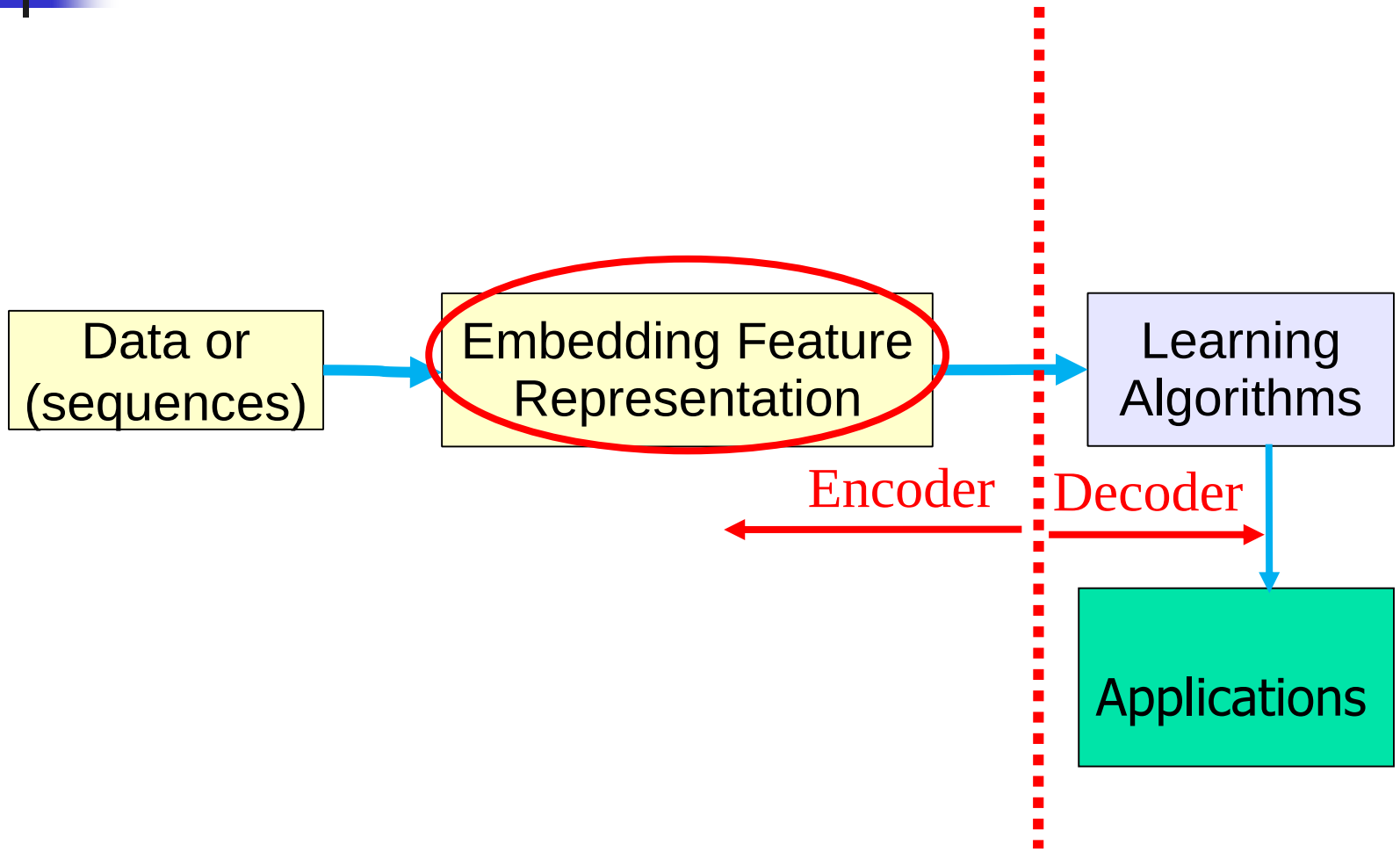
Transformers and Visual Applications

Jenq-Neng Hwang, Professor
Department of Electrical & Computer Engineering
University of Washington, Seattle WA
hwang@uw.edu

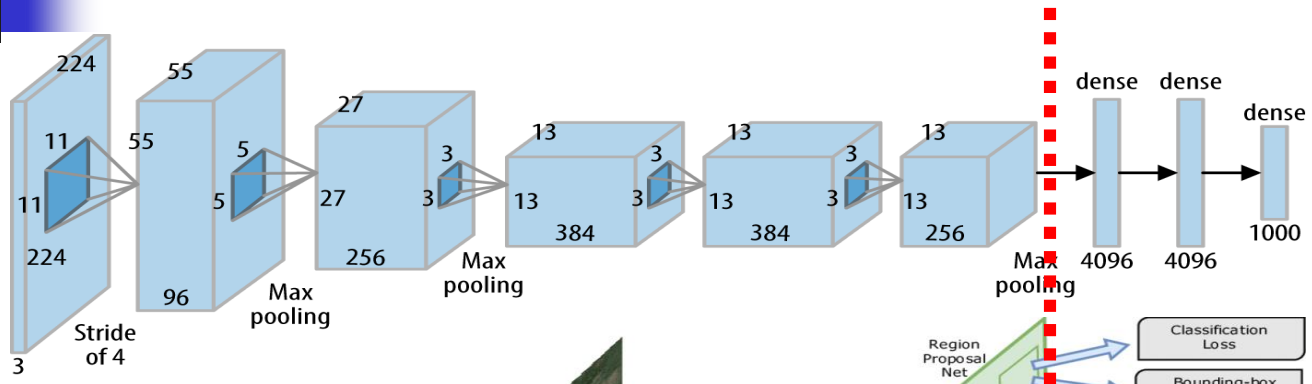
Transformer, Self-Attention and Applications

- [Transformer]: Ashish Vaswani, et al., "Attention Is All You Need," NIPS 2017, Google, https://arxiv.org/abs/1706.03762
- [BERT]: Jacob Devlin, et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," Google, https://arxiv.org/abs/1810.04805
- [ViT]: Alexey Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," Google, https://arxiv.org/pdf/2010.11929.pdf
- [DETR]: Nicolas Carion, et al., "End-to-End Object Detection with Transformers," FaceBook AI, https://arxiv.org/abs/2005.12872
- [CLIP]: Alec Radford, et al., "Learning Transferable Visual Models From Natural Language Supervision," https://arxiv.org/pdf/2103.00020.pdf

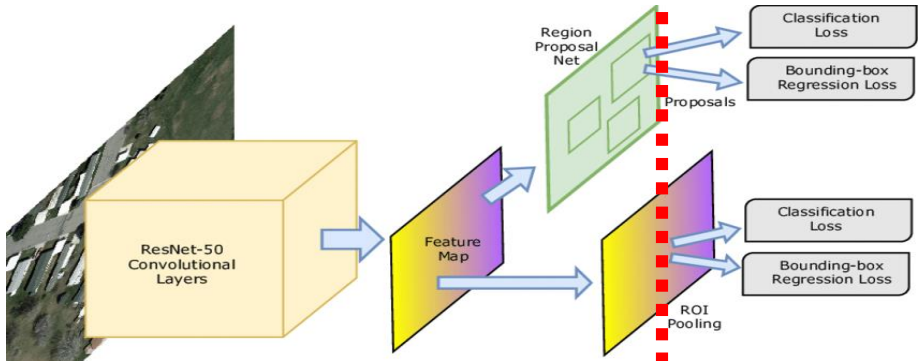
Machine Learning Tasks



Encoder and Decoder



Classification

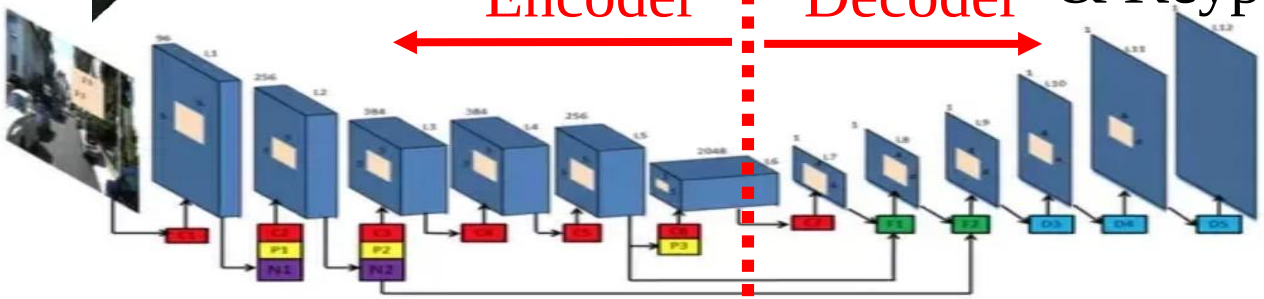


Detection

Segmentation & Keypoint

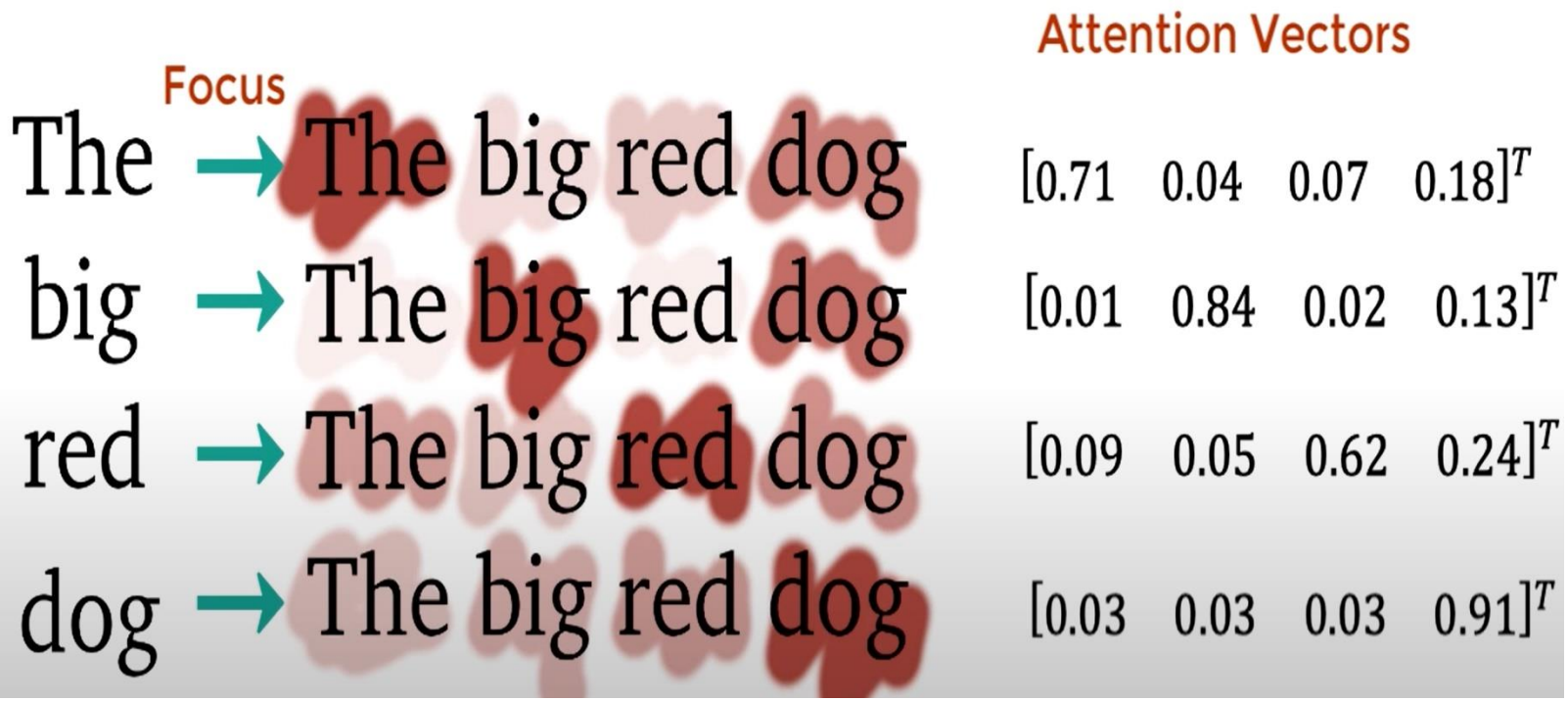
Encoder

Decoder

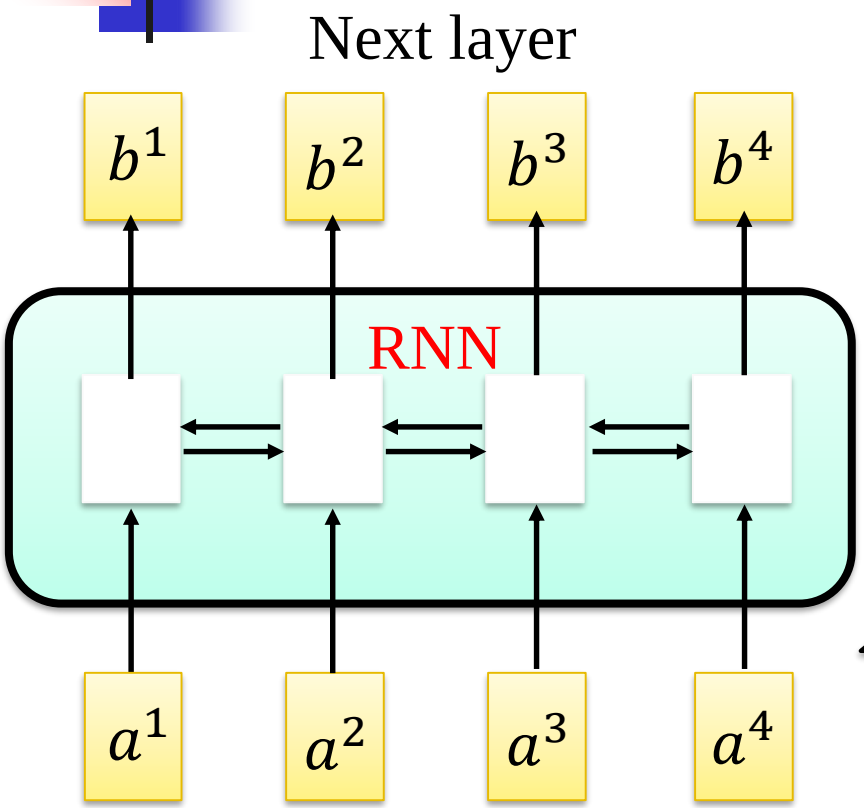


Attention in Natural Language Processing (NLP)

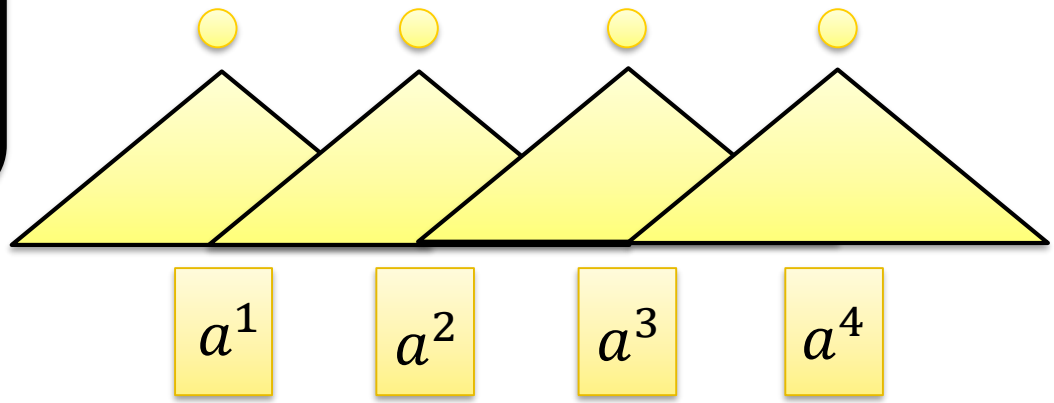
Attention : What part of the input should we focus?



Embedding Feature Extraction: RNN or CNN



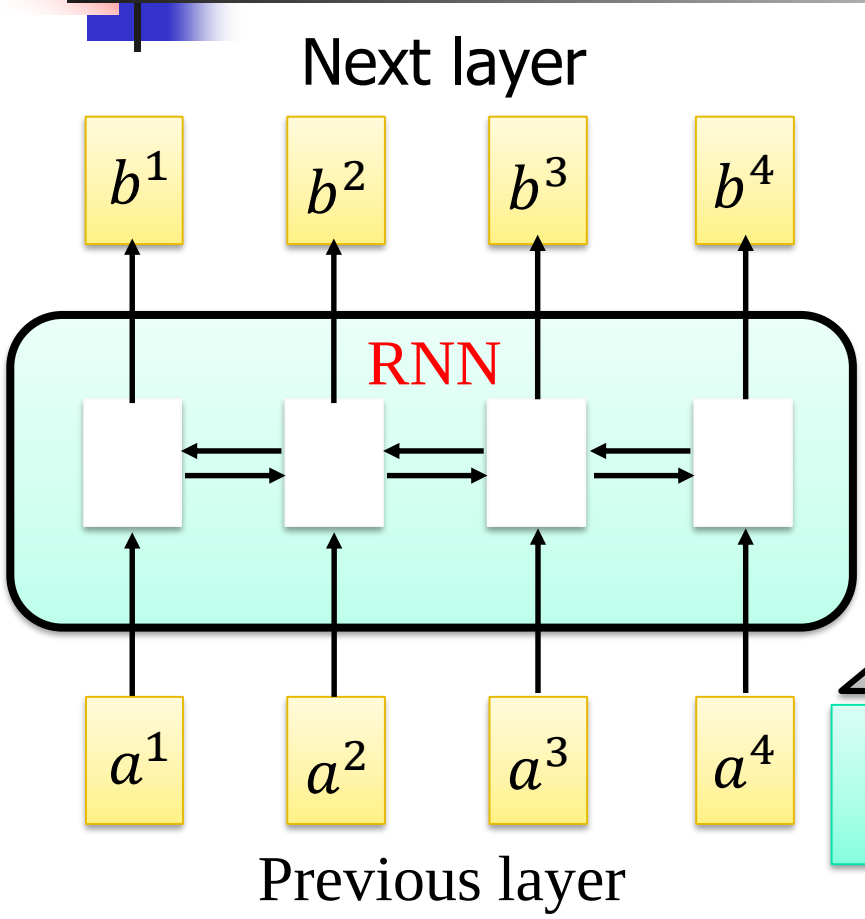
Hard to parallel !



Using CNN to replace RNN

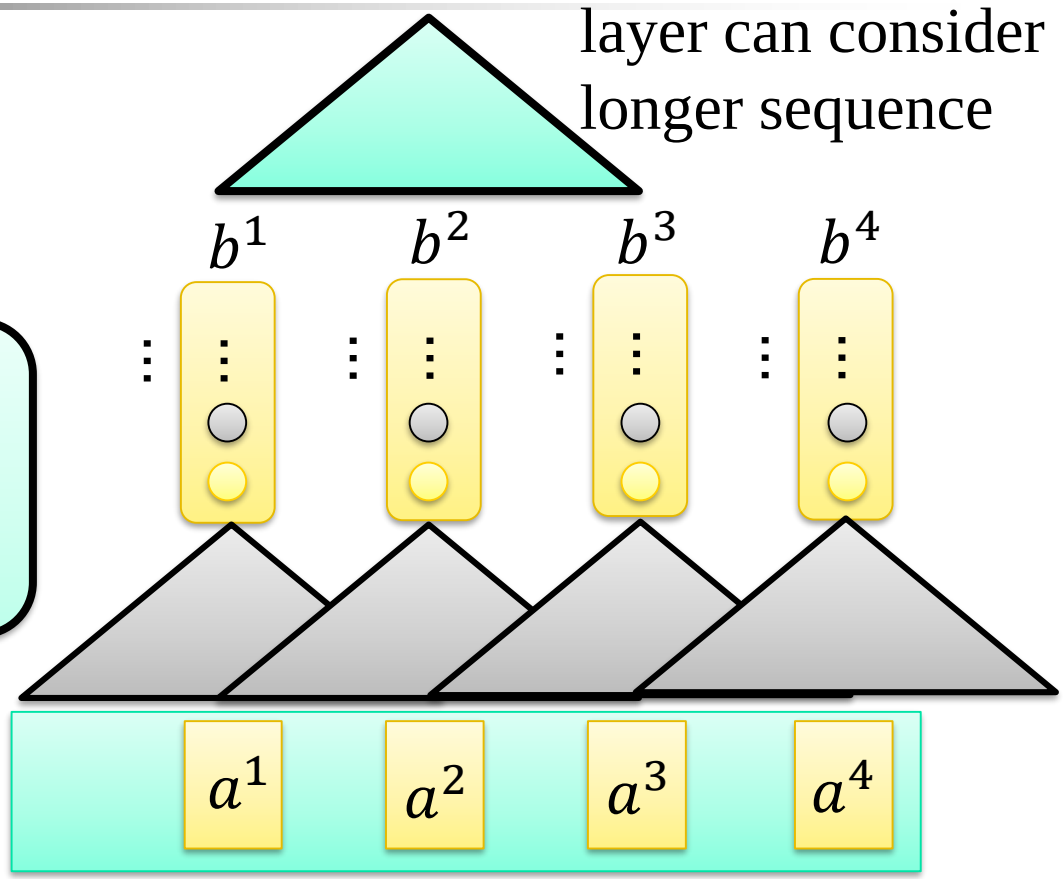
This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html
https://www.youtube.com/watch?v=ugWDIIOHtPA&t=2470s

Seq2Seq Transformation



Hard to parallel

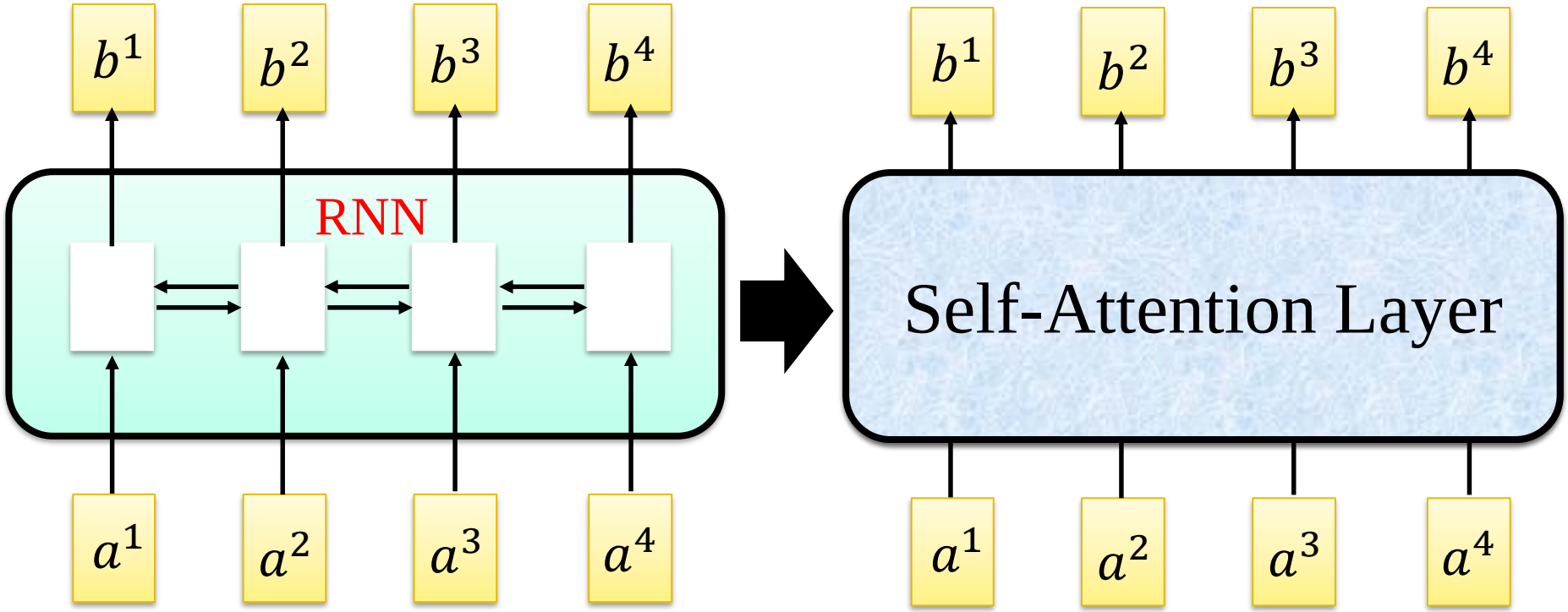
Filters in higher layer can consider longer sequence



Using CNN to replace RNN
(CNN can parallel, weights sharing)

Self-Attention for Seq2Seq

- b^i is obtained based on the whole input sequence.
- b^1, b^2, b^3, b^4 can be parallelly computed.

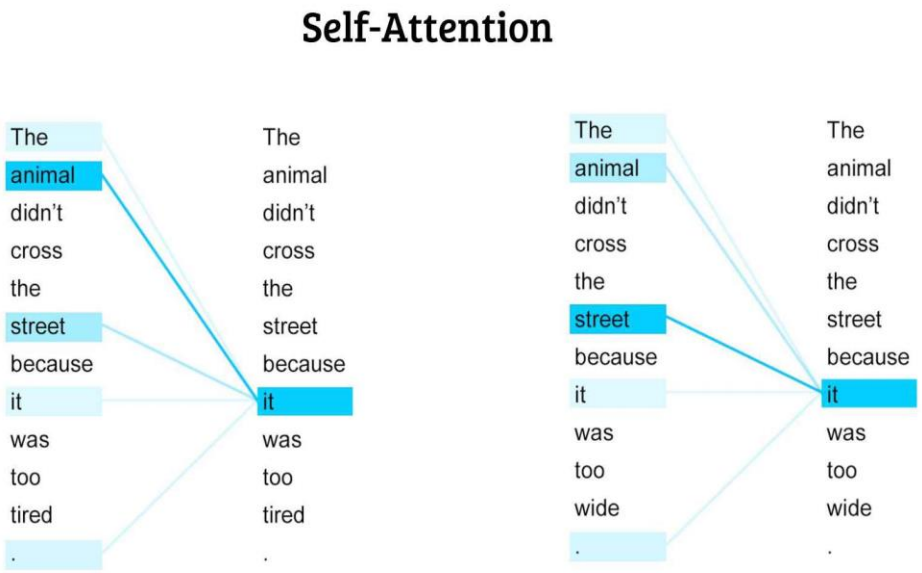


- Ashish Vaswani, et al., "Attention Is All You Need," NIPS 2017, Google

Why Self-Attention?

Self-attention: allows the input to interact with each other ("self") and find out who they should pay more attention.

