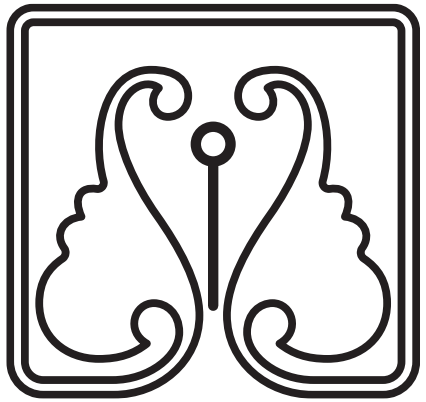
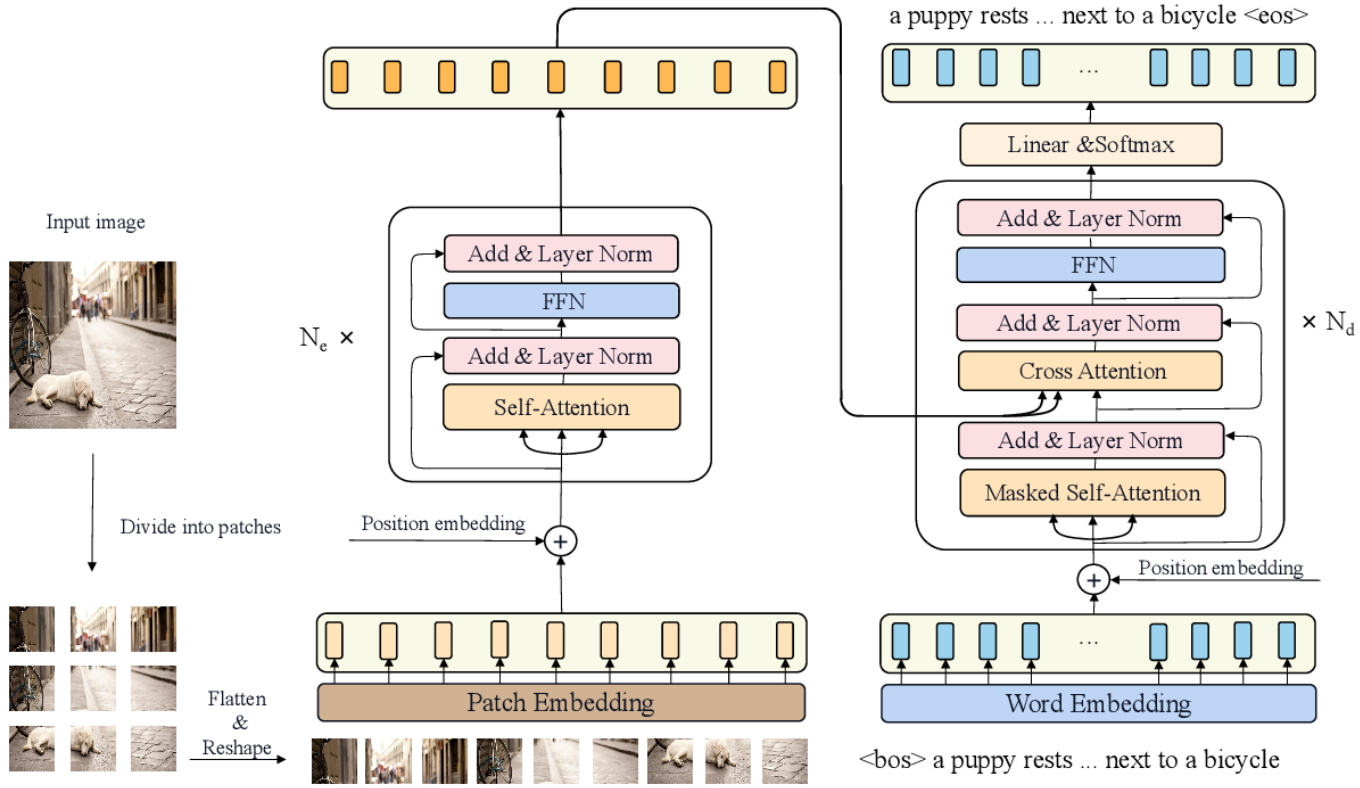
Image Captioning
Vision Transformer (ViT) and GPT2



A Shahbahrami, “Big Data”,

University of Guilan, 2022.

Mohammad Taghizadeh



Outline

1. Introduction and Motivation and challenges
    - Computer Vision tasks and image captioning applications
    - Deep neural networks
2. Previous related works on image captioning using natural language processing
    - Recurrent neural networks
    - Long short-term memory networks
    - Attention mechanism on image captioning
    - Drawbacks and Challenges
3. Vision Transformer
    - Attention is all you need (Transformers)
    - An Image is Worth 16x16 Words Transformers for Image Recognition at Scale (ViT) on image classification
4. Vision Transformer and image captioning
    - Vision Encoder Decoder image captioning
    - Language modeling and GPT2
5. Results and implementation

Computer Vision Tasks

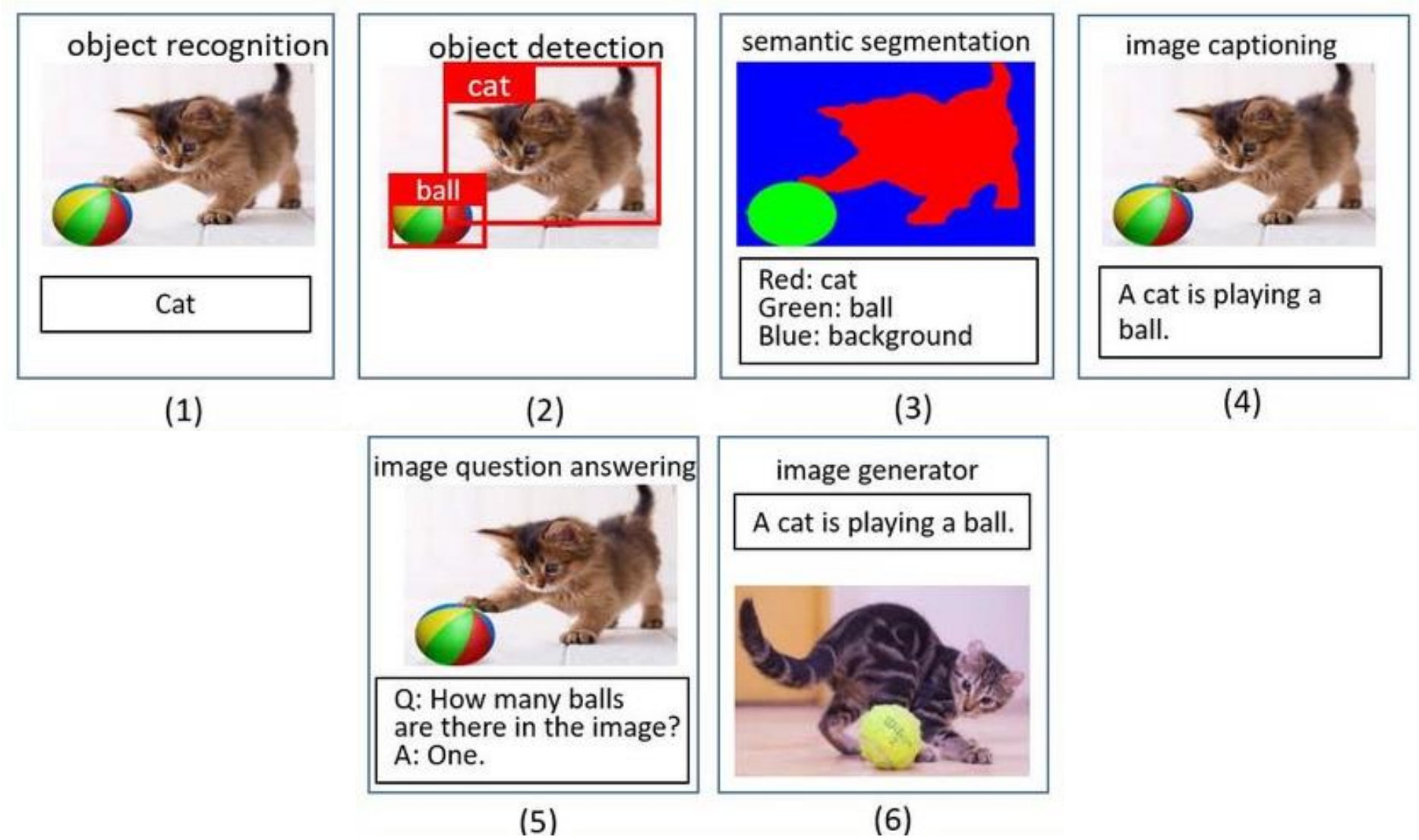


Figure 1: Computer Vision Tasks, https://www.researchgate.net/figure/The-differences-among-six-popular-computer-vision-tasks-1-Object-recognition-sometimes_fig1_335013237

Image Captioning, Motivation and Challenges

Some applications of image captioning

Social media posts and search in the bulk of images with captions
Assistance for visually impaired
Media and publishing houses
Recommendations for editing applications

Some of the challenges we have in the image captioning task

Challenges of the natural world: for example, the challenge of parallax error
The challenge of designing models and architectures

- How to construct grammar-based captions?
- Captions length?
- Which part of the image is the key part?
- Accurate dataset and corpus is very important

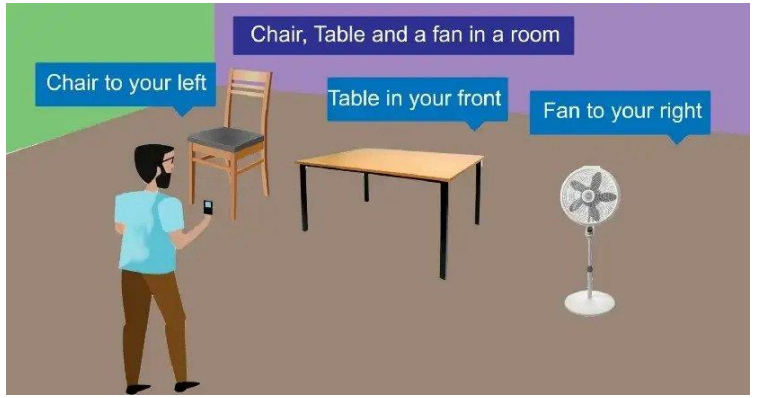
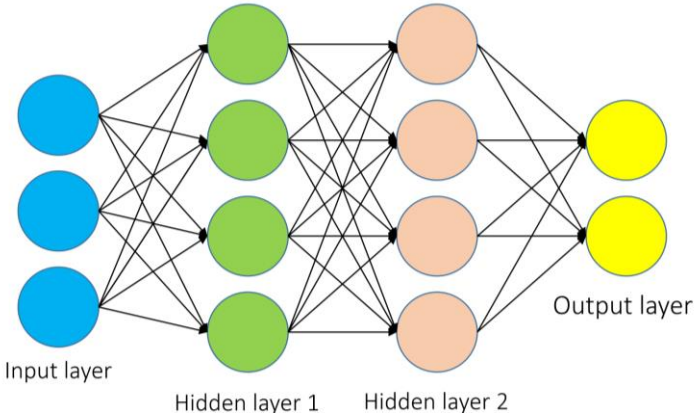
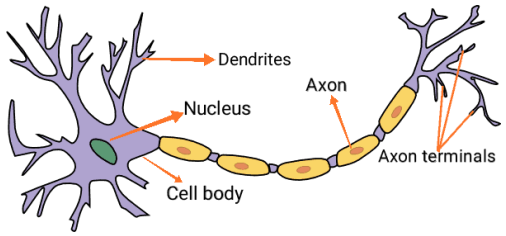


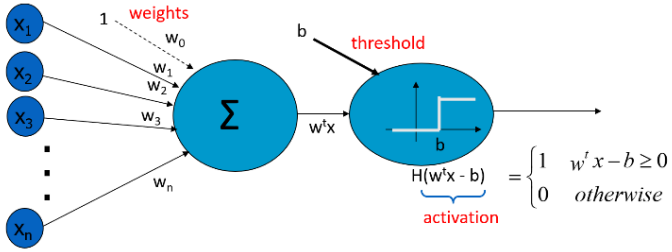
Figure 2: Image captioning can help visually impaired people and can be used in question-answering applications or social media

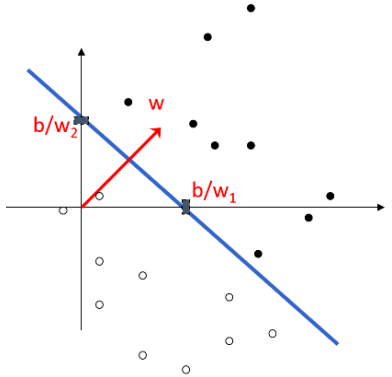
Types of common neural networks and their applications

1-Feed forward network









2-Deep neural network

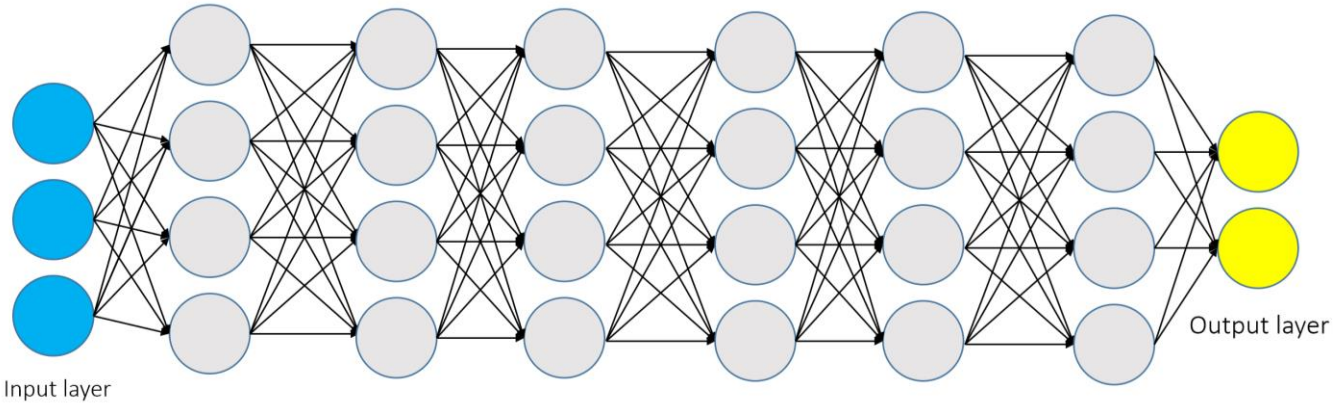


Figure 3: in deep neural networks, we have several hidden layers (at least more than 2 hidden layers), and these more hidden layers and greater depth make models much stronger with the ability to learn complex tasks.

Types of common neural networks and their applications
3-Convolutional neural network

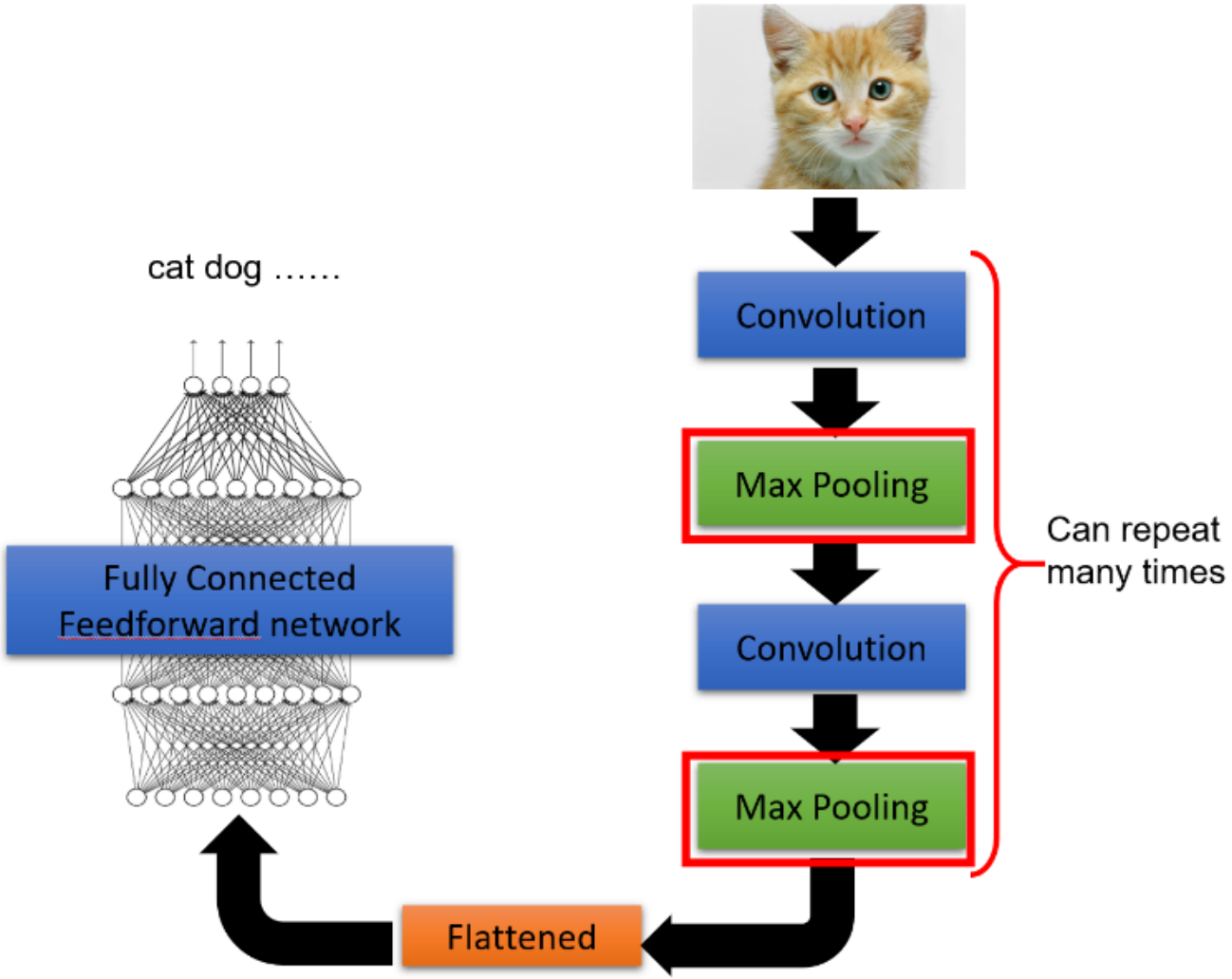


Figure 4: Convolutional neural networks are extremely suitable for computer vision tasks.

Types of common neural networks and their applications
4-Generative adversarial nets

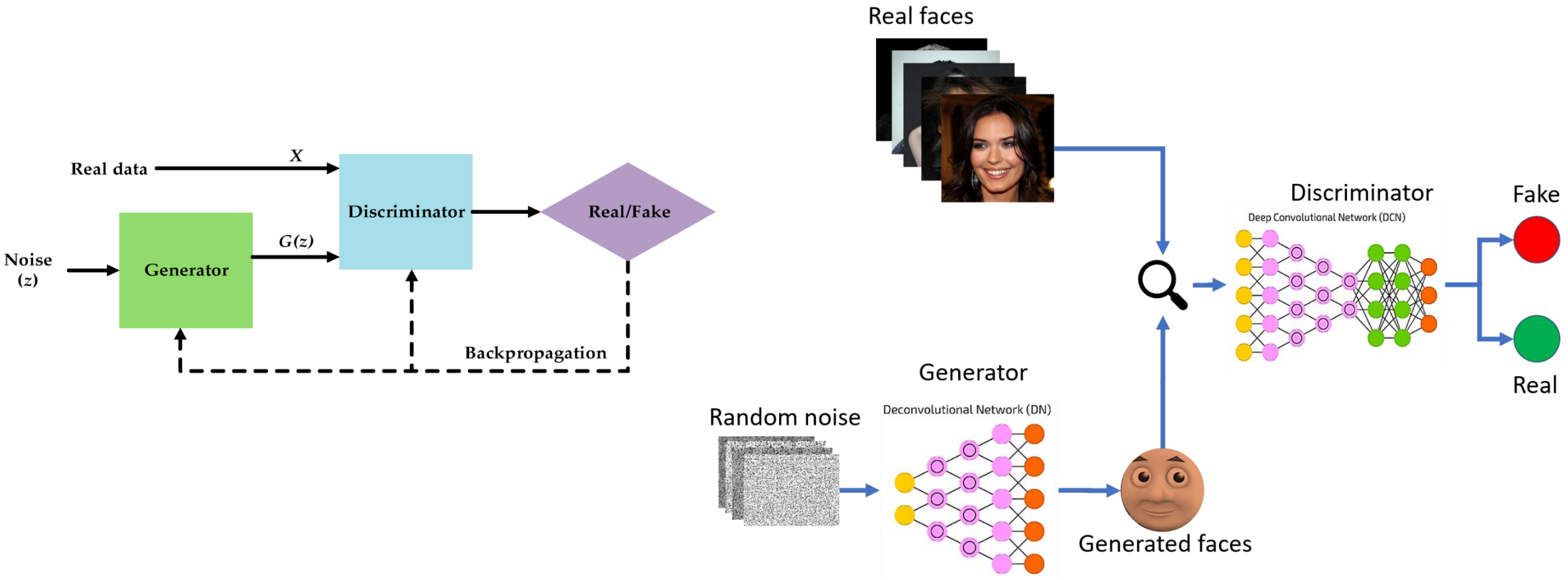


Figure 5: We have two parts in GANs and These networks are suitable for producing real-like fake data in small datasets

1. Generator: Produces fake samples
2. Discriminator: It detects that the generated

Types of common neural networks and their applications
5-Recurrent neural network

Some of applications

Image captioning

Time series prediction such as stock price prediction

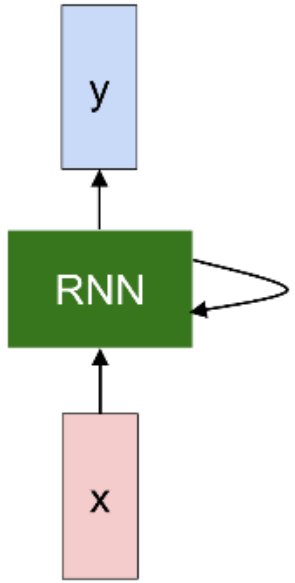
Natural language processing

Language modeling

Text classification and sentiment analysis

Natural machine translation

Speech recognition

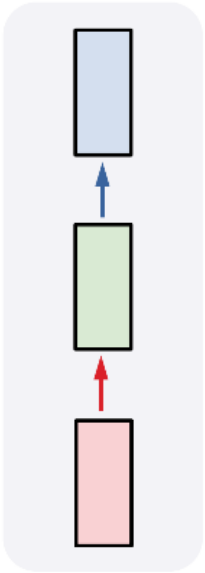


$$h_t = f_W(h_{t-1}, x_t)$$

حالت جدید — یک تابع با پارامترهای W — حالت قبلی — بردار ورودی در لحظه فعلی

Figure 6: Unlike feed-forward networks, have a hidden, which is practically the memory of these networks

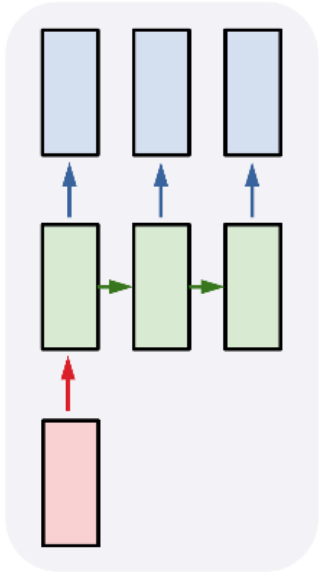
Different structures of Recurrent neural networks



