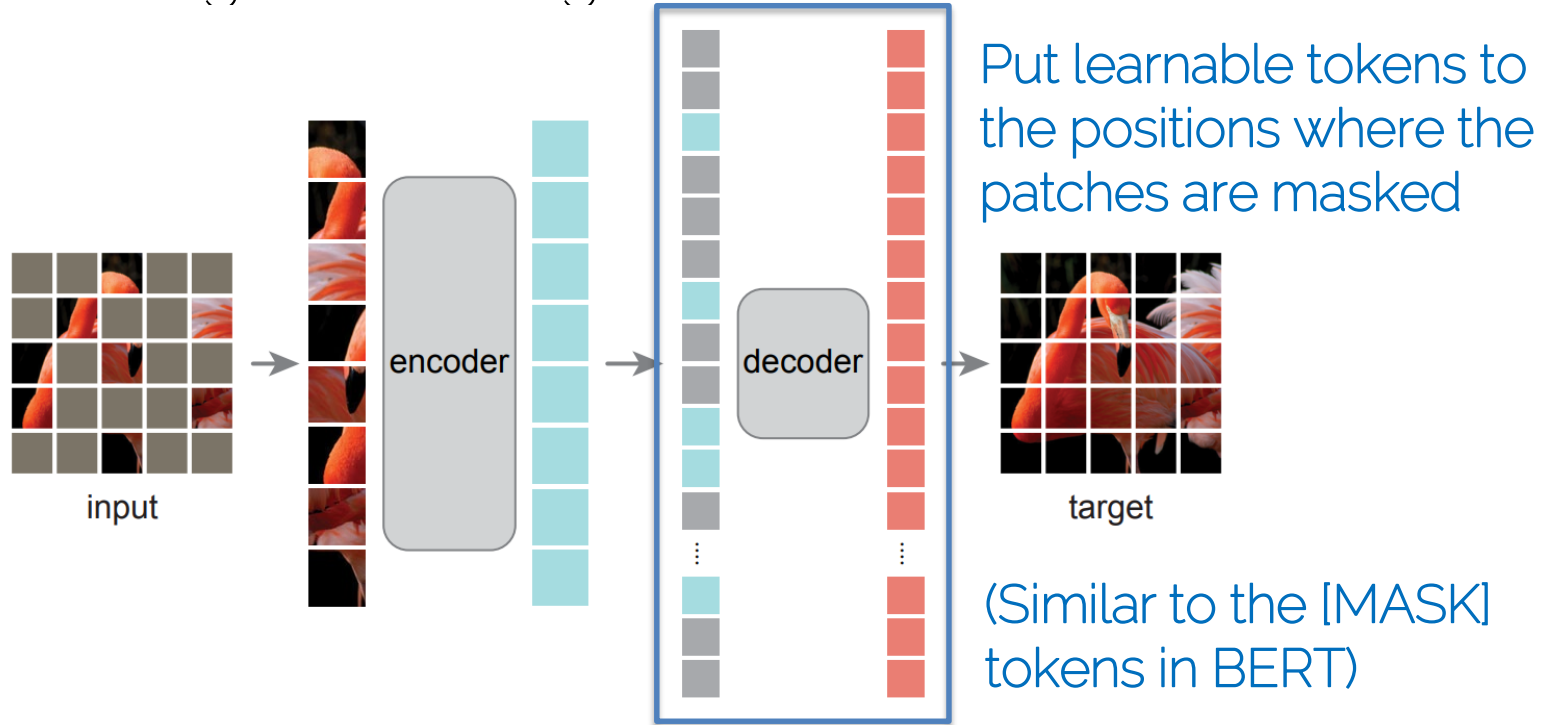
- Masked Image Modelling (MIM)

Only the unmasked patches are encoded



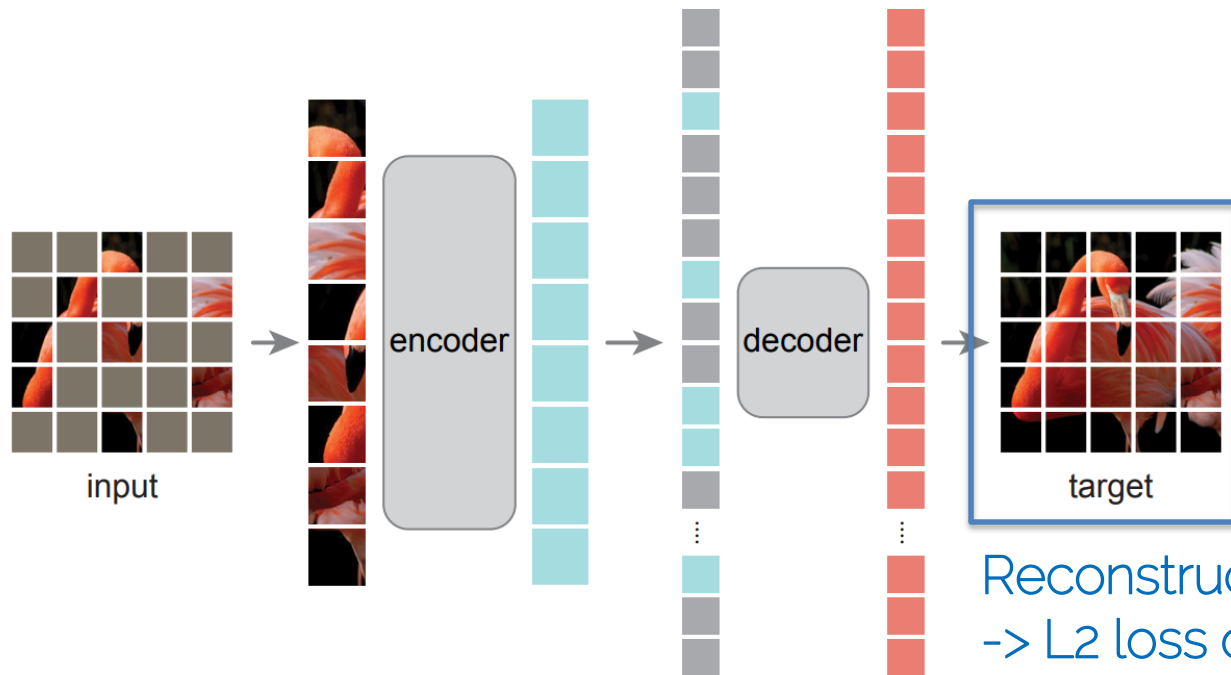
Masked Auto-encoder (MAE)