- All tokens can attend to each other equally well, no matter how far they are from each other
- Transformer captures long-range semantic dependencies
  - Creates global context into the output embeddings



e.g, 1. *"The **animal** didn't cross the street because **it** was too tired"*

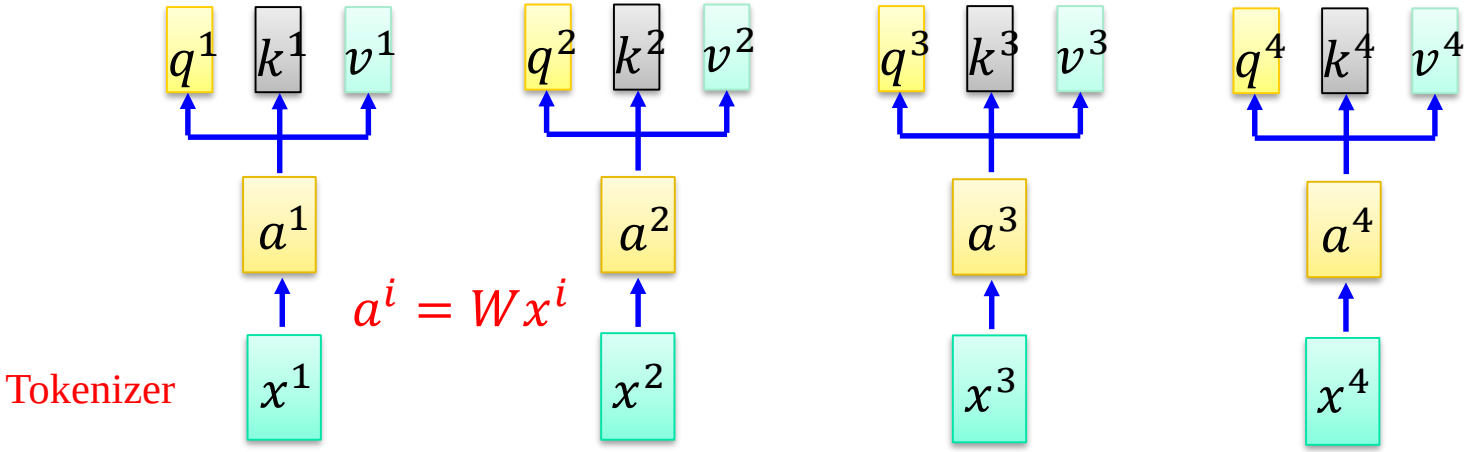
*e.g, 2. "The animal didn't cross the **street** because **it** was too wide"*

Query, Key and Value of A Transformer

$\boldsymbol{q}$: **query** (to match others) $q^i = W^q a^i$

$\boldsymbol{k}$: **key** (to be matched) $k^i = W^k a^i$

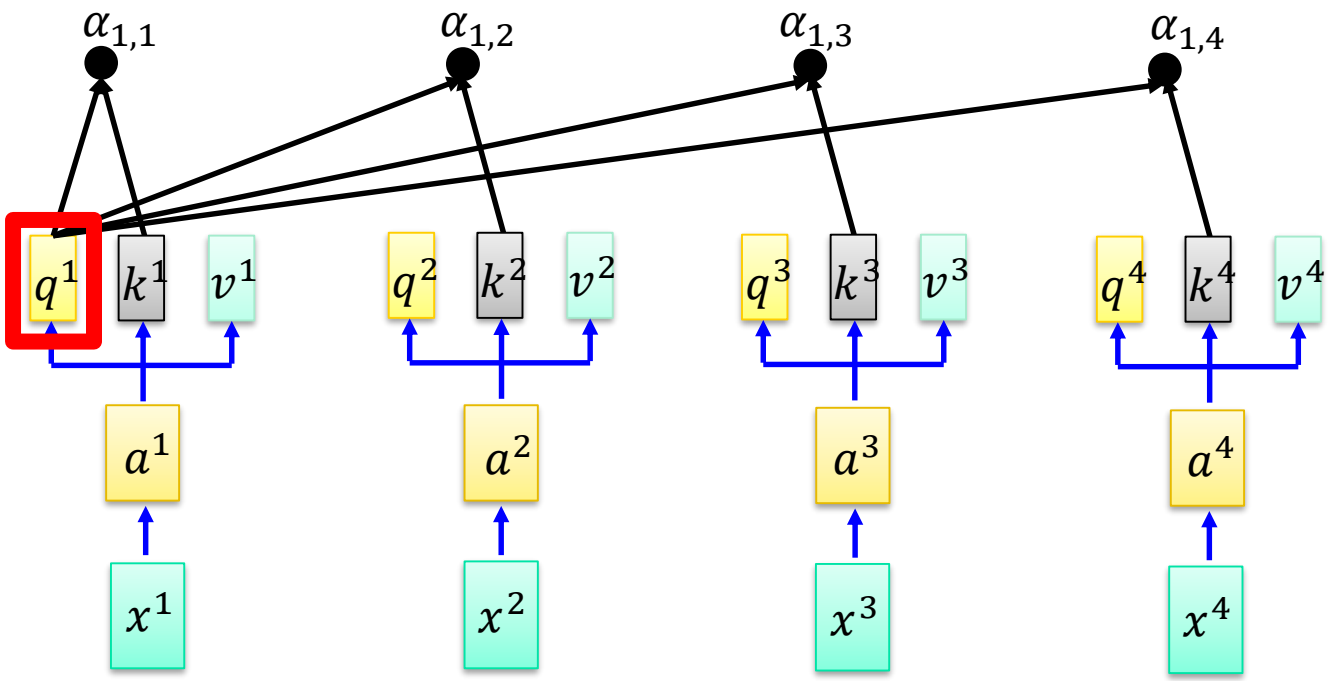
$\boldsymbol{v}$: **value** (information to be extracted) $v^i = W^v a^i$



This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

Correlation of Query & Key

Self-attention



Scaled Dot-Product Attention

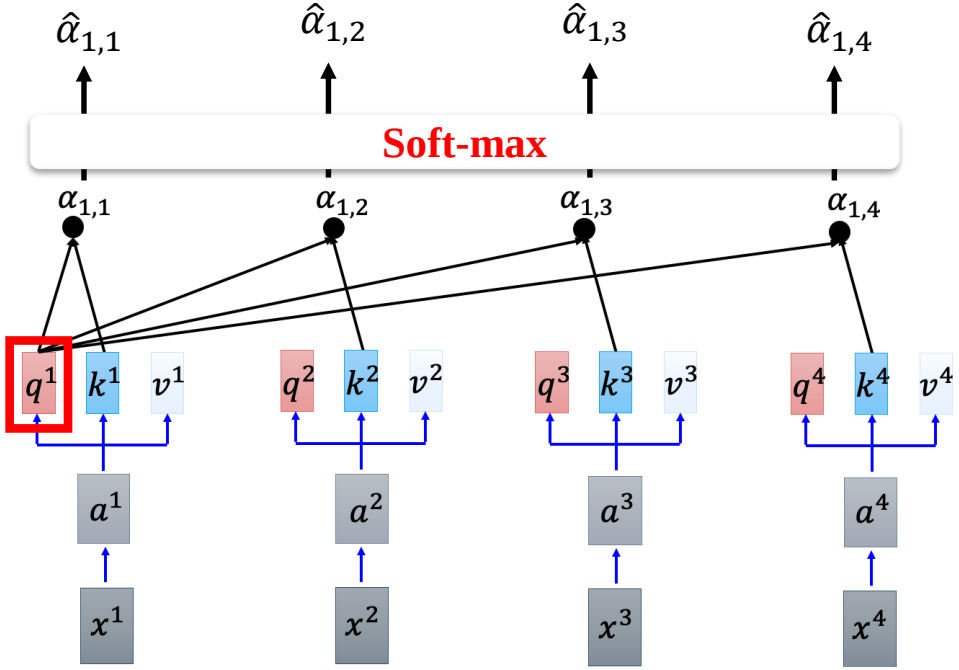
$$\alpha_{1,i} = \underbrace{q^1 \cdot k^i}_{\text{dot product}} / \sqrt{d}$$

d is the dim of q and k

This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

Self-Attention Weights

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j}), \qquad Softmax\ (attention\ weights)$$

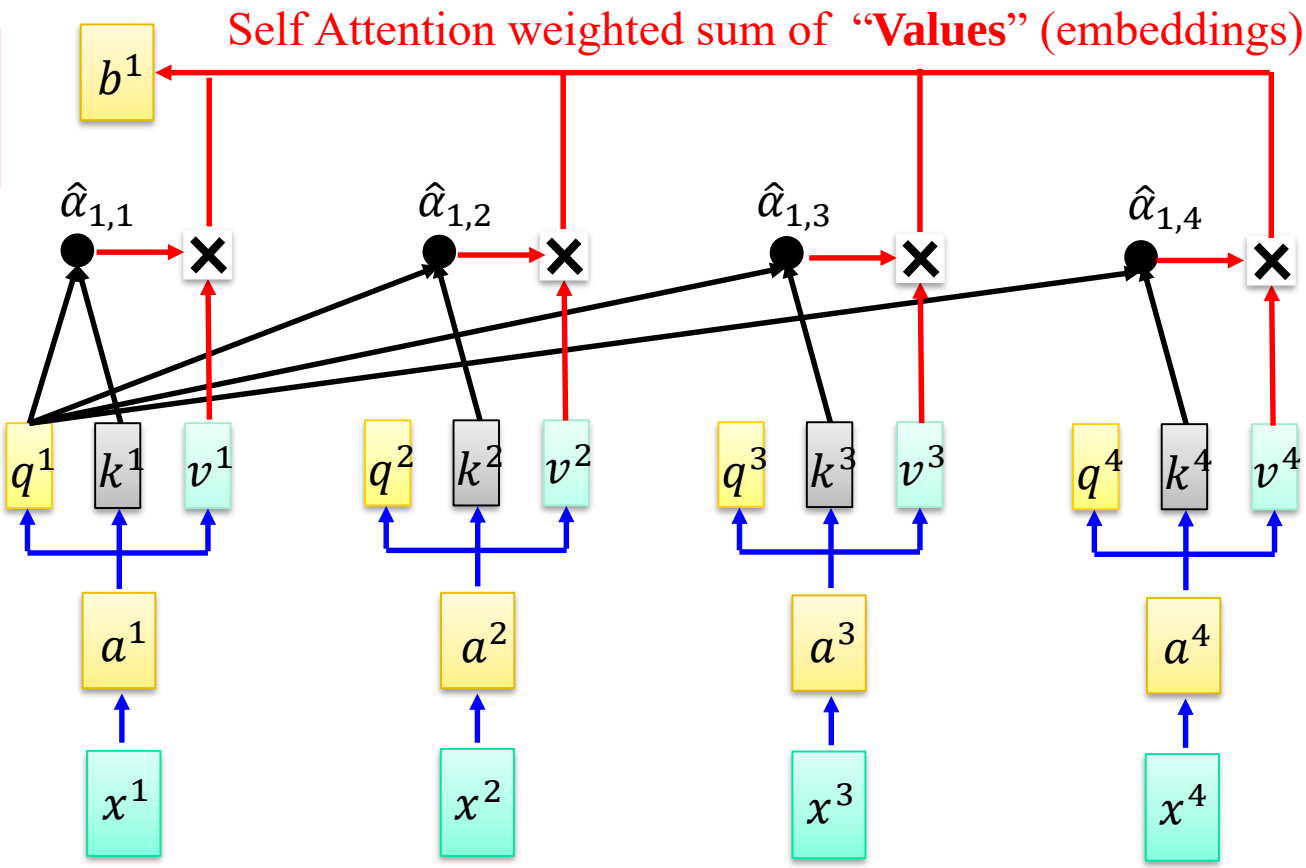


This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

Attention Weighted Outputs

$$b^1 = \sum_i \hat{\alpha}_{1,i} v^i$$

Considering the whole sequence

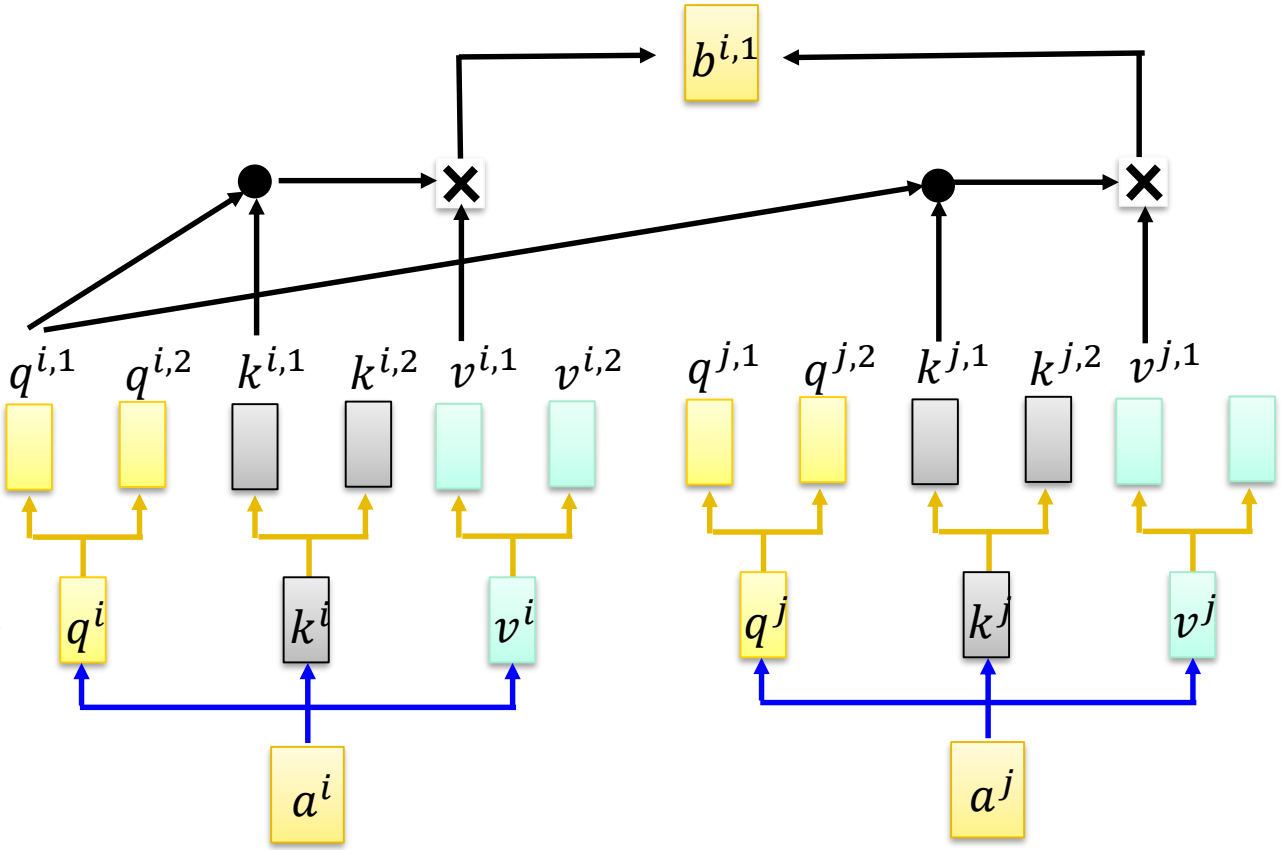


This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

Multi-Head Self Attention

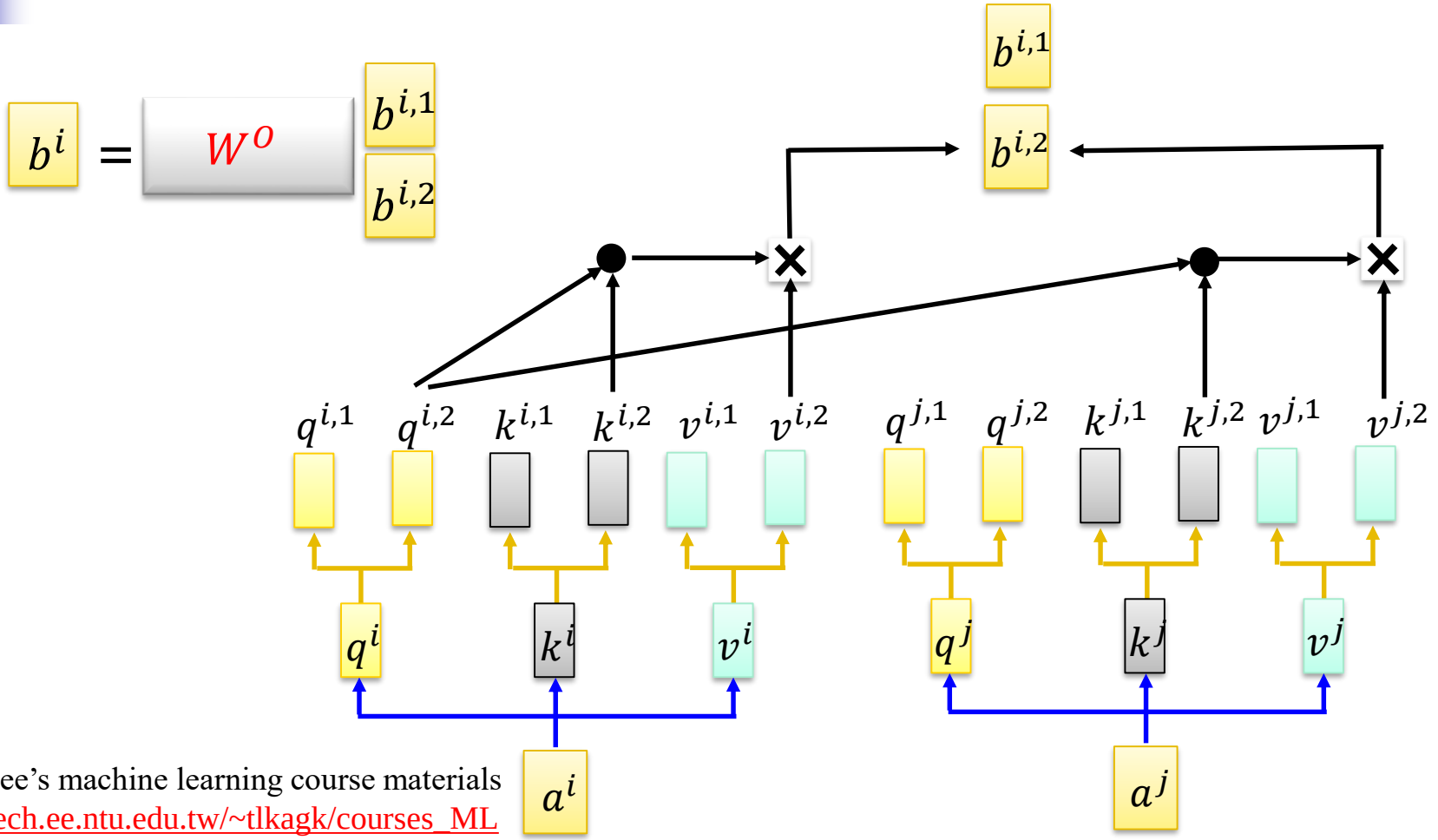
$$q^{i,1} = W^{q,1} q^i$$
$$q^{i,2} = W^{q,2} q^i$$

$$q^i = W^q a^i$$



This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

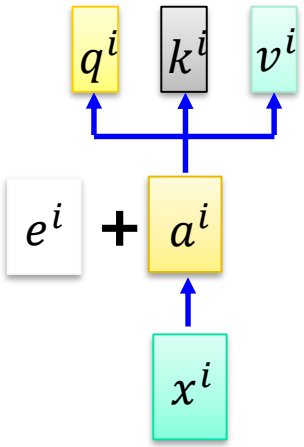
Multi-Head Self Attention



Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

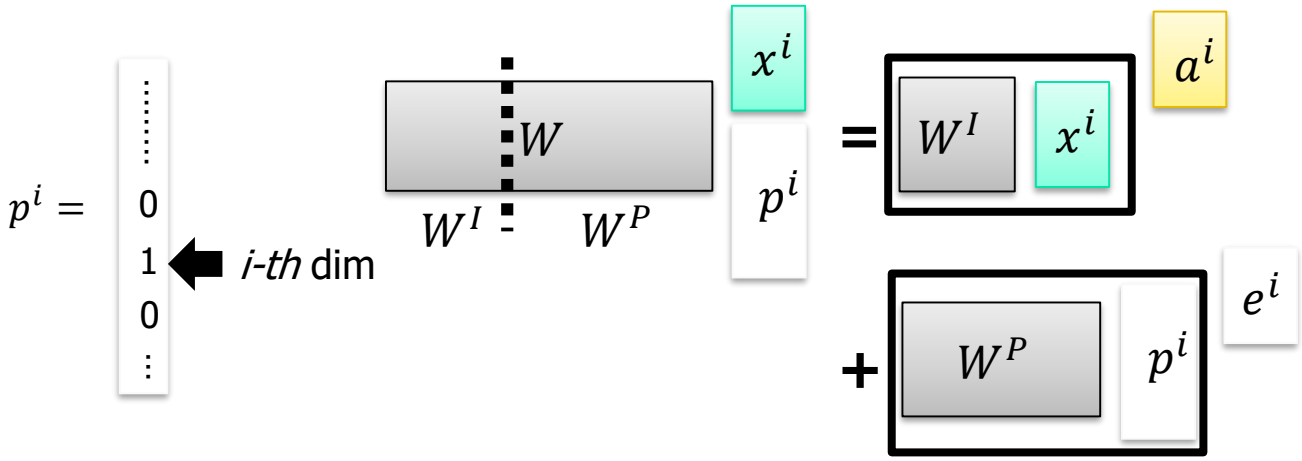
Positional Encoding

- No position information contained in the self-attention
- Positional encoding: each position has a unique positional vector e^i (manually picked, not learned from data)



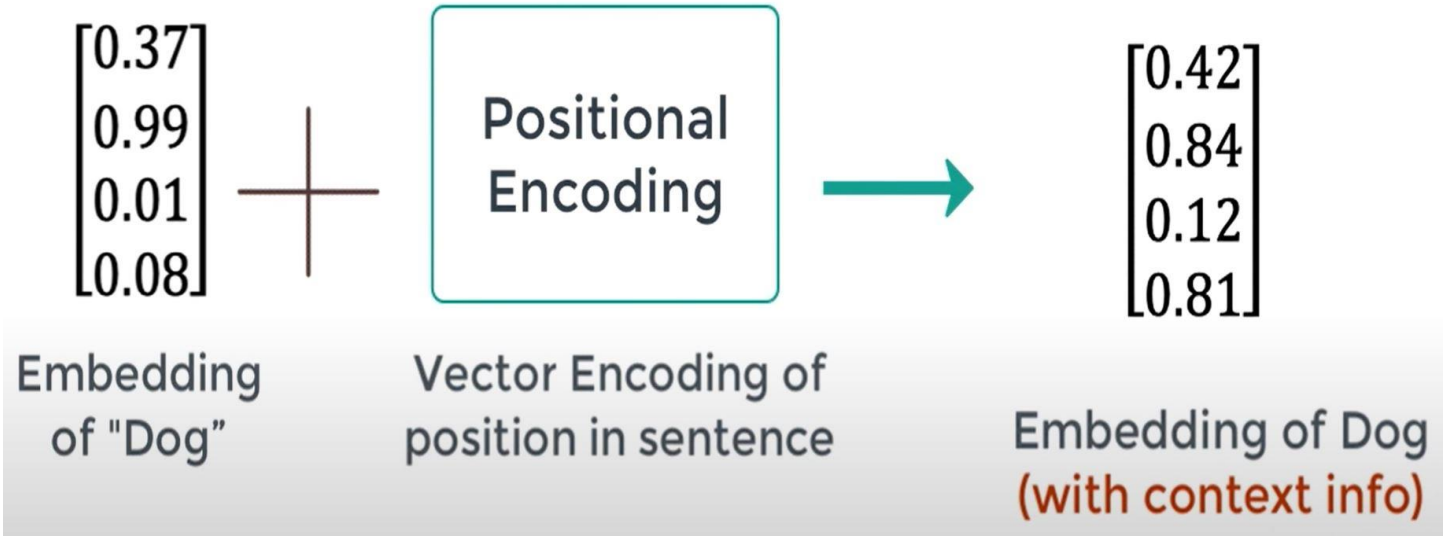
Why e + a instead of concatenation?

each x^i appends a one-hot vector p^i for position



This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

Positional Encoding

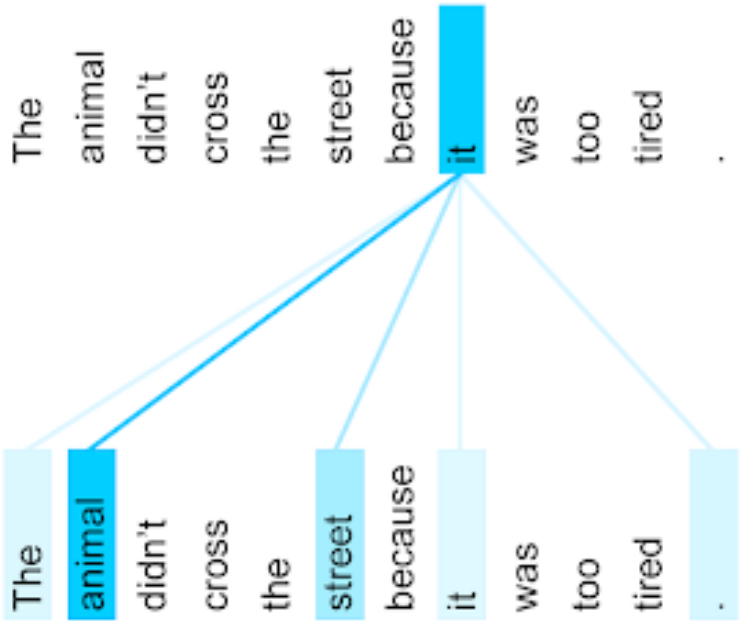


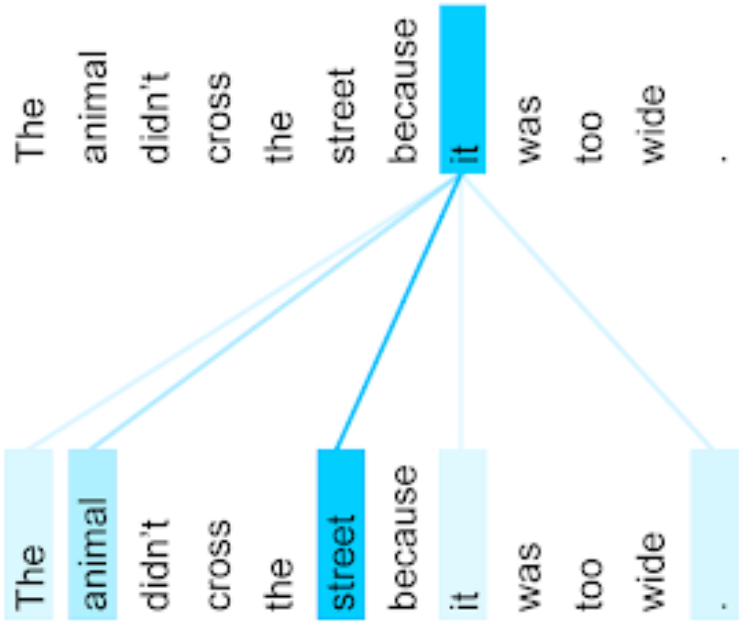
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