1:1

Feed-Forward Networks

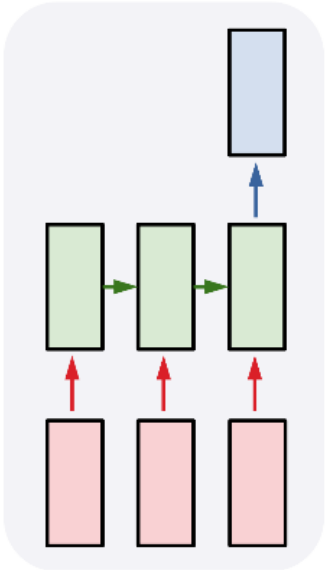
Example: Image Classification



1:N

RNN

Example: Image Captioning

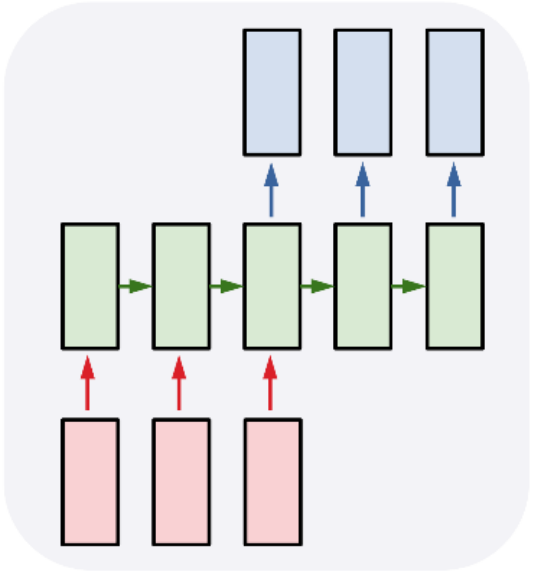


N:1

RNN

Example 1: Text Classification

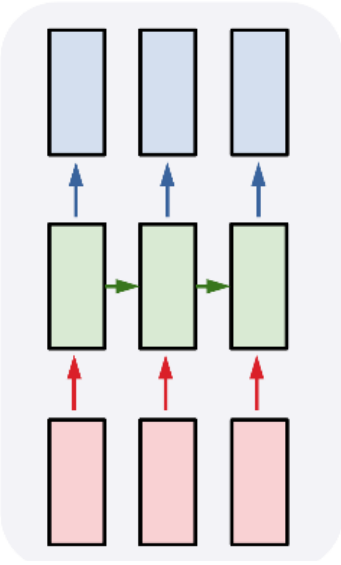
Example 2: Sentiment Analysis



N:N

RNN

Example: Machine Translation



N:N

RNN

Example: Processing of frame by frame in a video

Figure 7: Common Structures of RNNs

Why RNN? Normal feed-forward net VS RNN

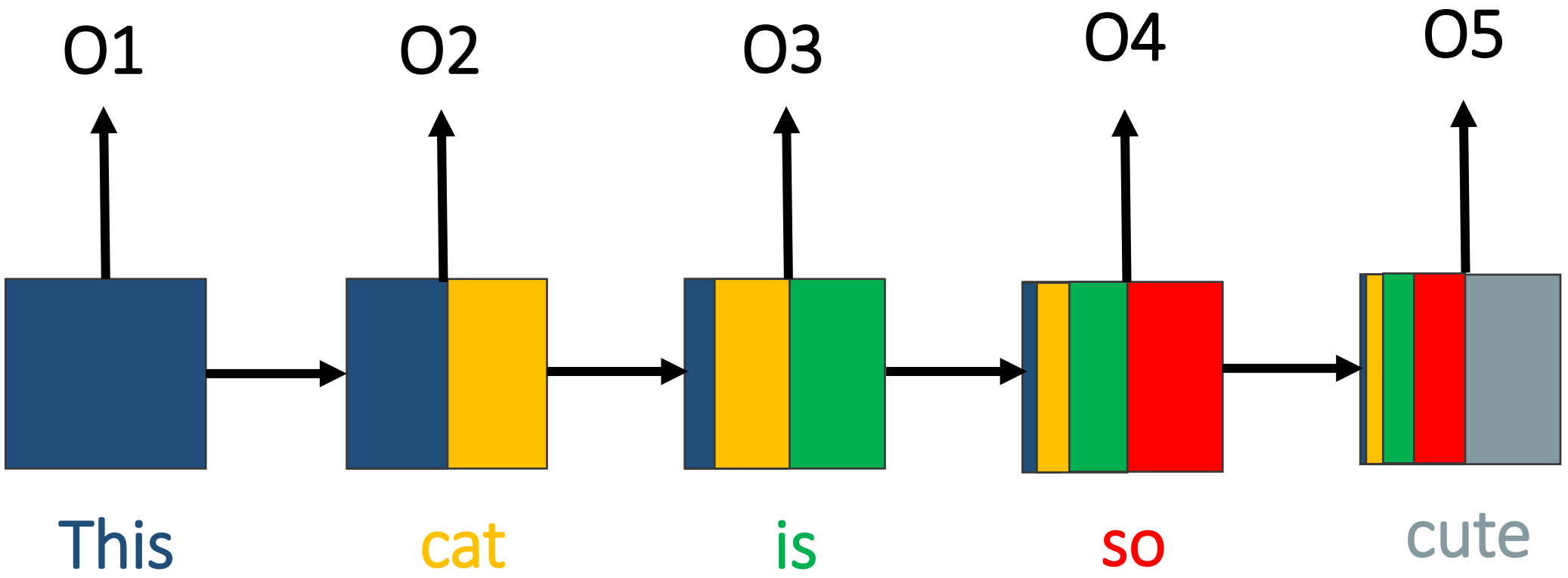


Figure 8: We gave a sentence to the RNN network, and in each time step or a time state, the network pays attention to the past inputs in addition to considering the current state input and the semantics of the sentence.

Deep dive to RNN Architecture

x_t : ورودی در لحظه t ام

y_t : خروجی در لحظه t ام

h_t : مقدار state در لحظه t ام

w_{xh} : وزن های لایه ورودی به لایه مخفی

w_{hh} : وزن های لایه مخفی یا state

w_{hy} : وزن های لایه خروجی یعنی از لایه مخفی به خروجی

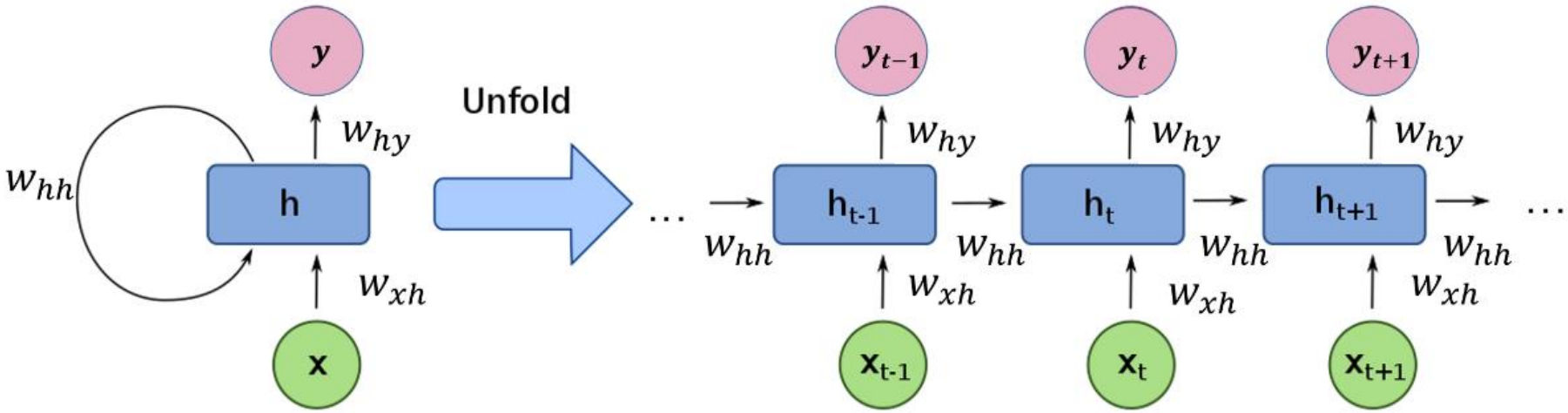


Figure 9: We see in this figure an Unfolded RNN in time steps. Recurrent neural networks generate a state or memory at each time step

RNN Models : Long term dependency problem

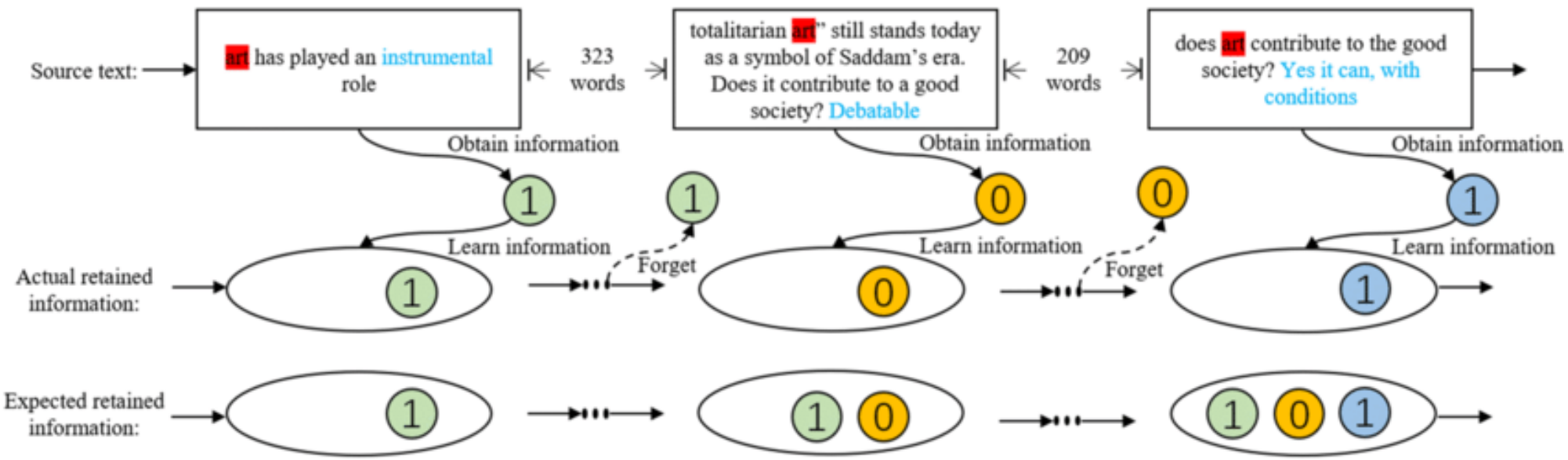
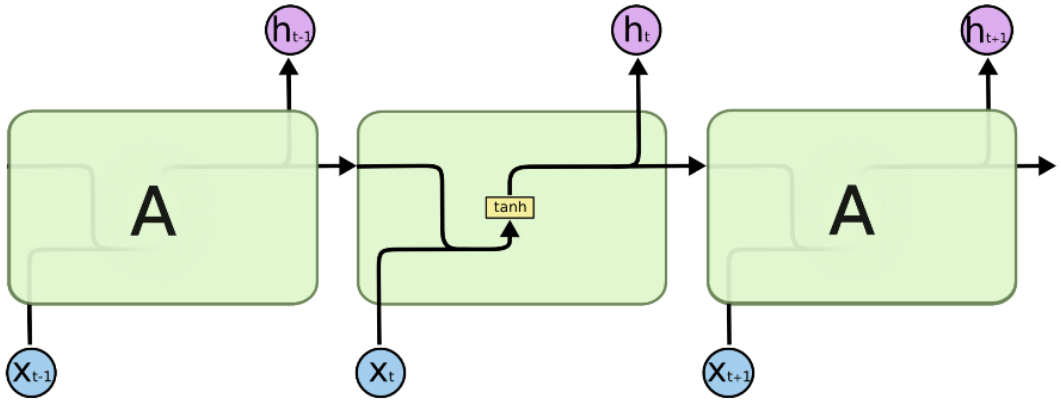


Figure 10: We see in this figure one of the most important problems of RNNs was the problem of long-term dependency, which was caused by the vanishing gradient challenge.

Long short-term memory



RNNs architecture

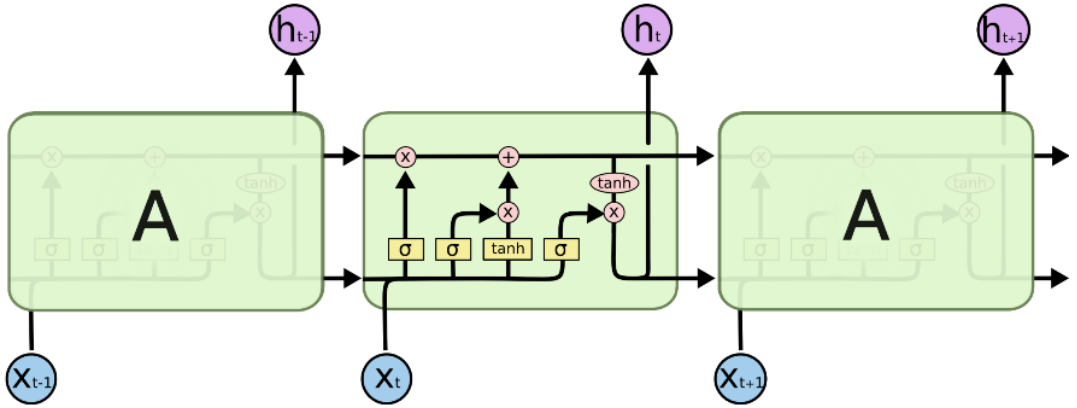


Figure 11: LSTM tries to learn long-term dependencies in sequences by controlling the flow of information input to memory or cell state using four internal gates

Long short-term memory architecture

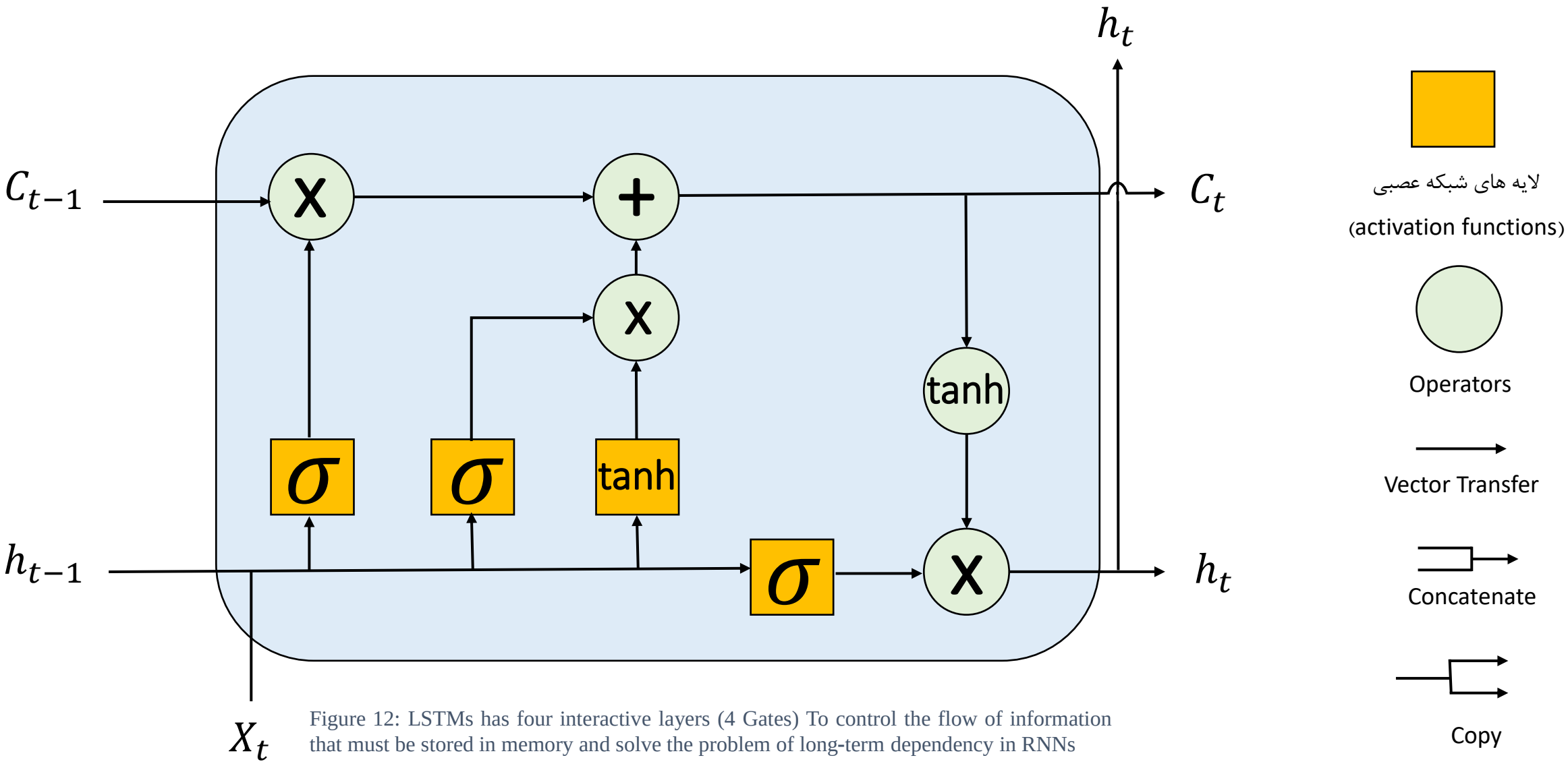


Figure 12: LSTMs has four interactive layers (4 Gates) To control the flow of information that must be stored in memory and solve the problem of long-term dependency in RNNs

Image captioning: Attention mechanism
Seq2Seq models (Encoder to Decoder or End to End)

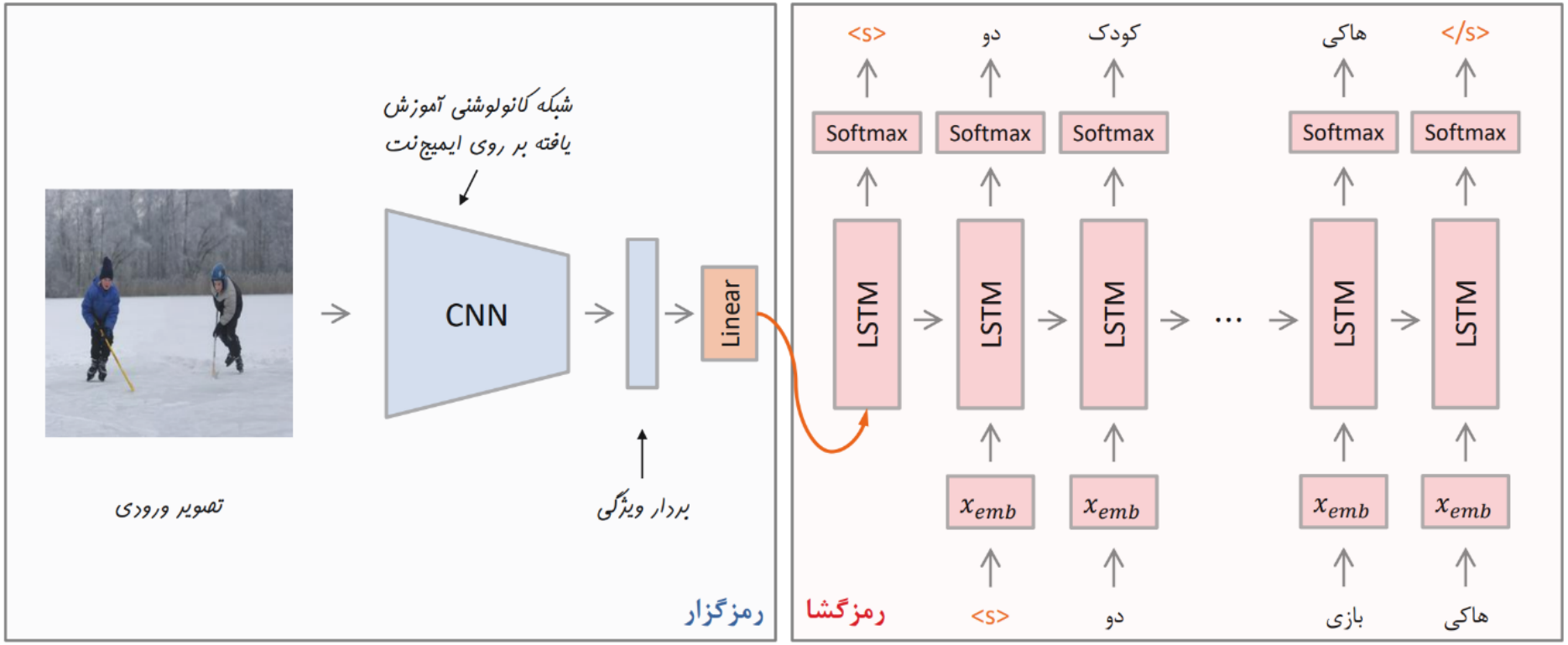


Figure 13: Image Captioning Using Seq2Seq LSTM models. In this model, we have two components, Encoder, and Decoder.

- Encoder: It receives the input image and creates an internal representation of the image and finally presents the image with an embedding vector.
- Decoder: The internal representation or embedding vector receives the image and creates words to describe the image by decoding it by an LSTM network.