- Masked Image Modelling (MIM)



Masked Auto-encoder (MAE)

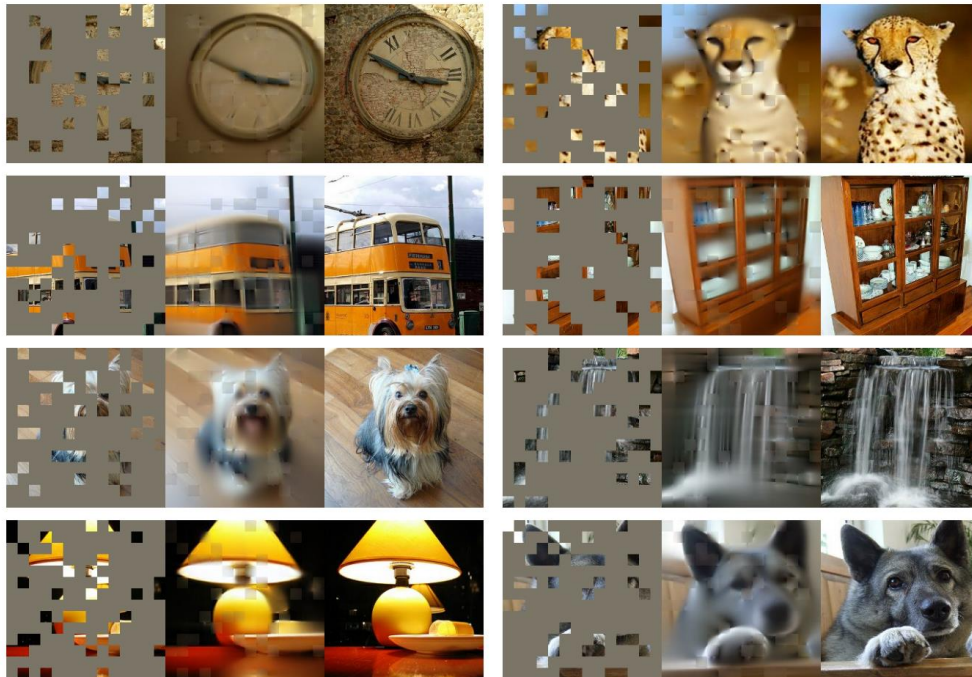
- Masked Image Modelling (MIM)



Reconstruct the image
→ L2 loss on the
masked patches

Masked Auto-encoder (MAE)

- Visualized reconstructions



Masked Auto-encoder (MAE)

- By inputting the unmasked patches into the ViT only, MAEs require less computes and training time
- Masked Image Modelling enforces the ViT encoder to learn the local and global context of the input images