D_{model} =dim of inputs (512)

where pos is the position and i is the dimension

Attention Visualization

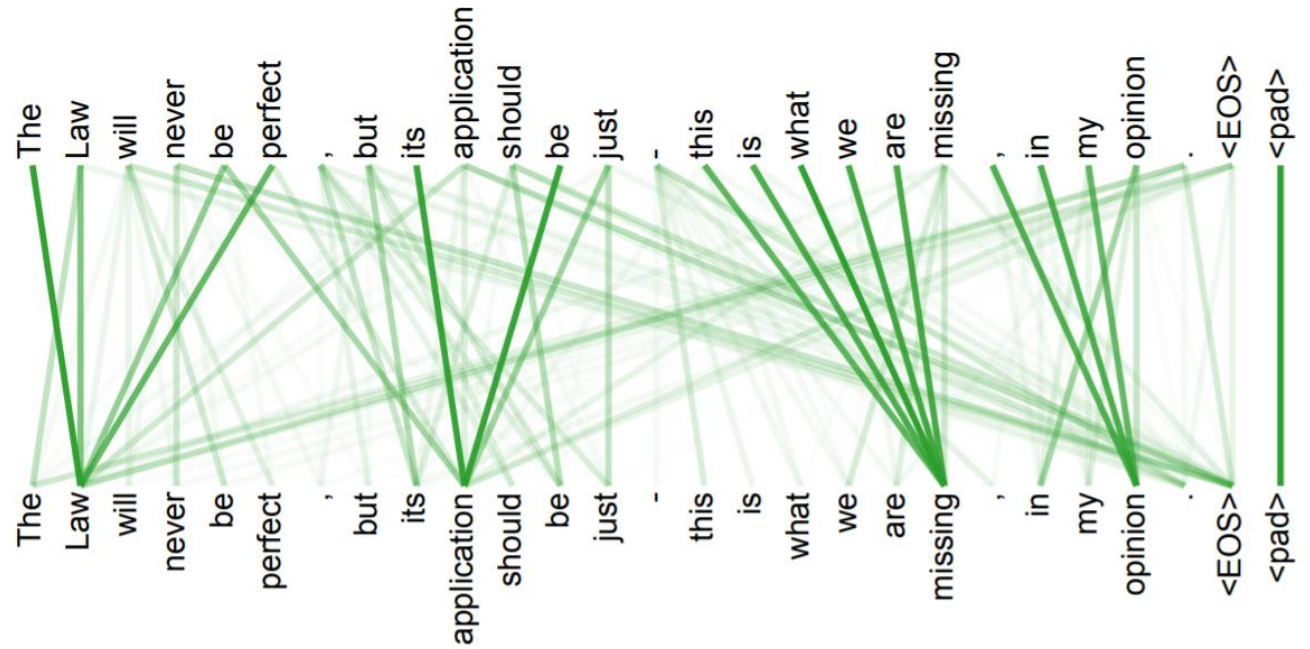


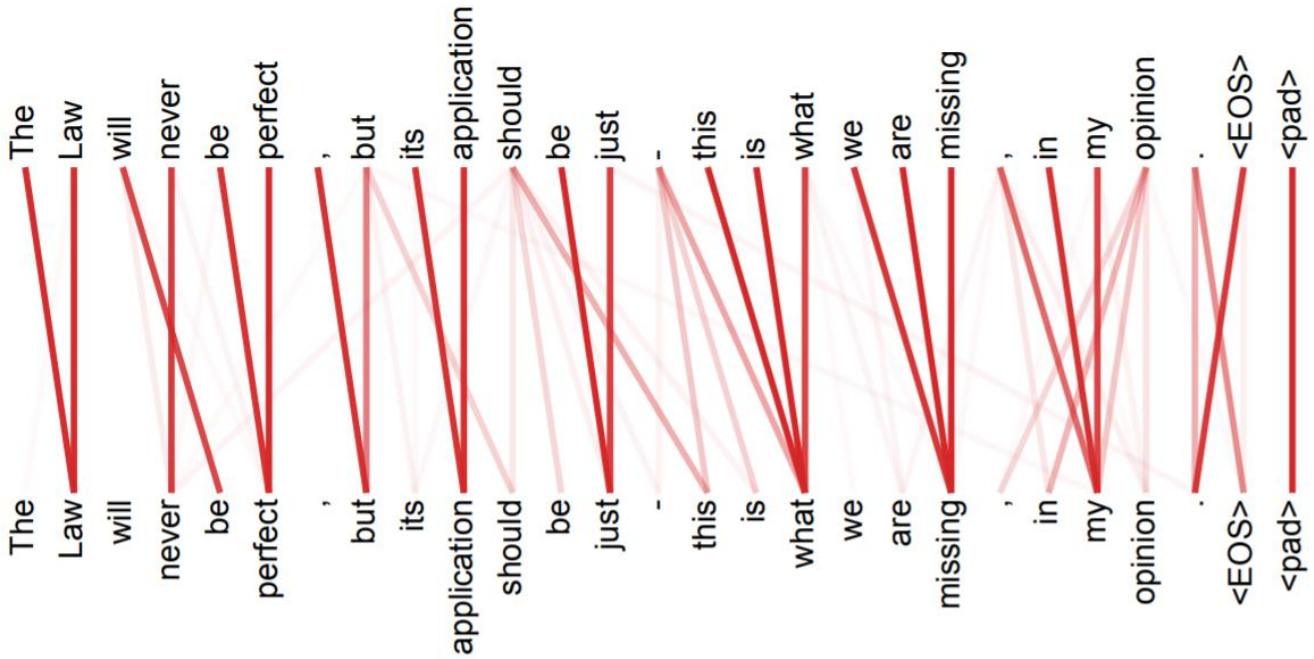


The encoder self-attention distribution for the word "it" from the 5th layer of a Transformer trained on English to French translation (one of eight attention heads).

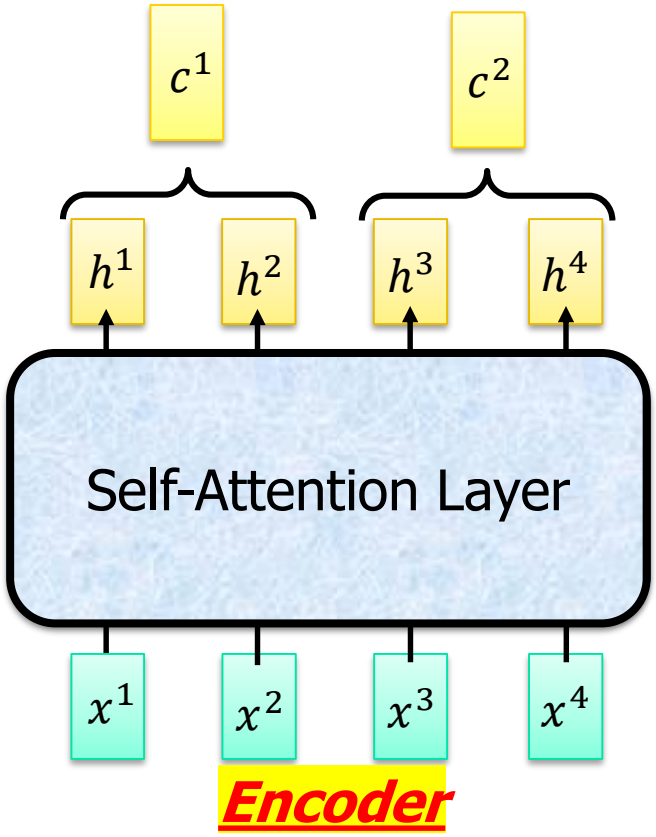
https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Multi-head Attention

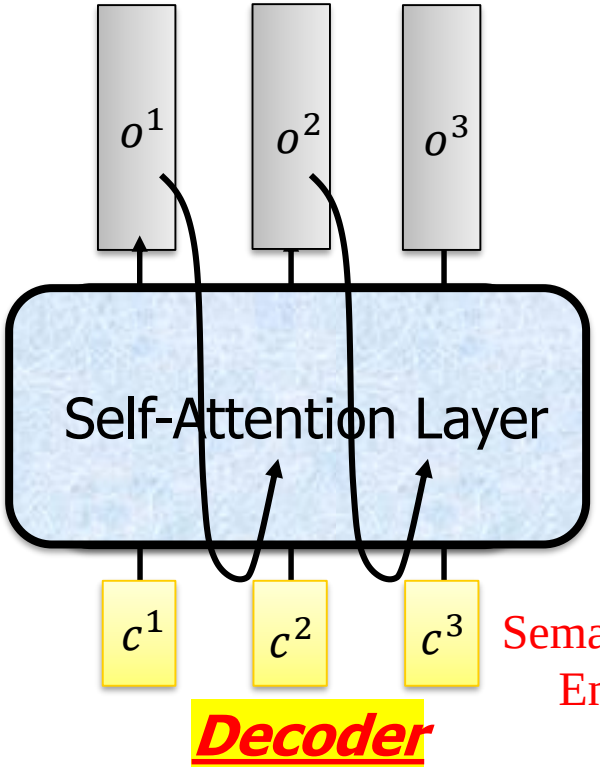




Ready for Seq2Seq Output



Bidirectional RNN

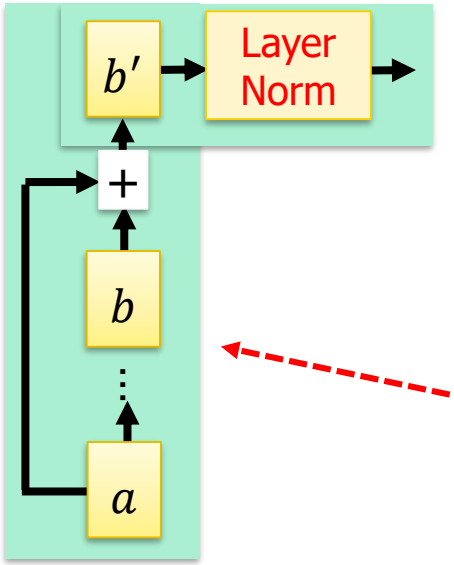


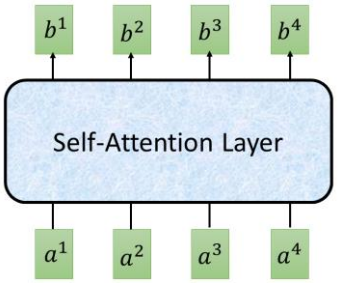
Unidirectional RNN

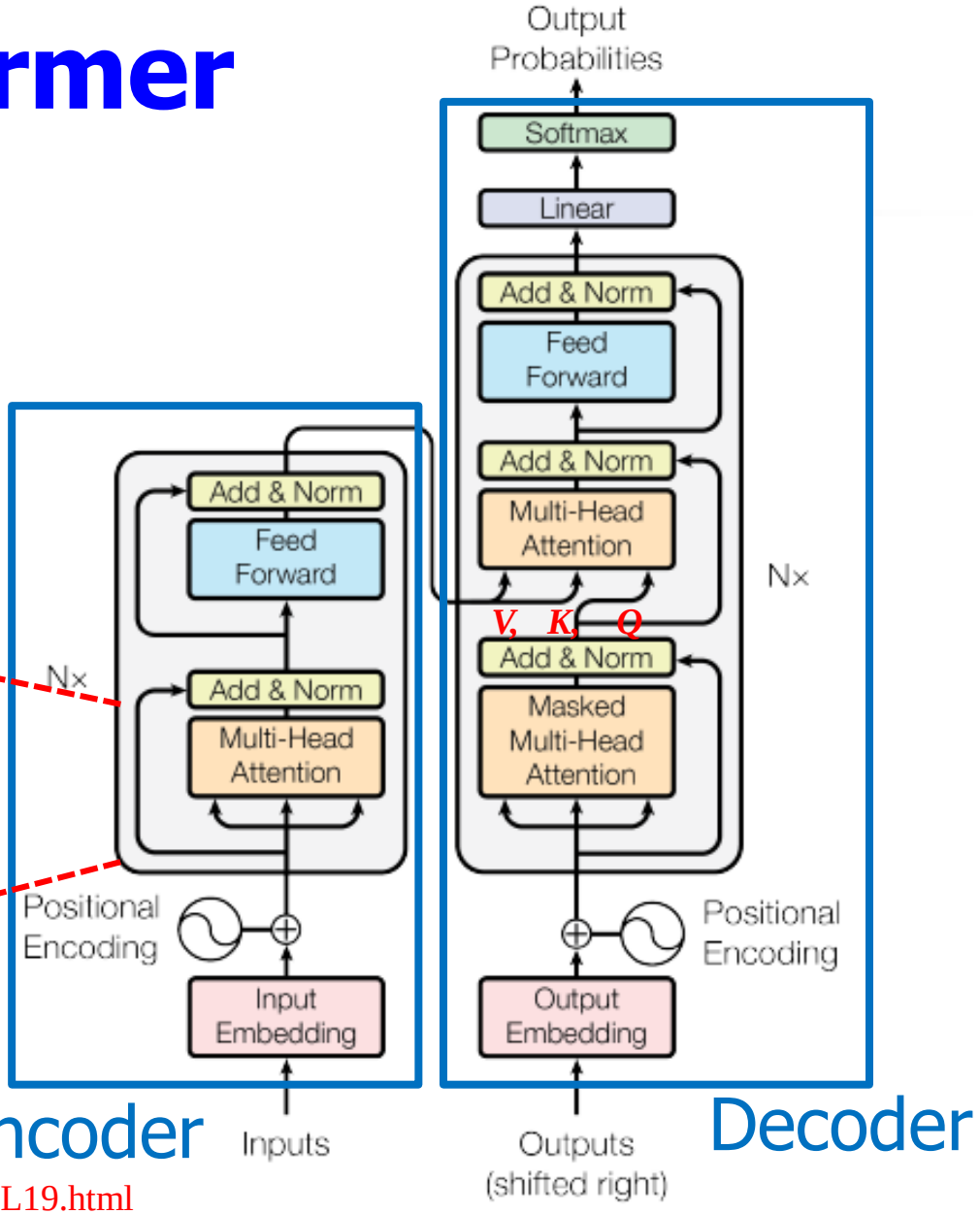
Semantic Feature Embedding

This slide contains Hung-Yi Lee's machine learning course materials
https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