Image captioning: Attention mechanism

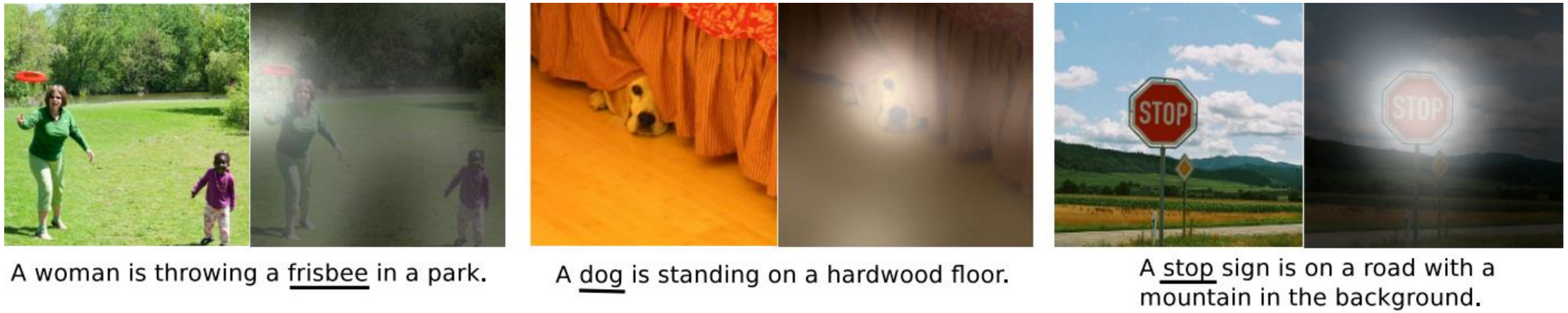
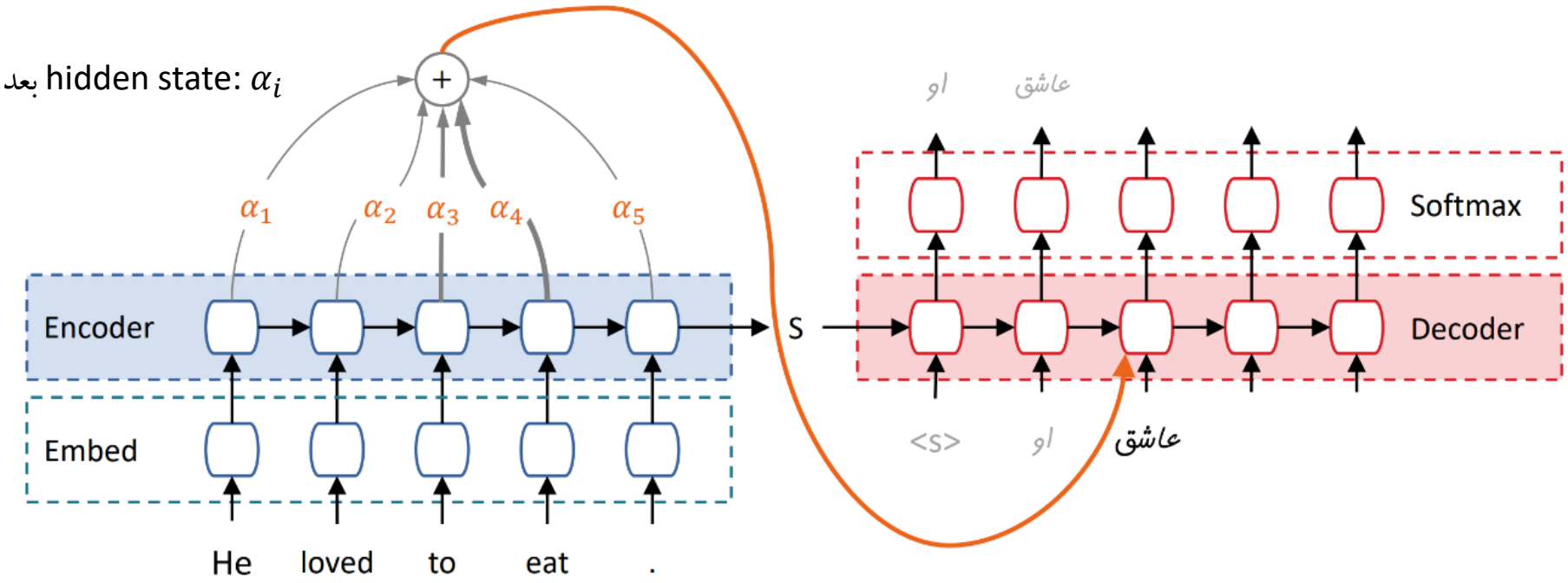


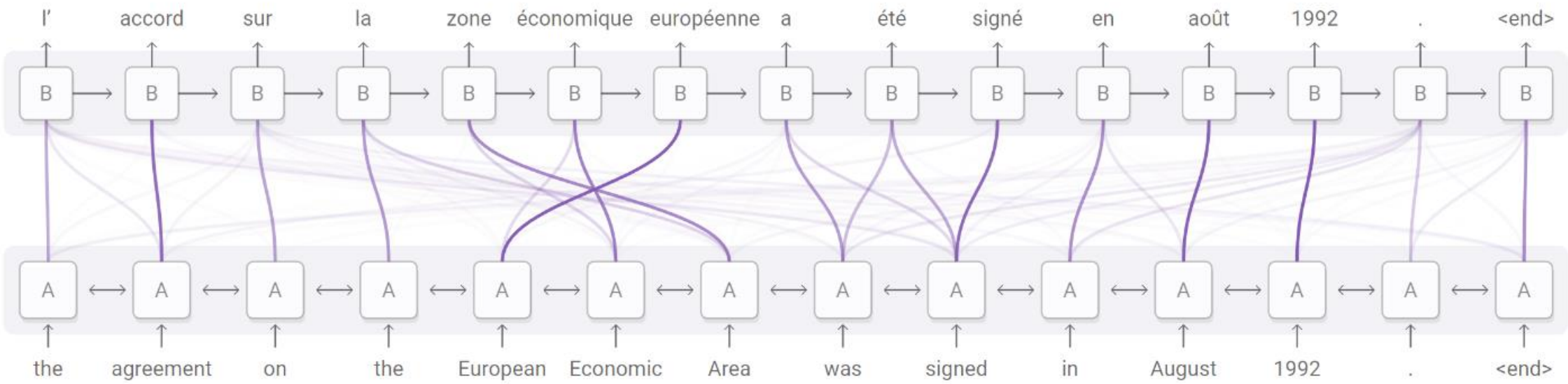
Figure 14: Attention Mechanism in Image captioning, https://distill.pub/2016/augmented-rnns/

Machine translation and Attention mechanism



Word embedding score: Dot Production

Machine translation and Attention mechanism



https://distill.pub/2016/augmented-rnns/

Word embedding and dot production

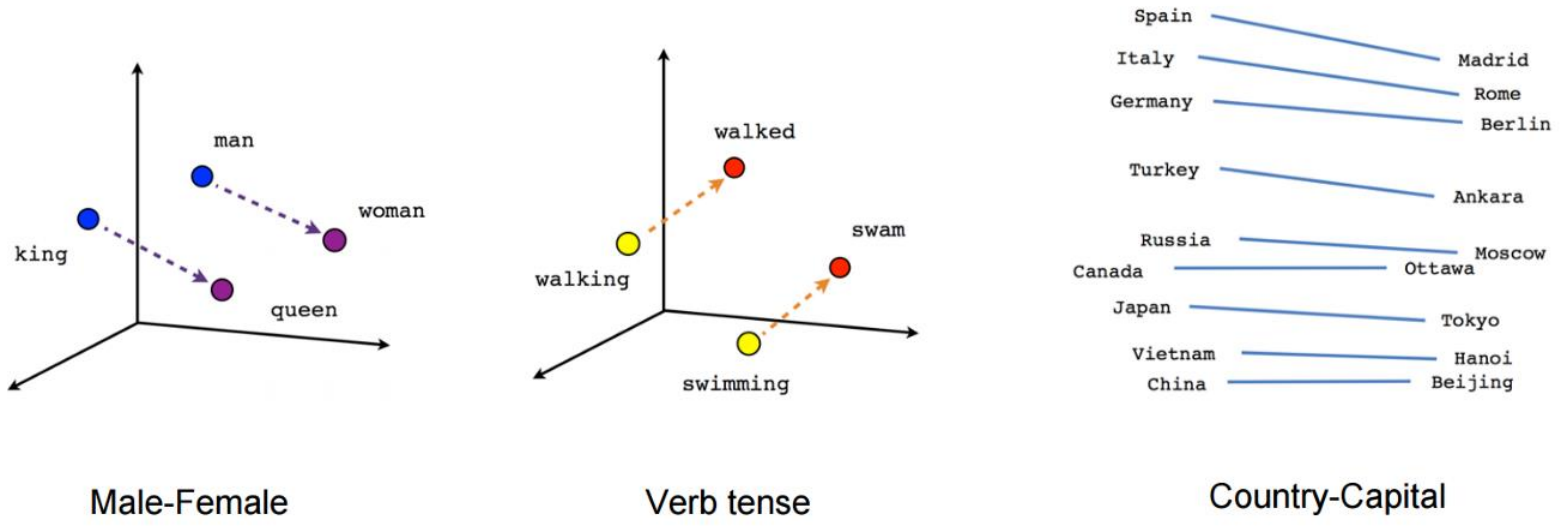
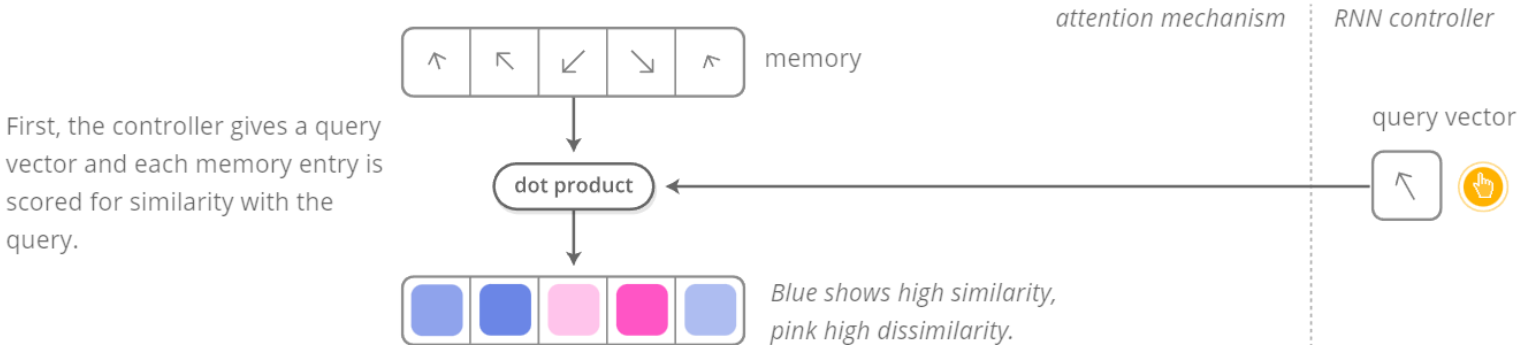


Figure 15: Word Embedding, https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8

dot production example



Transformer: Attention is all you need

RNN Challenges: Gradient Problem, Long term dependency, parallel computing

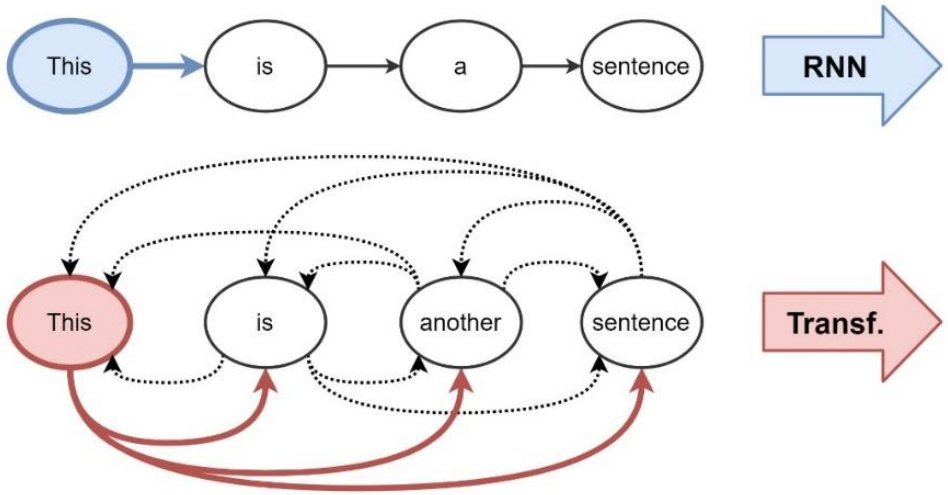
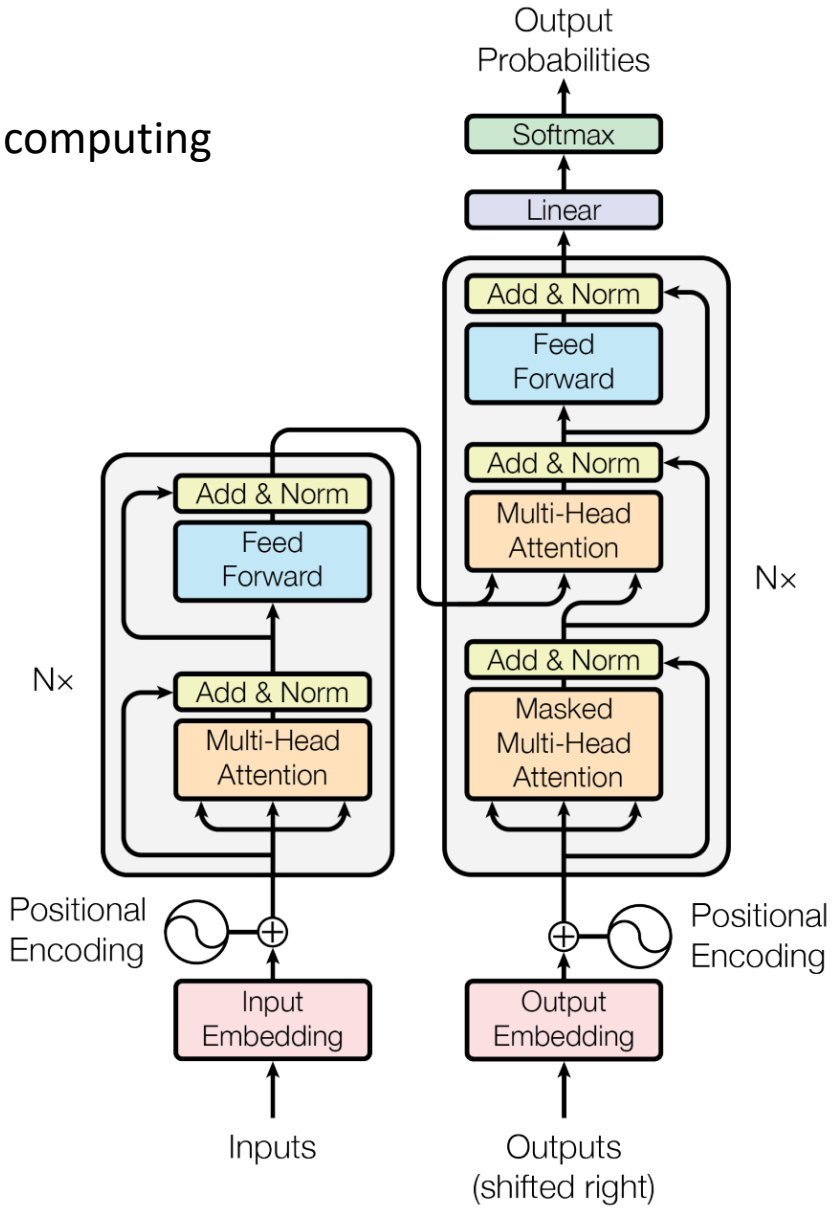
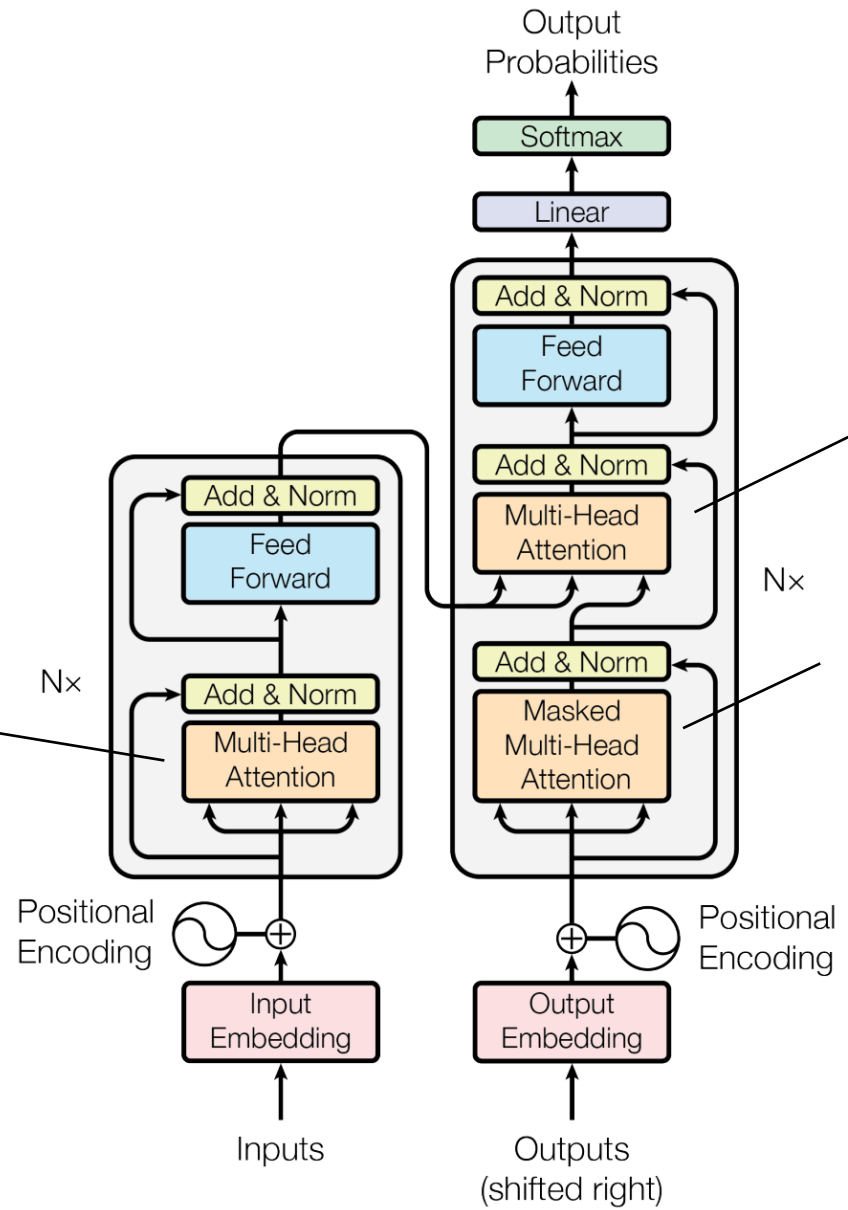


Figure 16: Transformer have an attention mechanism in the source and destination sentence unlike RNNs for example in this figure. This advantage makes the concept of a sequence with the full internal representation of its elements to be received in the best way. We will use this idea in the next section for the computer vision tasks and deal with an image like a sentence.



Transformer: Attention is all you need

Transformer Architecture



Attention mechanism

روی جملات هم مبدا و هم مقصد

Self-attention mechanism

روی جملات مبدا

Self-attention mechanism

روی جملات مقصد بصوت Mask شده

Transformer: Attention is all you need

Self attention mechanism

مقدار بردار ضرایب حاصل را با تقسیم بر d_k بصورت scale شده و نرمال در می‌آوریم.

هر Query با تک تک Key ها بصورت همزمان و موازی میشود.inner product

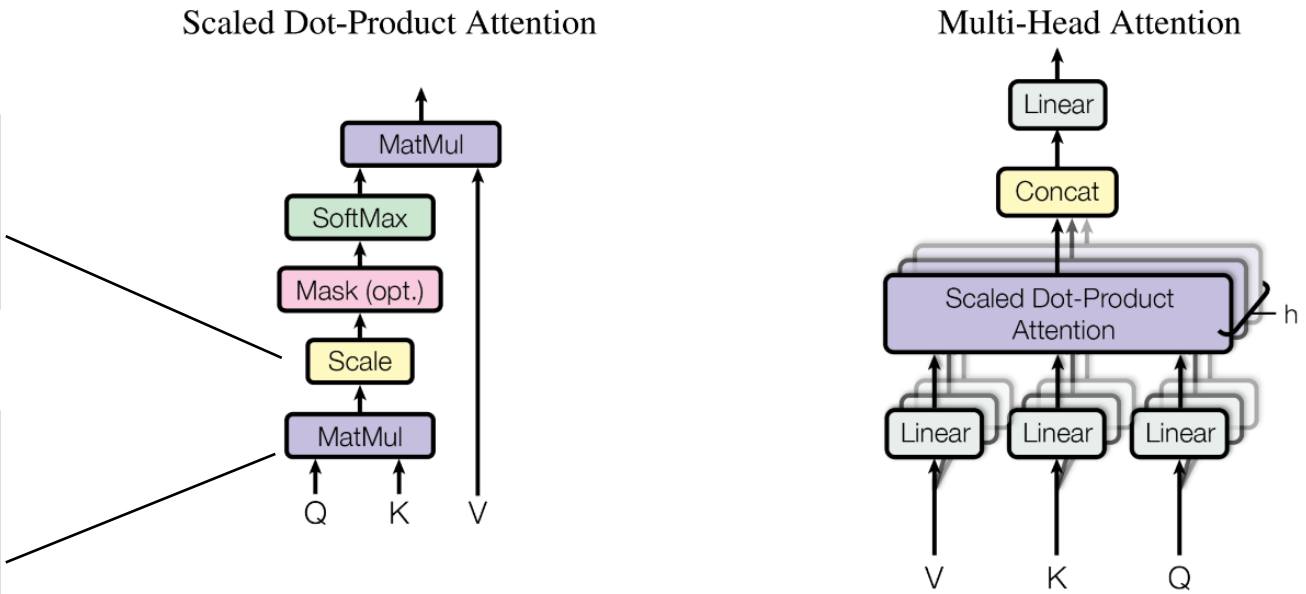


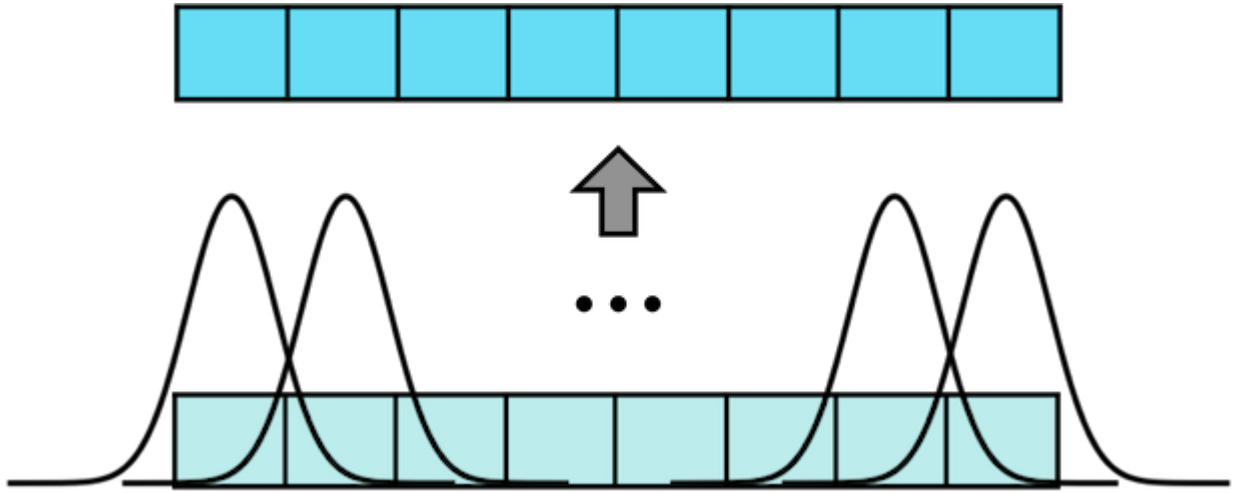
Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