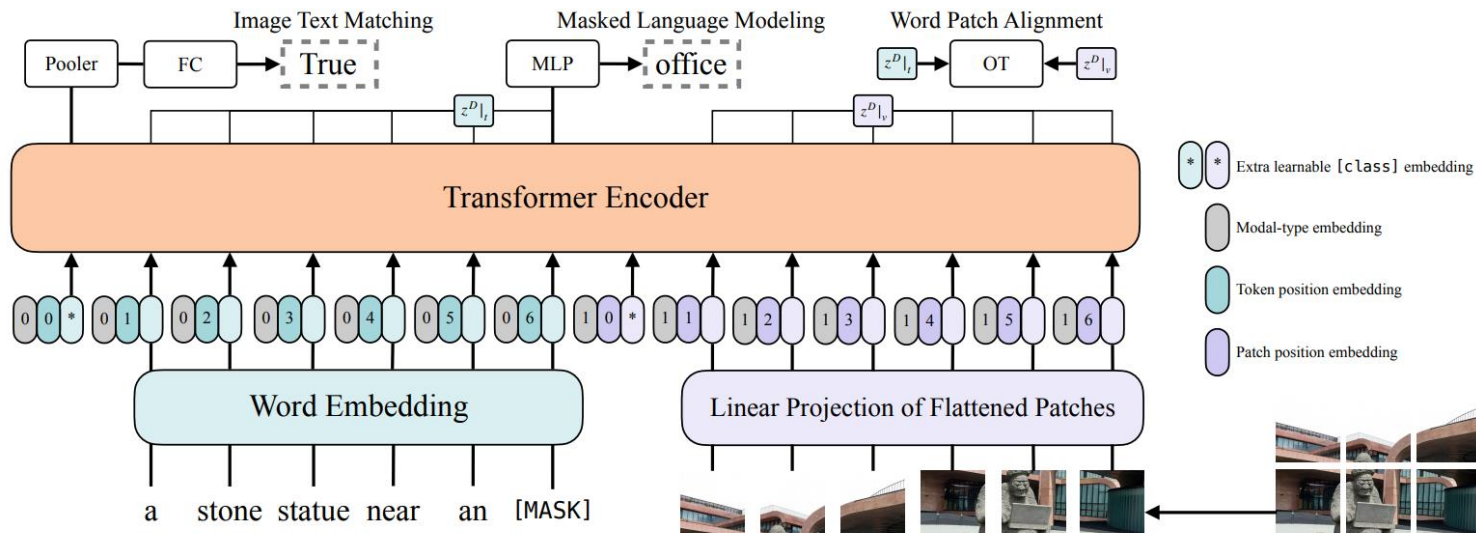
Transformers in Multimodal Learning

Vision-Language Transformers

- ViLT: **V**ision-and-**L**anguage **T**ransformer Without Convolution or Region Supervision
 - Concatenate image patches with text sequence as the input sequence
 - Pre-training with two self-supervised objectives
 - Image text matching
 - Masked language modelling