From Self-Attention to Transformer

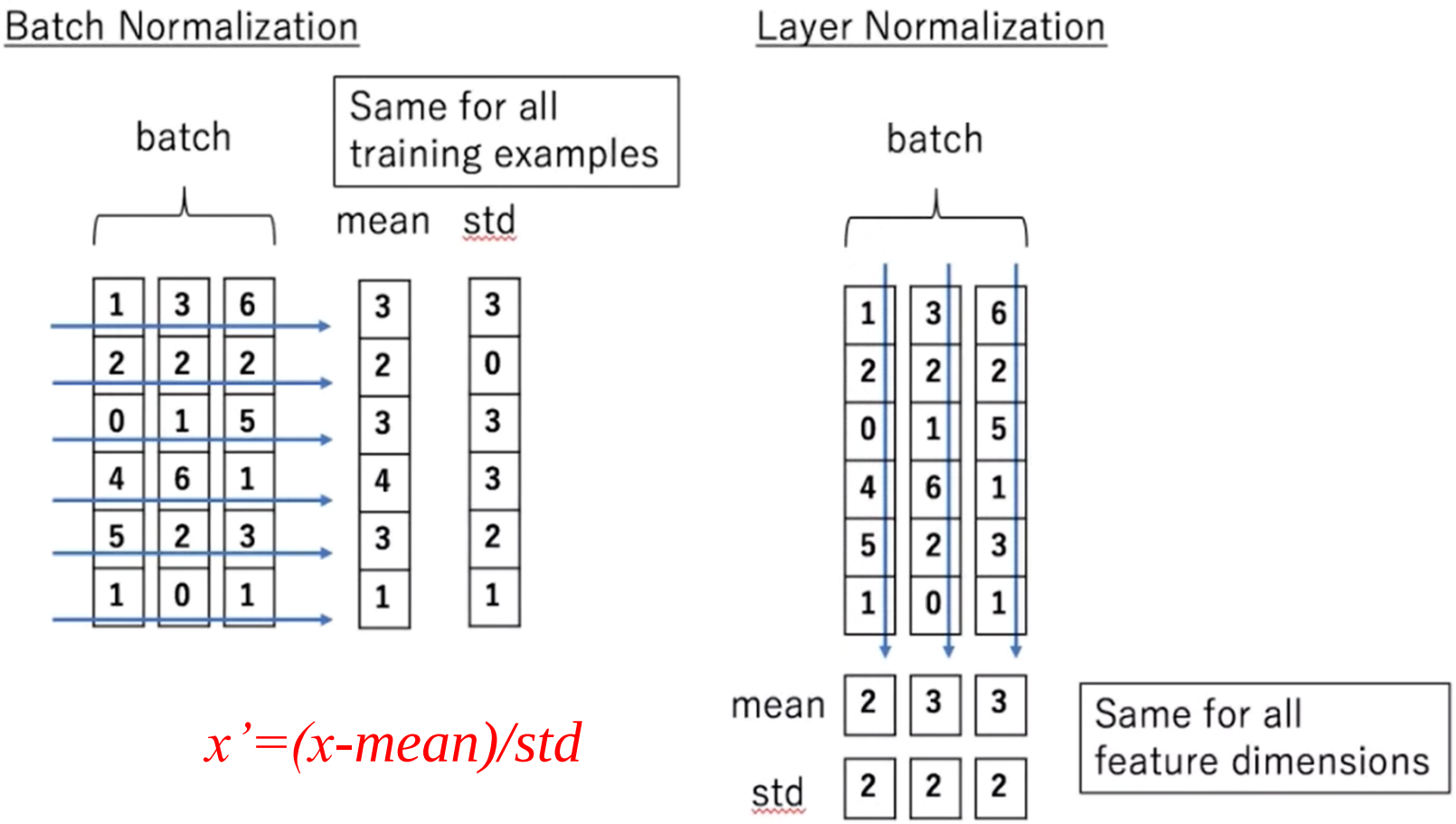




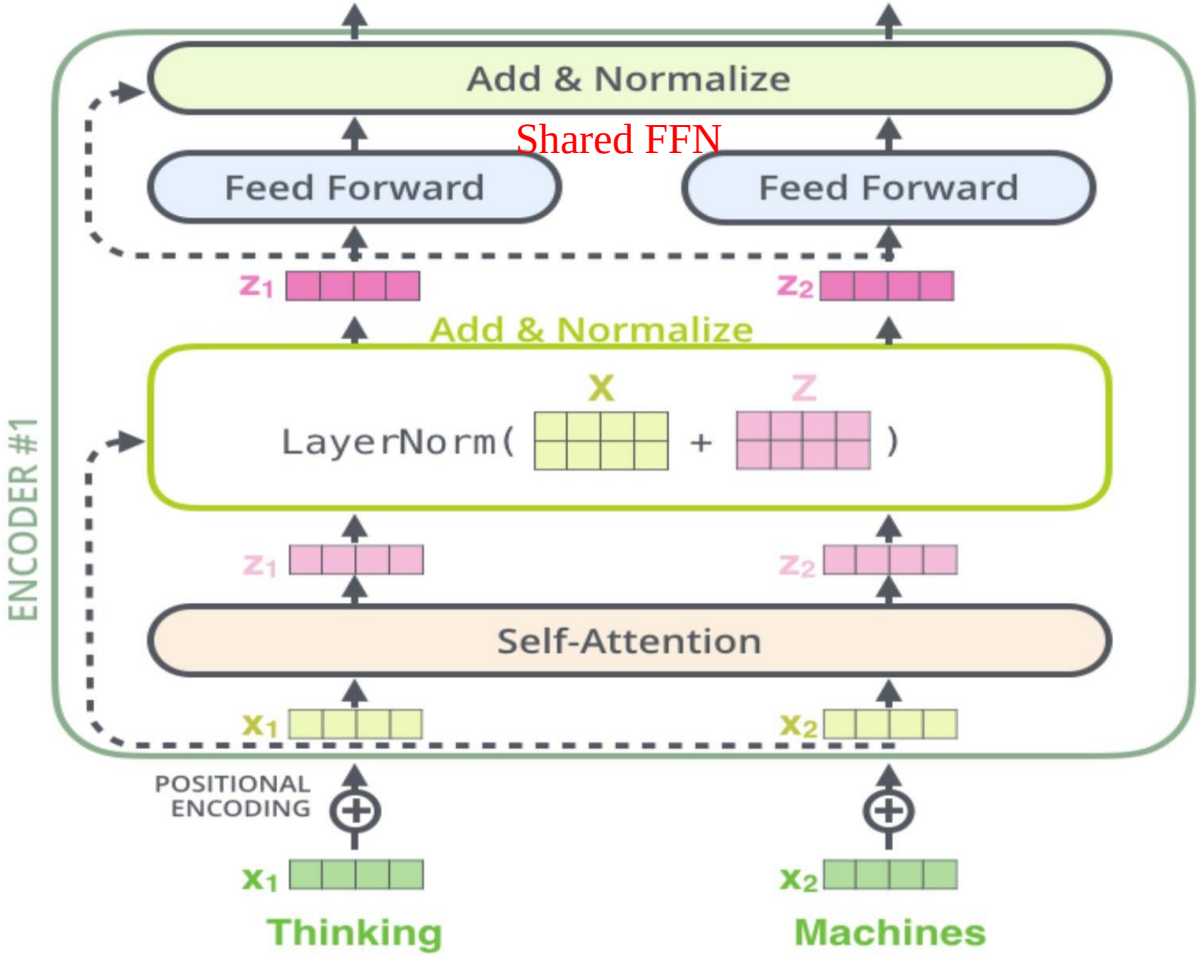


https://speech.ee.ntu.edu.tw/~tlkagk/courses_ML19.html

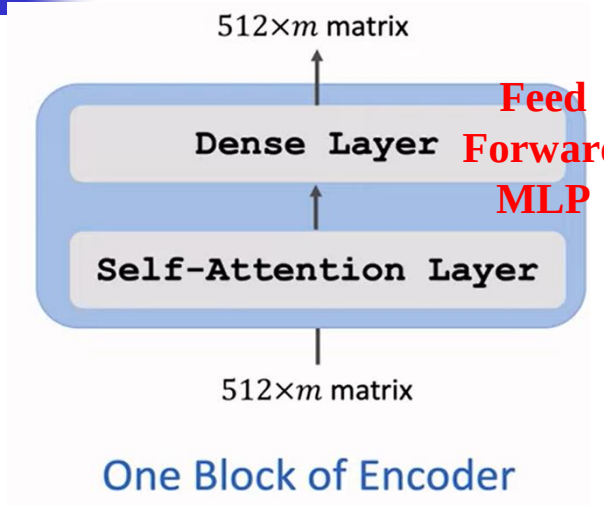
Layer Normalization

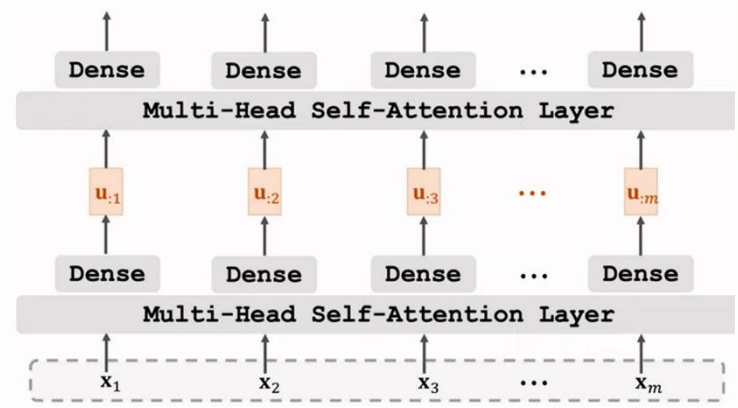


One Block of Self-Attention

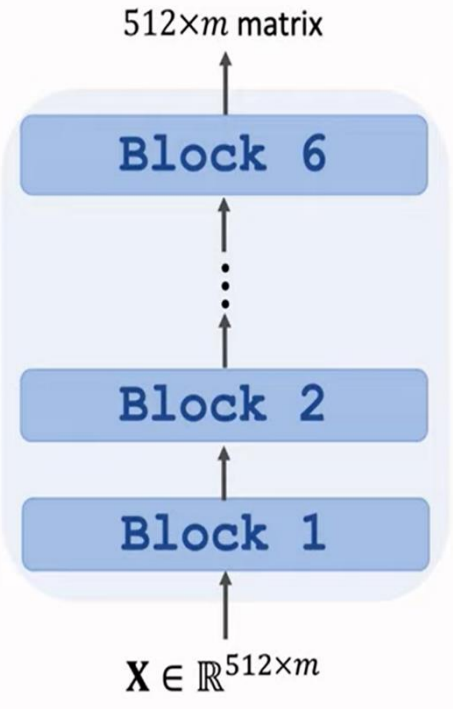


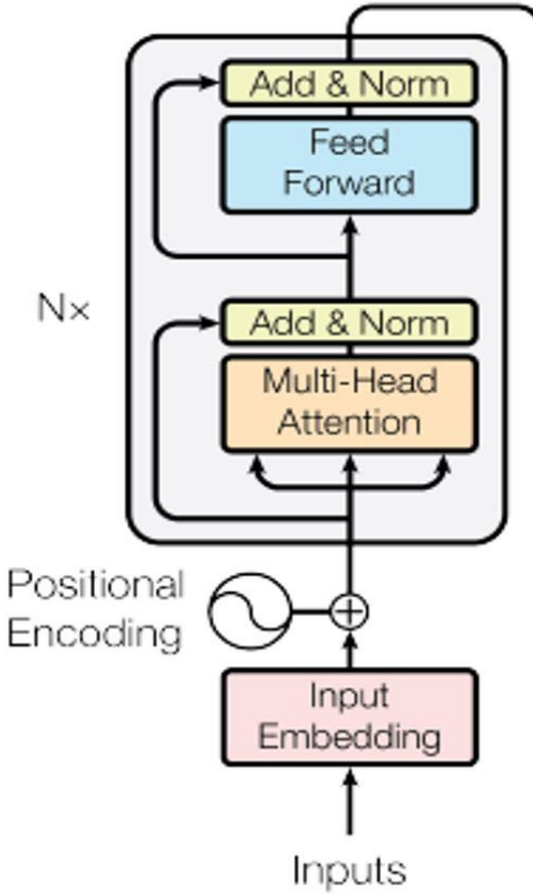
Stack Multi-Attention Blocks



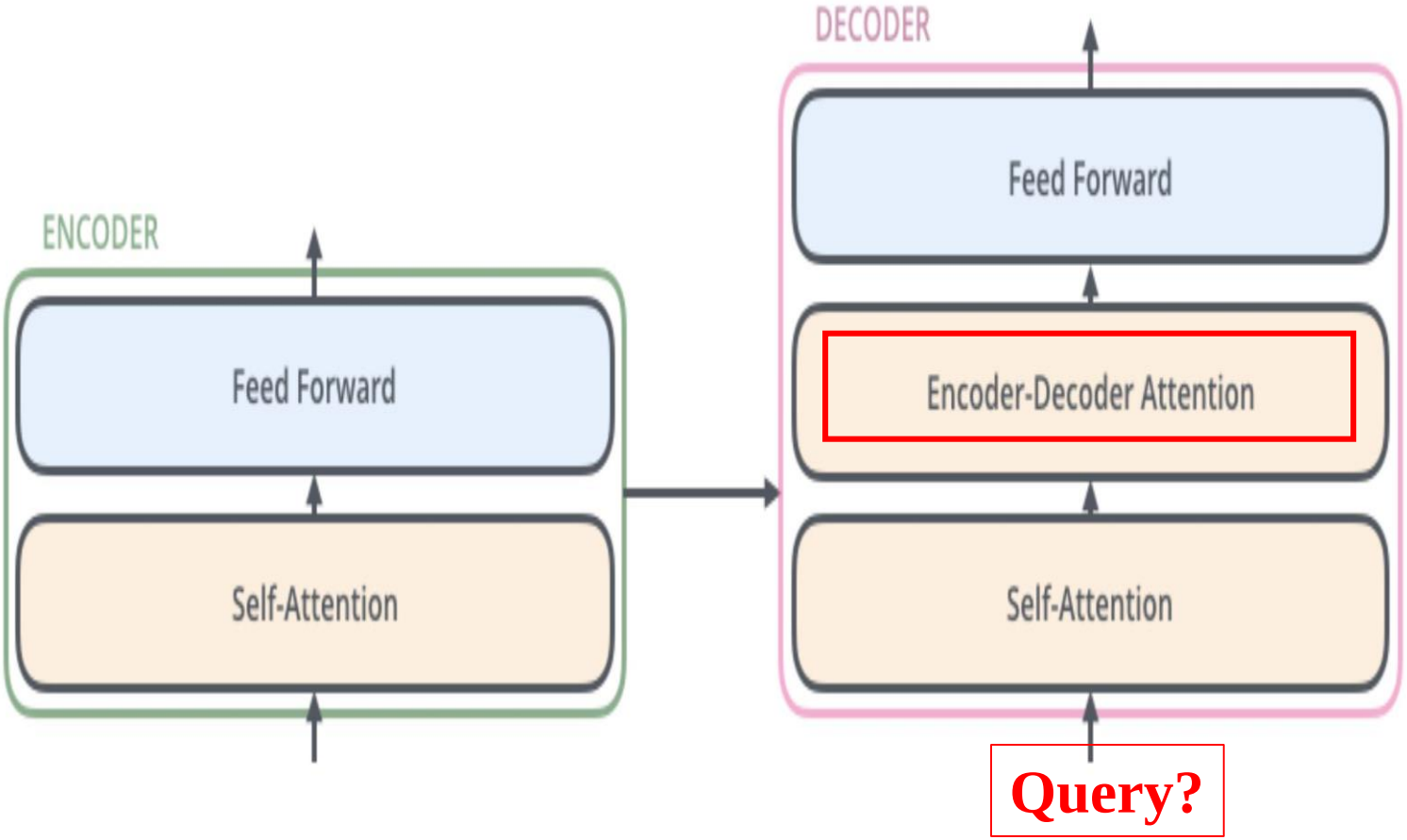


Stacking Two Blocks

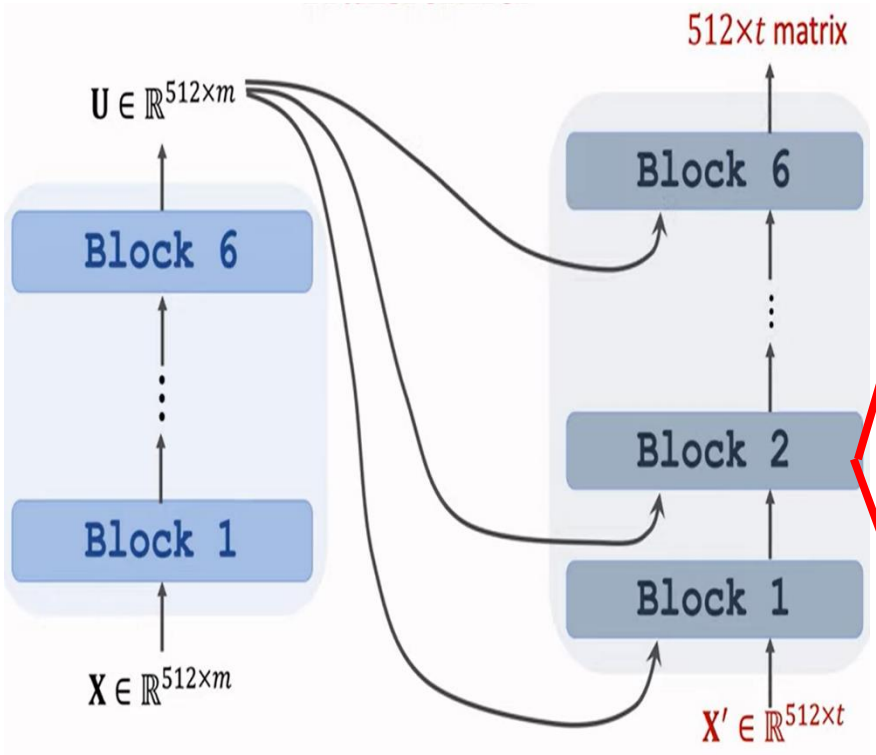


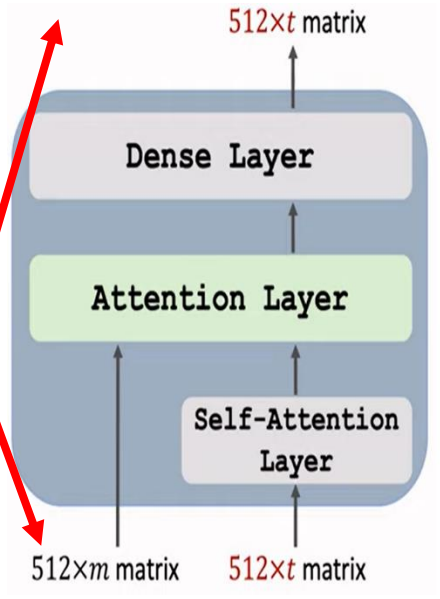


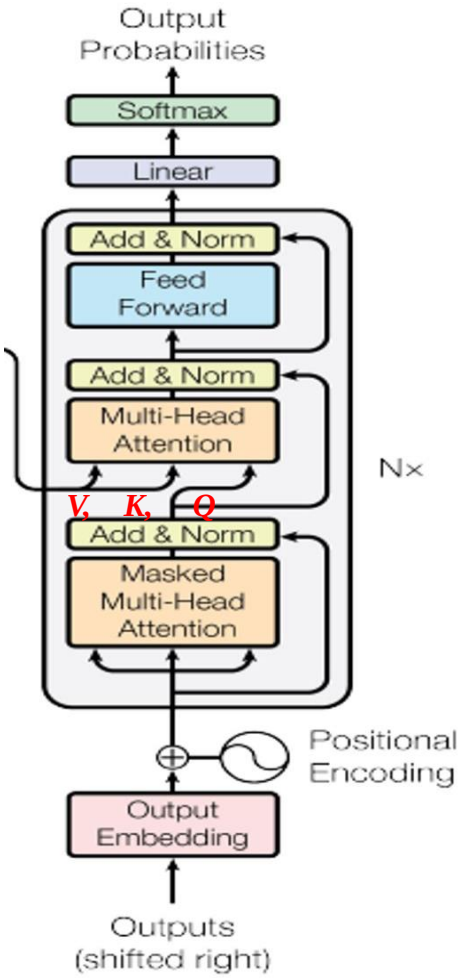
One Block of Decoder



Stacked Decoder Layers







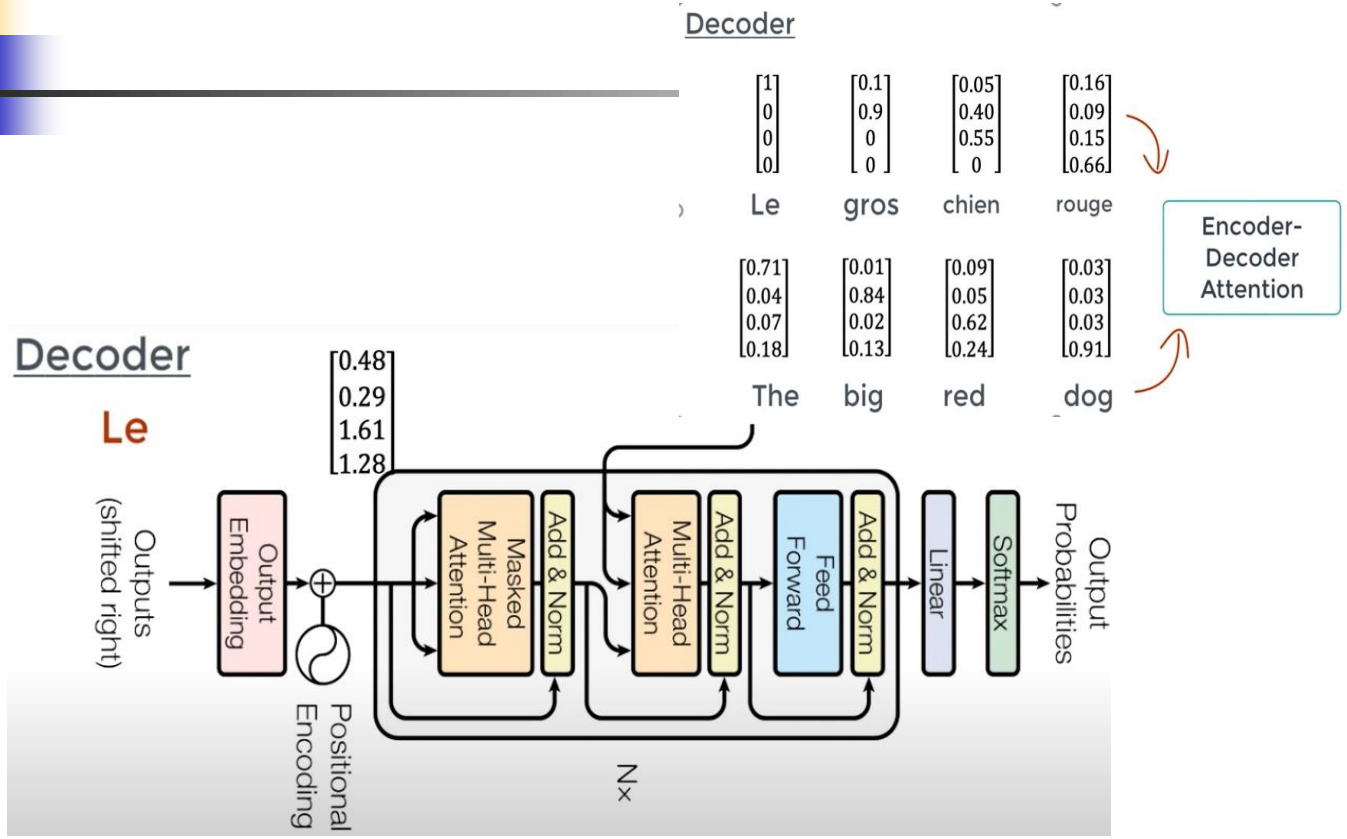
encoder

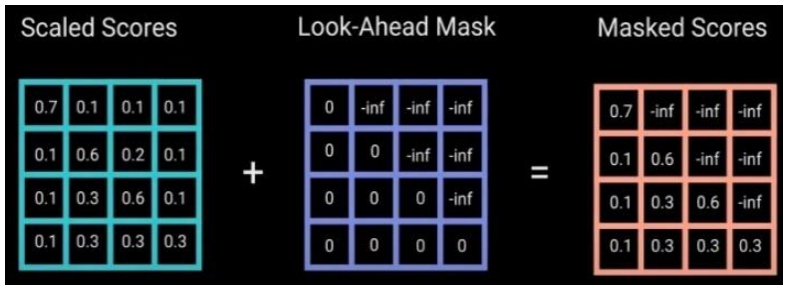
decoder

From encoder

From decoder

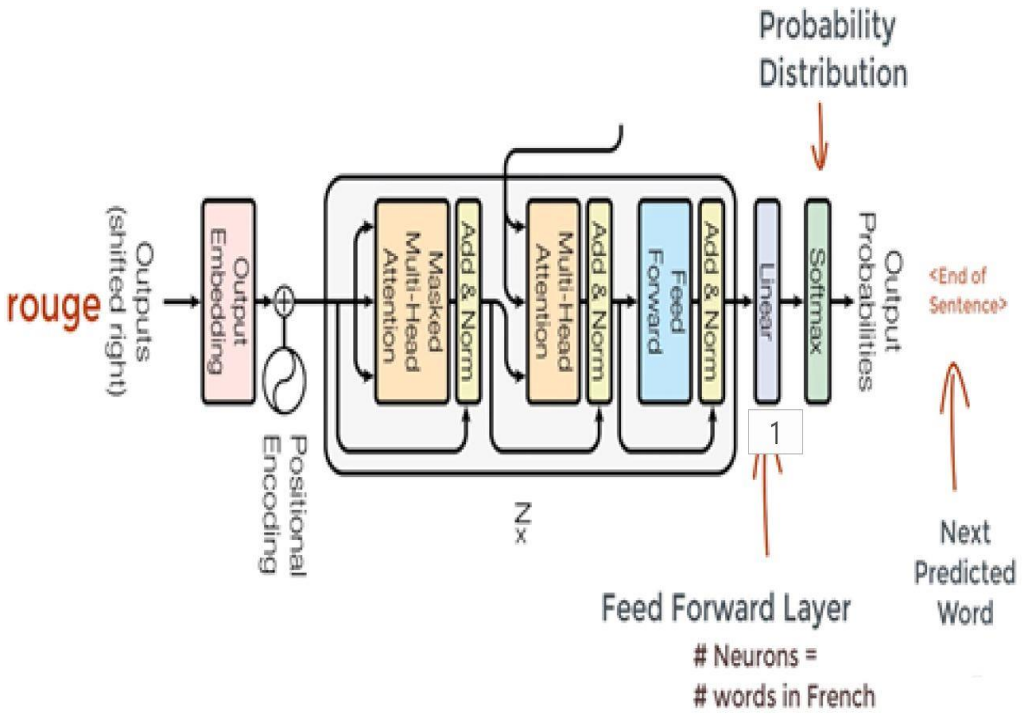
Transformer Decoder



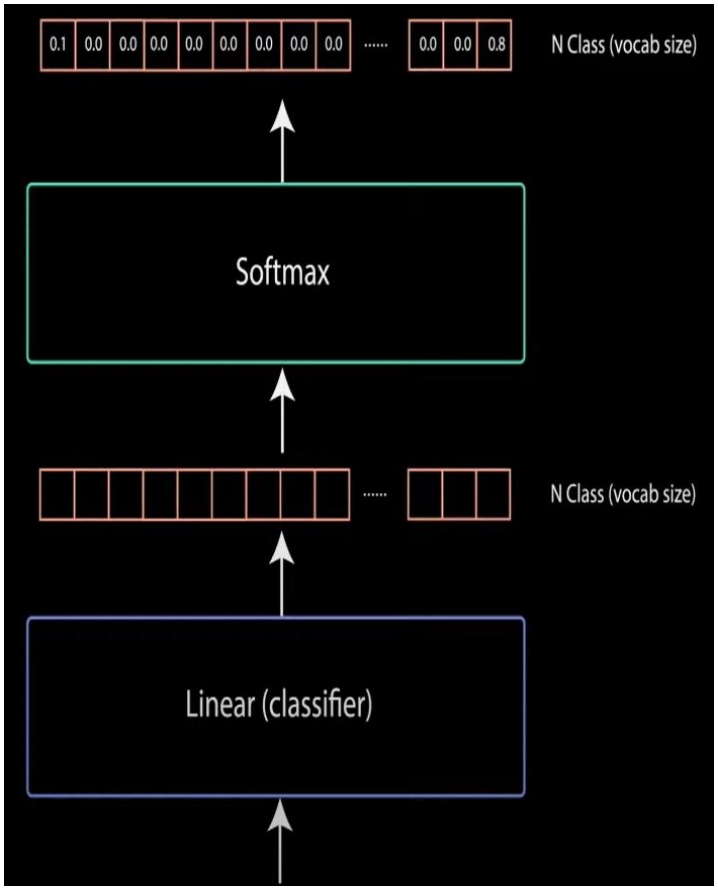


Only interested in attention with past data (softmax afterwards)

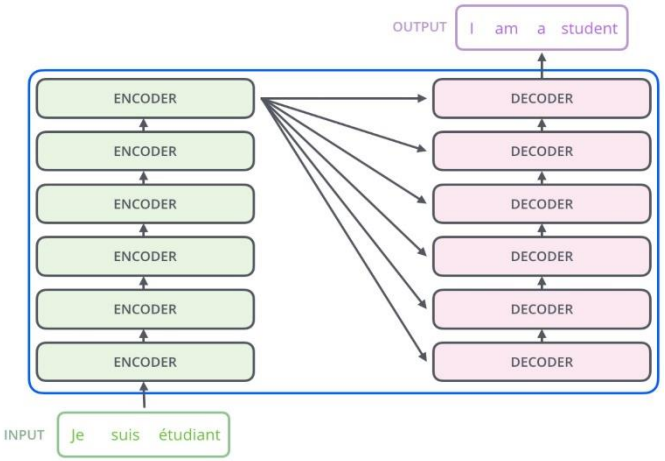
Transformer Decoder

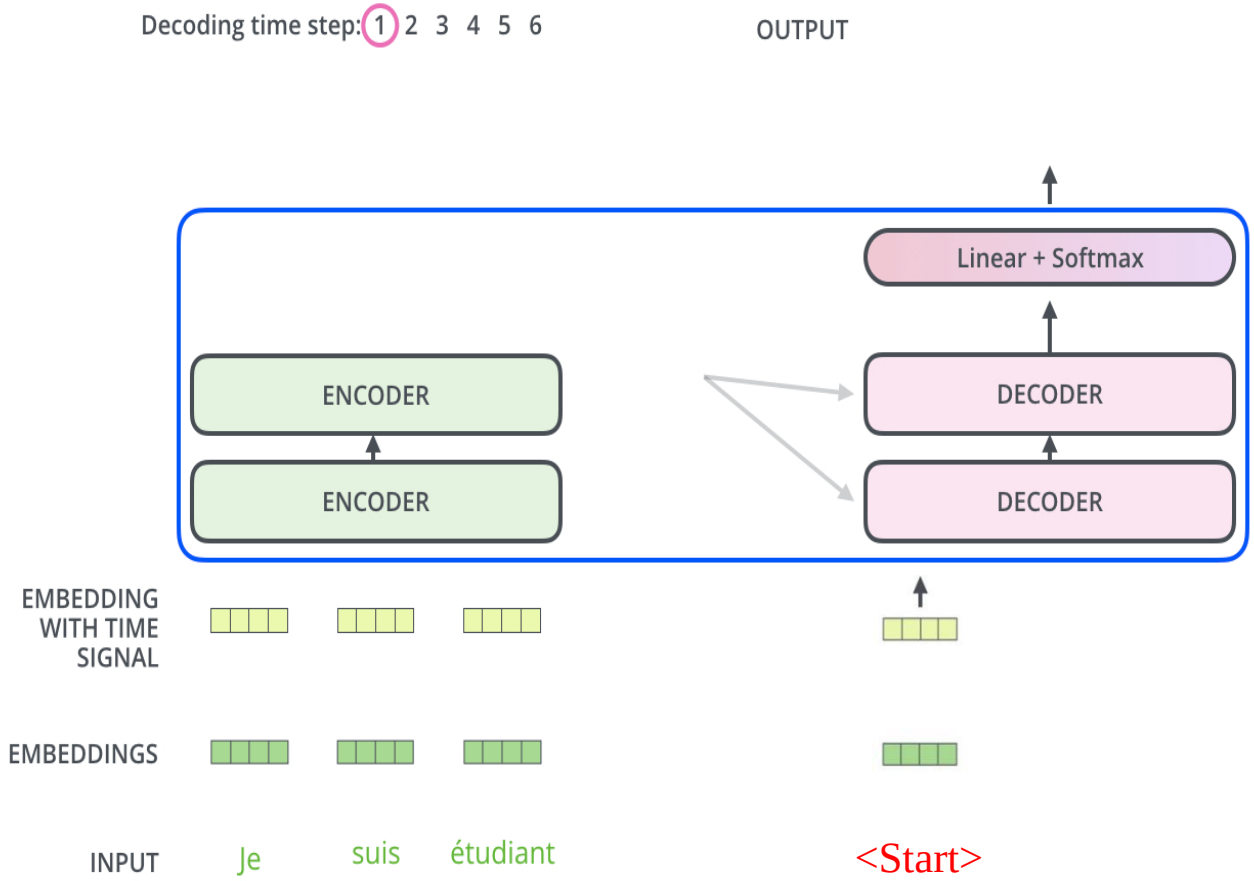


Vocabulary Size

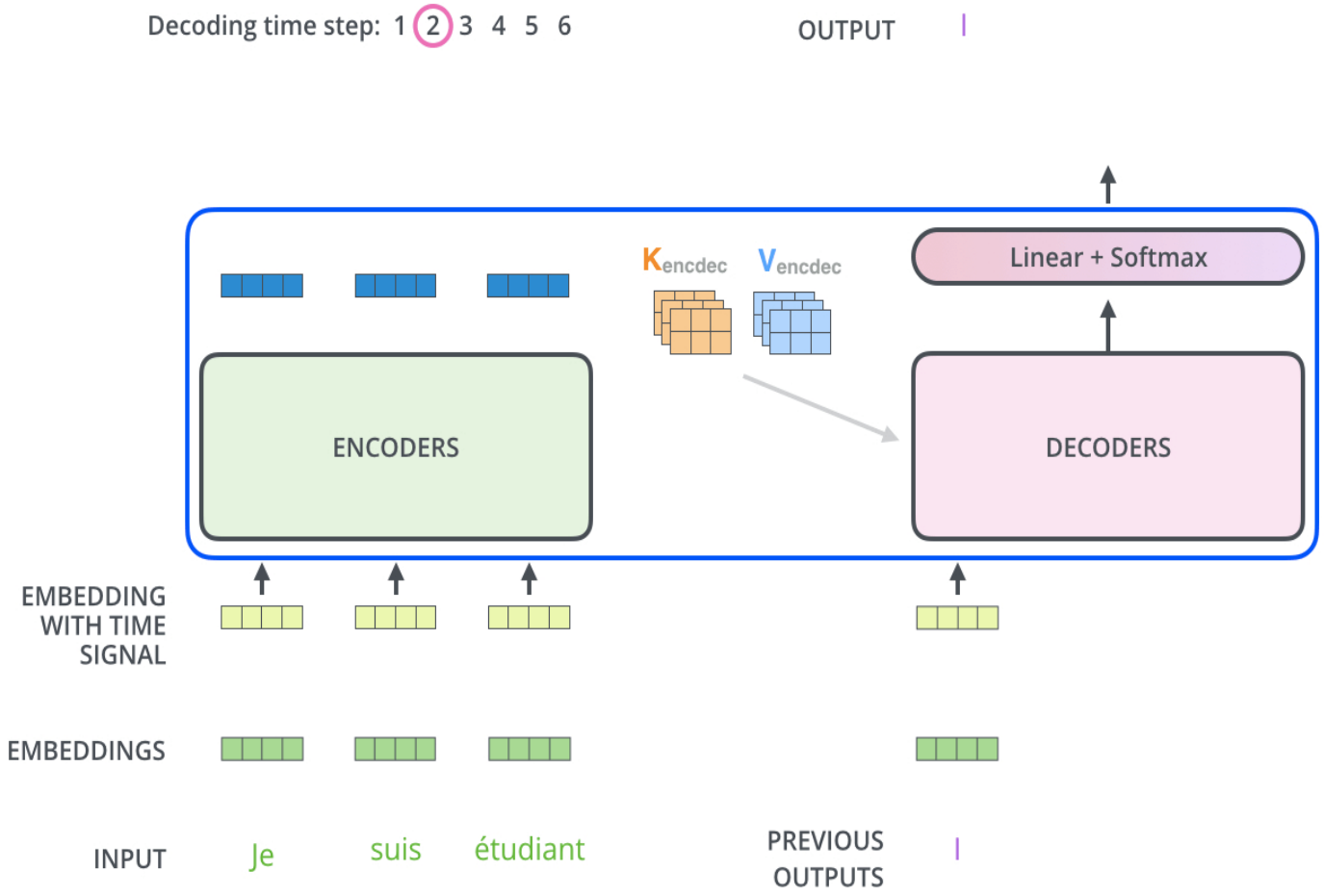


A Transformer Illustration

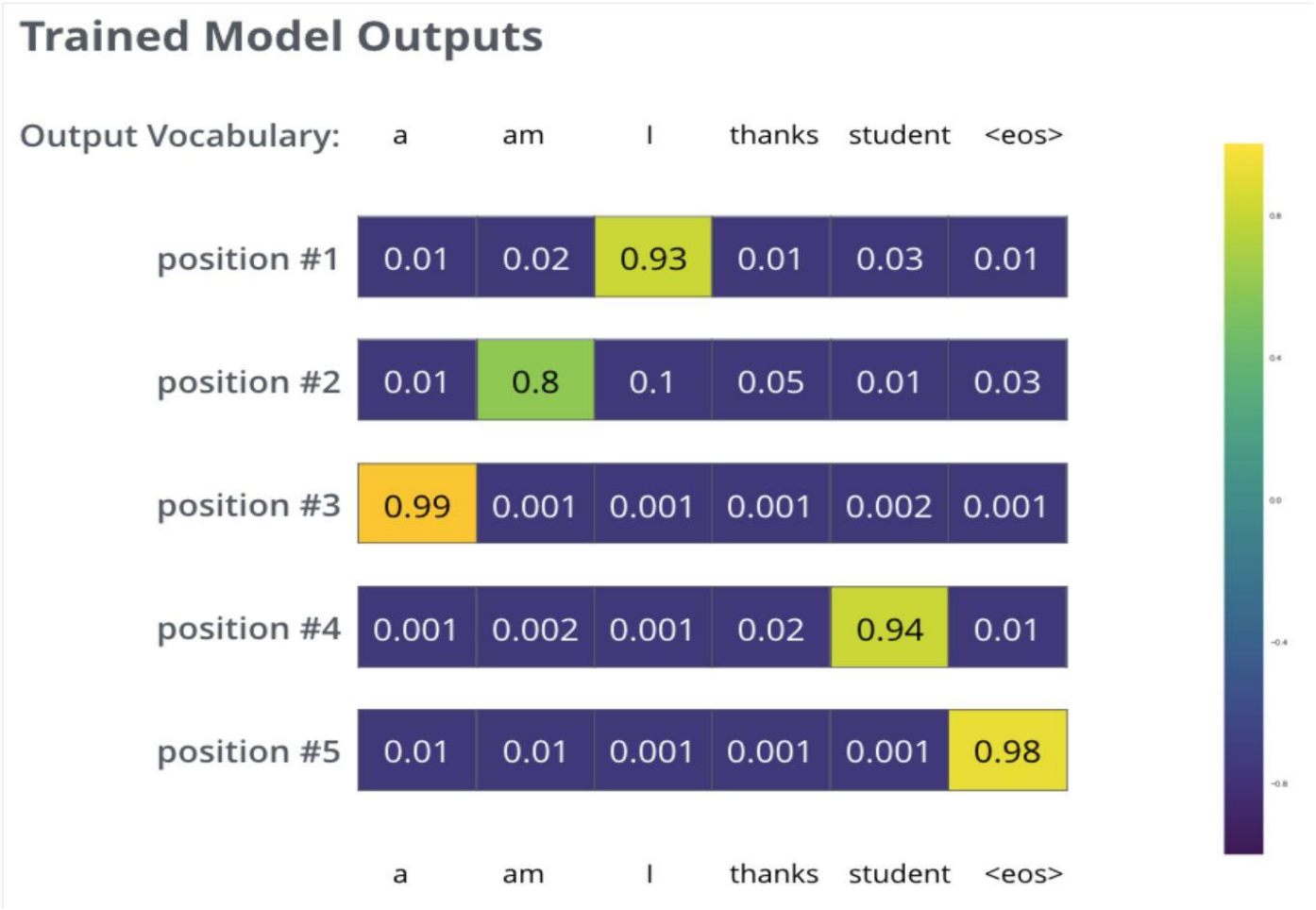




Decoder Illustration



Outputs of A Decoder

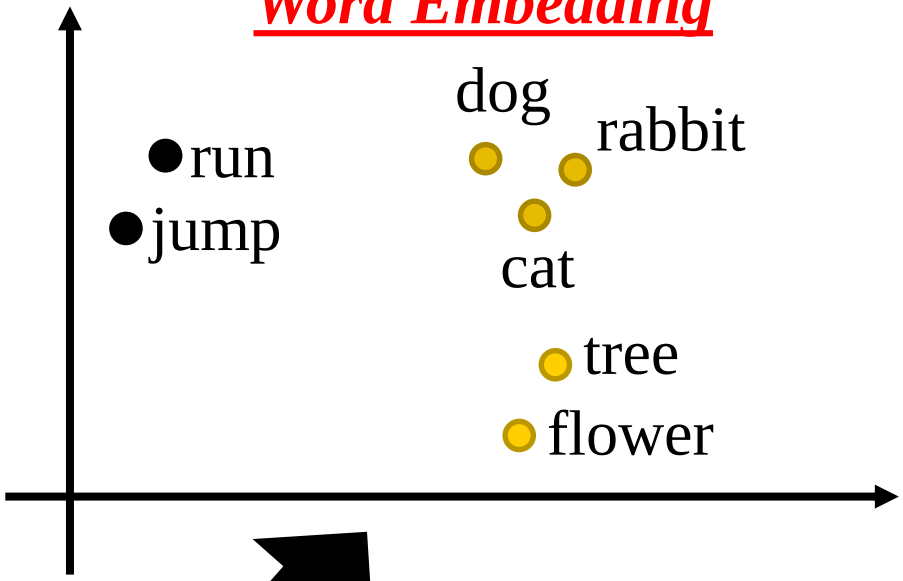


Word Embedding

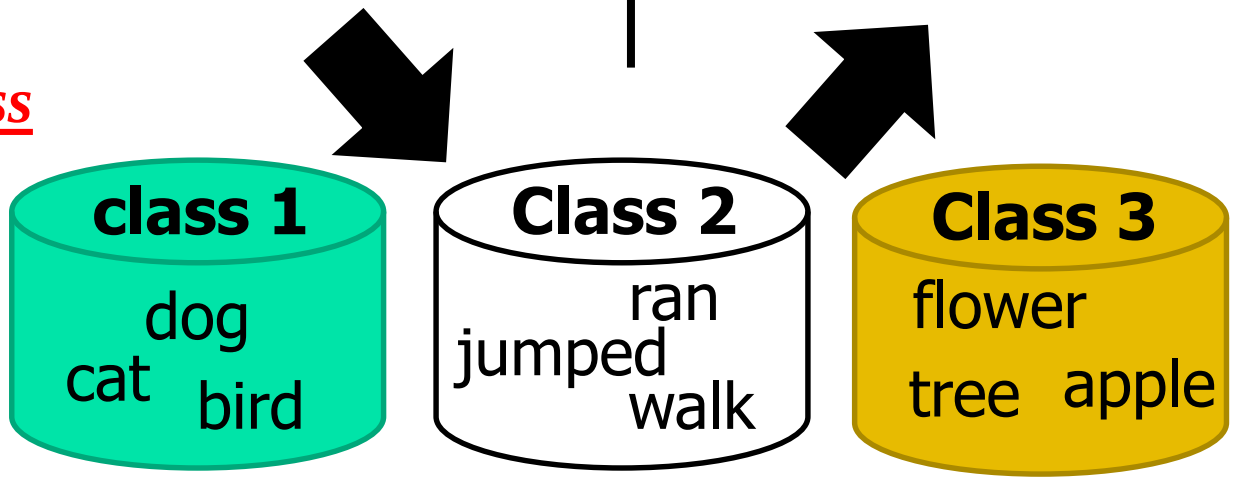
1-of-N Encoding

apple = [1 0 0 0 0]
bag = [0 1 0 0 0]
cat = [0 0 1 0 0]
dog = [0 0 0 1 0]
elephant = [0 0 0 0 1]