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

d_k: keys of dimension

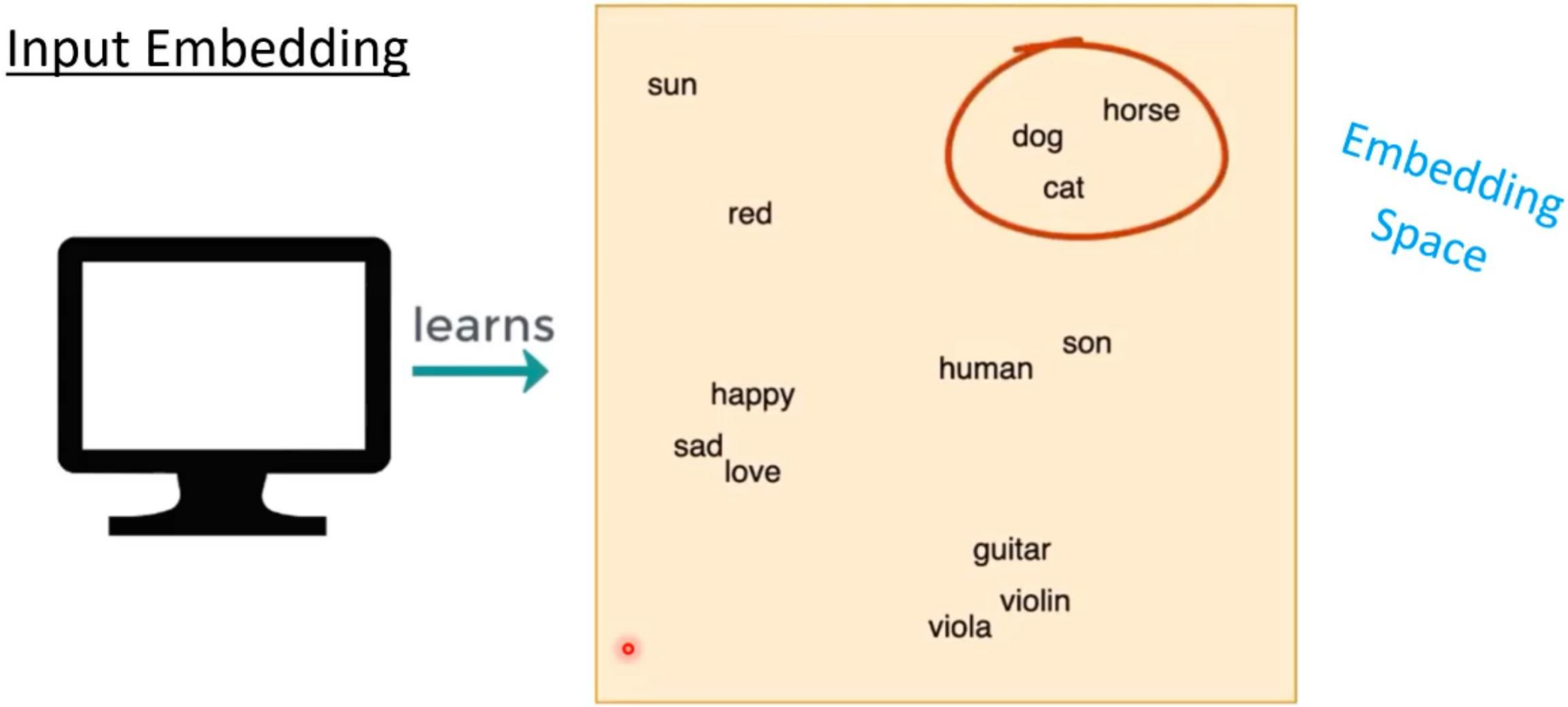
Transformer: Attention is all you need

Positional encoding mechanism for store the position of words in the sequence



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

Transformer: Input Embedding



Transformer: Input Embedding (GloVe, Word2Vec, FastText, …)

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning

Introduction

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Getting started (Code download)

- Download the latest latest code (licensed under the Apache License, Version 2.0).
  Look for "Clone or download"
- Unpack the files: unzip master.zip
- Compile the source: cd GloVe-master && make
- Run the demo script: ./demo.sh
- Consult the included README for further usage details, or ask a question

Download pre-trained word vectors

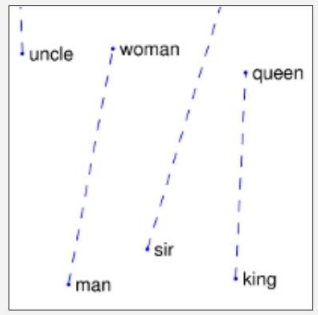
- Pre-trained word vectors. This data is made available under the Public Domain Dedication and License v1.0 whose full text can be found at: http://www.opendatacommons.org/licenses/pddl/1.0/.
  - Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): glove.6B.zip
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): glove.42B.300d.zip
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): glove.840B.300d.zip
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): glove.twitter.27B.zip
- Ruby script for preprocessing Twitter data

Citing GloVe

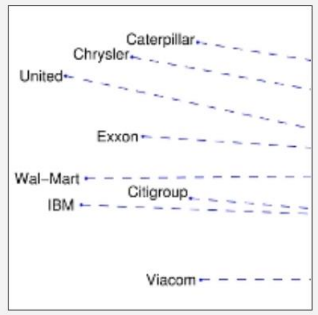
Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. [pdf] [bib]

Highlights

1. **Nearest neighbors**



man - woman



company - ceo

https://nlp.stanford.edu/projects/glove/

Transformer: Positional Encoding

Vectors that gives context based on position of word in sentence

AJ's dog is a cutie → Position 2

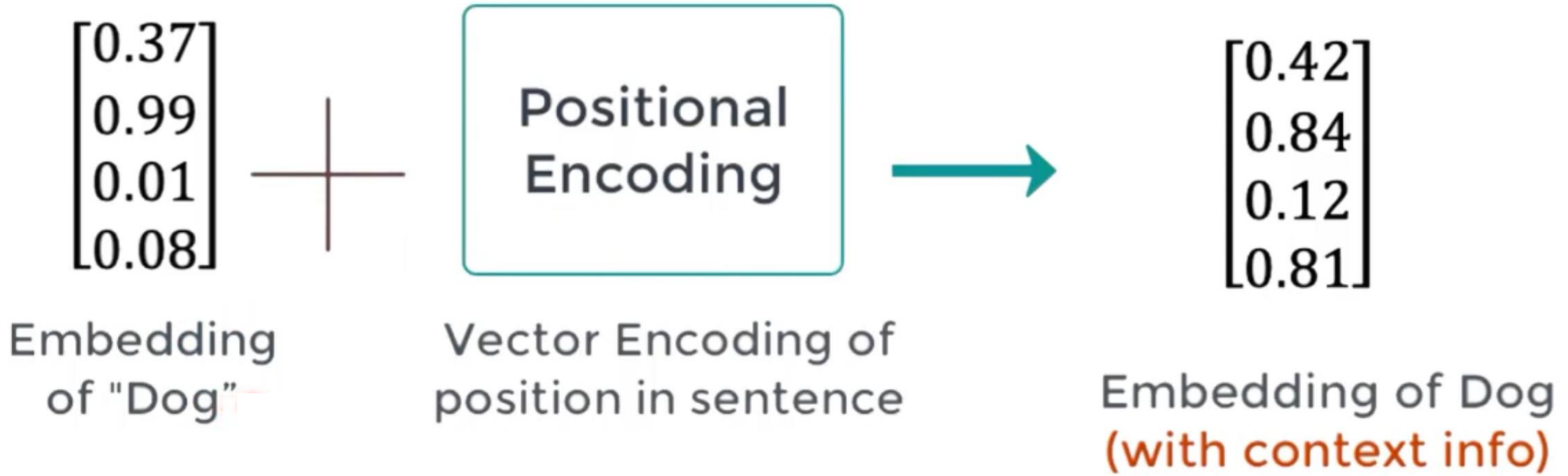
AJ looks like a dog → Position 5

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

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

Transformer: Positional Encoding

Vectors that gives context based on position of word in sentence



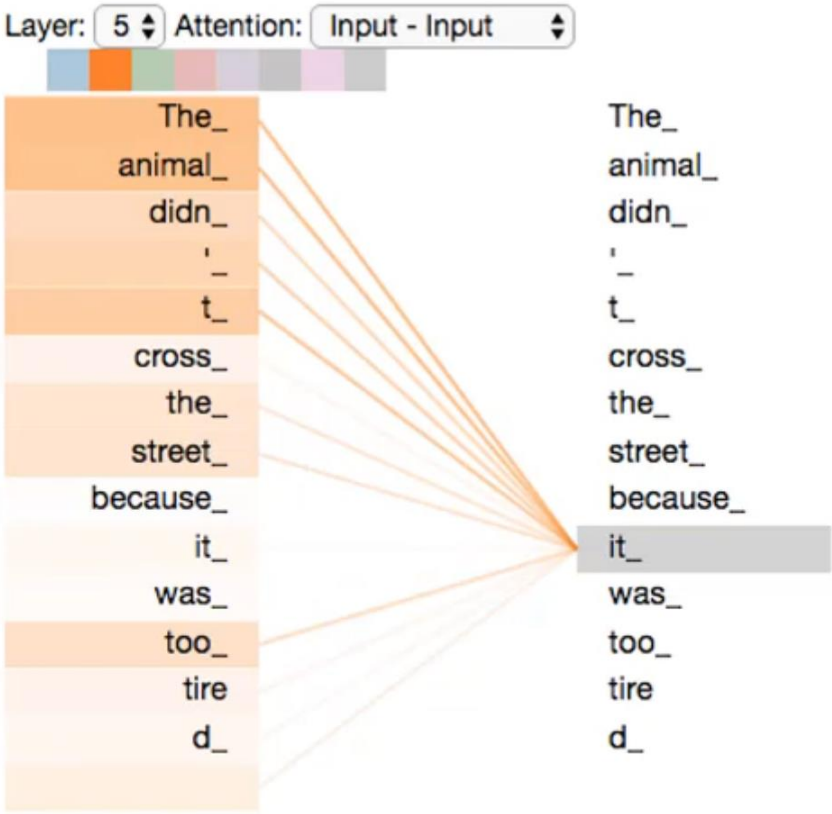
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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

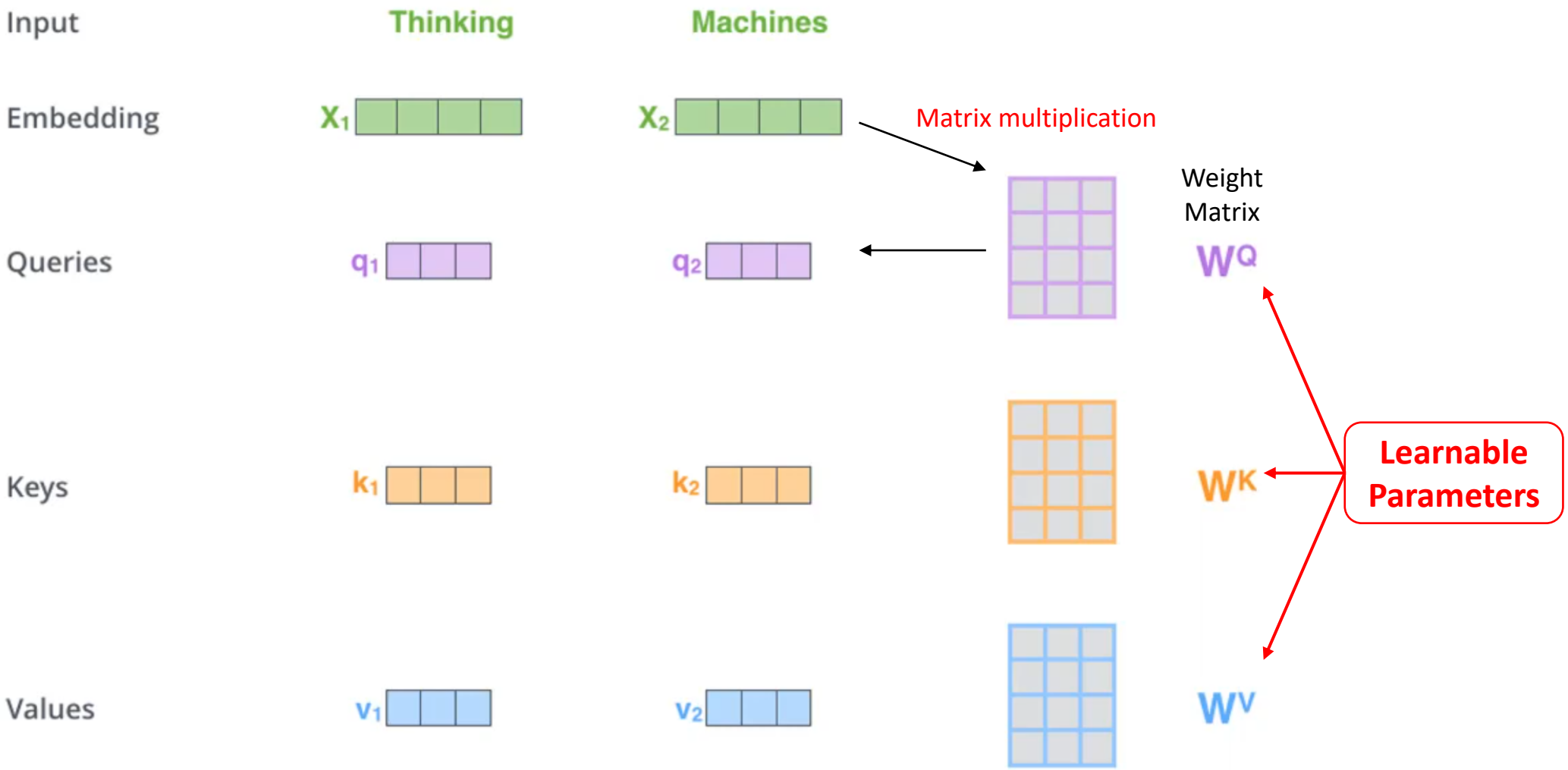
Transformer: Self Attention

What part of the input should we focus?

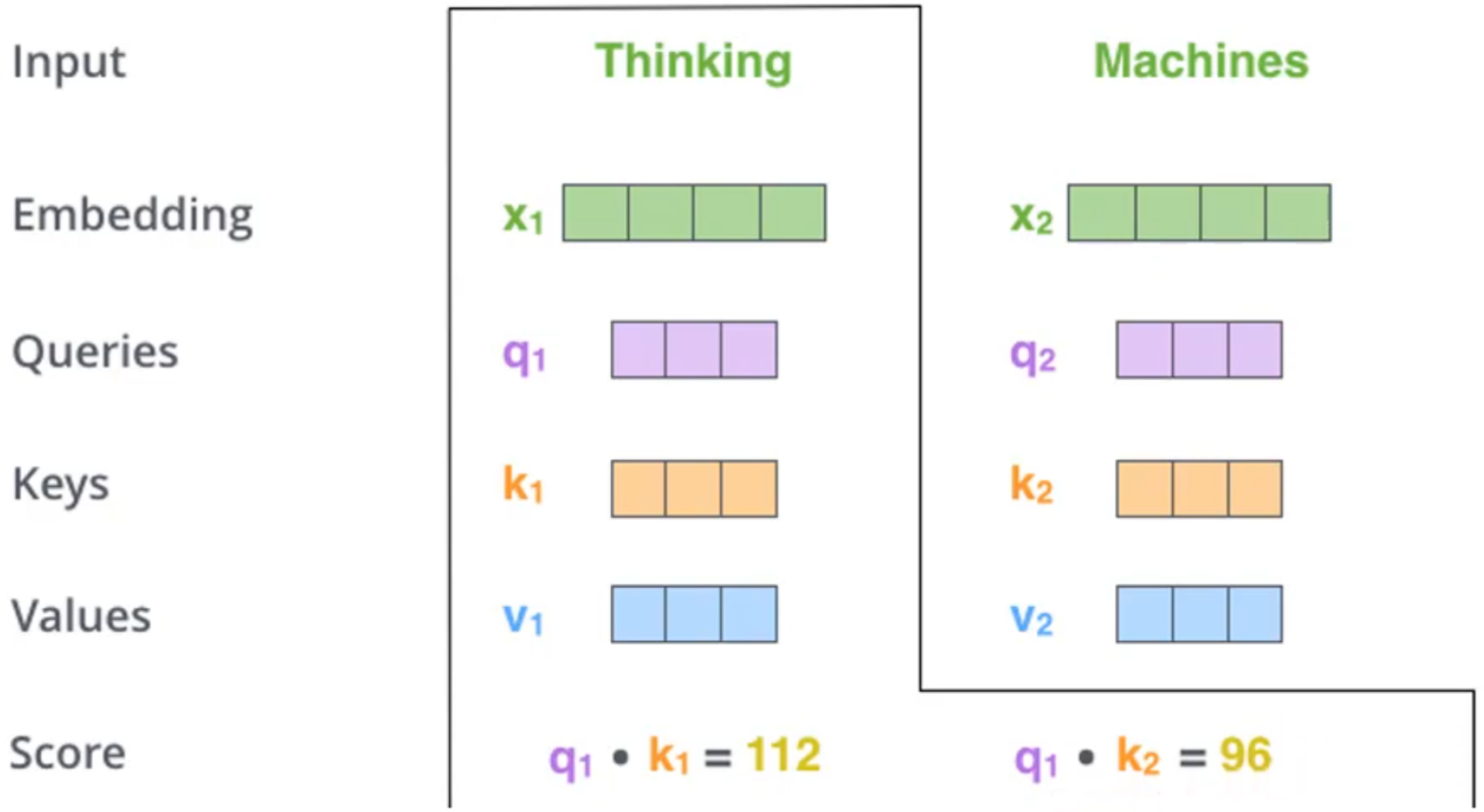
"The animal didn't cross the street because it was too tired"