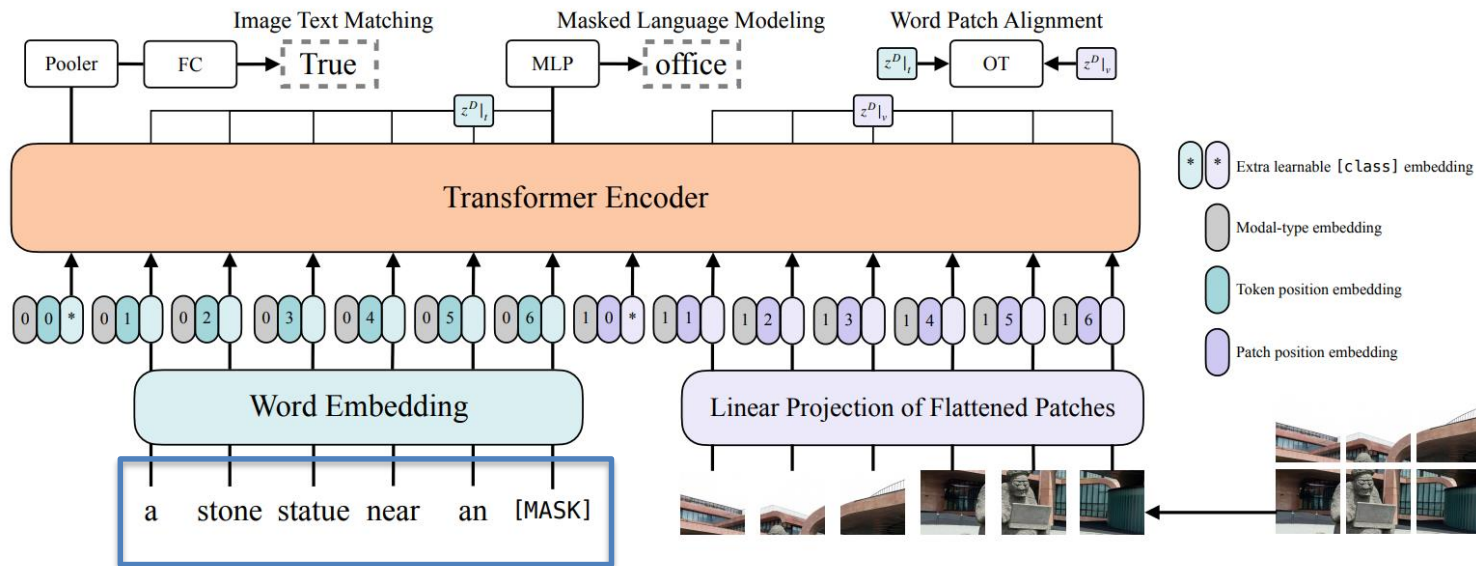
Vision-Language Transformers

- ViLT under the hood



Vision-Language Transformers

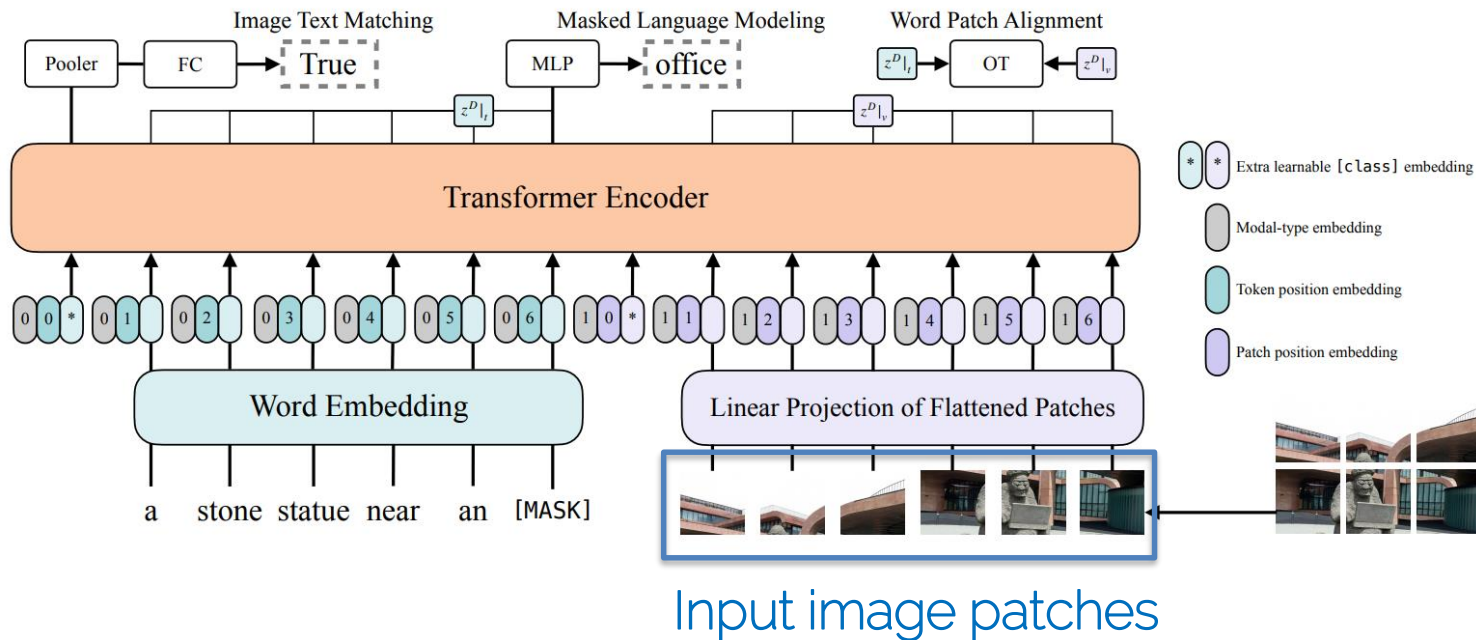
- ViLT under the hood



Masked input texts

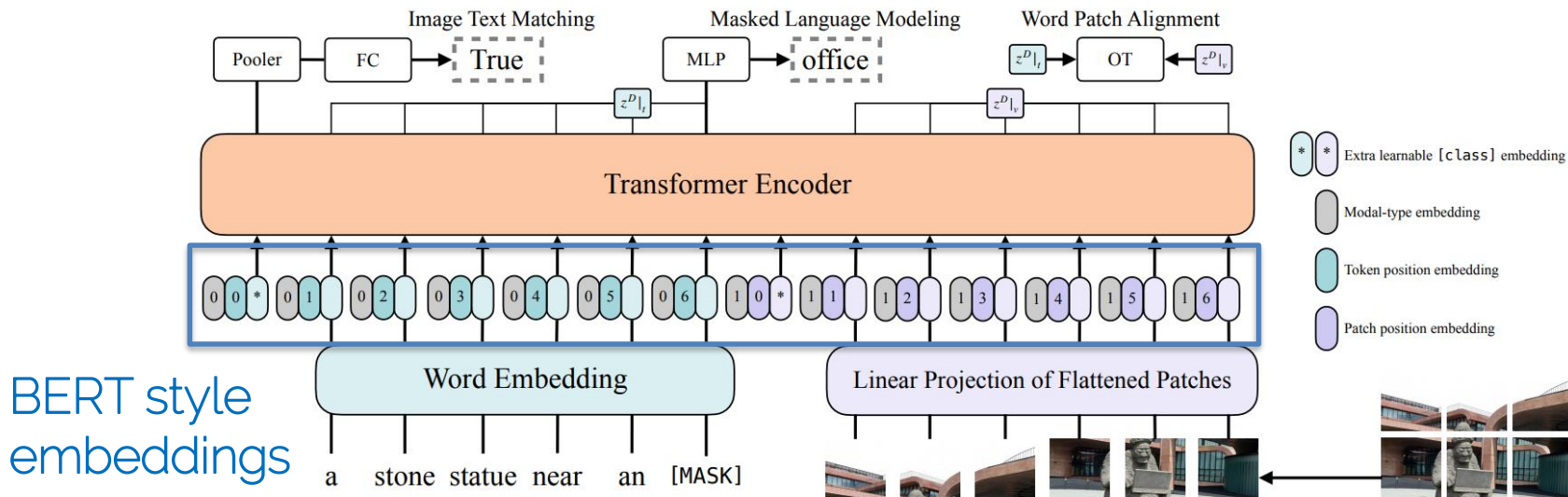
Vision-Language Transformers

- ViLT under the hood



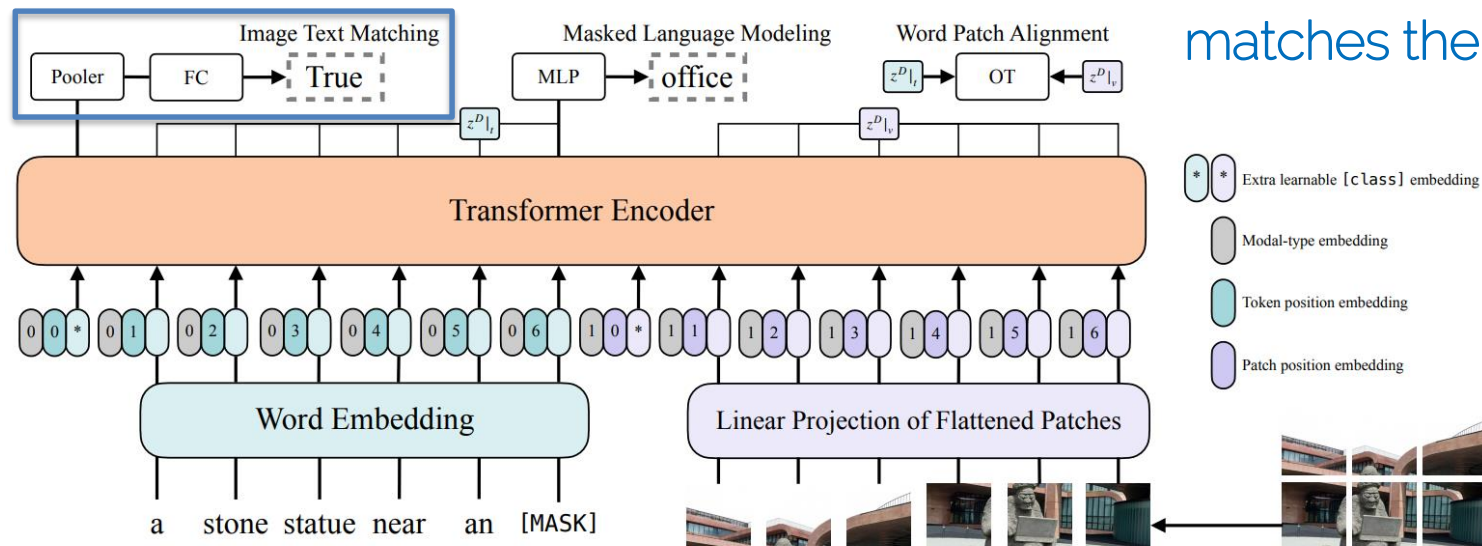
Vision-Language Transformers

- ViLT under the hood



Vision-Language Transformers

- ViLT under the hood

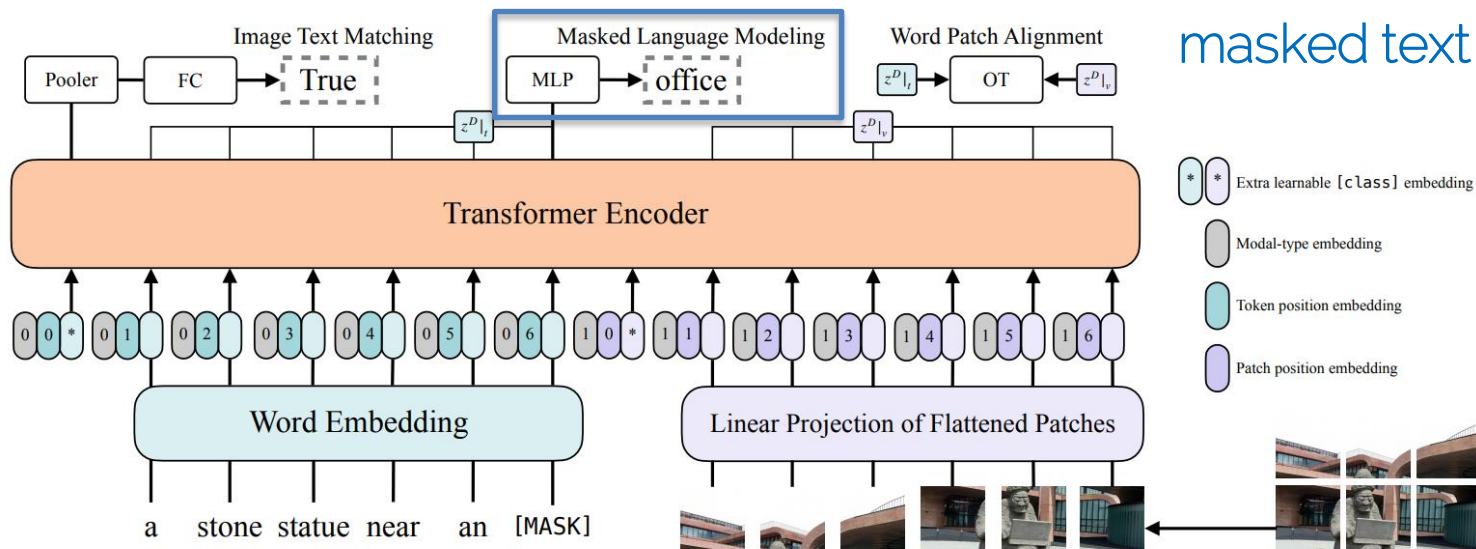


Predict if the image matches the text

(Similar to BERT's next sentence prediction)

Vision-Language Transformers

- ViLT under the hood



(Similar to BERT's masked language modelling)

Vision-Language Transformers

- ViLT is pre-trained on large-scale vision-language datasets in a self-supervised manner.
- Without any image region information (object detection bounding boxes), ViLT achieves strong performance in downstream tasks, e.g., text-to-image retrieval and visual question answering, by using the image patches only.
- ViLT is simple to implement and fast to train.