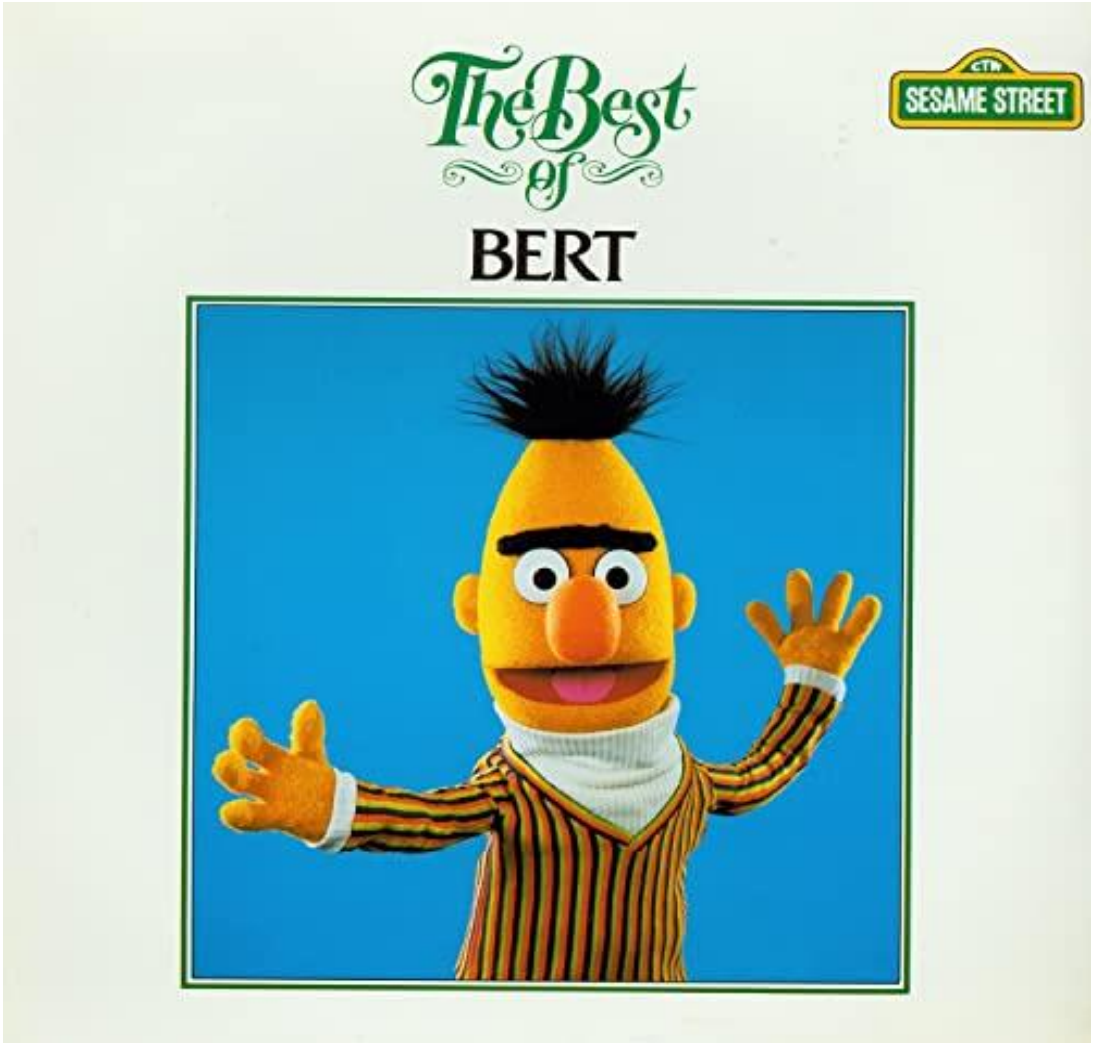
Word Embedding



Word Class

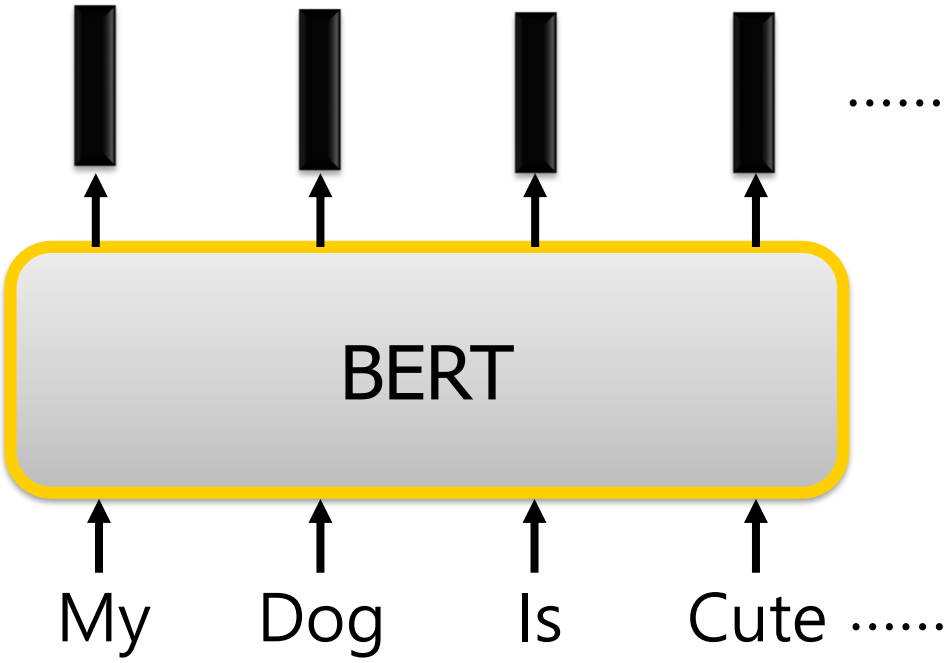


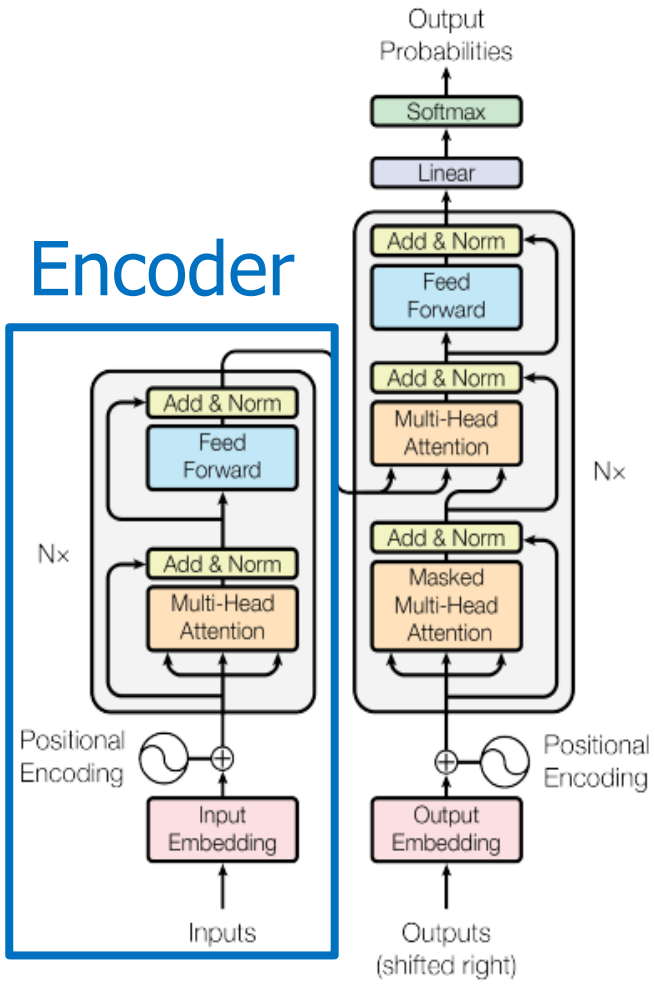
Bidirectional Encoder Representations from Transformers (BERT)



Bidirectional Encoder Representations from Transformers (BERT)

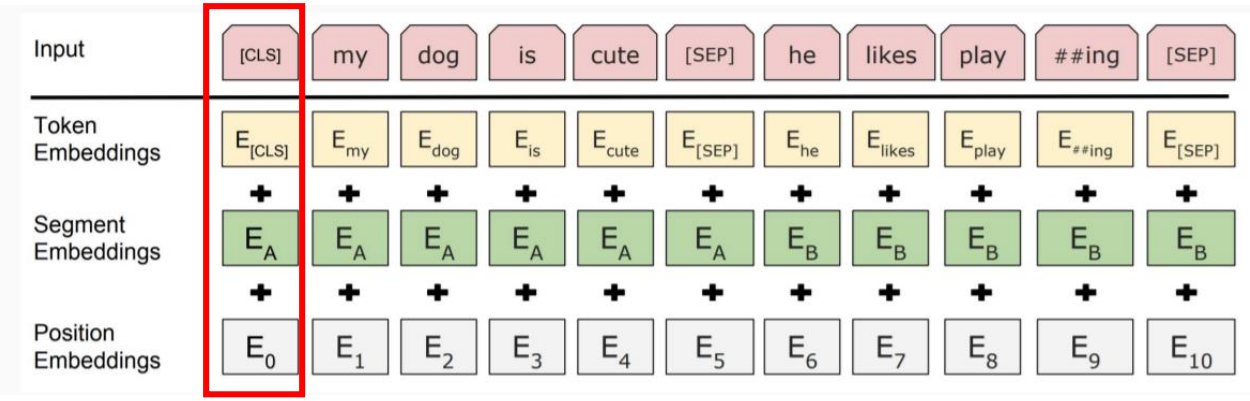
- BERT = Encoder of a Transformer
- Learned from a large amount of text without annotation





BERT Training

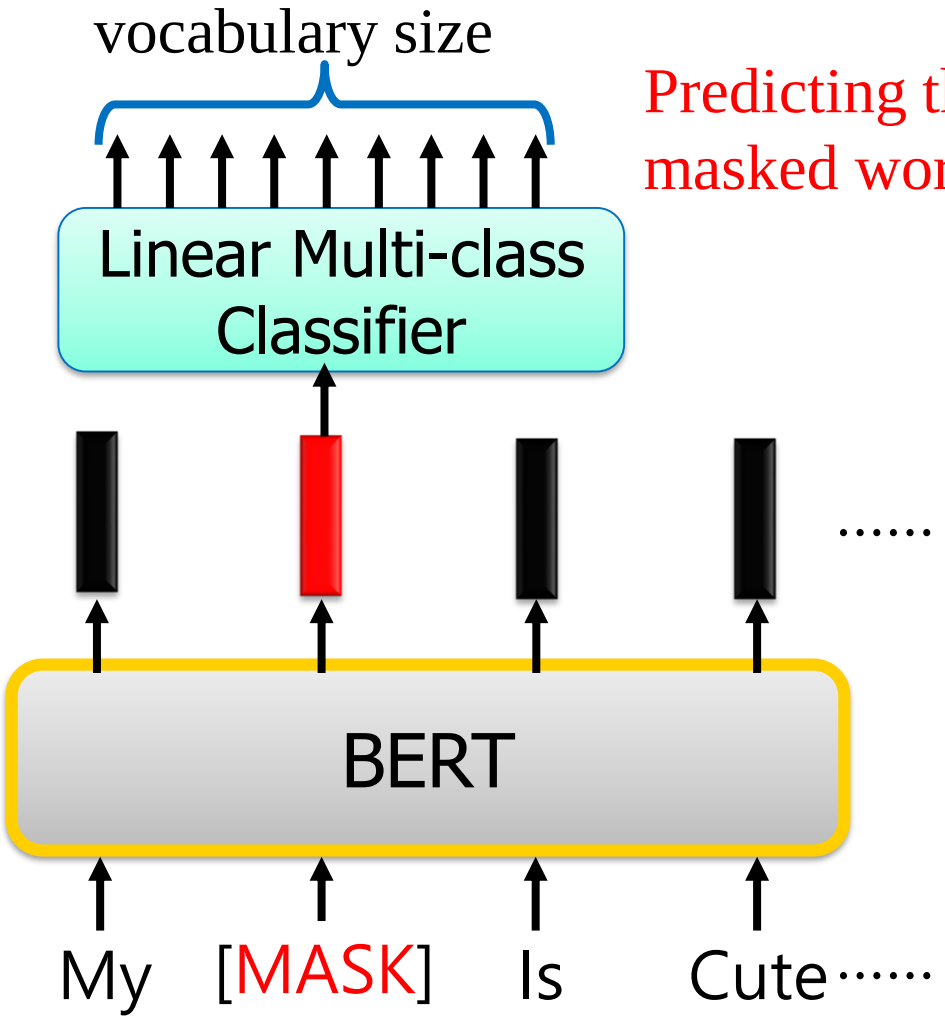
Input Representation



- The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks.
- Training Tasks:
  1) Mask Language Model (MLM)
  2) Next Sentence Prediction (NSP)

Training of BERT: 1) Predict the Masked Word

- Approach 1: Masked LM



Jacob Devlin, et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arxiv.org/abs/1810.04805

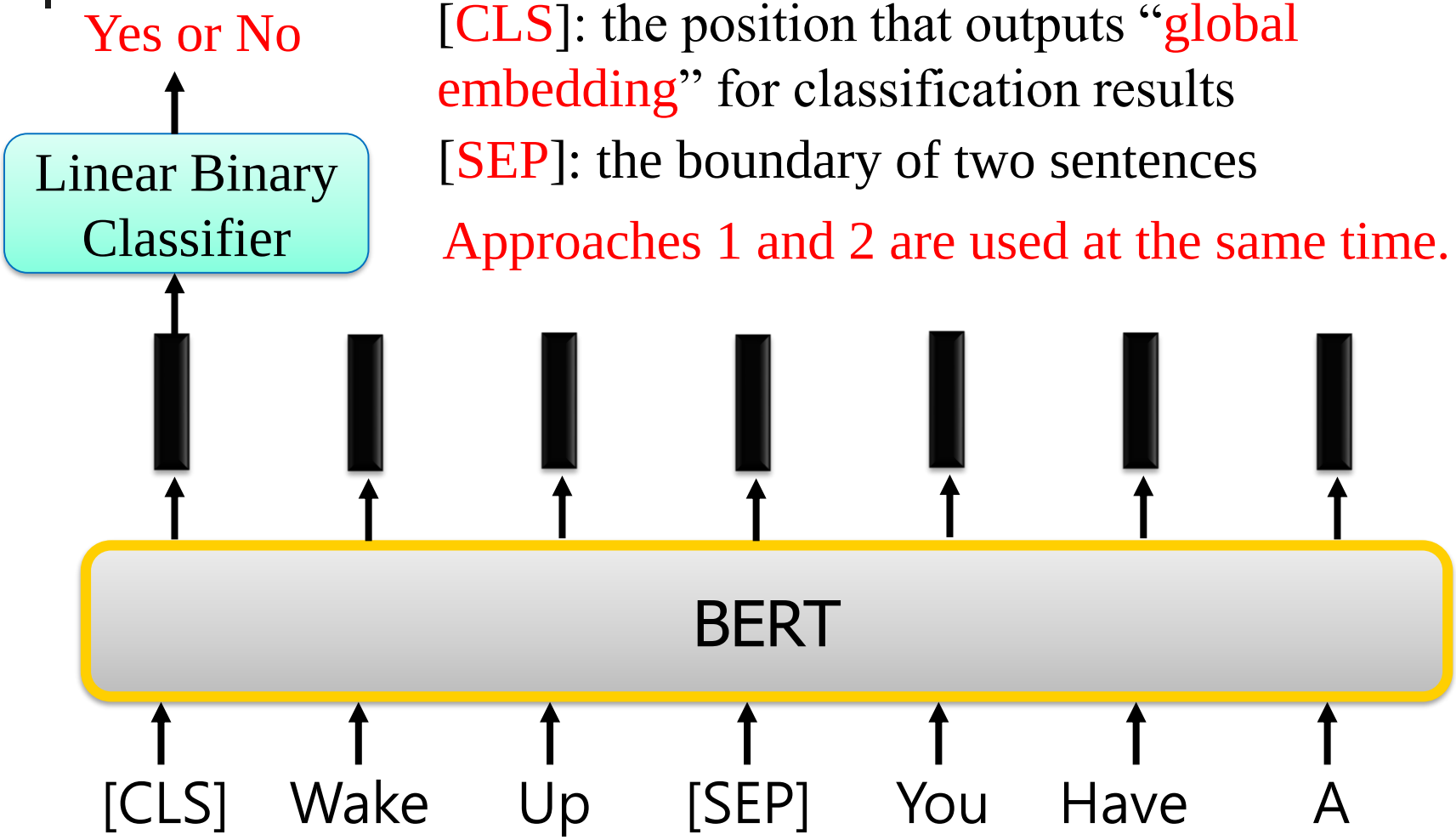
Transformed into 768-dim inputs by a Tokenizer

Training of BERT: 2) Predict the Next Sentence

[CLS]: the position that outputs "global embedding" for classification results

[SEP]: the boundary of two sentences

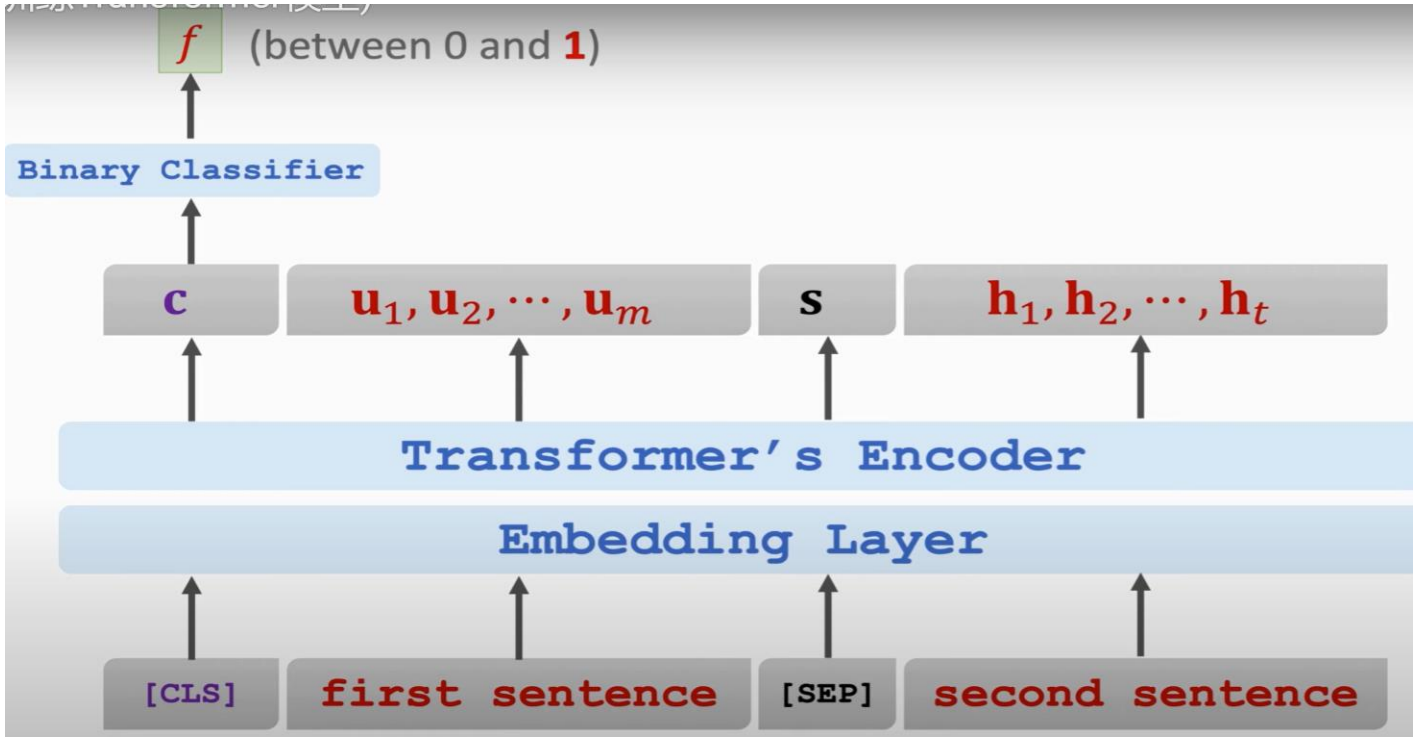
Approaches 1 and 2 are used at the same time.



The Use of [CLS] in BERT

- **Input:**
  [CLS] "calculus is a branch of math"
  [SEP] "it was developed by newton and leibniz"
- **Target:** true

- **Input:**
  [CLS] "calculus is a branch of math"
  [SEP] "panda is native to south central china"
- **Target:** false



Combing both Masked Words and Next Sentence Predictions

- **Input:**

"[CLS] calculus is a **[MASK]** of math [SEP] it **[MASK]** developed by newton and leibniz".

- **Targets:** true, "branch", "was".

- **Input:**

"[CLS] calculus is a branch of math [SEP] panda is native to **[MASK]** central china".

- **Targets:** false, "south".

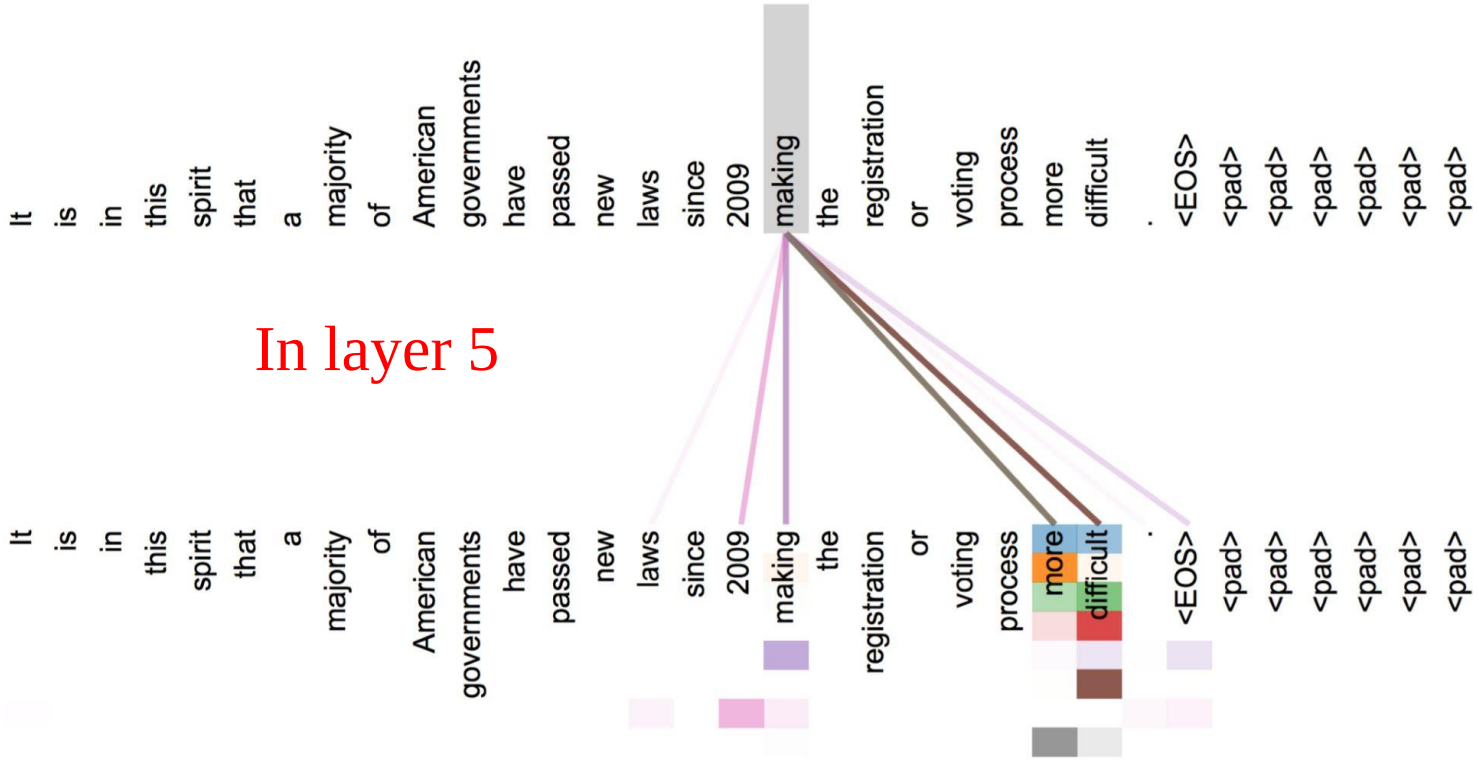
- Loss 1 is for binary classification (i.e., predicting the next sentence.)
- Loss 2 and Loss 3 are for multi-class classification (i.e., predicting the masked words.)
- Objective function is the sum of the three loss functions.
- Update model parameters by performing one gradient descent.