Transformer: Self Attention

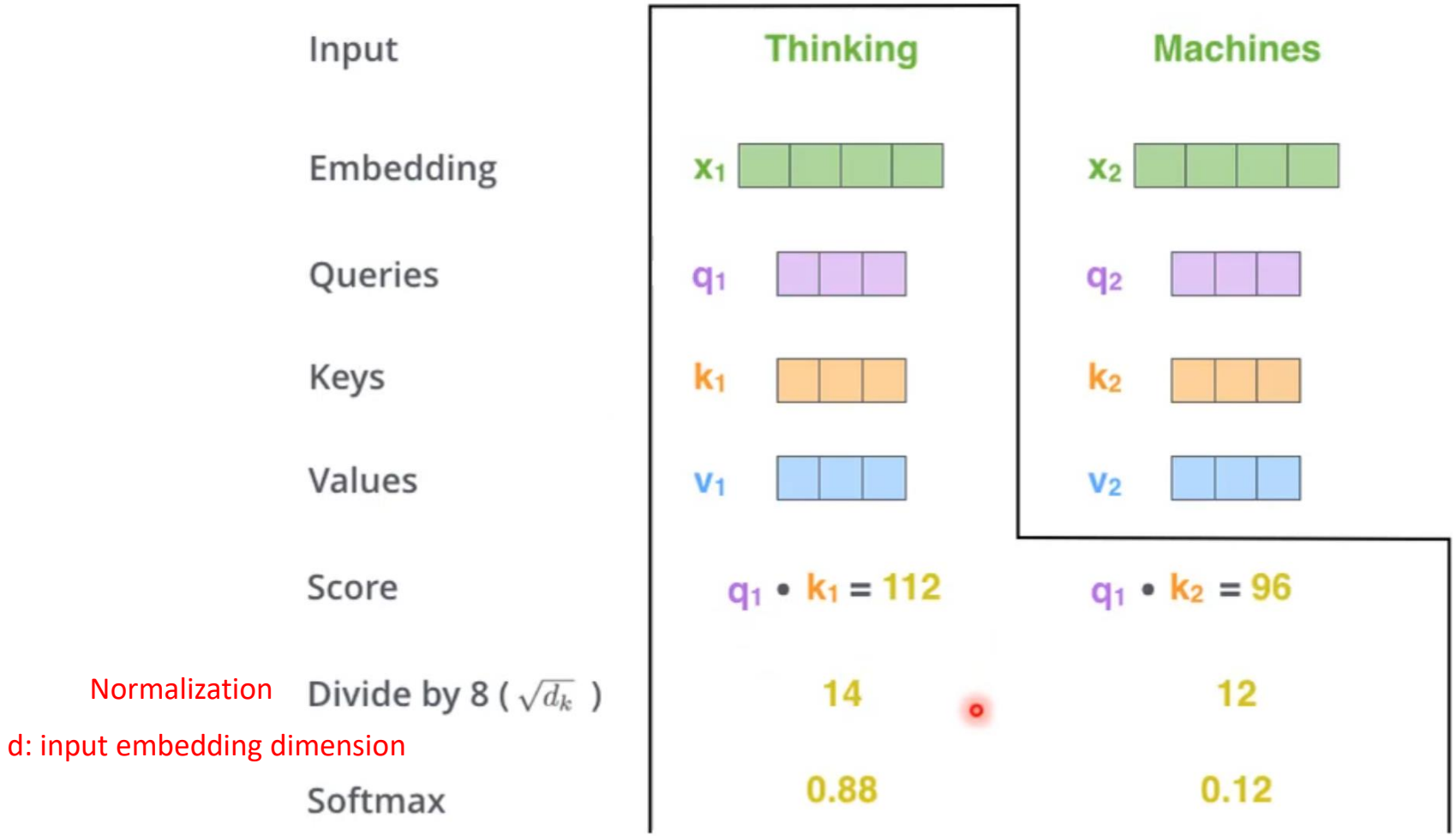


Transformer: Self Attention



Dot Product for Cosine Similarity

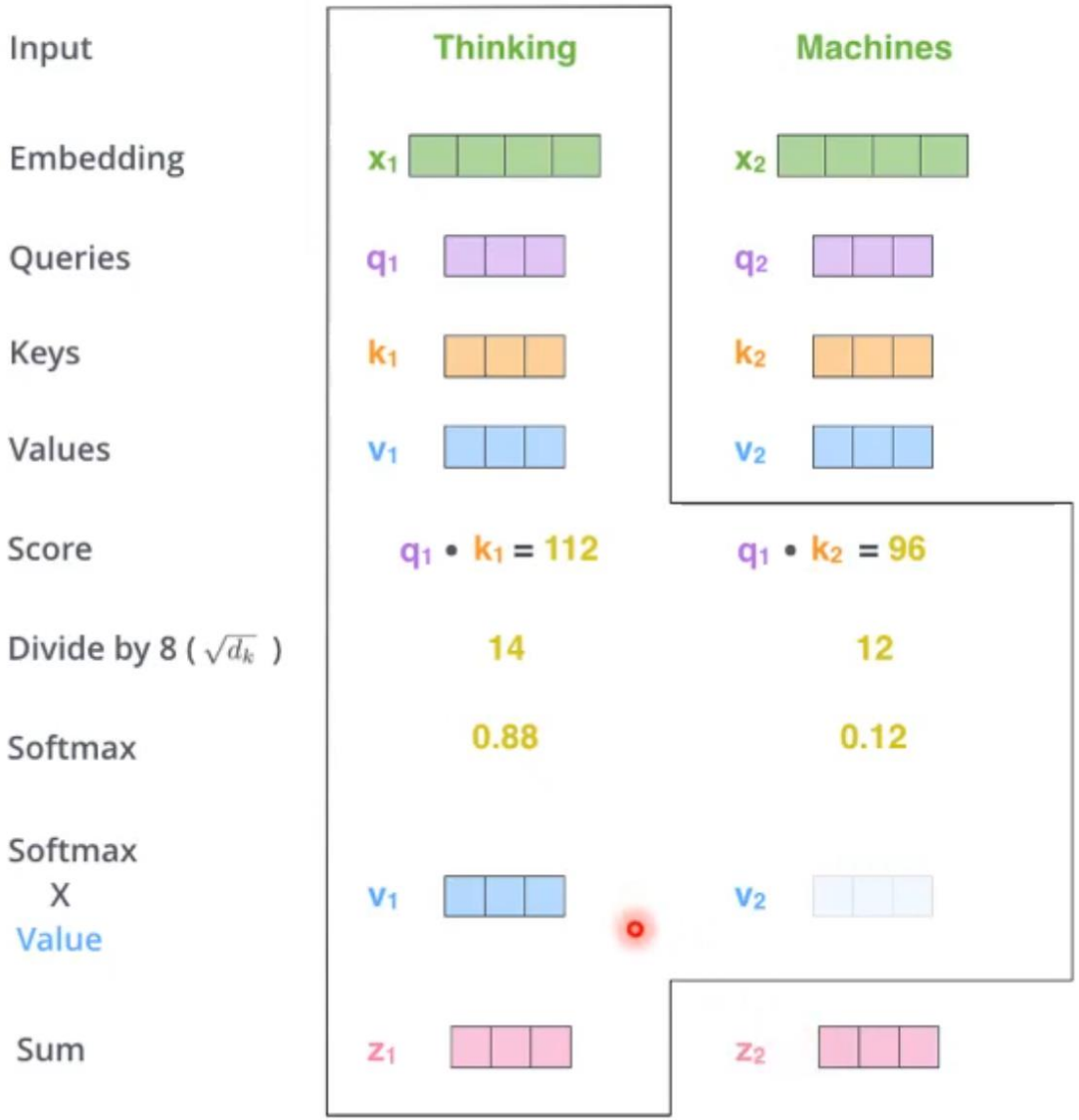
Transformer: Self Attention



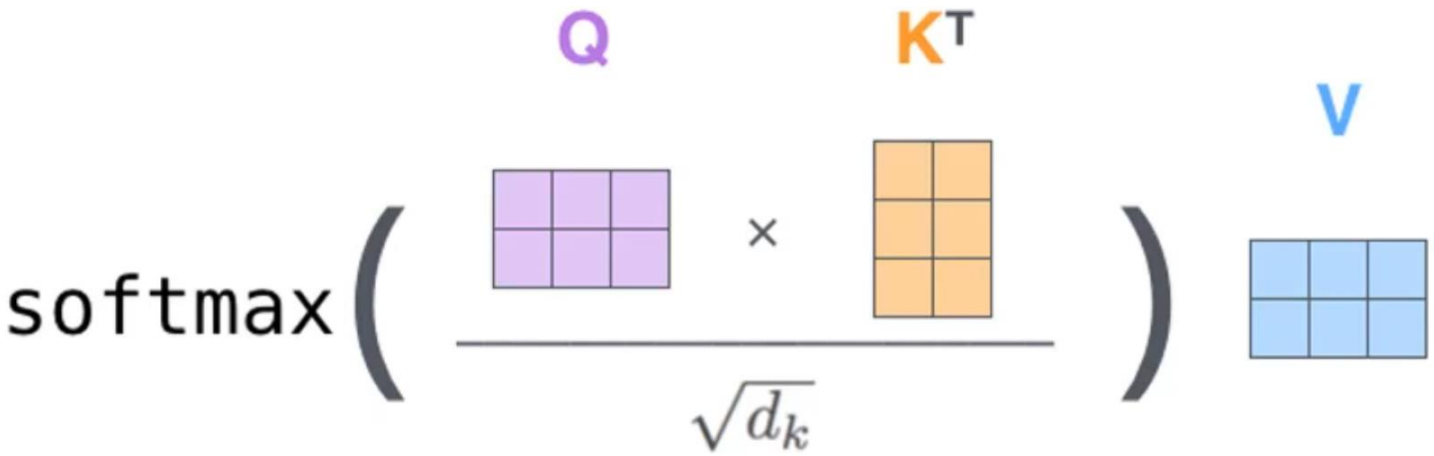
Normalization

d: input embedding dimension

Transformer: Self Attention

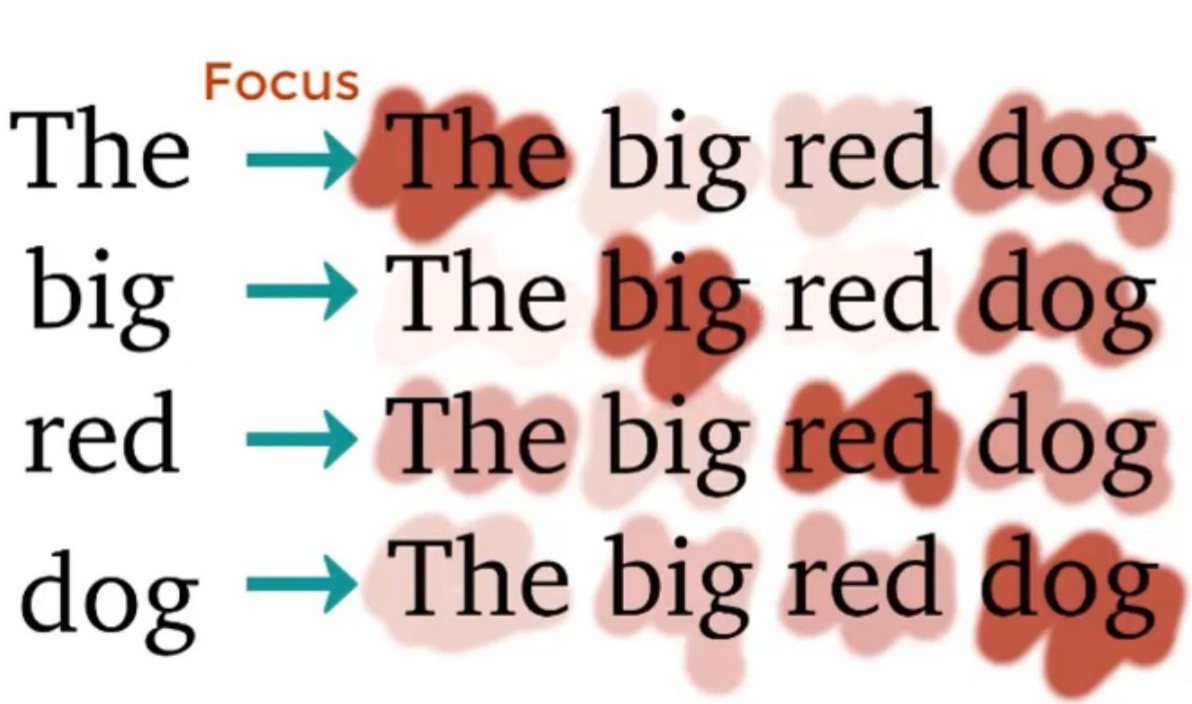


Transformer: Self Attention



هر سطر نشان دهنده بردار های نهایی Z برای یک کلمه است.
بردار Z : بردار توجه یا attention vector است.

Transformer: Self Attention Challenge : Why Multi-Head Attention?



Attention Vectors

$[0.71 \quad 0.04 \quad 0.07 \quad 0.18]^T$

$[0.01 \quad 0.84 \quad 0.02 \quad 0.13]^T$

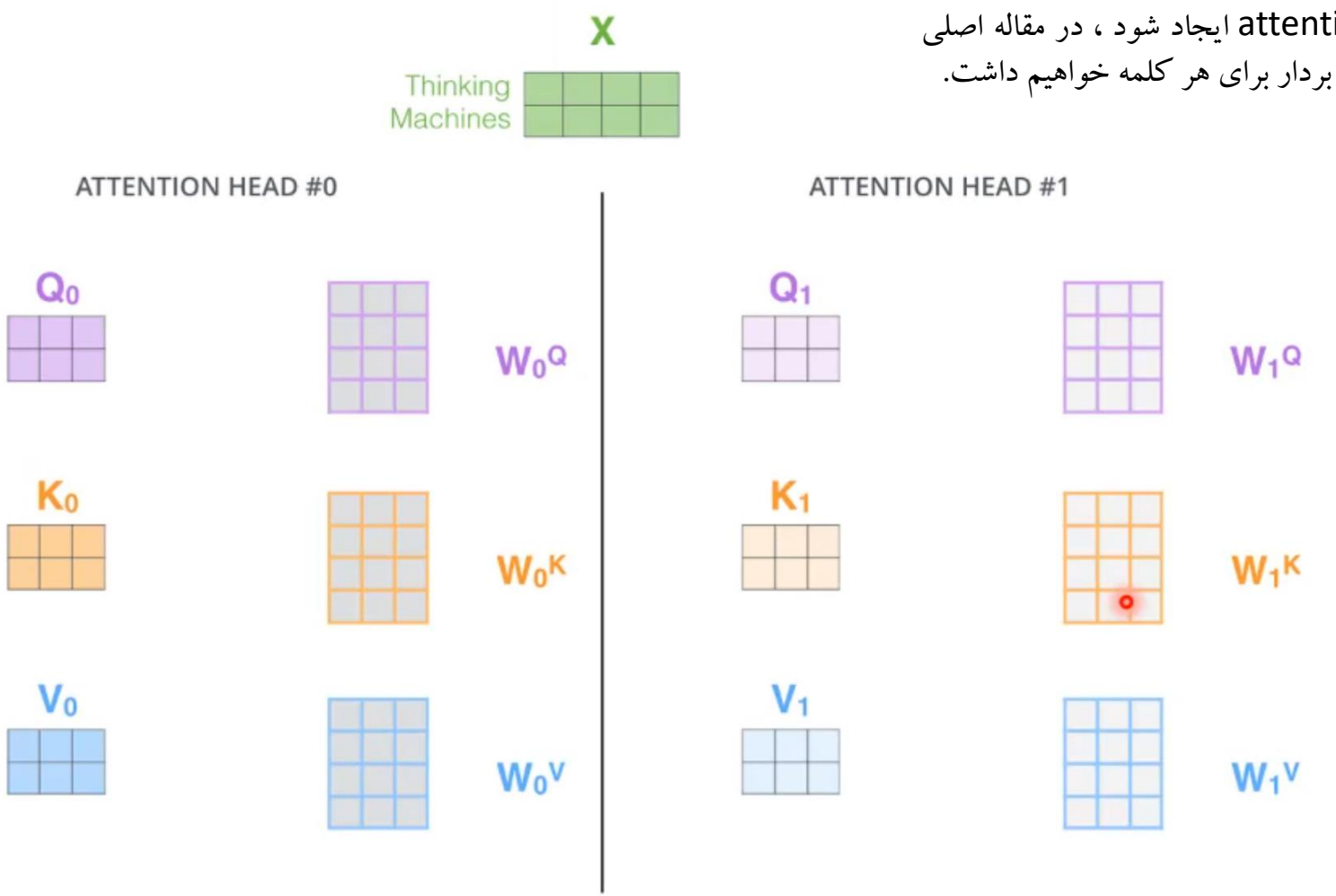
$[0.09 \quad 0.05 \quad 0.62 \quad 0.24]^T$

$[0.03 \quad 0.03 \quad 0.03 \quad 0.91]^T$

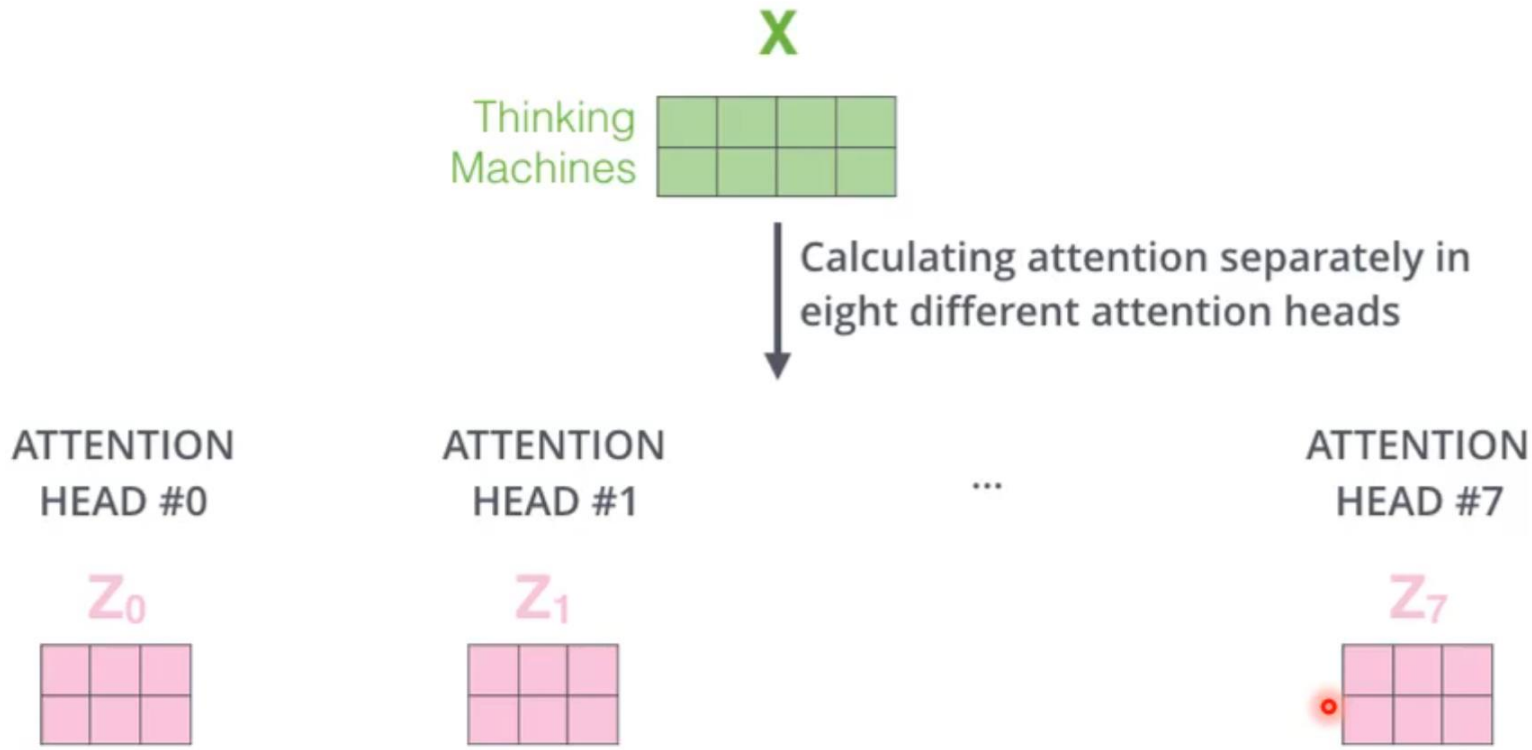
"The animal didn't cross the street because it was too tired"

Transformer: Multi-Head Attention

در این روش n بار عملیات self attention انجام میدهد ، با ایجاد ماتریس های مختلف Q ، K و V به مدل اجازه میدهیم فضاهای representation مختلفی را فرا بگیرد و هربار یک بردار z بعنوان attention vector ایجاد شود ، در مقاله اصلی 8 بار این کار انجام شده و 8 بردار برای هر کلمه خواهیم داشت.

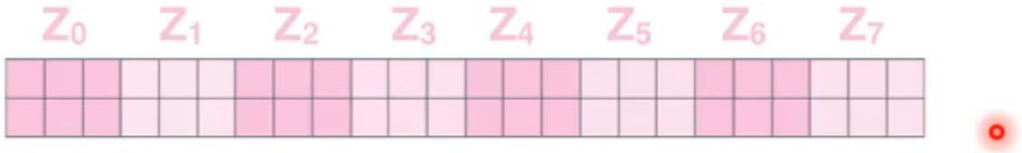


Transformer: Multi-Head Attention



Transformer: Multi-Head Attention (concatenate to feed forward)

1) Concatenate all the attention heads



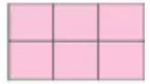
2) Multiply with a weight matrix W^O that was trained jointly with the model