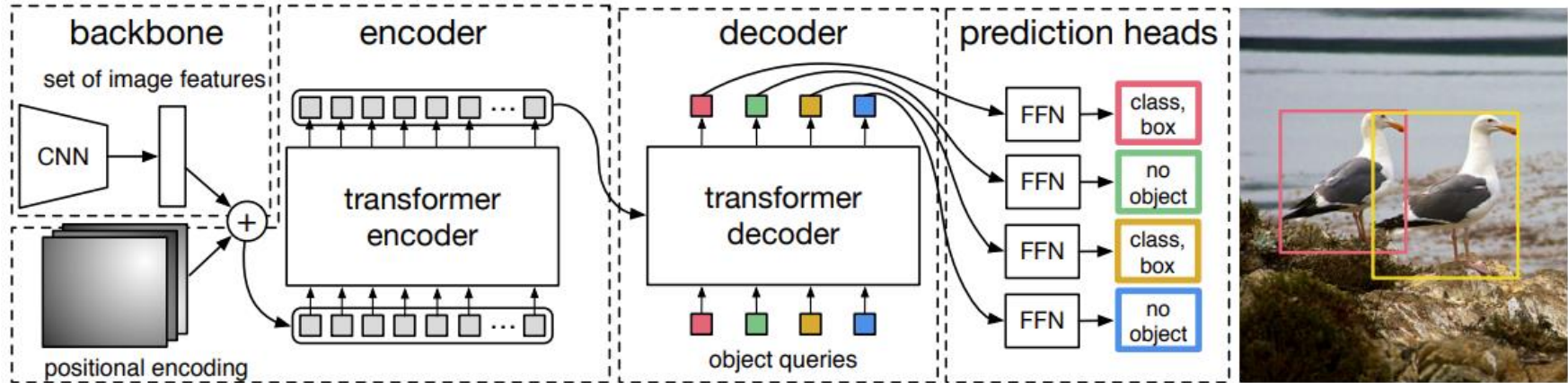
Transformers in Object Detection

DETR

- **D**etection **T**ransformer
 - A set-based object detection using a Transformer on top of a CNN backbone
 - A Transformer encoder encodes the CNN features with positional embeddings
 - A Transformer decoder “translates” the input positional embeddings (object queries) into object detections or “no object”

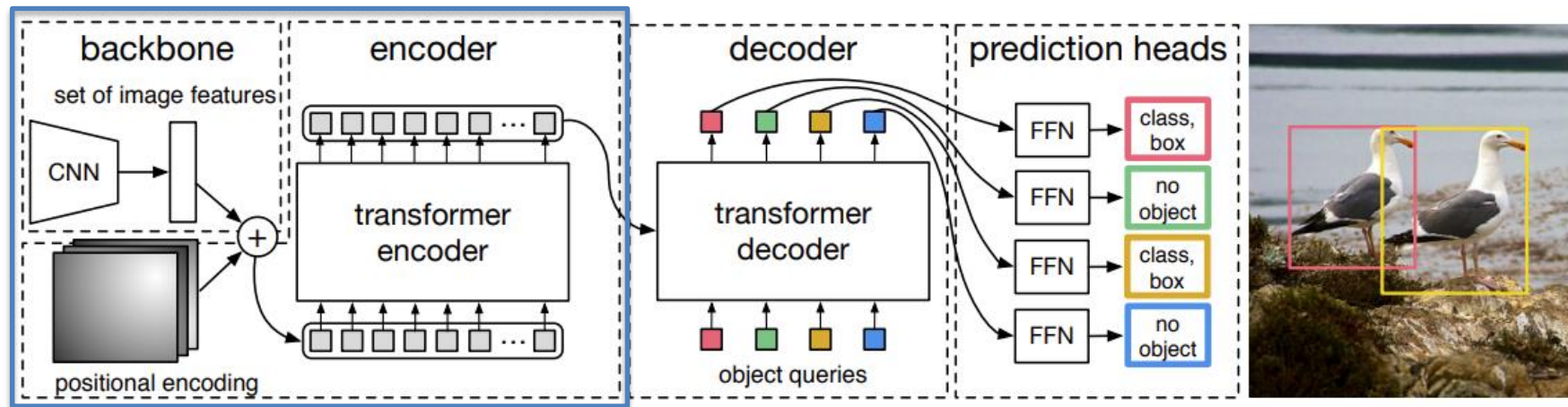
DETR

- DETR works in a similar way as Machine Translation



DETR

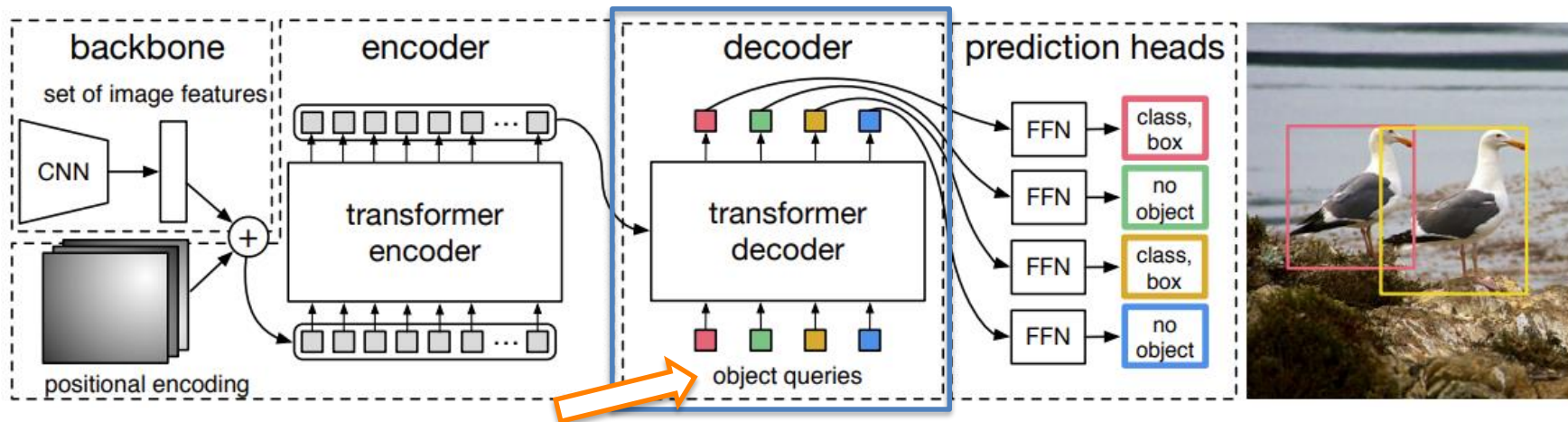
- DETR works in a similar way as Machine Translation