No Annotations of Data

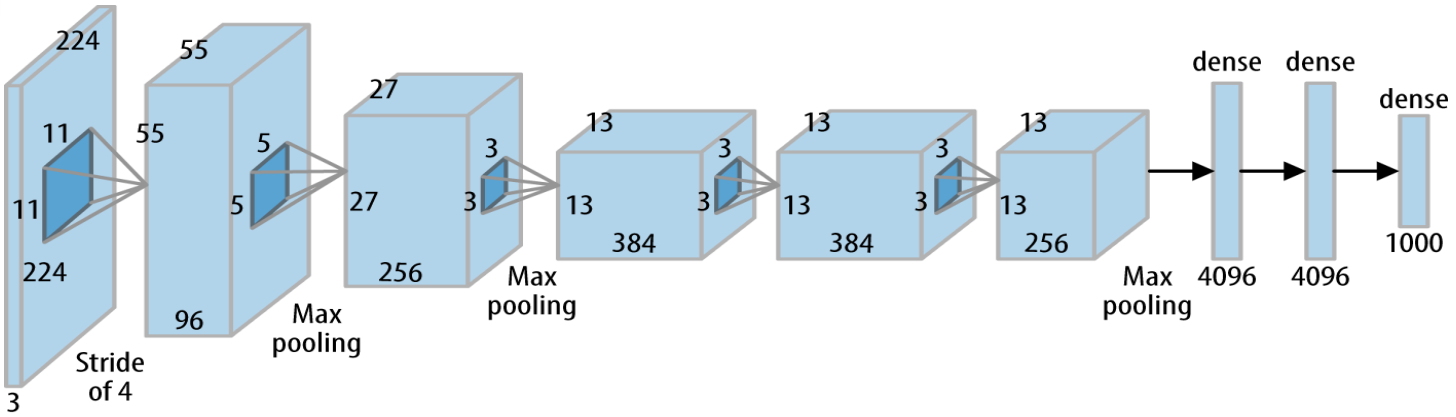
- BERT does not need manually labelled data for training (data annotation is expensive)
- Use large scale freely available data, e.g., English Wikepedia (2.5 billion words)
- Randomly masked words (15% with some tricks)
- 50% of the next sentences are real (the other 50% are fake, randomly chosen)
- BERT (base): 110M parameters, 12 blocks, 12 heads
- BERT (large): 235M parameters, 24 blocks, 16 heads

Visualization of Self Attention in BERT Word Embedding

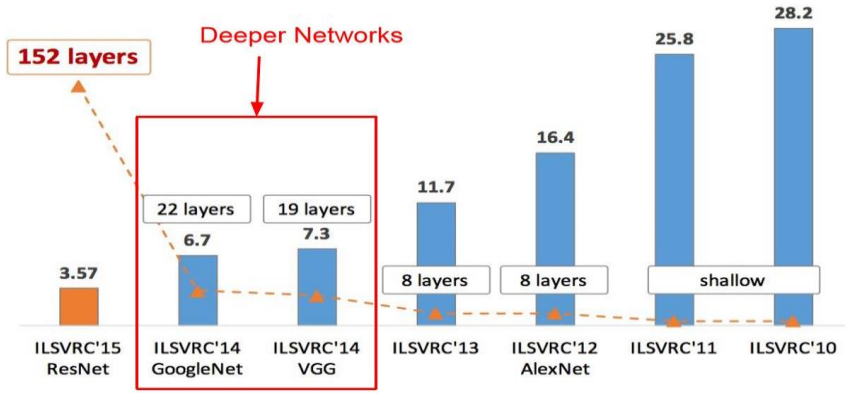
- Words start to pay attention to other words in sensible ways

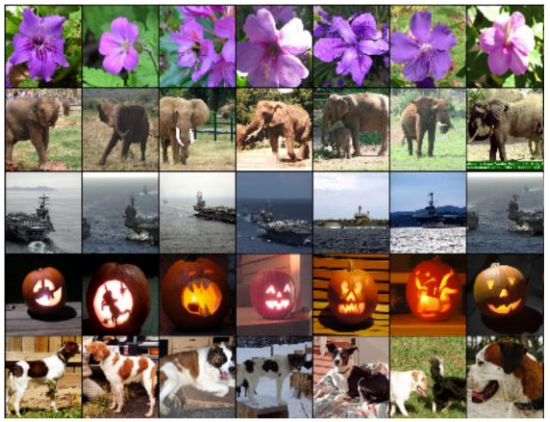


CNN Success in Image Object Classification



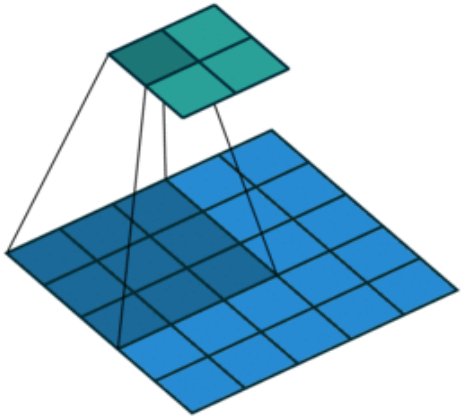
- These translational invariance and locality that have allowed the great success of CNNs（conv and pooling）





Self-Attention in NLP vs. Convolution in Vision

- Each pixel is an input "token"
- Traditionally, convolution is used to integrate pixels
  - Recognize patterns within a small window of pixels
  - Difficult to integrate non-local pixels
  - Make network extreme deep to "see" the whole picture
- Self-attention (Transformer) also integrates multiple pixels
  - Long distance context has "equal opportunity"
  - Works when the correlated pixels are non-local
  - Trunk, tail, legs → whole elephant



From NLP to Vision

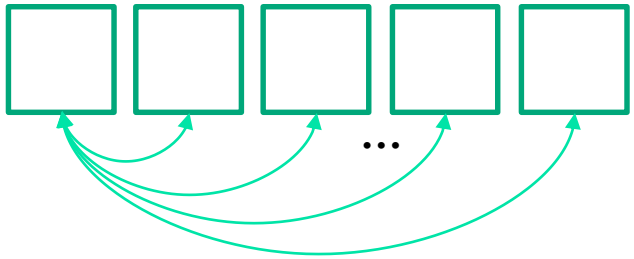
Transformers in Computer Vision:

Pros:

- No locality bias in Transformers (Self-attention based architectures)
- Self-supervised Tasks during the pre-training stage.

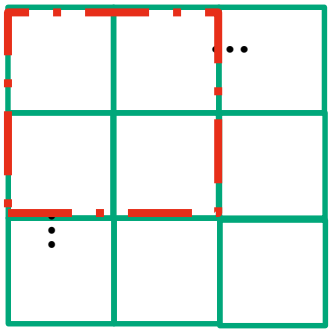
Cons:

For words:



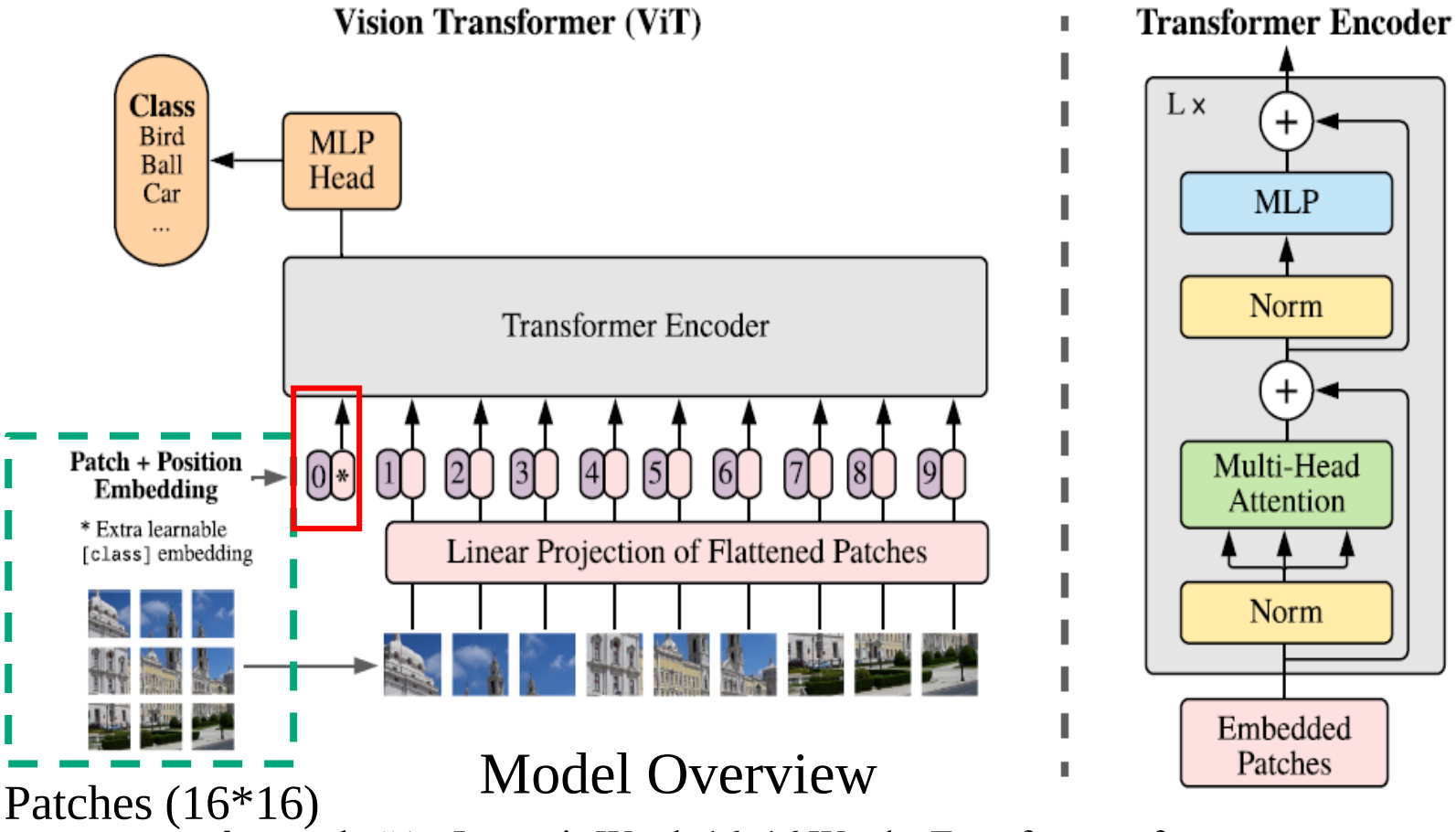
Space/Memory $O(N^2)$

For an image:



Space/Memory $O((N^2)^2)$

Vision image Transformer (ViT)

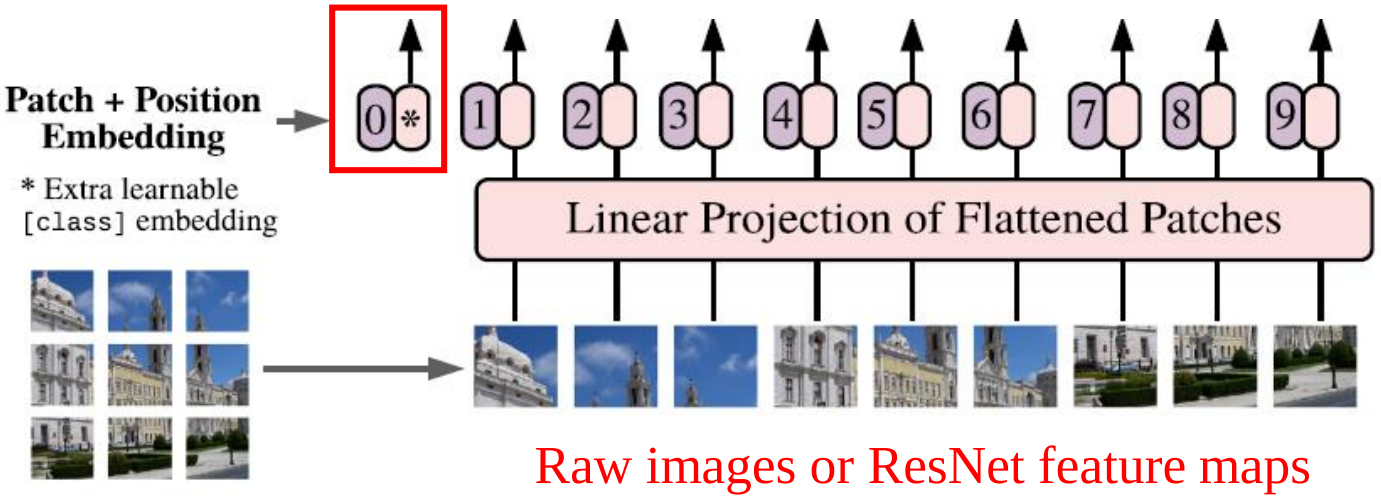


Patches (16*16)

Model Overview

Alexey Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," https://arxiv.org/pdf/2010.11929.pdf

Inputs to ViT



Raw images or ResNet feature maps

To handle 2D images,

- Reshape an image $x \in R^{H \times W \times C}$ into a sequence of flattened 2D patches
- $(H \times W)$ is the resolution of the original image, (P, P)=(16,16) is resolution of each image patch, $N = HW/P^2$ is the effective length for the transformer $x_p \in R^{N \times (P^2 \cdot C)}$
- Linear Projection Layer: a trainable linear projection maps each vector to the model dimension D $(= PxPxC = 16x16x3 = 768)$

Vision image Transformer (ViT)

Hybrid Architecture

ResNet50 (output of stage 4 feature maps)

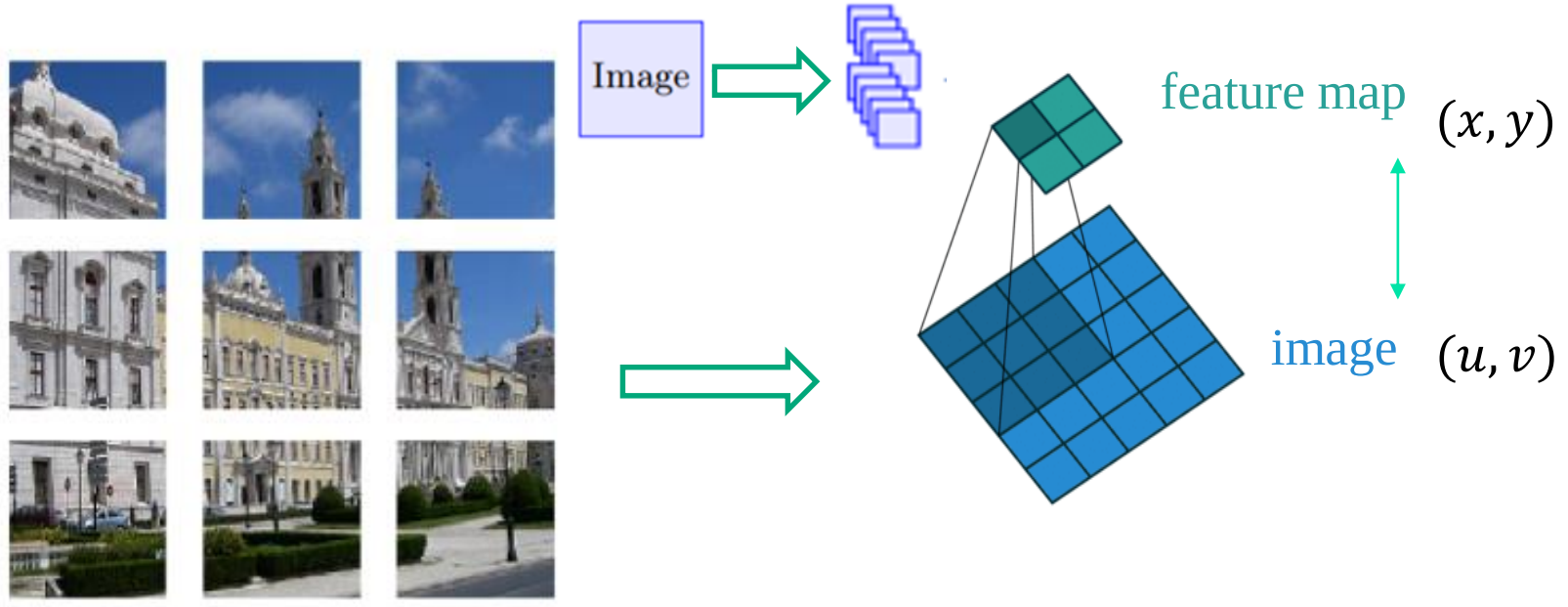
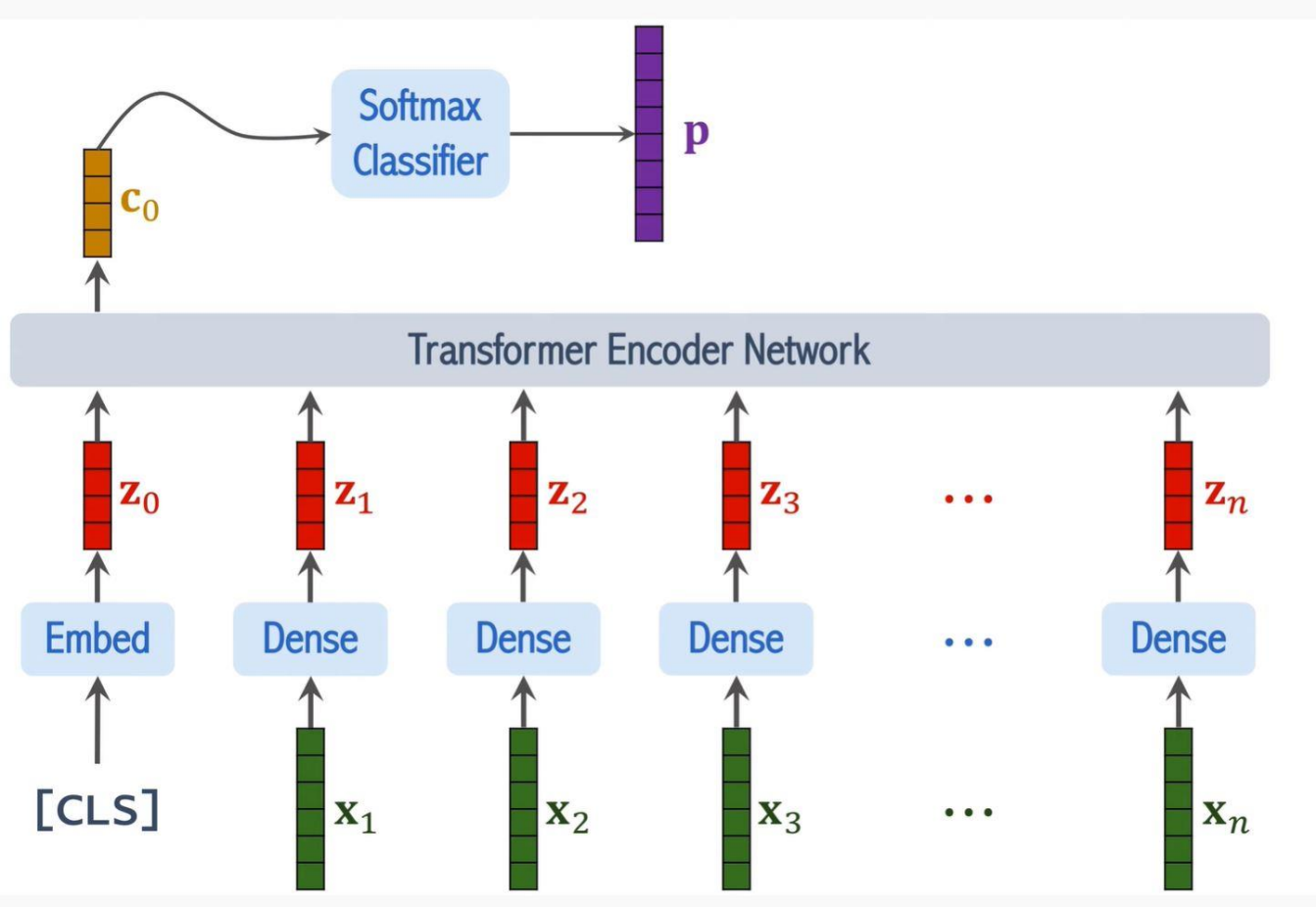


Image patches

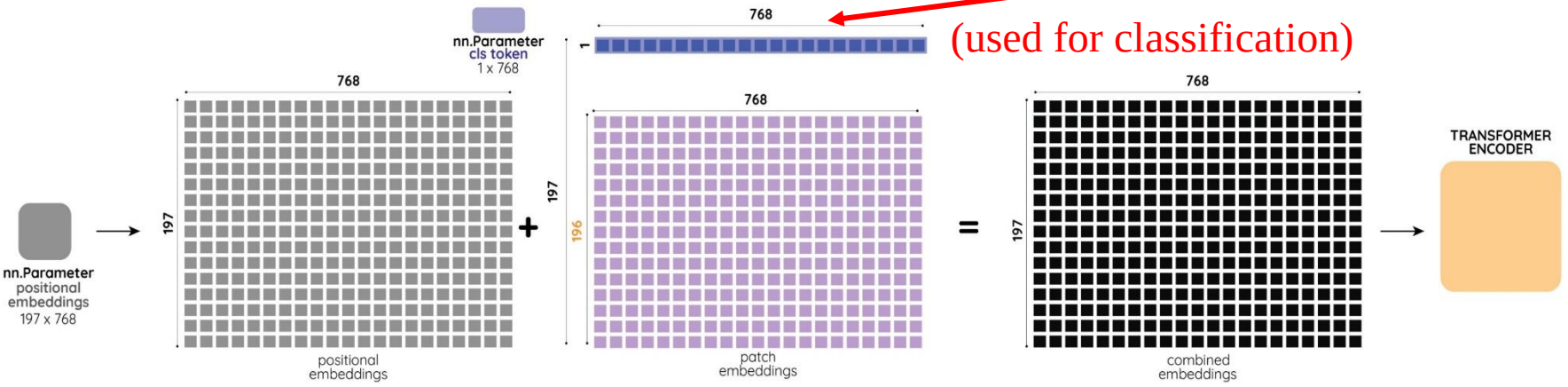
Feature map as inputs

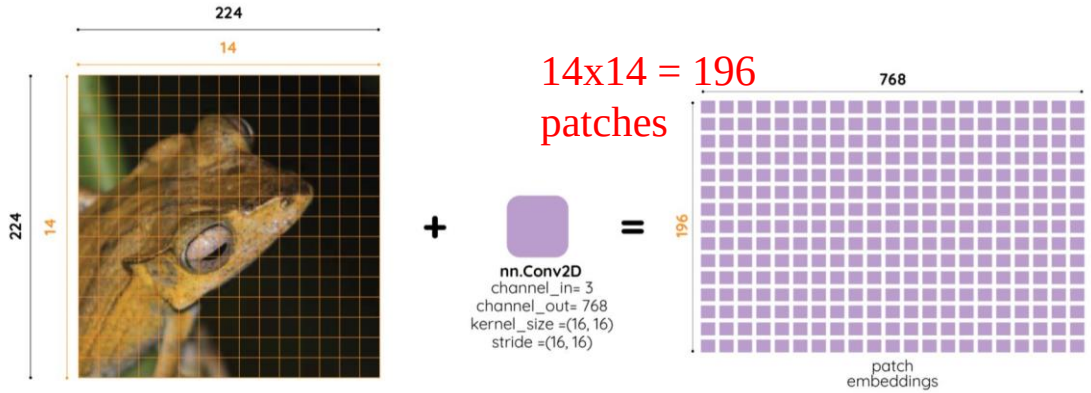
ViT Classification



Patch, Position and Class <CLS> Token Embeddings

- Similar to BERT's classification token <CLS> , a learnable embedding to the sequence of embedded patches $(\mathbf{z}_0^0 = \mathbf{x}_{\text{class}})$





14x14 = 196 patches

d=768 channels of 2D convolution feature maps

Vision image Transformer (ViT) Configurations

Model Configurations

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

Table 1: Configuration of our different model variants.

Same as the BERT configurations,
Base = “B”, Large = “L”, Huge = “H”

Pretrained Datasets in ViT

	# of Images	# of Classes
ImageNet (Small)	1.3 Million	1 Thousand
ImageNet-21K (Medium)	14 Million	21 Thousand
JFT (Big)	300 Million	18 Thousand

- Pretrain the model on Dataset A, fine-tune the model on Dataset B, and evaluate the model on Dataset B.

- Pretrained on ImageNet (small), ViT is slightly worse than ResNet.
- Pretrained on ImageNet-21K (medium), ViT is comparable to ResNet.
- Pretrained on JFT (large), ViT is slightly better than ResNet.