X

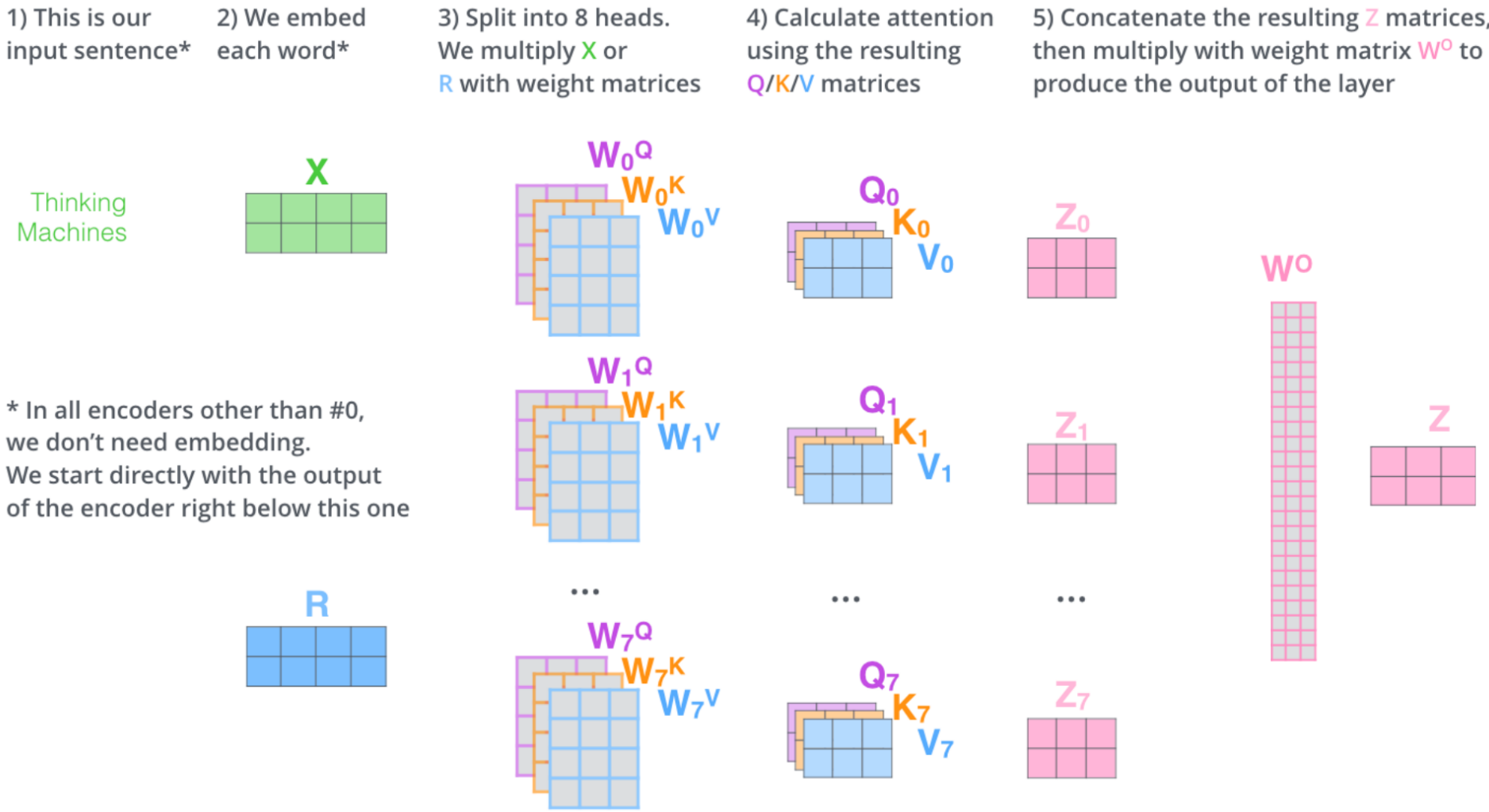
Learnable Parameters

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

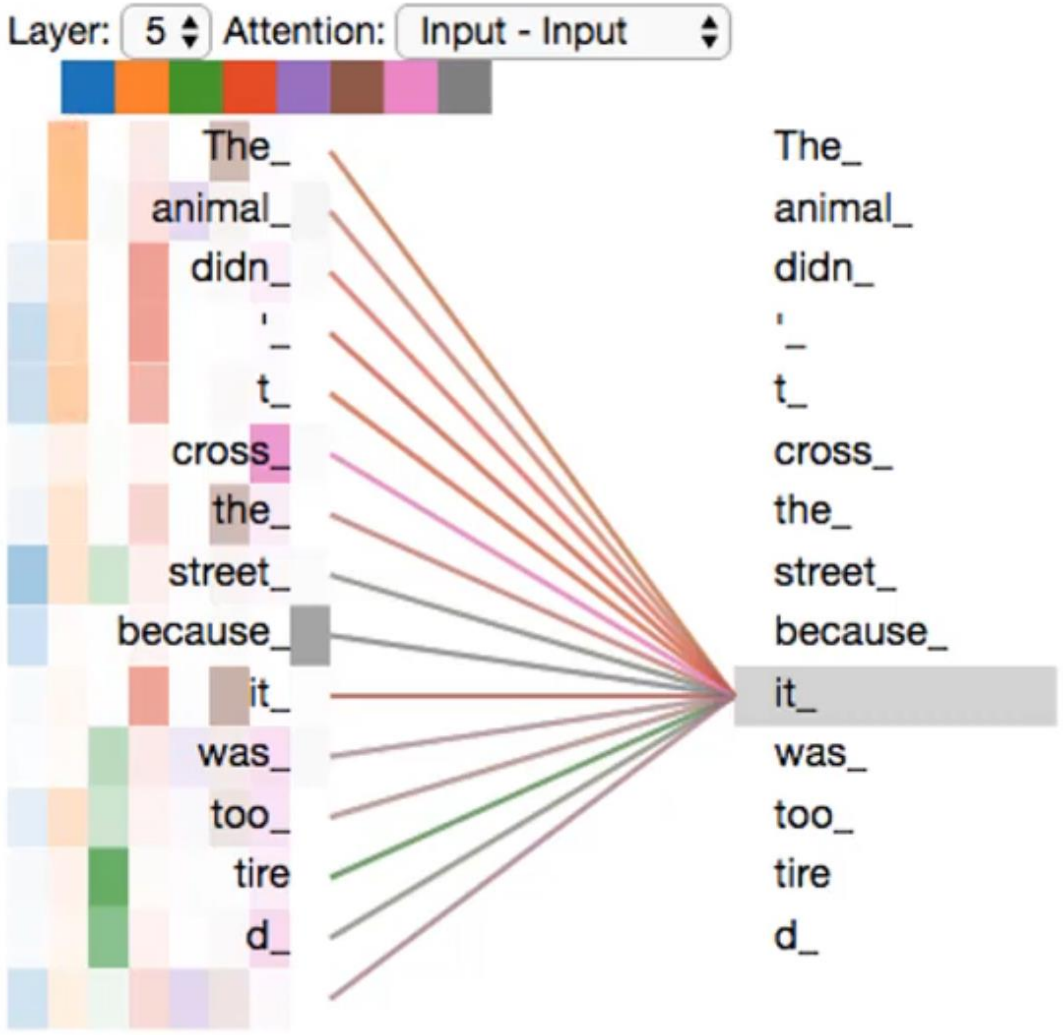
=



Transformer: Multi-Head Attention

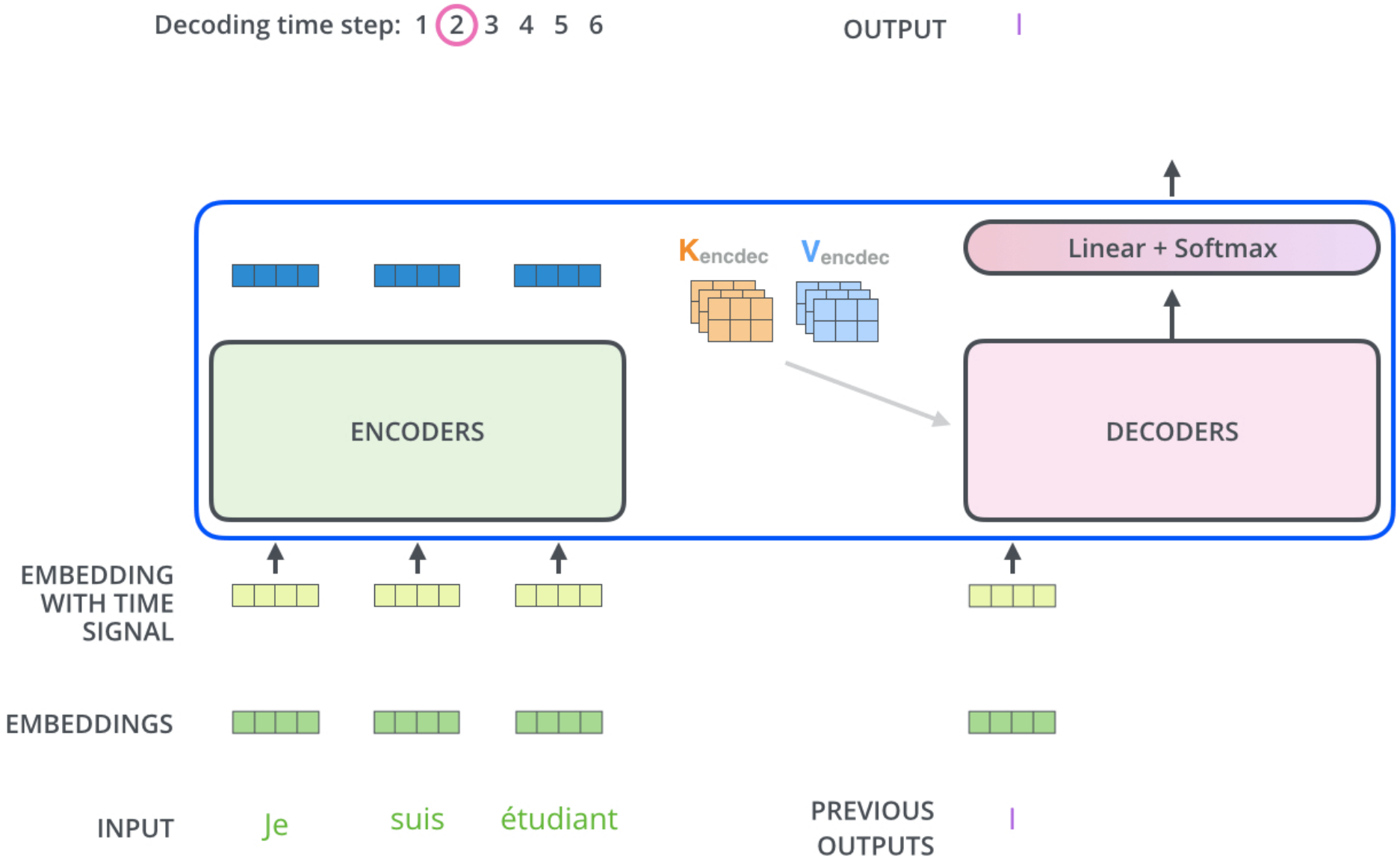


Transformer: Multi-Head Attention



هر رنگ نشان دهنده یک بردار Z یا Attention vector یعنی توجه در فضاهای مختلف است

Transformer



Transformer: Experiment

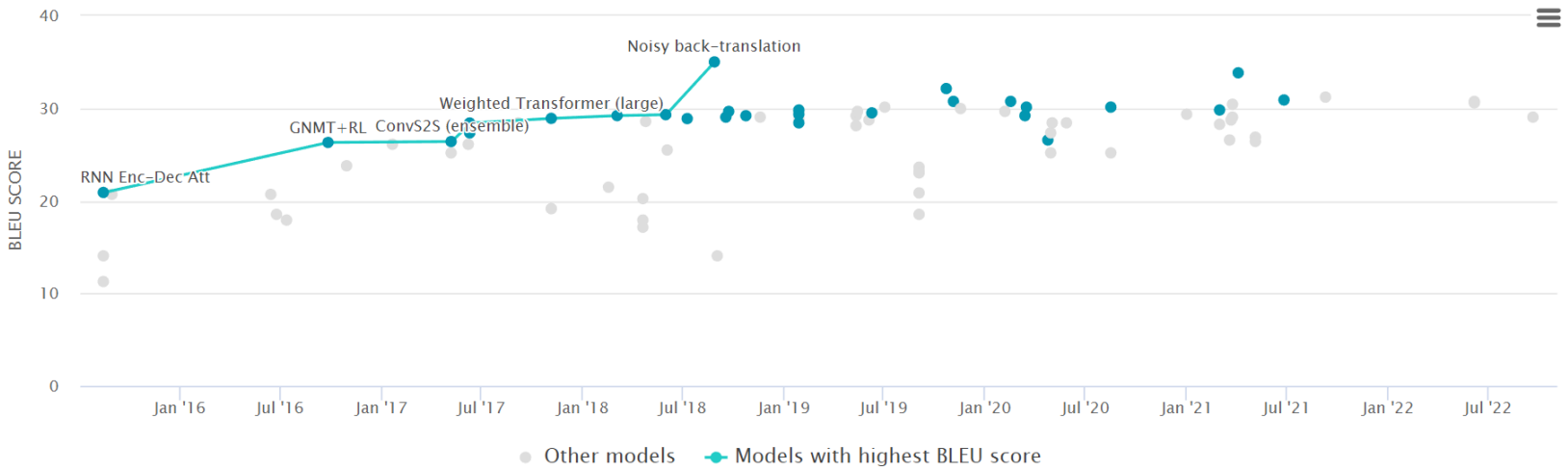
Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Machine Translation on WMT2014 English-German

Leaderboard Dataset

View BLEU score by Date for All models



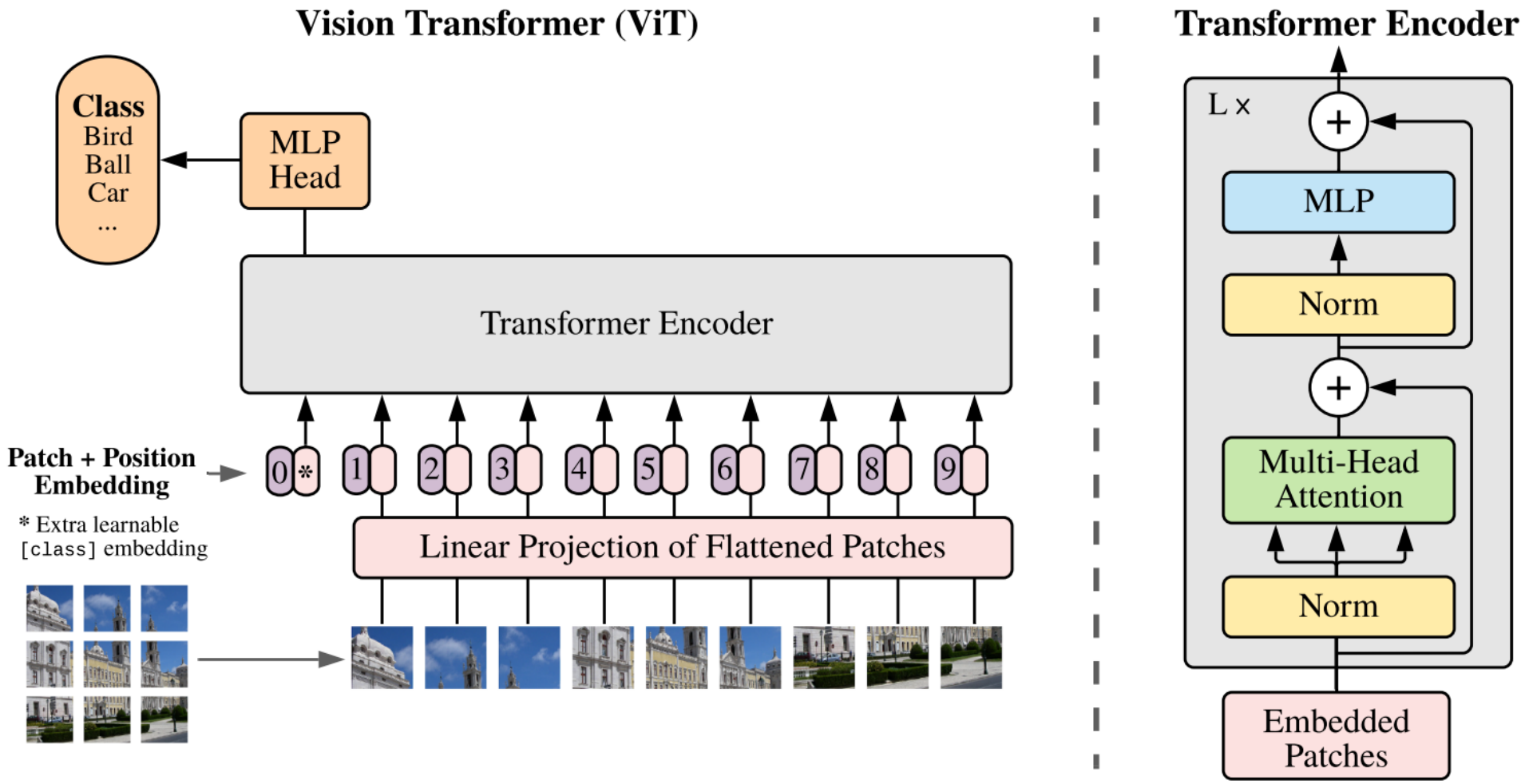
Filter: Transformer LSTM CNN multiscale untagged Hardware Burden Operations per network pass

Edit Leaderboard

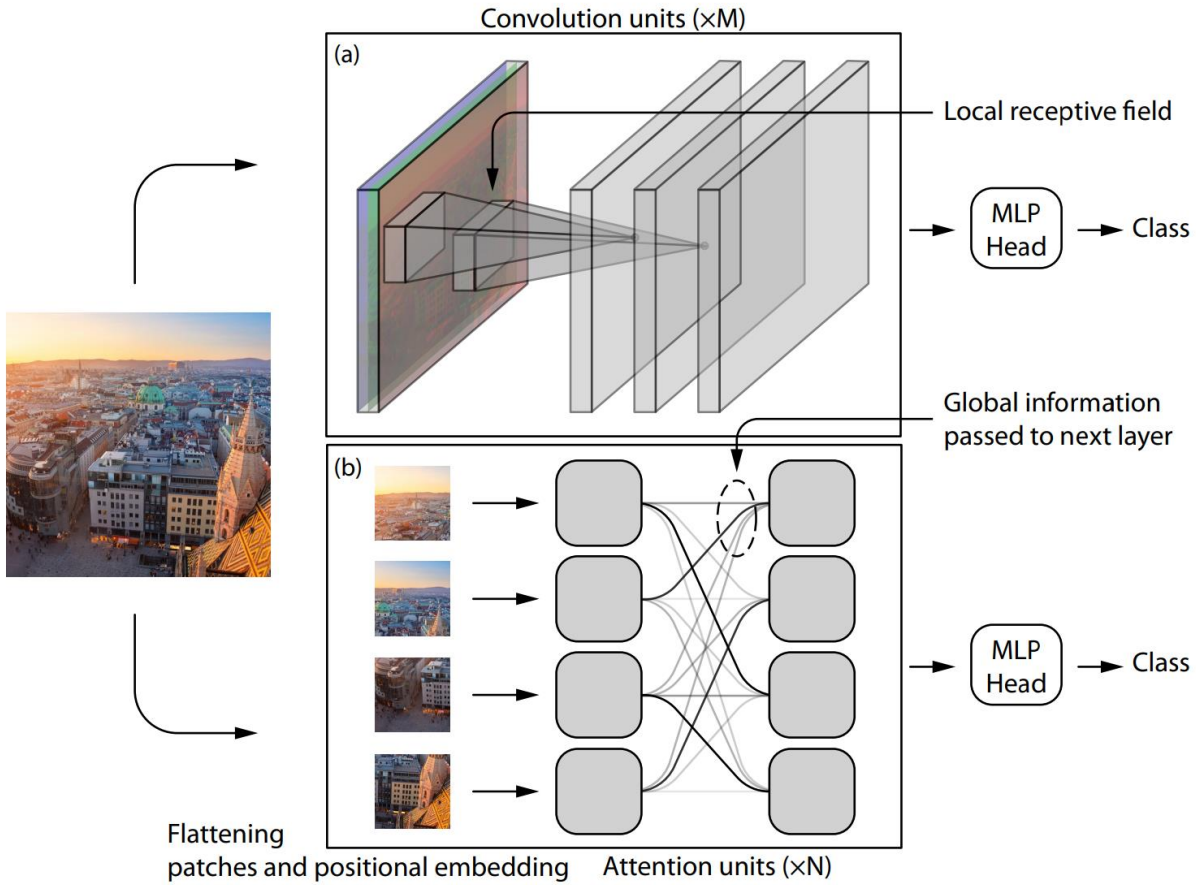
Rank	Model	BLEU score	SacreBLEU	Number of Params	Extra Training Data	Paper	Code	Result	Year	Tags
1	**Noisy back-translation**	35.0	33.8		✓	Understanding Back-Translation at Scale			2018	
2	**Transformer+Rep** (Uni)	33.89	32.35		✓	Rethinking Perturbations in Encoder-Decoders for Fast Training			2021	Transformer
3	**T5-11B**	32.1		11110M	✓	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer

Figure 17: English to German Translation Benchmark, as you can see all of top ranks of the models are transformer base.

Vision Transformer

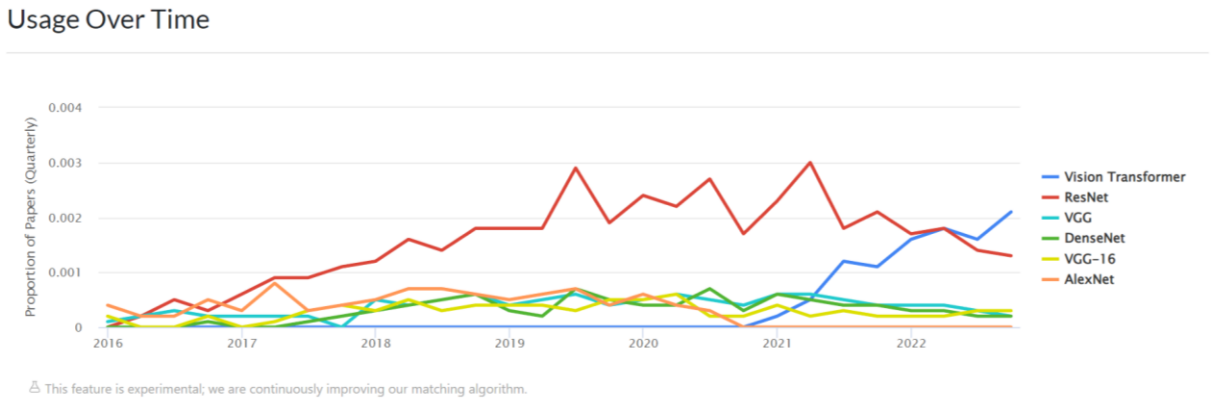


Vision Transformer VS CNN



Bird's eye view of (a) convolutional and (b) attention-based networks.

[Are Convolutional Neural Networks or Transformers more like human vision?](), 2021



https://paperswithcode.com/method/vision-transformer

Vision Transformer VS CNN

CNN uses pixel arrays, whereas ViT splits the images into visual tokens.

CNN have local receptive information with CNN local windows.

And the purpose of the transformer is to extract general knowledge from the whole image by using the attention mechanism so that each patch of the image pays attention to all other patches.

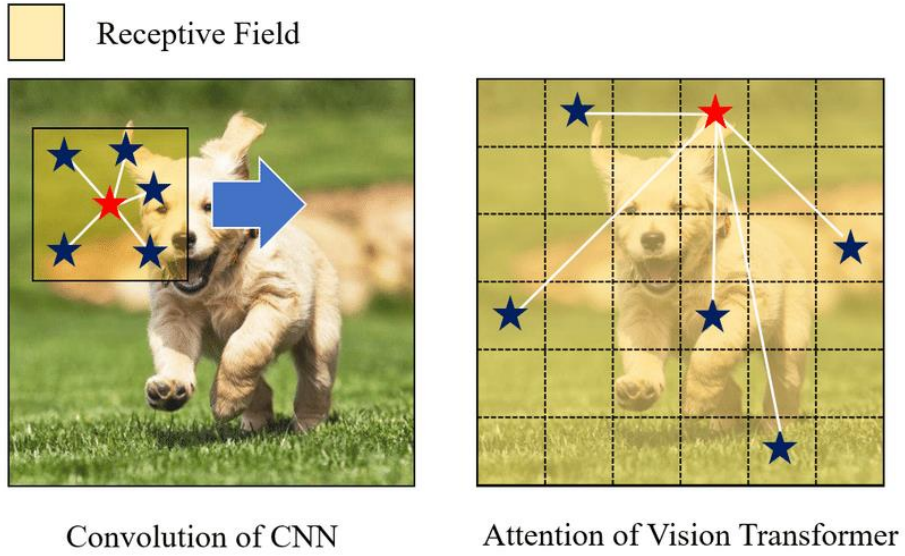


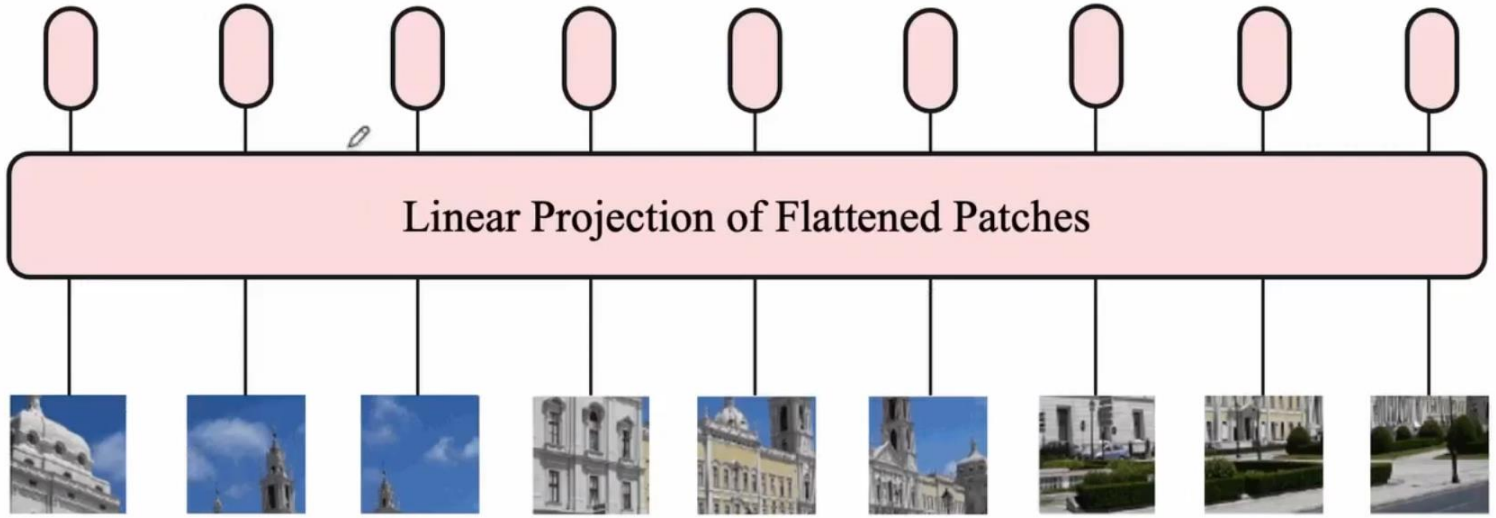
Figure 18: **CNNs Vs Vision Transformers:** As you can see in the figure, as we have reviewed vision transformers before, we saw that, unlike CNN which had local receptive field and local vision, vision transformer architecture has been able to tokenize image patches with an image similar to a sentence in NLP can reach global information using the Attention mechanism, and this has caused a significant growth in its popularity after its introduction ViT Model in 2021 by Google.