Encode the input image into a sequence of image tokens (similar to ViT's encoding process)

DETR

- DETR works in a similar way as Machine Translation

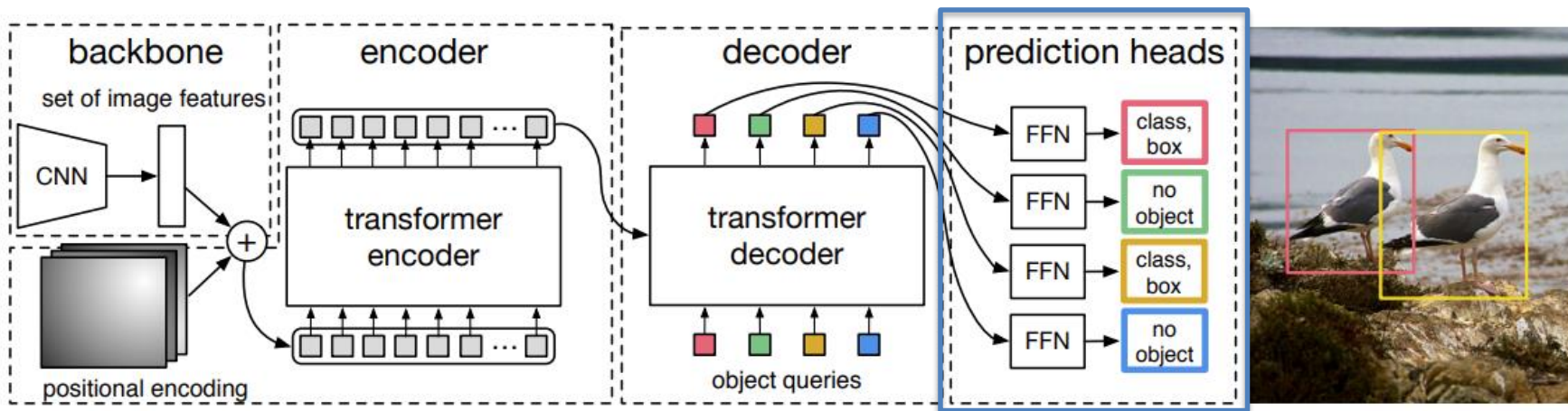


New positional embeddings
as object queries

decode the object queries into output
embeddings, while referring to the
encoded image tokens

DETR

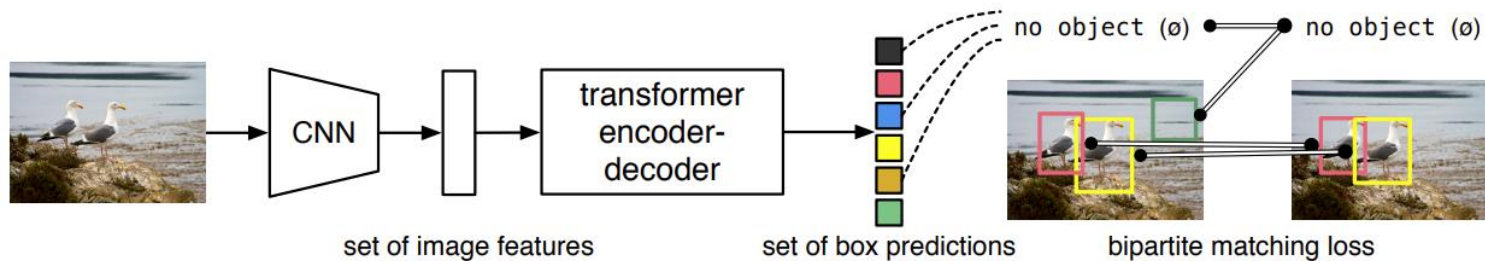
- DETR works in a similar way as Machine Translation



An output layer predicts the object classes and bounding boxes, or "no object"

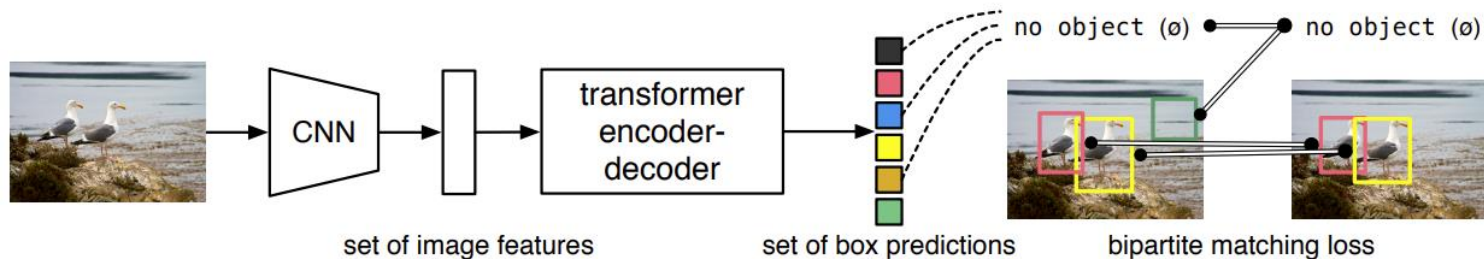
DETR

- Object detection as set prediction
 - Each box prediction matches with only **ONE** ground truth (one-to-one assignment), i.e., only match the best prediction out of duplicated ones to the GT, the others will be assigned to “no object”