ViT Performance

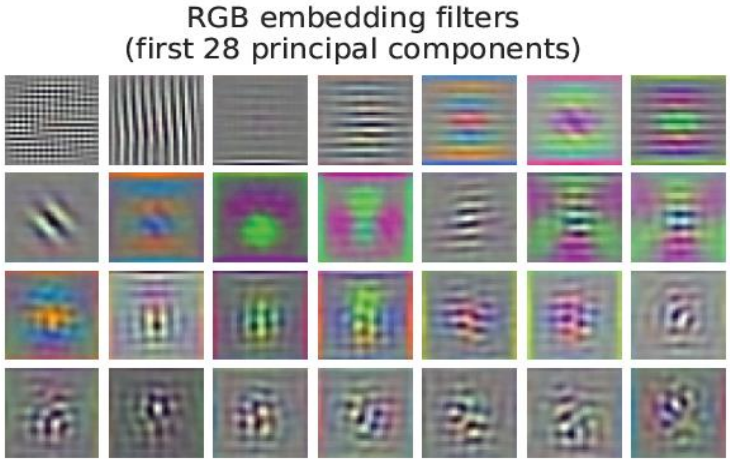
Model	Epochs	ImageNet	ImageNet ReaL	CIFAR-10	CIFAR-100	Pets	Flowers	exaFLOPs
ViT-B/32	7	80.73	86.27	98.61	90.49	93.40	99.27	164
ViT-B/16	7	84.15	88.85	99.00	91.87	95.80	99.56	743
ViT-L/32	7	84.37	88.28	99.19	92.52	95.83	99.45	574
ViT-L/16	7	86.30	89.43	99.38	93.46	96.81	99.66	2586
ViT-L/16	14	87.12	89.99	99.38	94.04	97.11	99.56	5172
ViT-H/14	14	88.08	90.36	99.50	94.71	97.11	99.71	12826
ResNet50x1	7	77.54	84.56	97.67	86.07	91.11	94.26	150
ResNet50x2	7	82.12	87.94	98.29	89.20	93.43	97.02	592
ResNet101x1	7	80.67	87.07	98.48	89.17	94.08	95.95	285
ResNet152x1	7	81.88	87.96	98.82	90.22	94.17	96.94	427
ResNet152x2	7	84.97	89.69	99.06	92.05	95.37	98.62	1681
ResNet152x2	14	85.56	89.89	99.24	91.92	95.75	98.75	3362
ResNet200x3	14	87.22	90.15	99.34	93.53	96.32	99.04	10212
R50x1+ViT-B/32	7	84.90	89.15	99.01	92.24	95.75	99.46	315
R50x1+ViT-B/16	7	85.58	89.65	99.14	92.63	96.65	99.40	855
R50x1+ViT-L/32	7	85.68	89.04	99.24	92.93	96.97	99.43	725
R50x1+ViT-L/16	7	86.60	89.72	99.18	93.64	97.03	99.40	2704
R50x1+ViT-L/16	14	87.12	89.76	99.31	93.89	97.36	99.11	5165

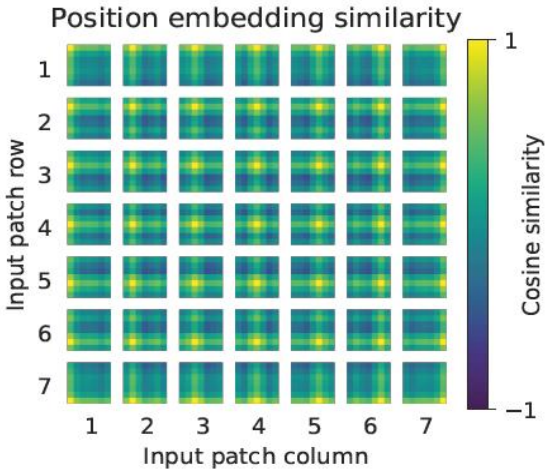
hybrid

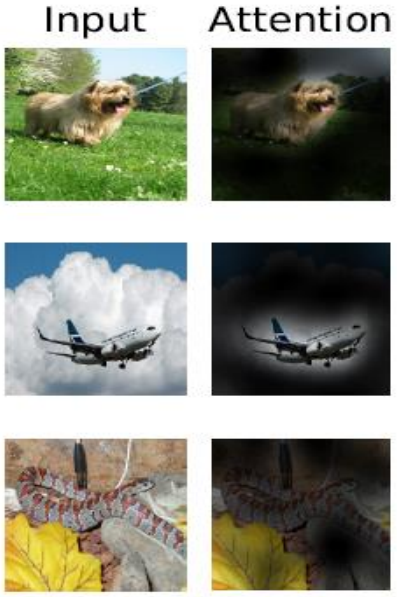
- ViT-L/16 means the “Large” variant with 16×16 input patch size
- BiT-L is ResNet50 with 4 stages

Global Self Attention of ViT

- Unlike CNNs, where only the later layers are able to aggregate information from different parts of the image.
- Filters of the initial linear embedding of RGB values of ViT-L/32.



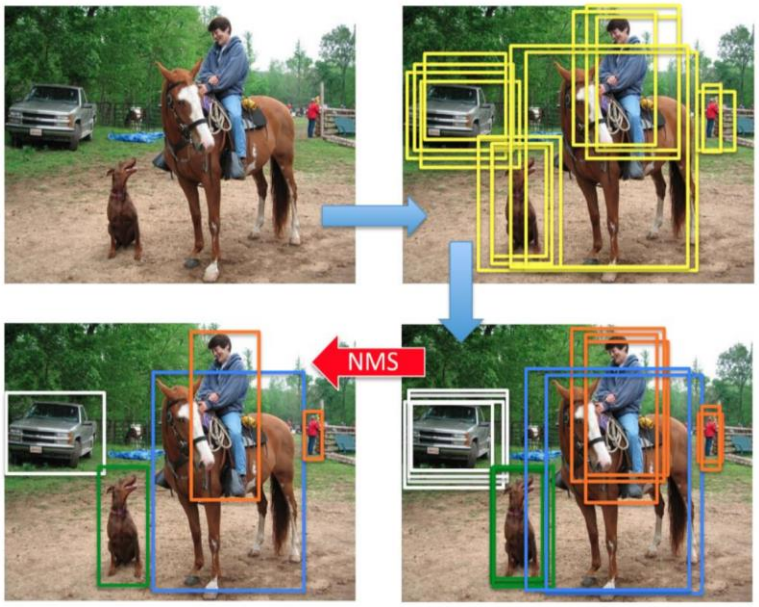


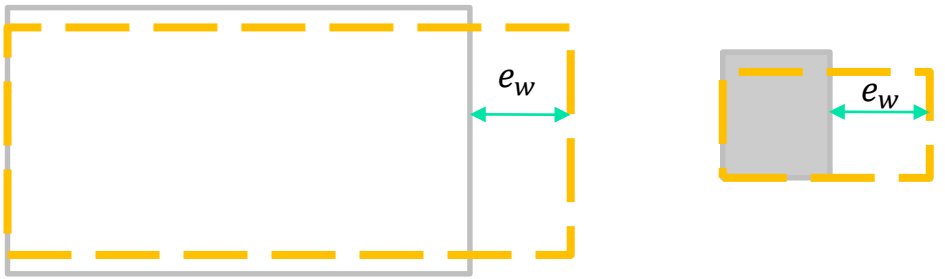


CLS token to the rest of tokens attention maps

Issues of Object Detection

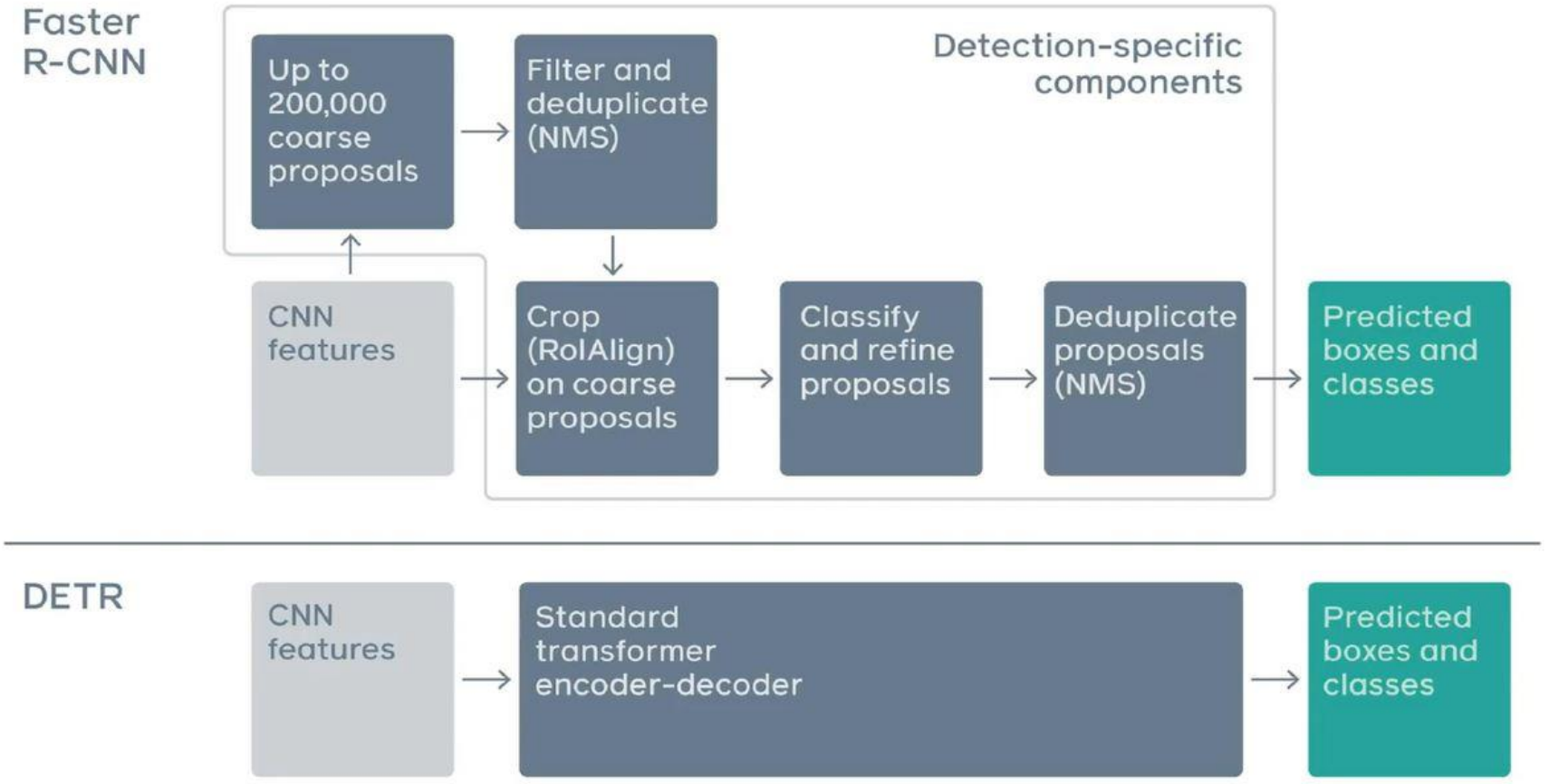
- Final performance highly depends on the initial anchor guess (location, number, size)
- Post-processing step to collapse duplicated detections (NMS) significantly influence the performance
- The location regression error has an issue with relative scaling
  - l_1 loss will have different scales for small and large bboxes even if the relative errors are similar



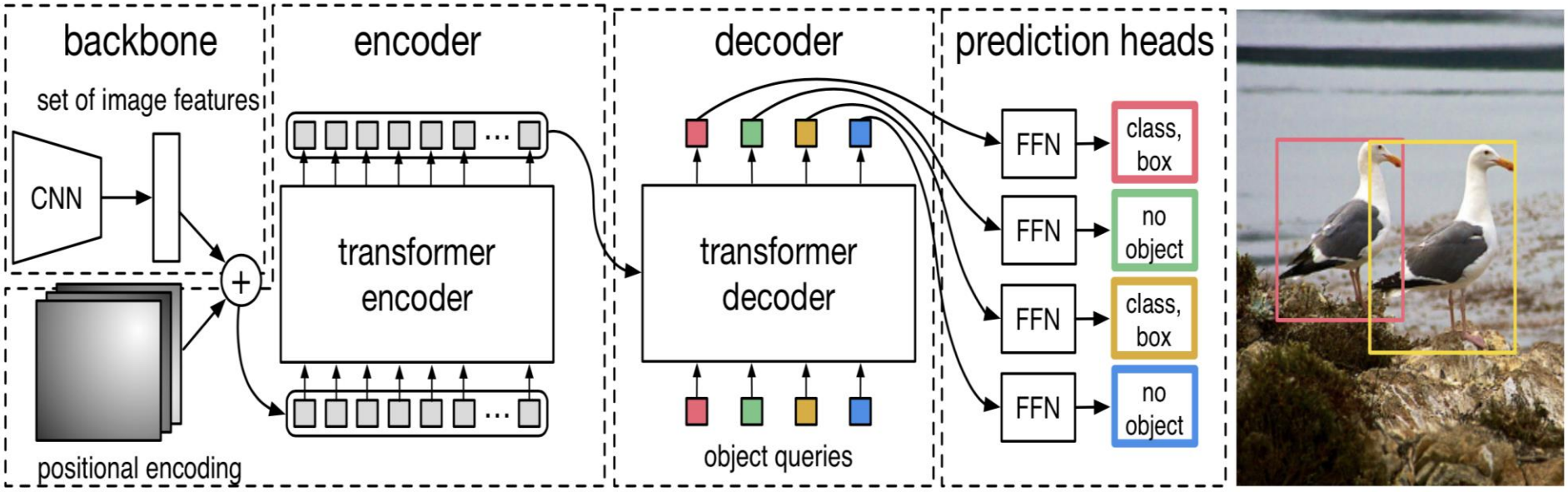


BBox regression in Yolo 2

Detector Pipelines



Detection Transformers (DETR)



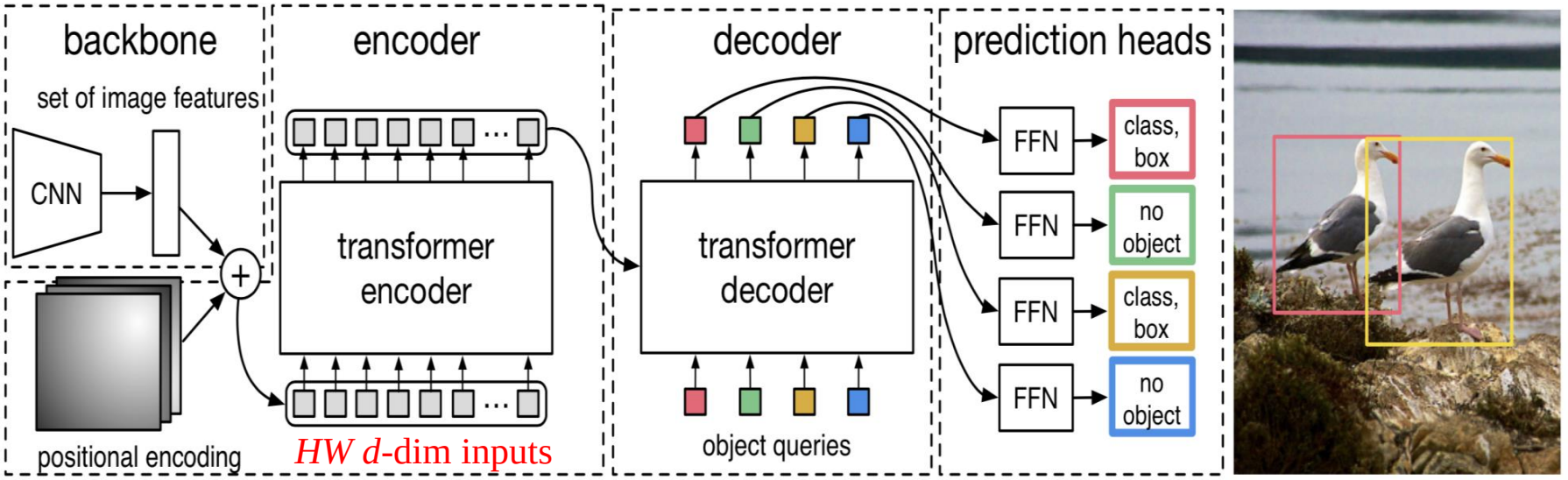
CNN for feature extraction

Encoder-decoder Transformer

Feed-Forward Network for classification/regression

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

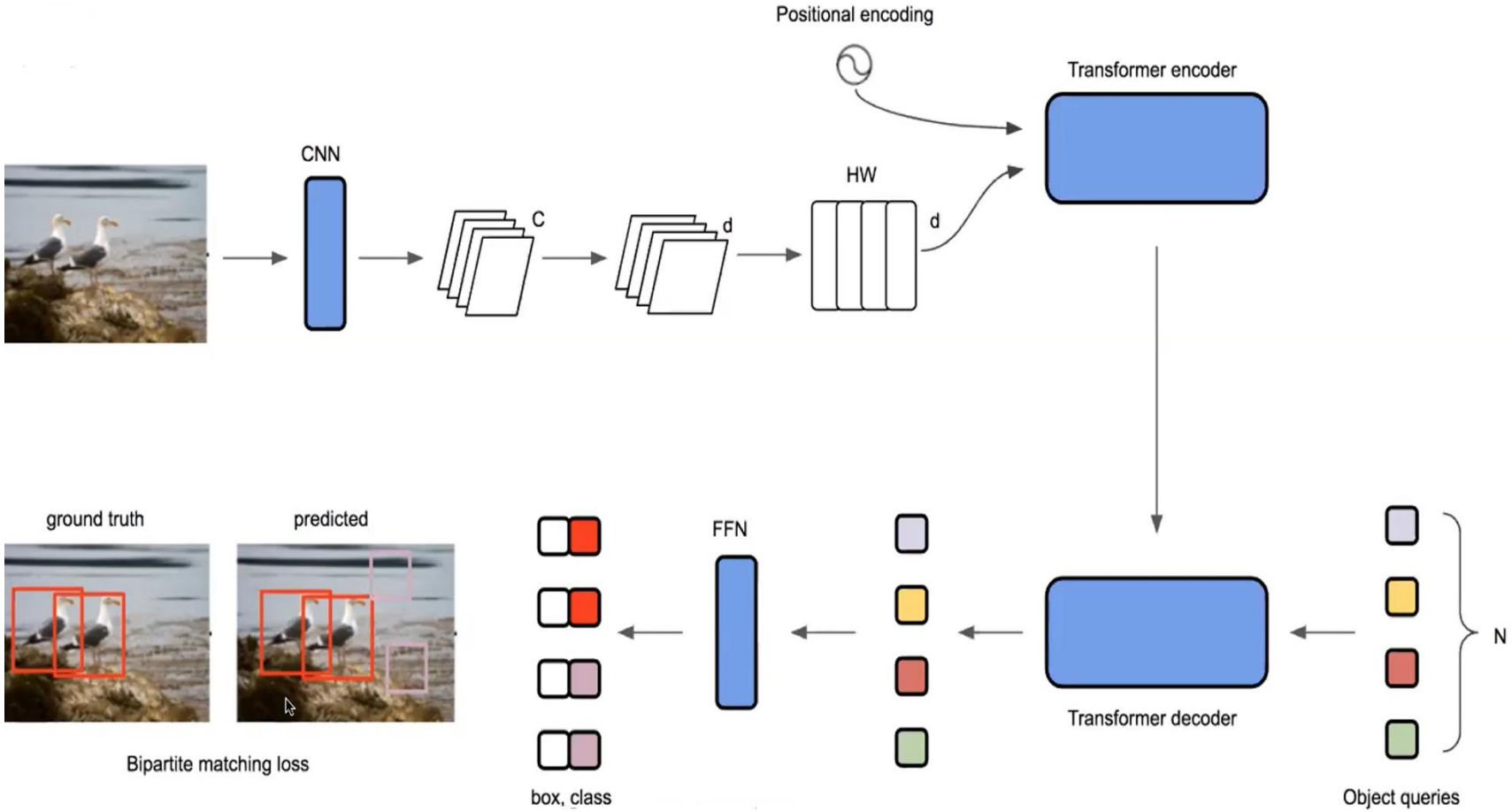
Detection Transformers (DETR)



$x_{img} \in \mathbb{R}^{3\times H_0\times W_0}$ —CNN→ $f \in \mathbb{R}^{C\times H\times W}$, $C = 2048, H, W = \frac{H_0}{32}, \frac{W_0}{32}$ —1x1 Conv→ $z_0 \in \mathbb{R}^{d\times H\times W}$, d: *a smaller dimention* —flattened→ $d \times HW$

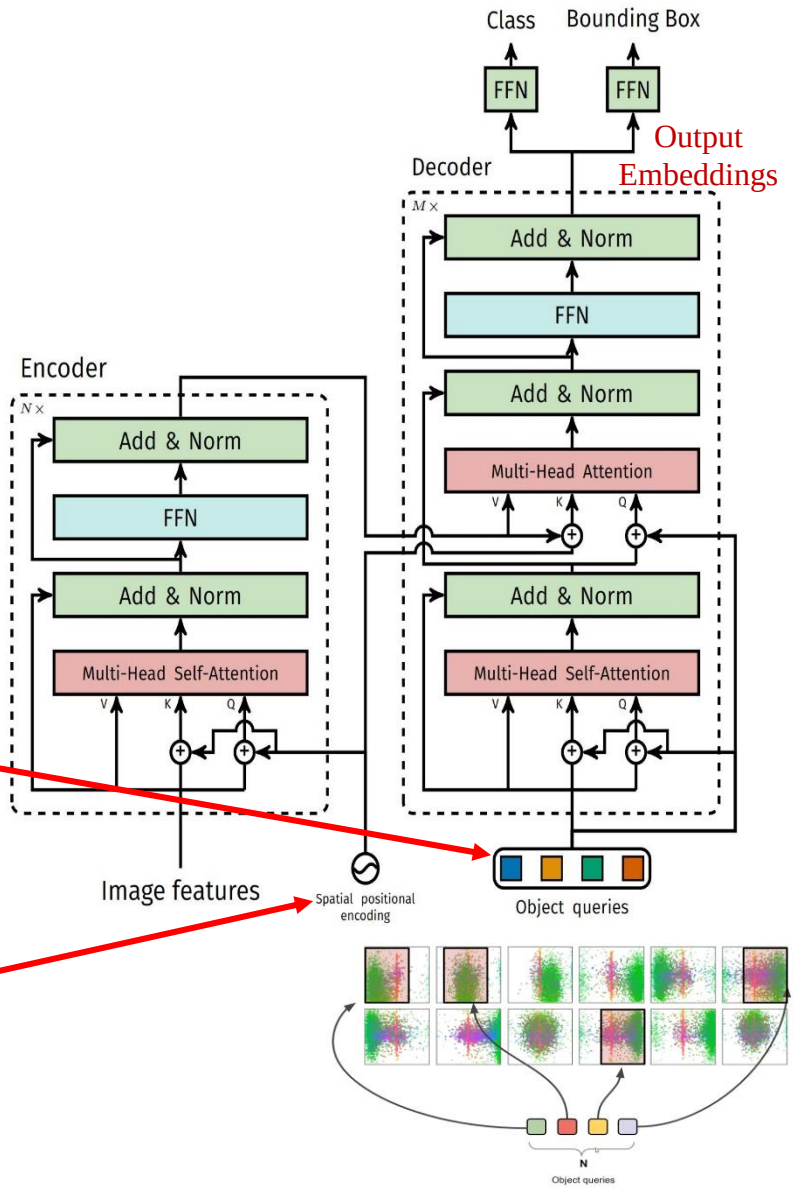
HW *emb of size* d —encoder→ N *learned positionsal* encodings (100 objecy queries) —decoder→ N *emb* —FFN (Feedforward Net)→ N *prediction*

Complete Pipeline of DETR

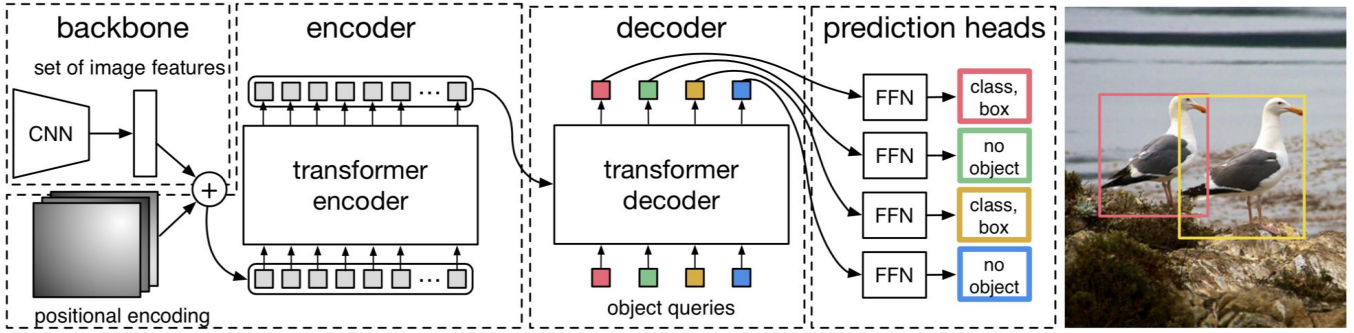


DETR Decoder

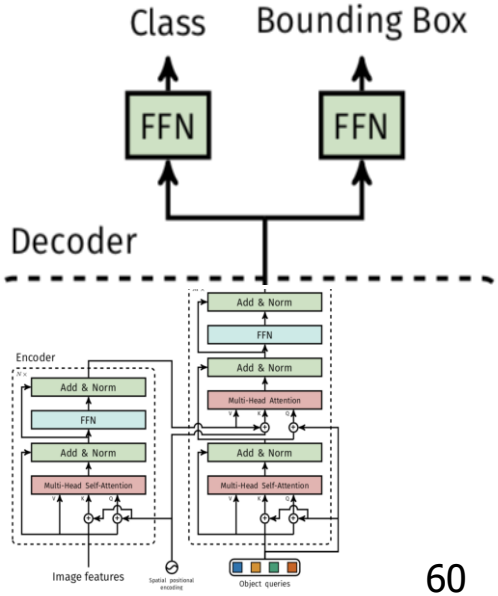
- DETR decodes *N* targets in parallel, no sequential autoregressive prediction.
- Decoder uses the *N* learned positional embeddings (called object queries), (equivalent to learned anchor).
- Spatial positional encodings are also used



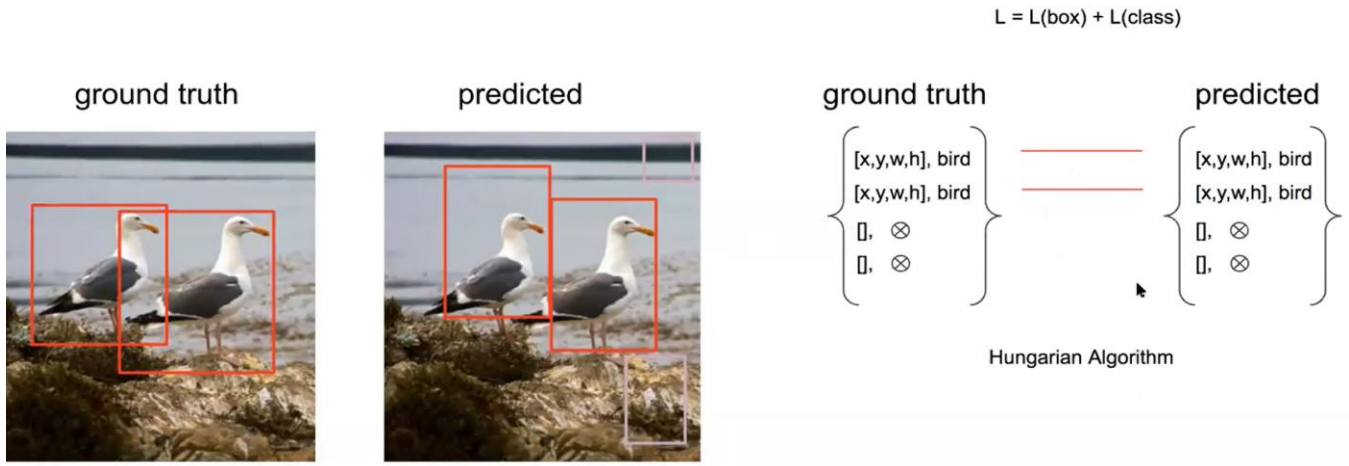
Shared FFN in DETR



- 3-layer perceptron with ReLU and hidden dimension *d*
- A linear projection layer for class prediction with softmax
- Predict a fixed-size set of *N* results, *N* is much larger than the number of objects in an image
- Predicts [center_x, center_y, height, width] for location
- A Ø class label for non-object (like background label)
- FFNs share parameters



Loss Function in DETR



y : *ground truth set of objects, pad y to size N with* $\emptyset$

$\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$: *prediction set of N objects*