Vision Transformer

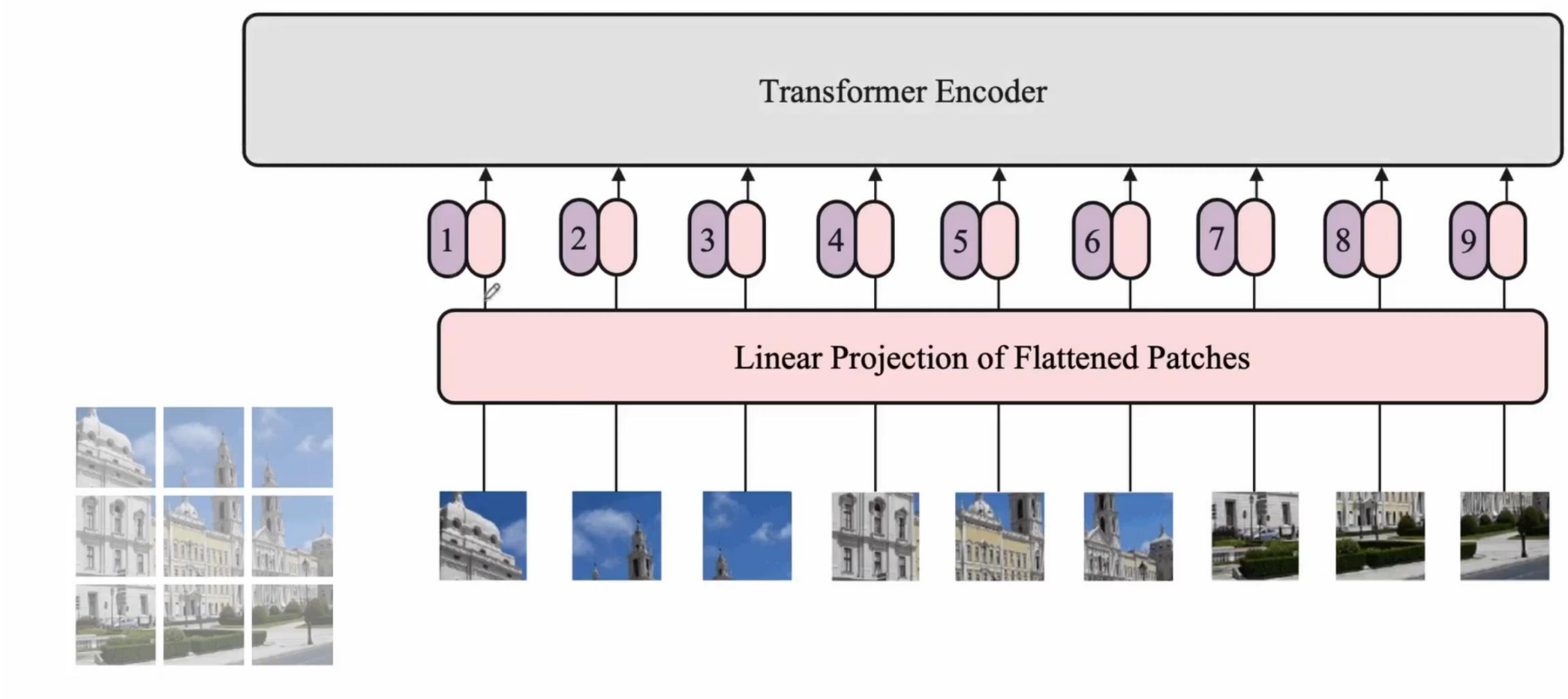
Resize input image to 224x224

Patch size: 16x16

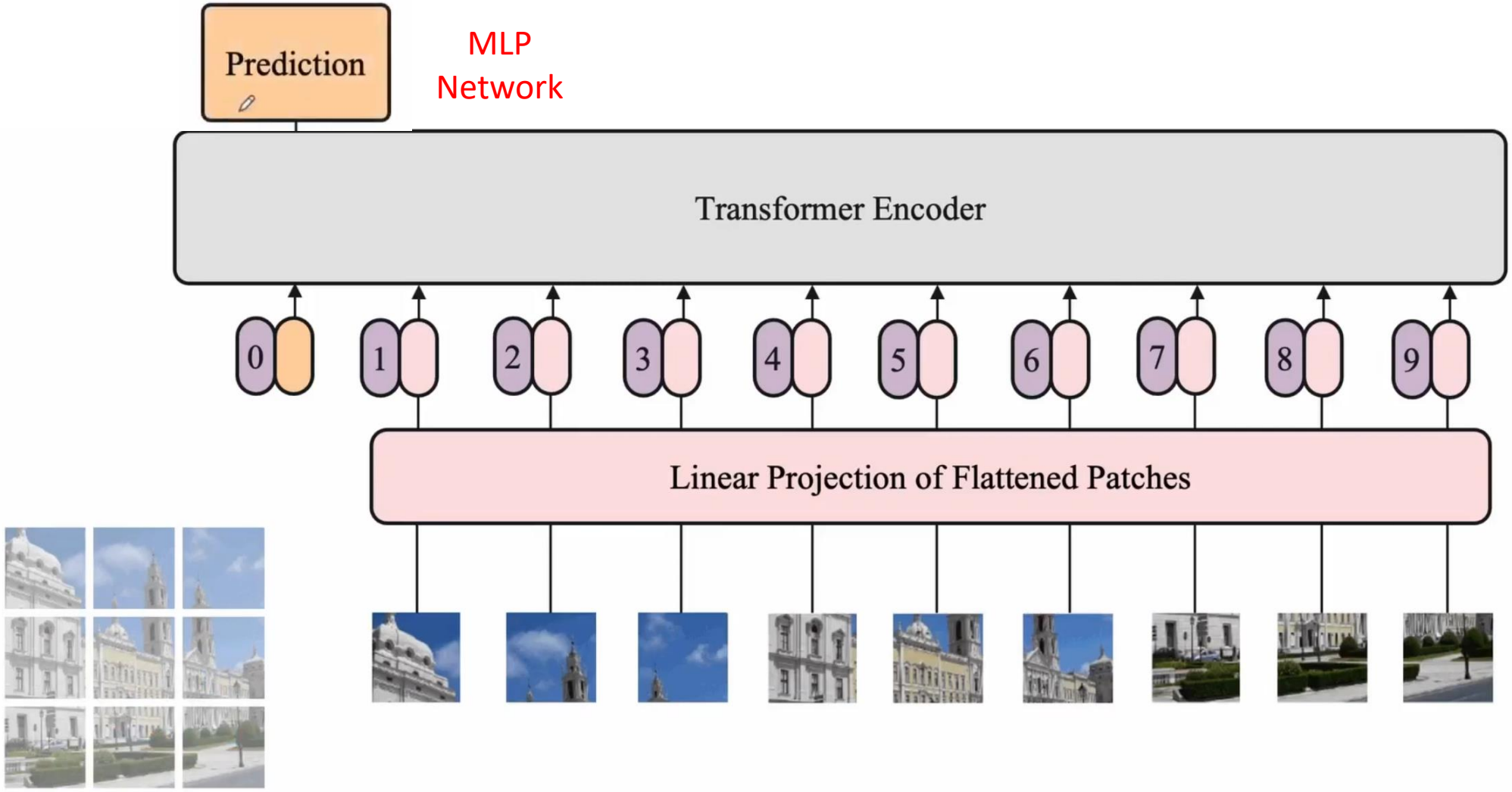
Vision Transformer



Vision Transformer

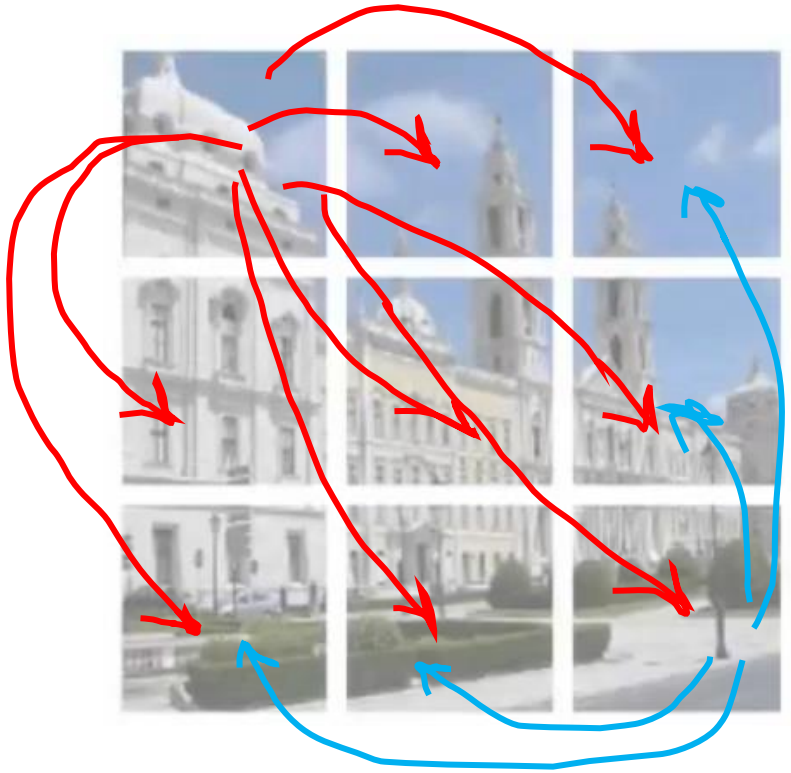


Vision Transformer



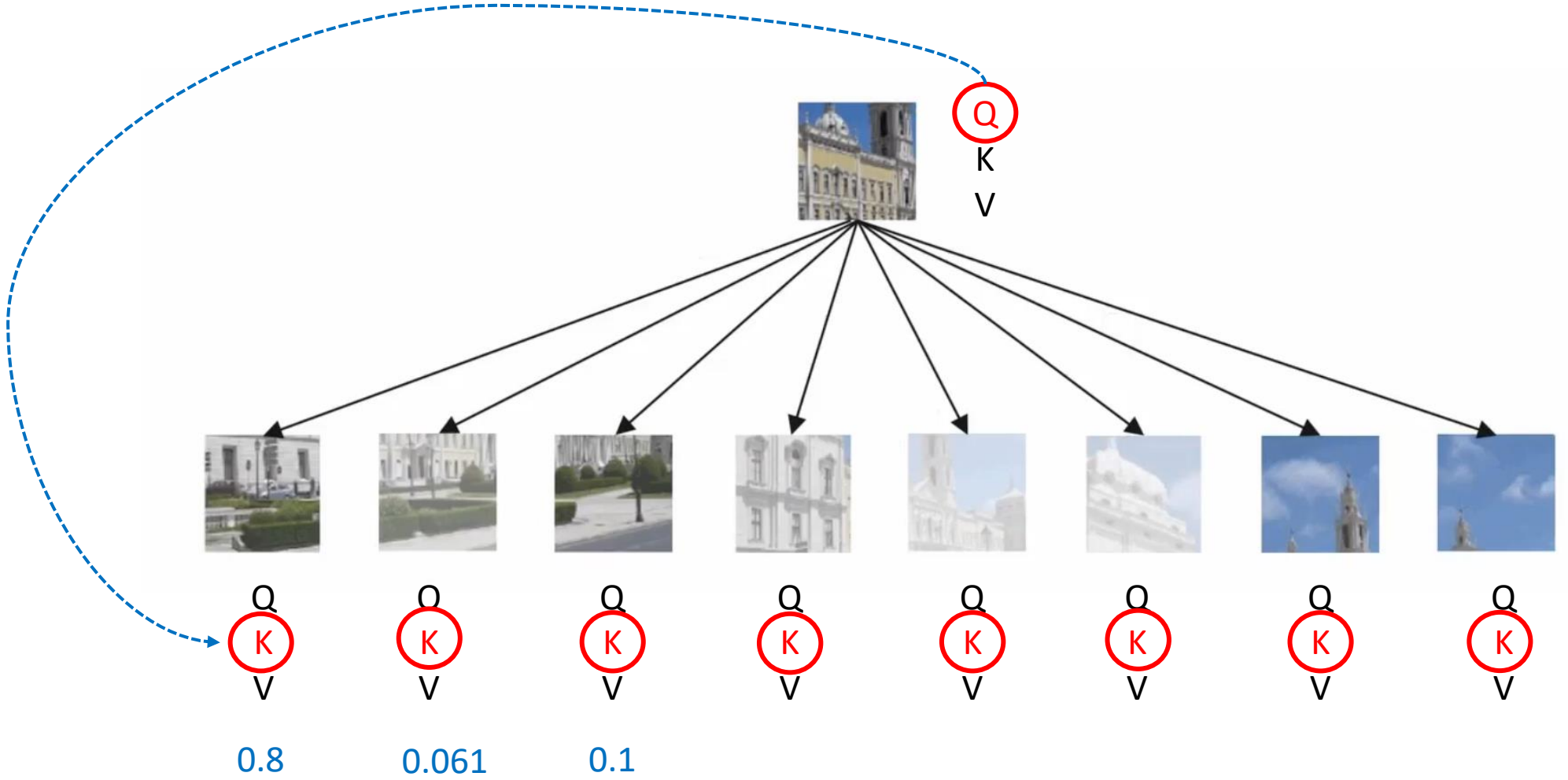
Vision Transformer: Attention in vision

Global Receptive Field

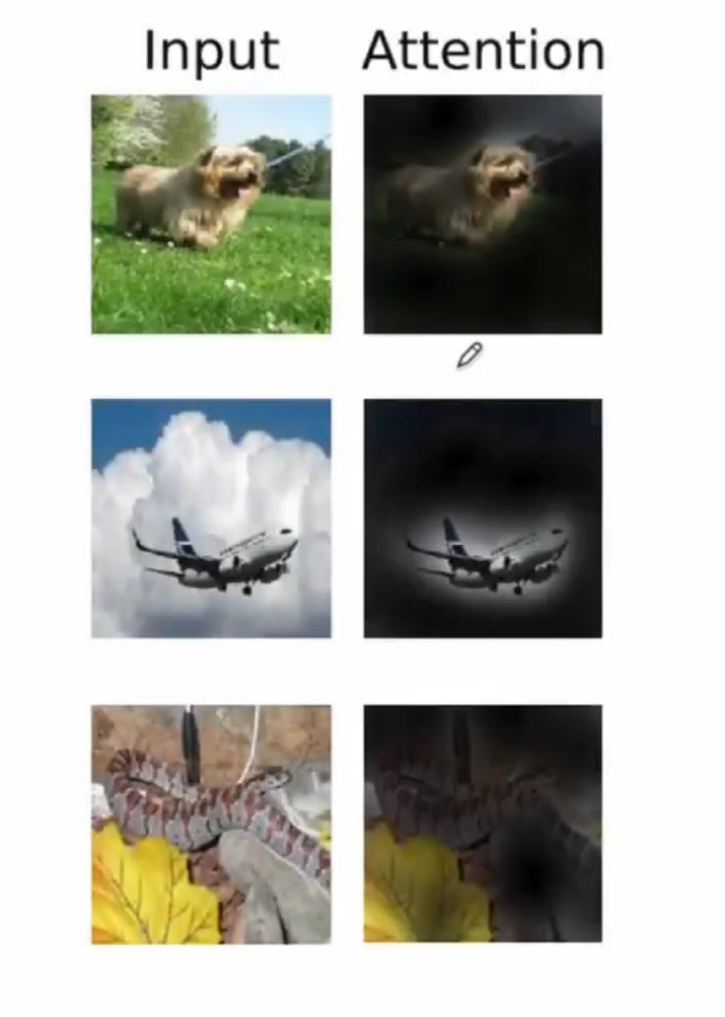


Patch Size? Large or Small?

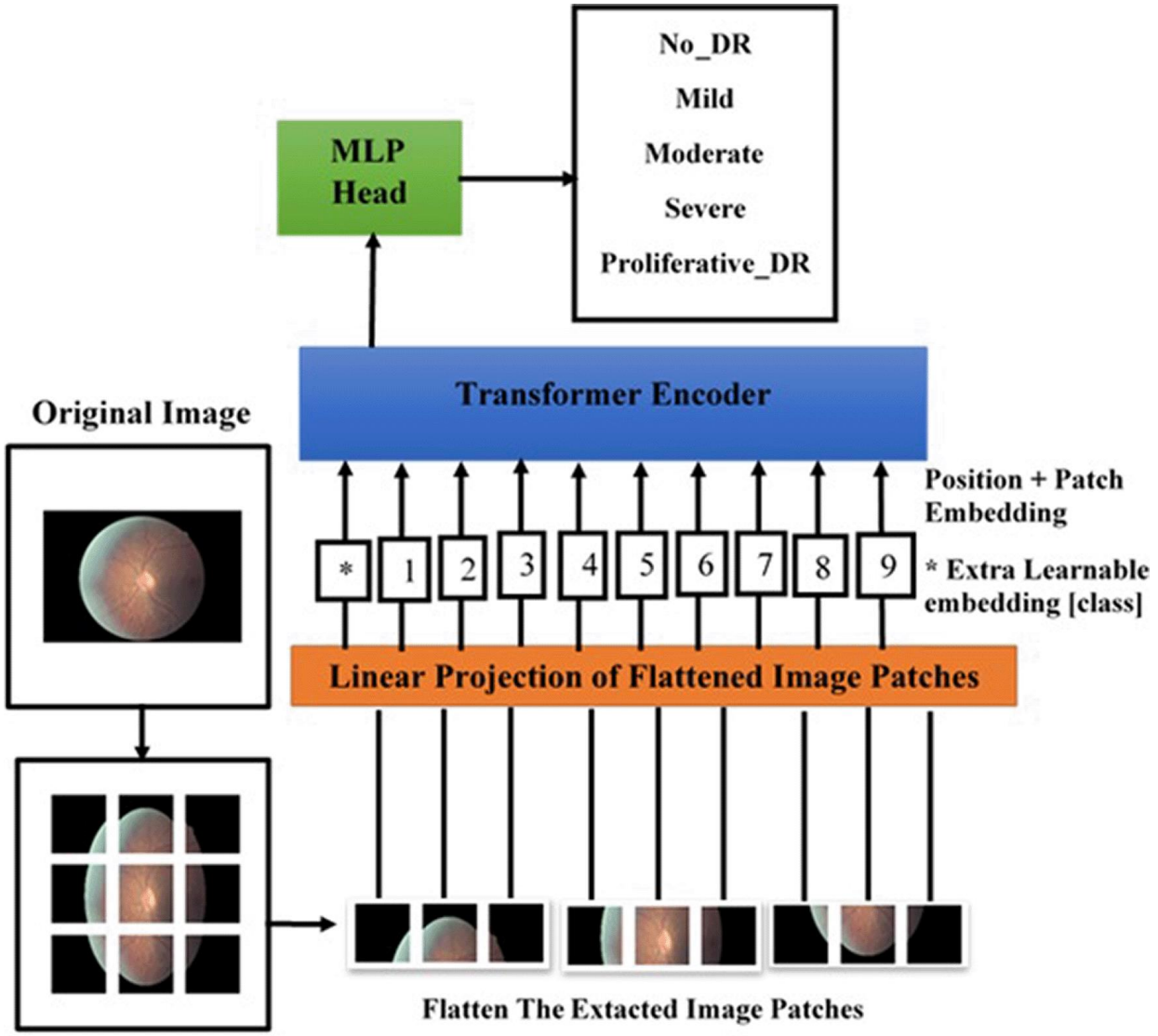
Vision Transformer: Attention in vision



Vision Transformer: Attention in vision



Vision Transformer: Image Classification



Vision Transformer: Drawbacks

- Quadratic Complexity
- Being data hungry due to multi-head attention

Vision Transformer: Image Captioning using Encode Decoder Model

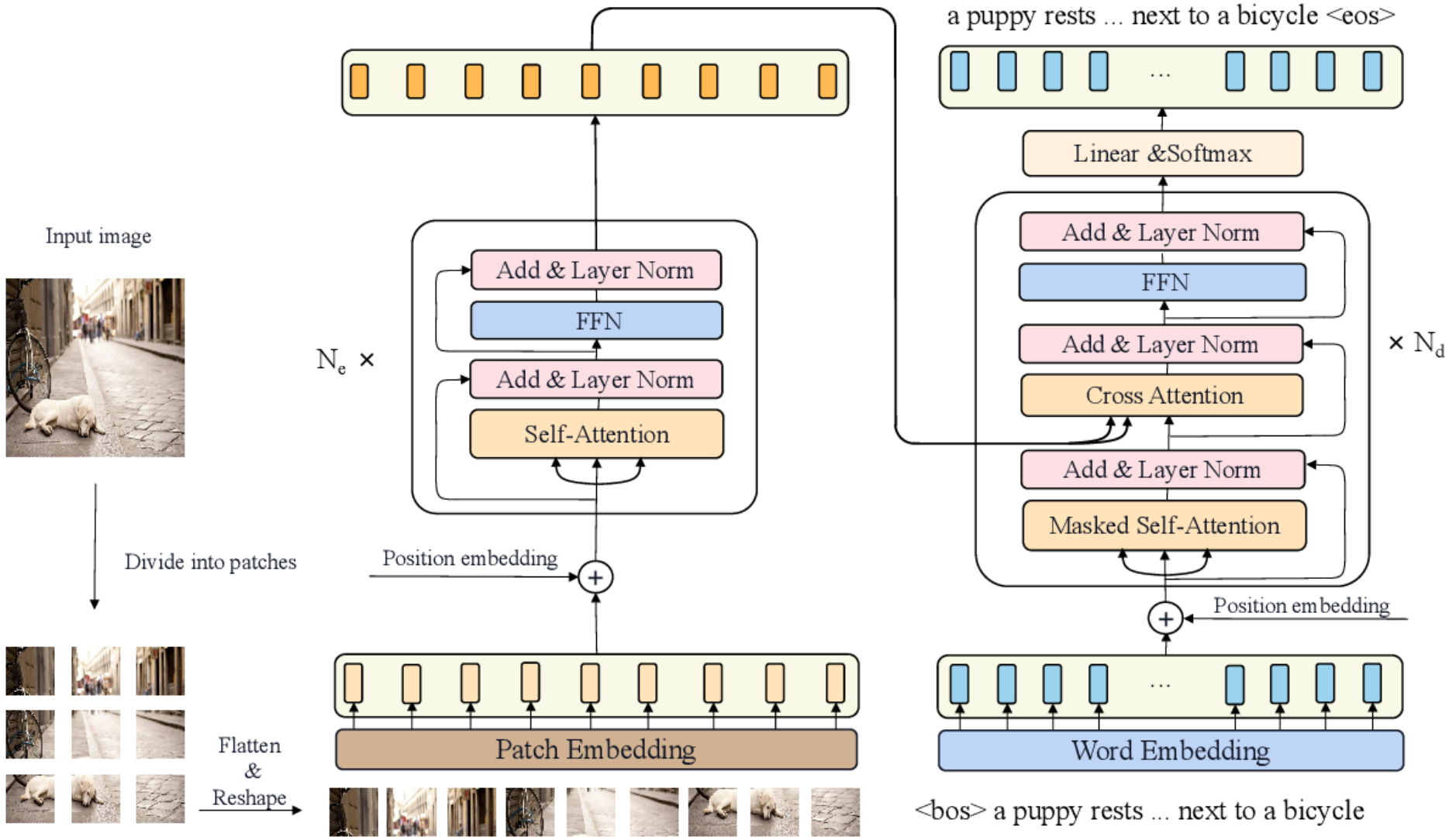
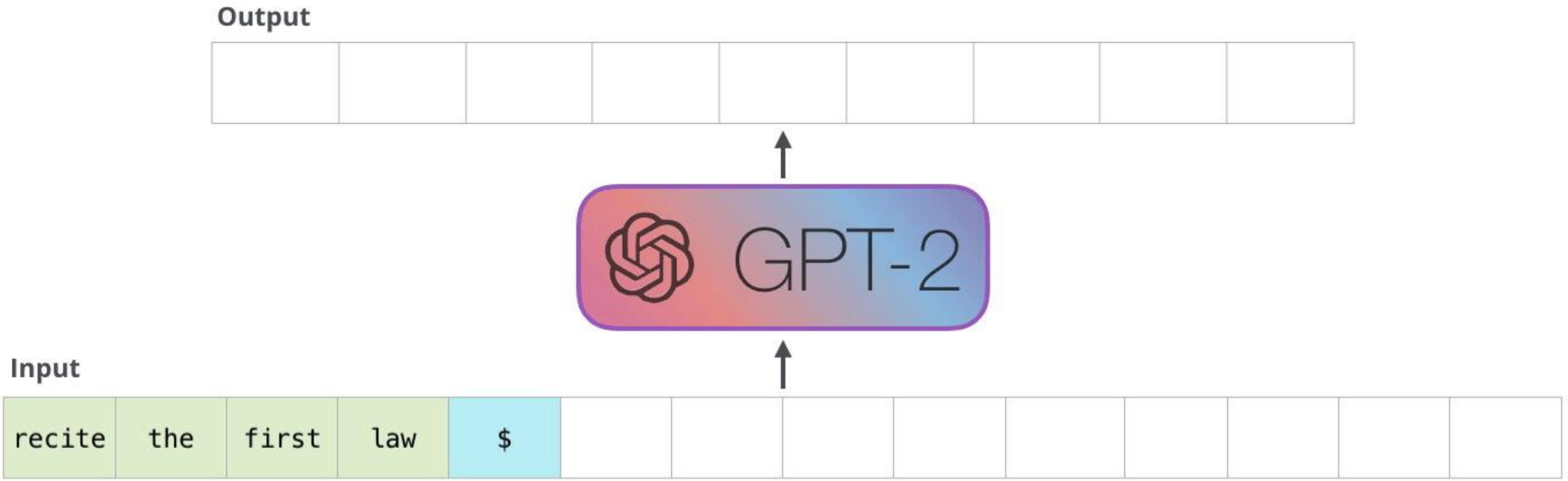


Figure 19: CPTR: Full Transformer Network for Image Captioning https://arxiv.org/pdf/2101.10804.pdf

Vision Transformer: Language Modeling



Vision Transformer: Language Modeling

$$P(w_1, w_2, \dots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \times \cdots \times P(w_n|w_1, w_2, \dots, w_{n-1})$$

میخواهیم احتمال اینکه این زنجیره کلمات از کلمه اول یعنی w1 تا کلمه آخر این دنباله یعنی w2 با این توالی بیایند را محاسبه کنیم

Some of other applications : machine translation, speech recognition and so on.