DETR

- Object detection as set prediction
 - Bipartite matching with pair-wise matching cost



L1 loss and generalized IoU loss

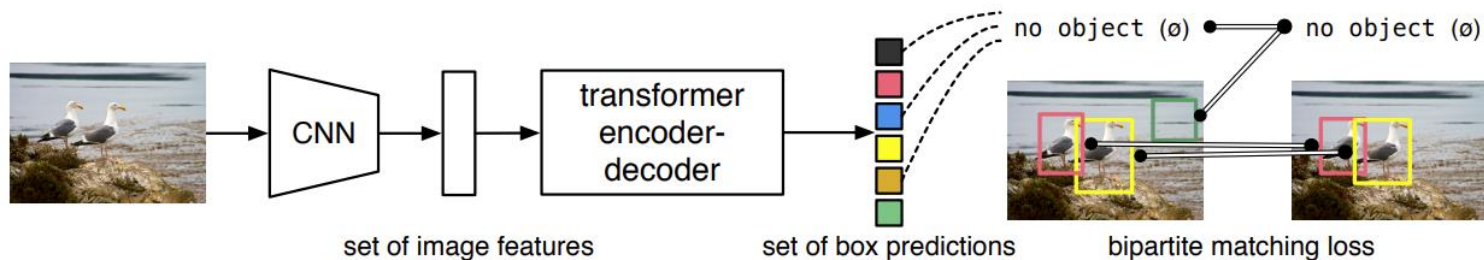
$$\hat{\sigma} = \arg \min_{\sigma \in \mathfrak{S}_N} \sum_i^N \boxed{-\mathbb{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i)} + \boxed{\mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})}$$

Find prediction-GT assignments
that minimize this cost

Predicted class probability
("no object" class excluded)

DETR

- Object detection as set prediction
 - Bipartite matching with pair-wise matching cost

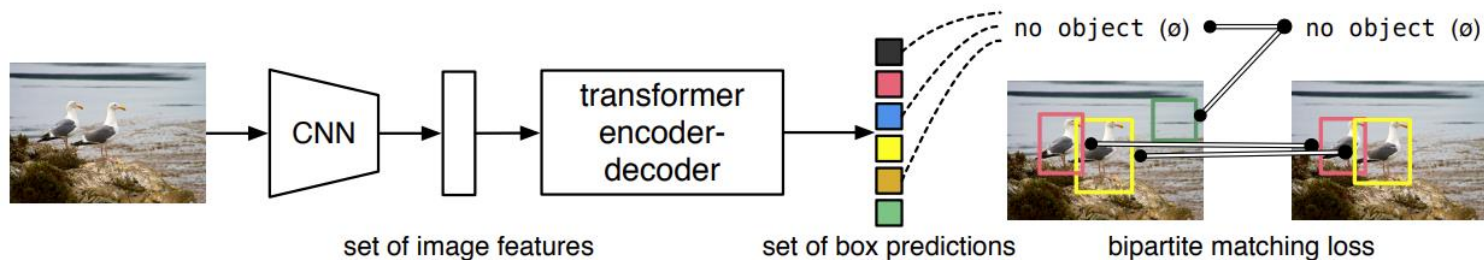


Matching via [Hungarian Algorithm](#), non-matched predictions will be assigned to "no object" class

$$\hat{\sigma} = \arg \min_{\sigma \in \mathfrak{S}_N} \sum_i^N -\mathbb{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$$

DETR

- Object detection as set prediction
 - Hungarian loss (after bipartite matching)



L1 loss and generalized IoU loss

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^N \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}(i)}) \right]$$

Predicted class probability
(including the "no object" class)

DETR

- DETR works in a similar way as Neural Machine Translation.
- The bipartite matching enforces DETR to predict non-overlapping bounding boxes; DETR does not need non-maximum suppression (NMS) by design.
- Can be easily extended to panoptic segmentation by adding a mask head on top of the decoder outputs.

Reading Homework

- Understanding BERT
 - <https://jalammr.github.io/illustrated-bert/>
- BEiT: [Bao et al. 2021] BEiT: BERT Pre-Training of Image Transformers
 - <https://arxiv.org/pdf/2106.08254.pdf>
- Pix2Seq: [Chen et al. 2021] Pix2seq: A Language Modeling Framework for Object Detection
 - <https://arxiv.org/pdf/2109.10852.pdf>

Literature

- Attention: [Xu. 2015] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention
 - <https://arxiv.org/pdf/1502.03044.pdf>
- Transformer: [Vaswani et al. 2017] Attention is All You Need
 - <https://arxiv.org/pdf/1706.03762.pdf>
- BERT: [Devlin et al. 2020] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
 - <https://arxiv.org/pdf/1810.04805.pdf>
- ViT: [Dosovitskiy et al. 2020] An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
 - <https://arxiv.org/pdf/2010.11929.pdf>
- MAE: [He et al. 2022] Masked Autoencoders Are Scalable Vision Learners
 - <https://arxiv.org/pdf/2111.06377.pdf>
- ViLT: [Kim et al. 2019] ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision
 - <https://arxiv.org/pdf/2102.03334.pdf>
- DETR: [Carion et al. 2022] End-to-End Object Detection with Transformers
 - <https://arxiv.org/pdf/2005.12872.pdf>

Thanks for watching!