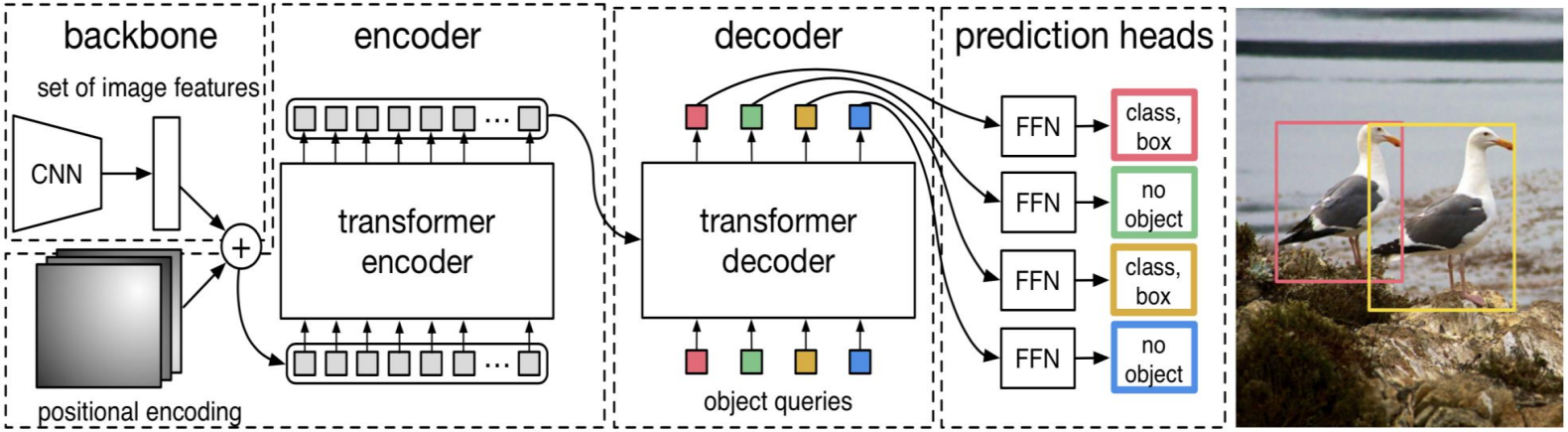
Bipartite matching

search for a permutation of N elements $\sigma \in \mathfrak{S}_N$

$$\hat{\sigma} = argmin_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} L_{match}(y_i, \hat{y}_{\sigma(i)})$$

- Match is done by Hungarian matching considering class probability and bbox (same as other detectors)

Loss Function in DETR

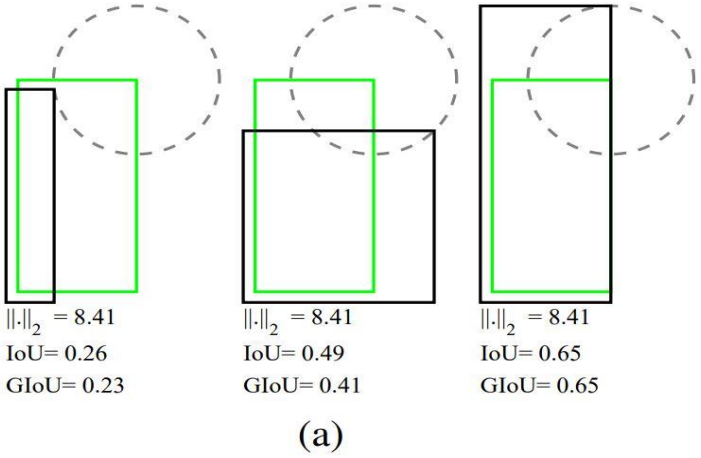


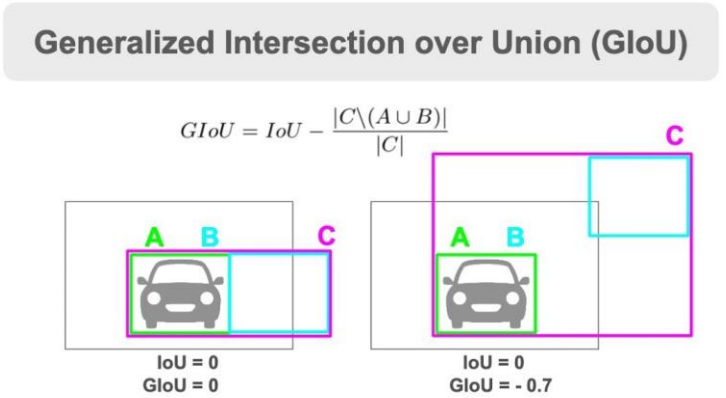
$$L_{Hungarian}(y_i, \hat{y}_{\sigma(i)}) = \sum_{i=1}^{N} [-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{I}_{\{c_i \neq \emptyset\}} L_{box}(b_i, \hat{b}_{\hat{\sigma}}(i))]$$

$$L_{box}(b_i, \hat{b}_{\hat{\sigma}(i)}) = \lambda_{iou} L_{iou}(b_i, \hat{b}_{\hat{\sigma}(i)}) + \lambda_{L1} \|b_i - \hat{b}_{\hat{\sigma}(i)}\|_1$$

Generalized IoU loss L1 regression loss

Generalized IoU for Metric and Loss Function





$$IOU = \frac{|A \cap B|}{|A \cup B|}$$

C is the smallest convex hull that encloses both A and B

$$GIOU = \frac{|A \cap B|}{|A \cup B|} - \frac{|C \backslash (A \cup B)|}{|C|}$$

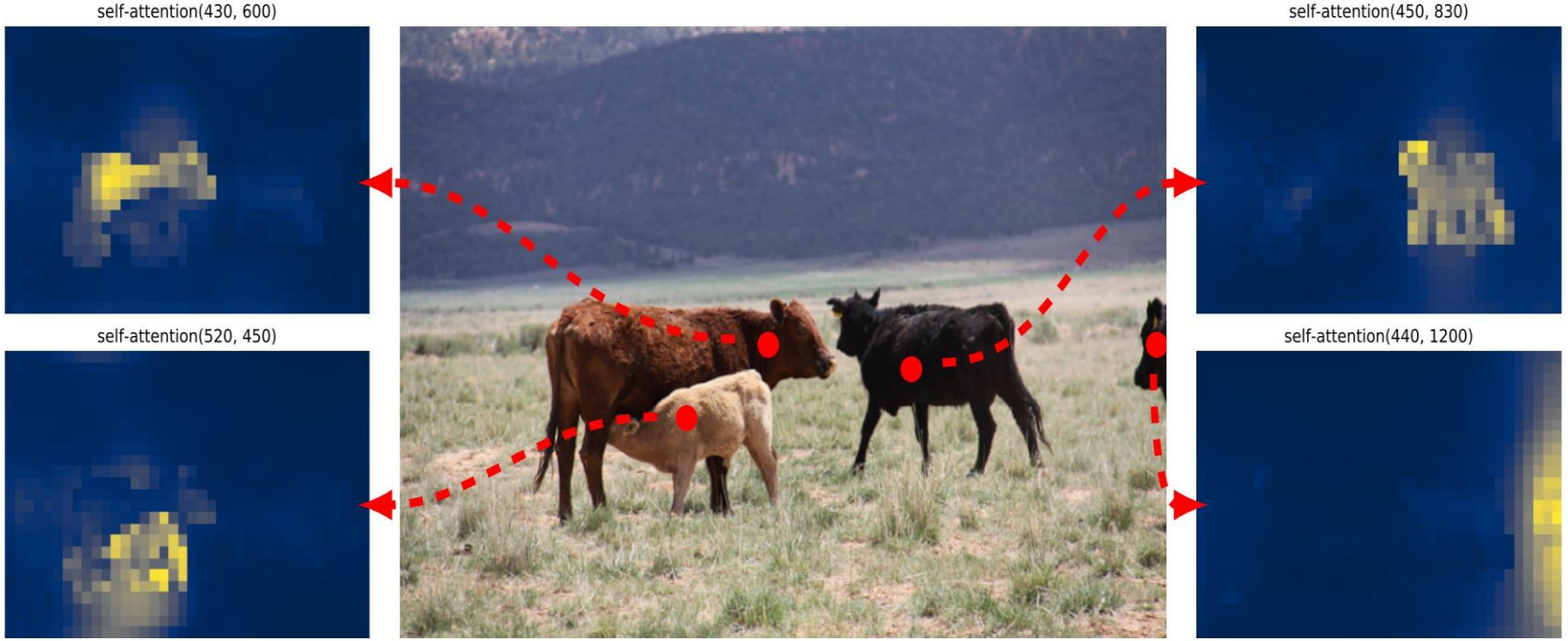
$$IoU = \frac{\mathcal{I}}{\mathcal{U}}, \text{ where } \mathcal{U} = A^p + A^g - \mathcal{I}.$$

$$GIoU = IoU - \frac{A^c - \mathcal{U}}{A^c}.$$

$$\mathcal{L}_{IoU} = 1 - IoU, \quad \mathcal{L}_{GIoU} = 1 - GIoU$$

Self Attention in DETR

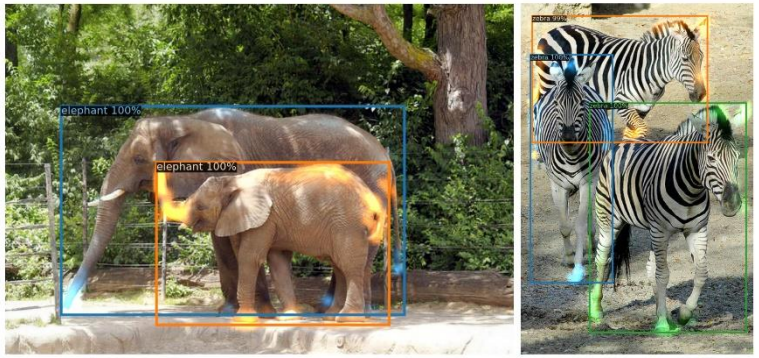
- The encoder is able to separate individual instances. Predictions are made with baseline DETR model on a validation set image.



attention maps of the last encoder layer of a trained model with respect to tokens associated with 4 red dots

Detection Transformers (DETR) Performance

- Training the baseline model for 300 epochs on 16 V100 GPUs takes 3 days, with 4 images per GPU (hence a total batch size of 64).

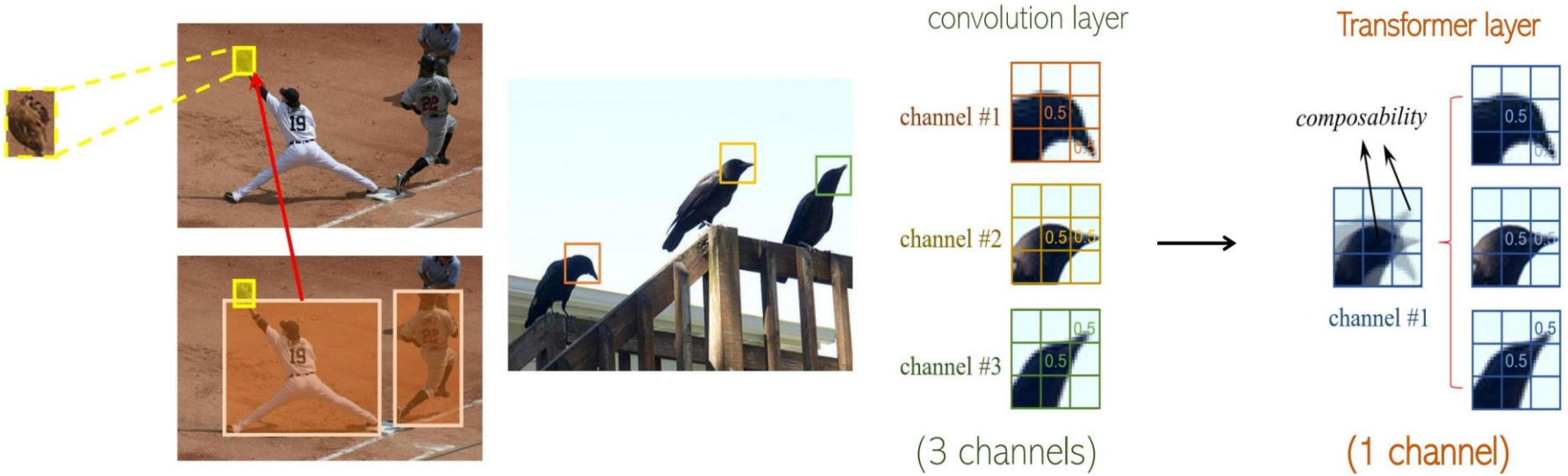


All use a ResNet 50 backbone

Model	GFLOPS/FPS	#params	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-DC5	320/16	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180/26	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-R101-FPN	246/20	60M	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-DC5+	320/16	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180/26	42M	42.0	62.1	45.5	26.6	45.4	53.4
Faster RCNN-R101-FPN+	246/20	60M	44.0	63.9	**47.8**	**27.2**	48.1	56.0
DETR	86/28	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5 Dilated C5	187/12	41M	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	152/20	60M	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	253/10	60M	**44.9**	**64.7**	47.7	23.7	**49.5**	**62.3**

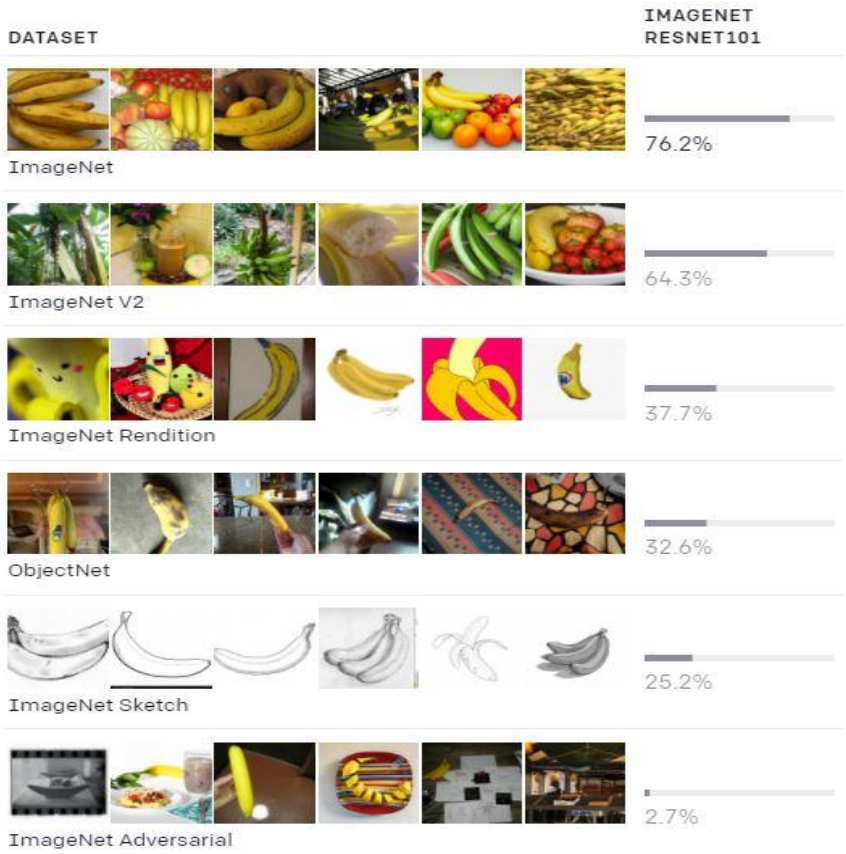
Why Transformer in Vision?

- A glove can be detected easier if we know a baseball player
- Features are composed of a linear combination of attention-related features in a transformer, unlike we need 3 separate convolution kernels (3 separate feature maps) in CNN



Han Hu, "Swin Transformer and 5 Reasons to Use Transformer/Attention in Computer Vision," tutorial talk in CVPR 2021, https://ancientmooner.github.io/doc/SwinTransformer_FiveReasons.pdf

Poor Generalization of Vision Tasks to Different Datasets



The ImageNet dataset required over 25,000 workers to annotate 14 million images for 22,000 object categories

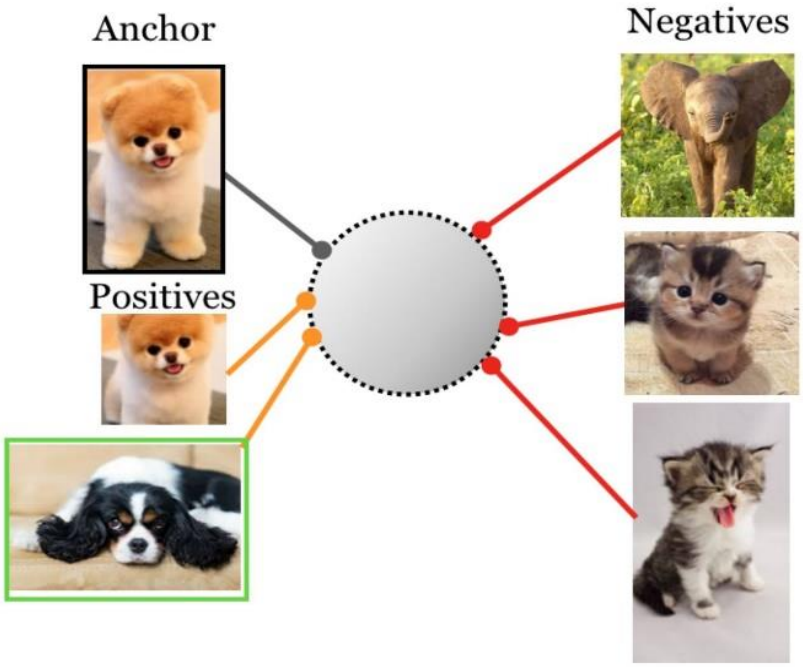
- Typical vision datasets are labor intensive and costly to create while teaching only a narrow set of visual concepts
- Standard vision models are good at one task and one task only, and require significant effort to adapt to a new task;
- Models that perform well on benchmarks have disappointingly poor performance on stress tests

Contrastive Language-Image Pre-training (CLIP)

- Builds on zero-shot transfer, natural language supervision, and multimodal learning
- Given an image, predict which out of a set of 32,768 randomly sampled text snippets, was actually paired with it in our dataset
- In small to medium scale experiments, we found that the contrastive loss used by CLIP is 4x to 10x more efficient at zero-shot ImageNet classification (cross-entropy).
- The second choice was the adoption of the Vision Transformer, which gave us a further 3x gain in compute efficiency over a standard ResNet.

Supervised Contrastive Loss

- Generalization to an arbitrary number of positives
- Contrastive power increases with more negatives.



Data augmentation to 2N to ensure we can always find positives

$$\mathcal{L}^{sup} = \sum_{i=1}^{2N} \mathcal{L}_i^{sup}$$

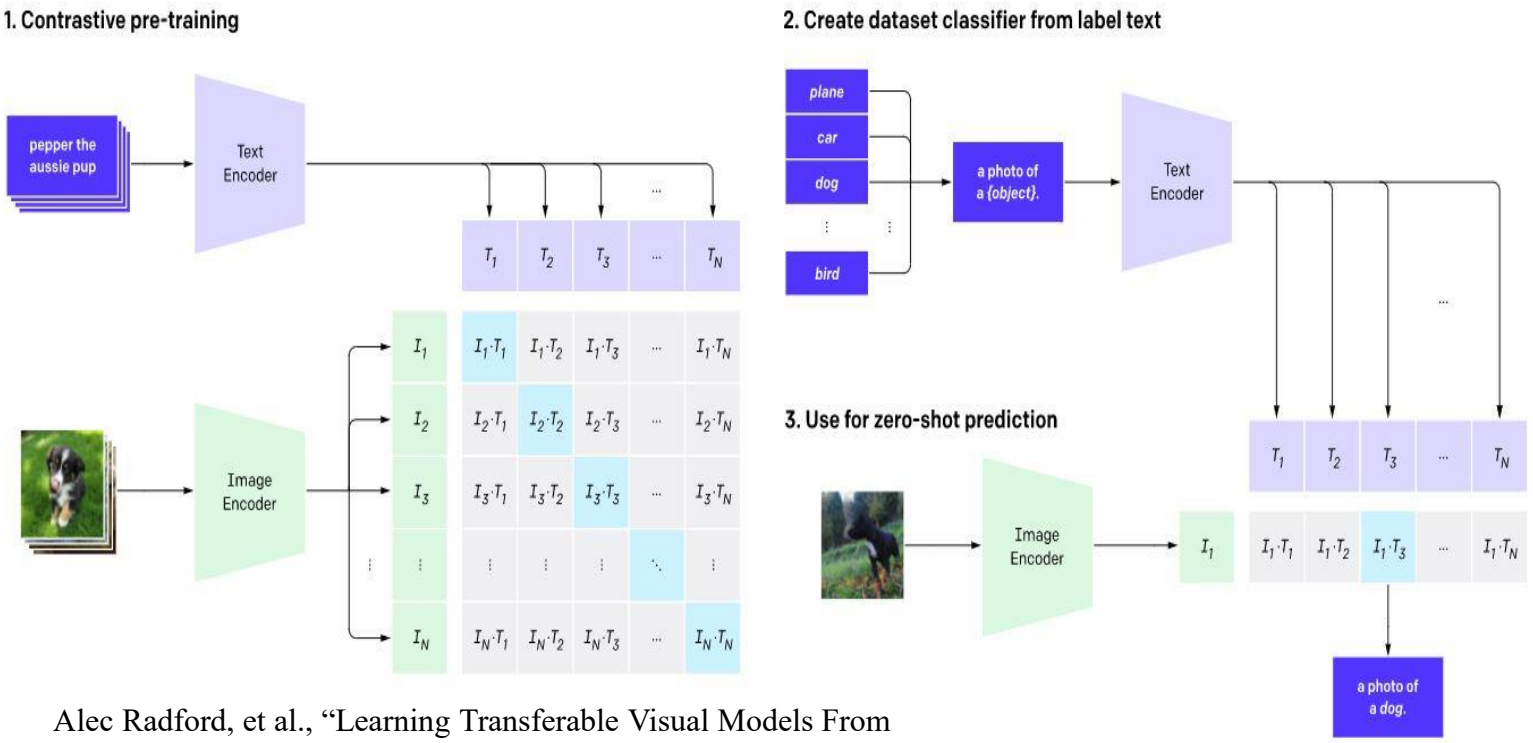
Normalized embedding

$$\mathcal{L}_i^{sup} = \frac{-1}{2N_{\tilde{\boldsymbol{y}}_i} - 1} \sum_{j=1}^{2N} \mathbf{1}_{i \neq j} \cdot \mathbf{1}_{\tilde{\boldsymbol{y}}_i = \tilde{\boldsymbol{y}}_j} \cdot \log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_j / \tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{i \neq k} \cdot \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_k / \tau\right)}$$

where $N_{\tilde{\boldsymbol{y}}_i}$ is the total number of images in the minibatch that have the same label, $\tilde{\boldsymbol{y}}_i$, as the anchor, i.

Training and Use of CLIP

- 400+ millions image-text pairs (images and descriptions scraped from the Internet), trained on 256 GPUs for 2 weeks

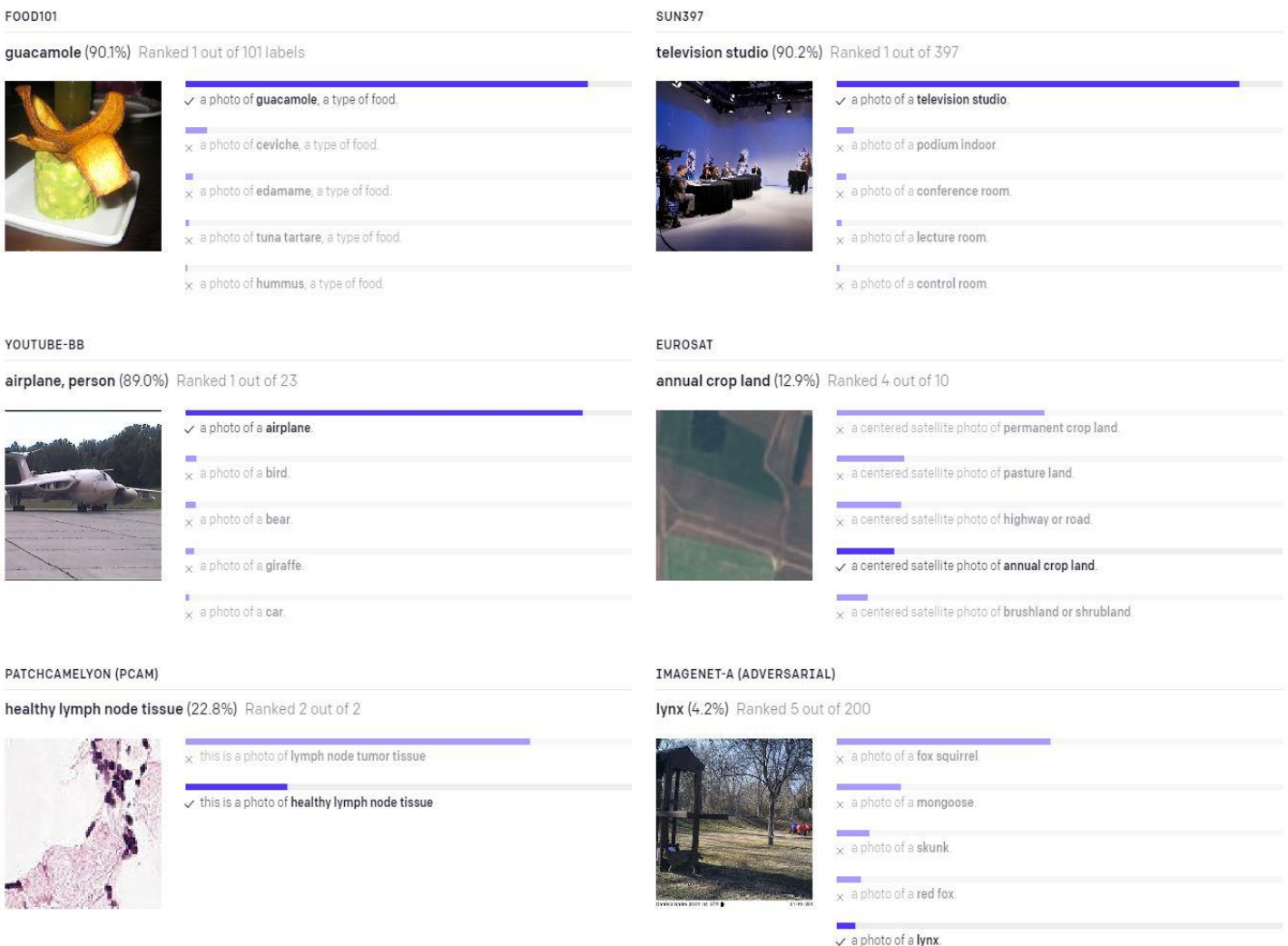


Alec Radford, et al., "Learning Transferable Visual Models From Natural Language Supervision," https://arxiv.org/pdf/2103.00020.pdf

CLIP Performance

DATASET	IMAGENET RESNET101	CLIP VIT-L
 ImageNet	76.2%	76.2%
 ImageNet V2	64.3%	70.1%
 ImageNet Rendition	37.7%	88.9%
 ObjectNet	32.6%	72.3%
 ImageNet Sketch	25.2%	60.2%
 ImageNet Adversarial	2.7%	77.1%

CLIP Zero-Shot Performance



Happy Holidays

Wish You an Enjoyable and Productive Graduate Study