$p(\text{he likes apple}) > p(\text{apple likes he})$

$p(\text{he likes apple}) > p(\text{he licks apple})$

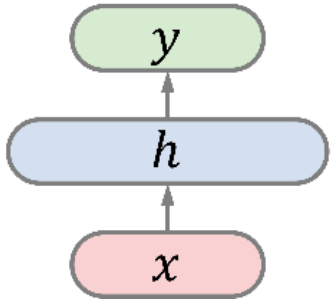
Language modeling
Approach1 : Count based N-Gram

مدل دو-بخشی →

$$P(w_1, w_2, \ldots, w_n) \approx P(w_1)P(w_2|w_1)P(w_3|w_2) \times \cdots \times P(w_n|w_{n-1})$$

$$P(w_i|w_{i-1}) = \frac{P(w_{i-1}, w_i)}{P(w_{i-1})} = \frac{N(w_{i-1}, w_i)}{N(w_{i-1})}$$

NLP: Language modeling and RNN Models
Approach2 : Deep neural net N-Gram



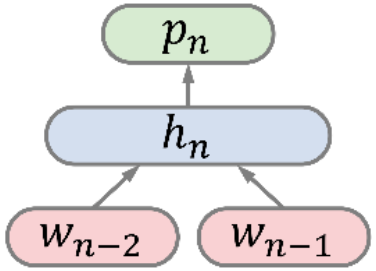
$$y = W_y h + b_y$$

$$h = f(W_h x + b_h)$$

$$x = [w_1; w_2; \dots; w_{n-1}]$$

تاریخچه

Example : 3-gram model



$$p_n = softmax(W_y\ h_n + b_y)$$

$$h_n = f(W_h\ [w_{n-2}; w_{n-1}] + b_h)$$

$$x_n = [w_{n-2}; w_{n-1}]$$

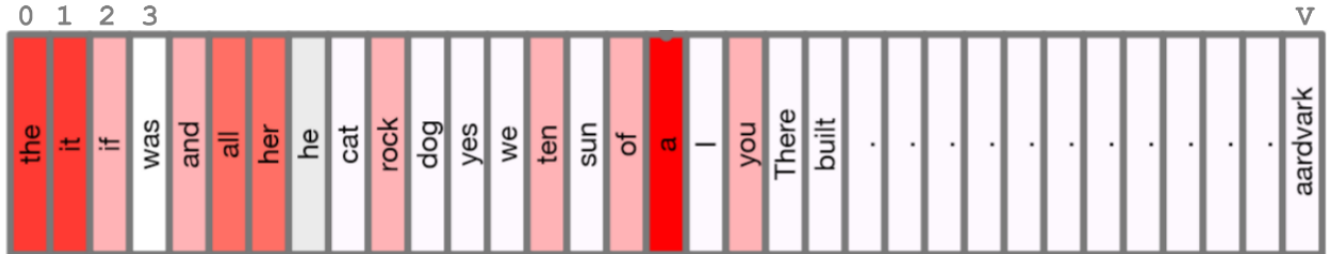
"w": word embedding
";": concatenate

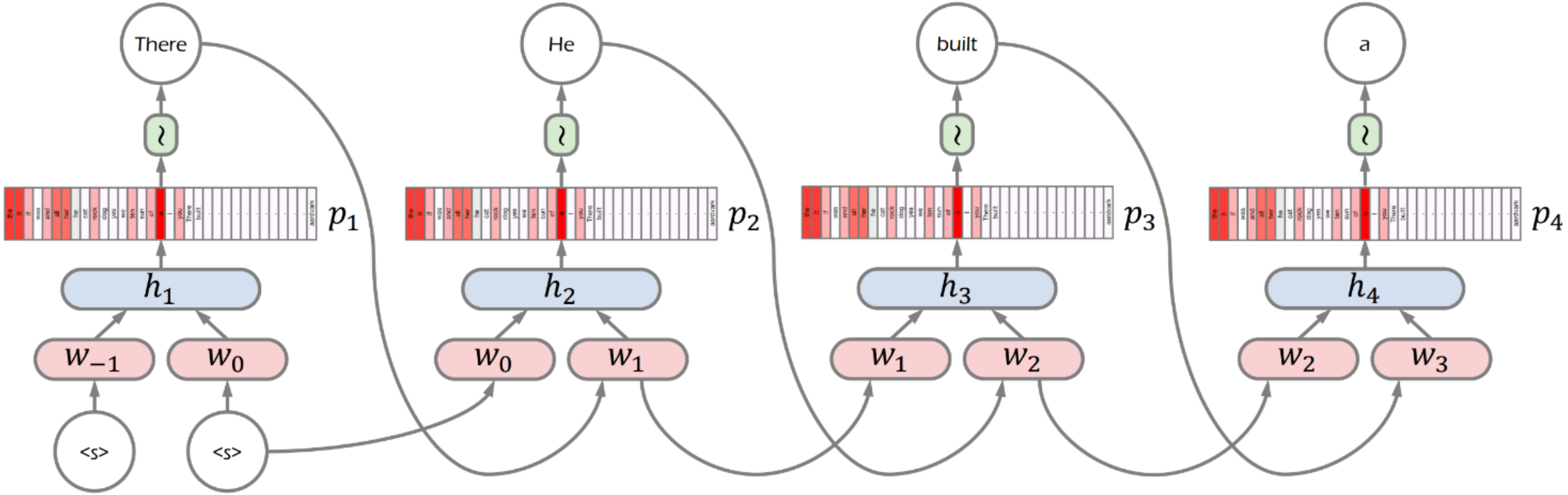
NLP: Language modeling and RNN Models

Approach2 : Deep neural net N-Gram

Example

Build a new sequence from Corpus
by learning the language model

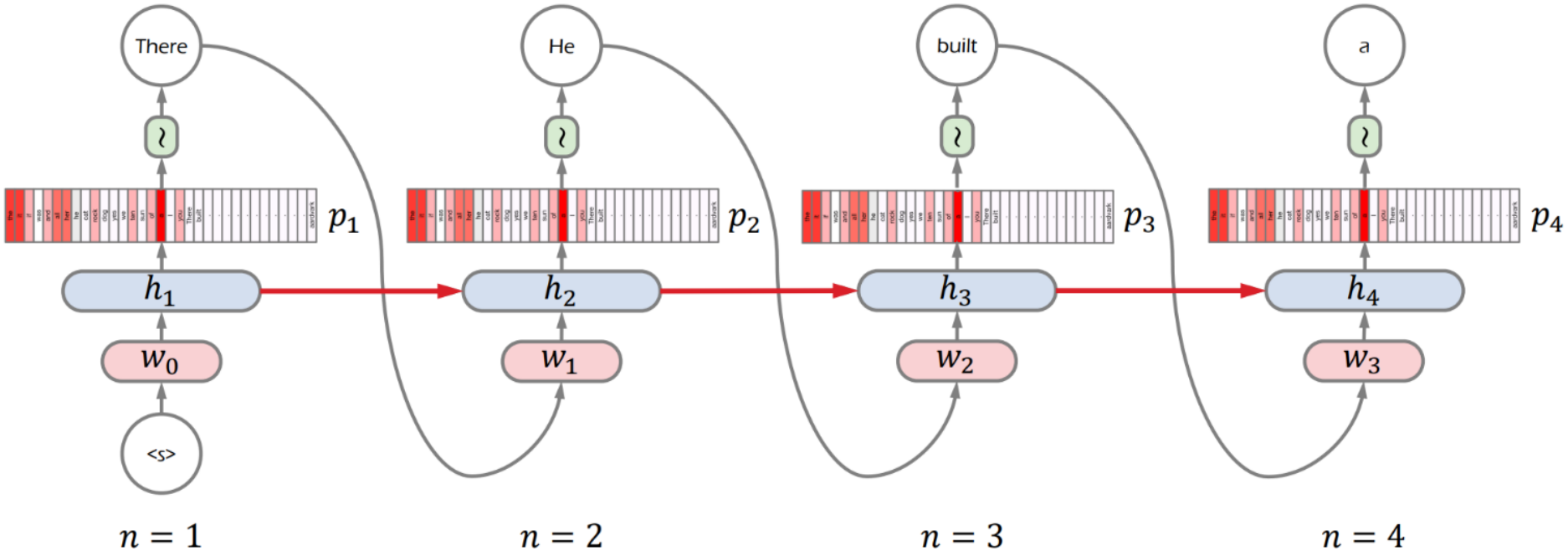




NLP: Language modeling and RNN Models

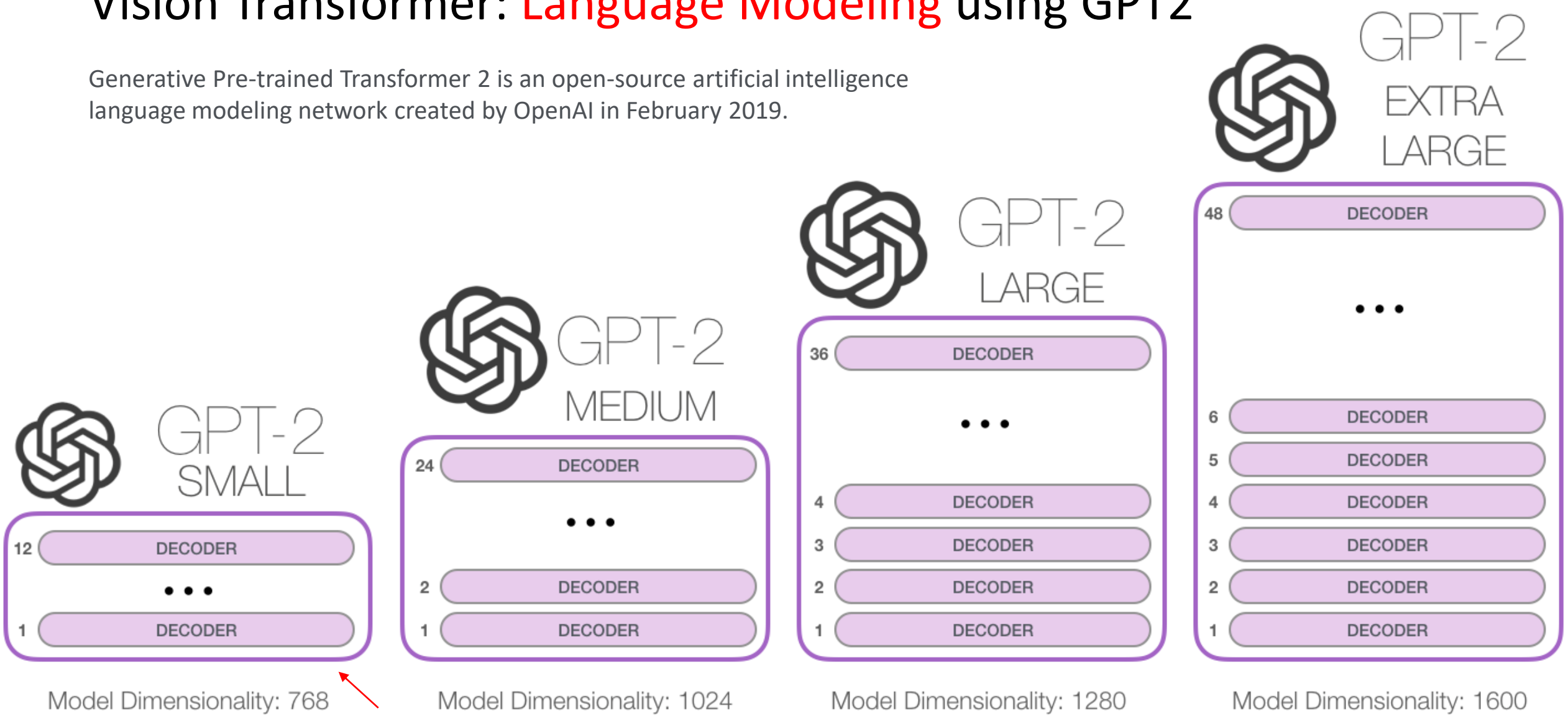
Approach3: RNN Models

$$h_n = f(W_h\,[x_n; h_{n-1}] + b_h) \qquad x_n = w_{n-1} \;\rightarrow\; y_n = w_n = x_{n+1}$$

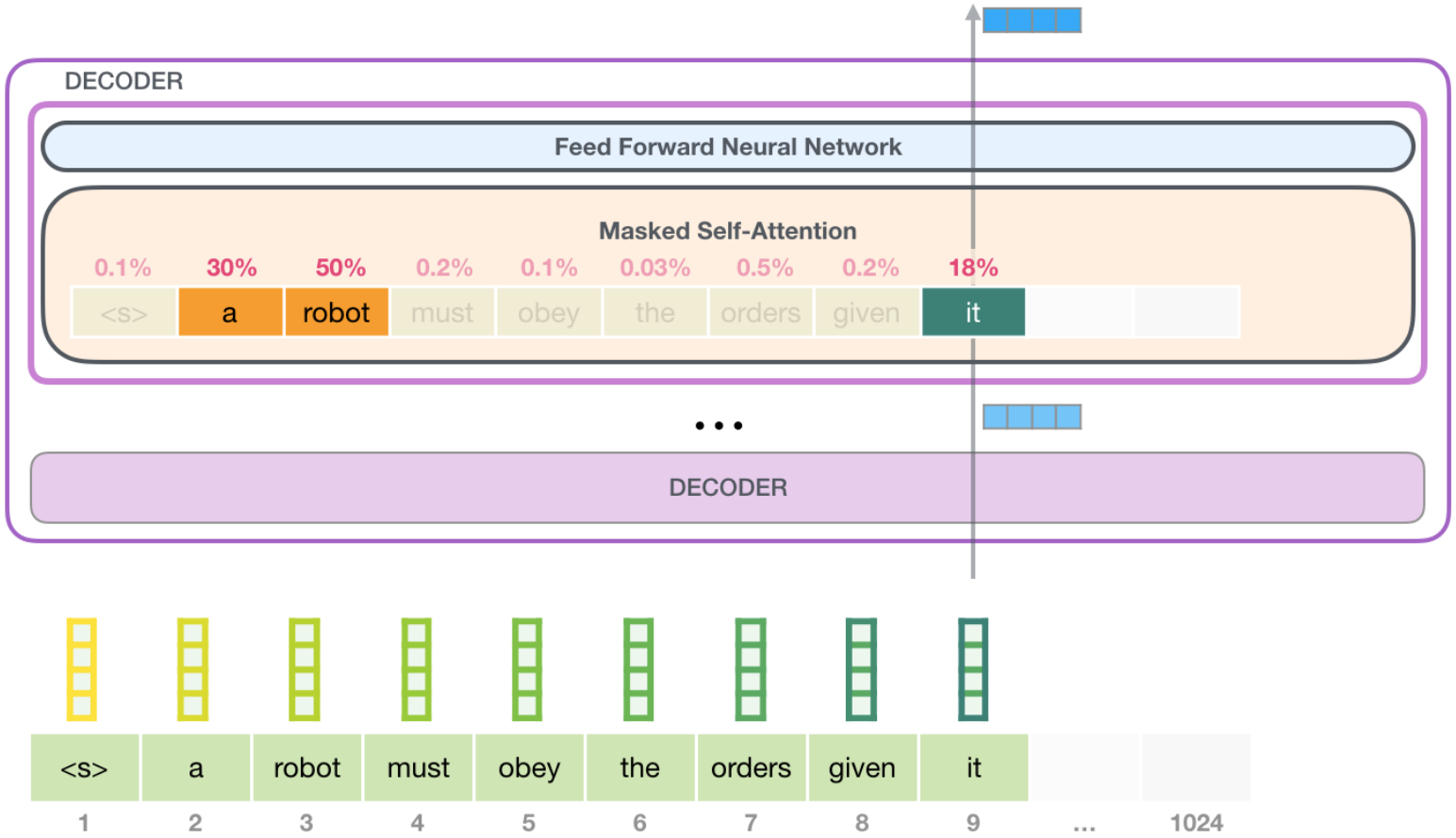


Vision Transformer: Language Modeling using GPT2

Generative Pre-trained Transformer 2 is an open-source artificial intelligence language modeling network created by OpenAI in February 2019.



Vision Transformer: GPT2 Decoder



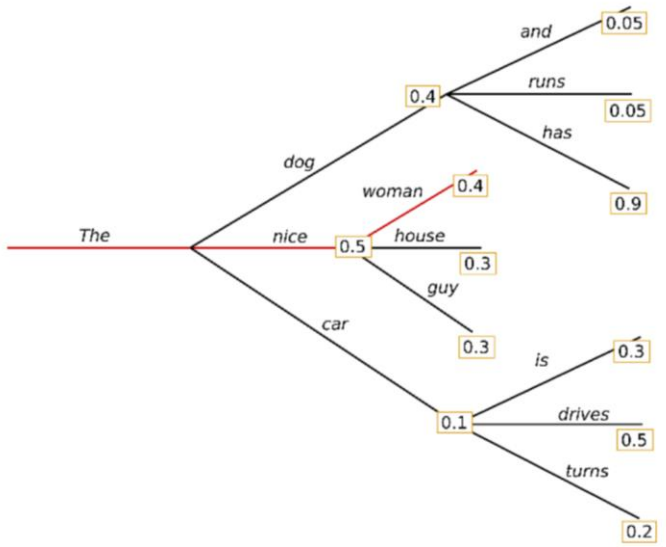


Figure 3.15: An example of a greedy search.
Image source: [9]

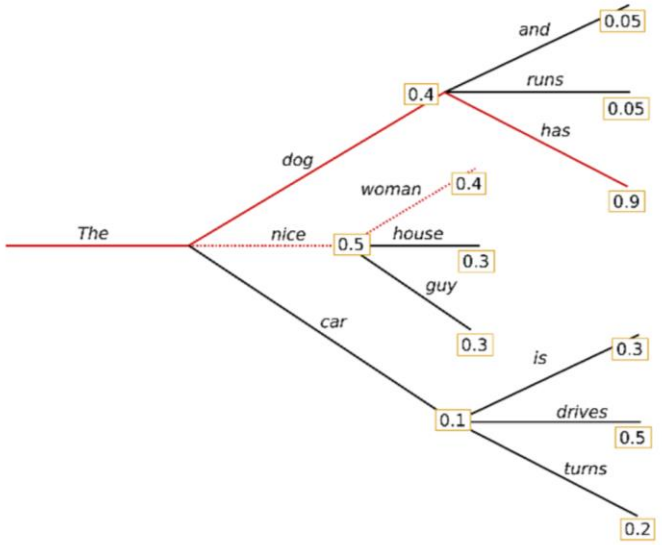


Figure 3.16: An example of a beam search.
Image source: [9]

Vision Transformer: Results on ImageNet Dataset

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Figure 20: Comparison with state of the art on popular image classification benchmarks on ImageNet.

Vision Transformer: Language Modeling using GPT2

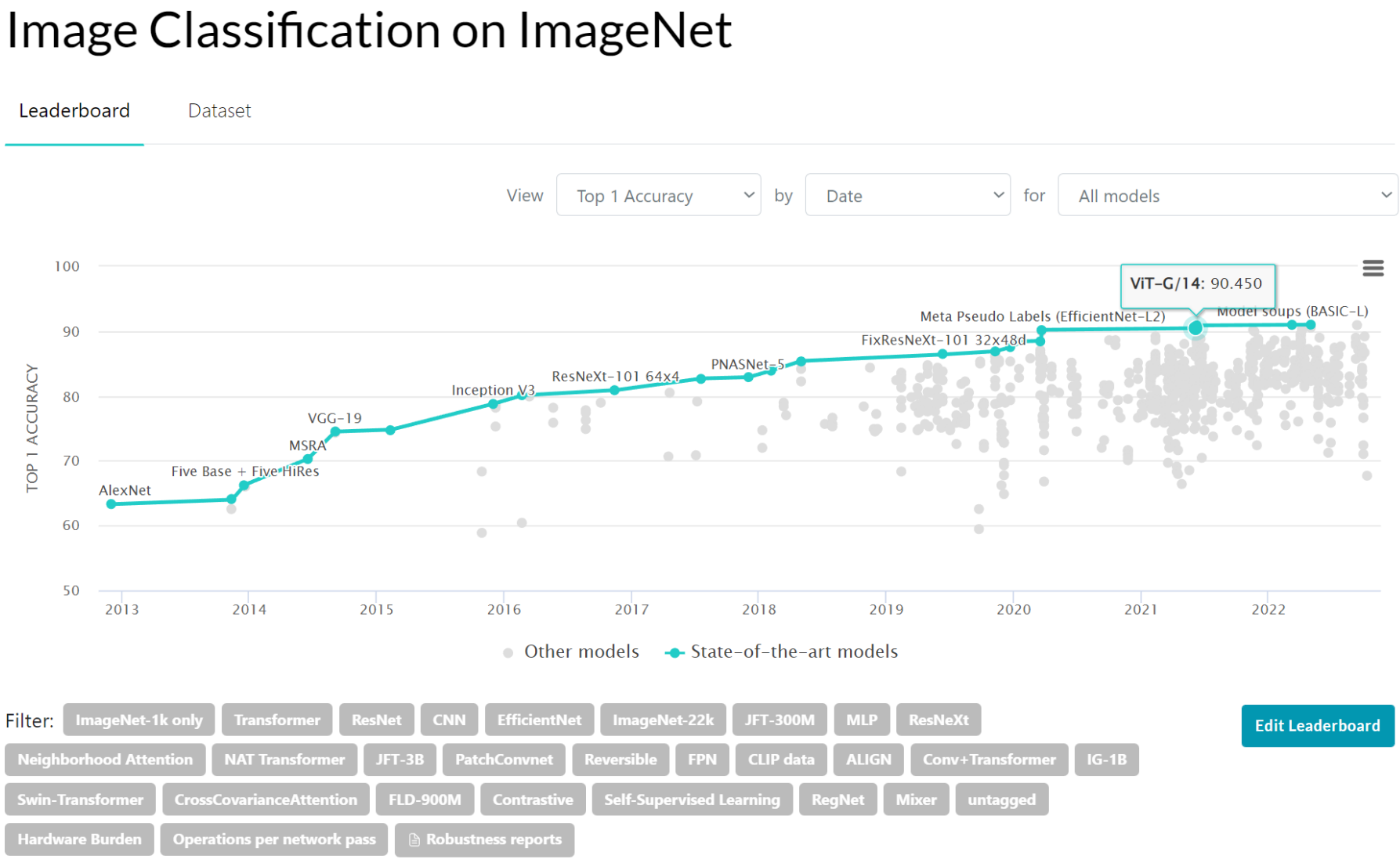


Figure 21: Comparison with state of the art on popular image classification benchmarks on ImageNet.

Vision Transformer: Language Modeling using GPT2

Rank	Model	Top 1 Accuracy	Top 5 Accuracy	Number of params	GFLOPs	Extra Training Data	Paper	Code	Result	Year	Tags
1	**CoCa** (finetuned)	91.0%		2100M		✓	CoCa: Contrastive Captioners are Image-Text Foundation Models			2022	ALIGN, Transforme, JFT-3B
2	**Model soups** (BASIC-L)	90.98%		2440M		✓	Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time			2022	Conv+Transformer, JFT-3B, ALIGN
3	**Model soups** (ViT-G/14)	90.94%		1843M		✓	Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time			2022	JFT-3B, Transforme
4	**ViT-e**	90.9%		3900M		✓	PaLI: A Jointly-Scaled Multilingual Language-Image Model			2022	Transformer, JFT-3I

Vision Transformer: COCO Dataset and Results

[Microsoft COCO Dataset]
http://cocodataset.org/#captions-2015

More than 80k training images and 40k validation images.
At least 5 captions for every image.

Yih-Dar SHIEH 17:15 (39 minutes ago)
to me

Hi Mohammad Taghizadeh,

The COCO dataset is used, but I might have some pre-processing performed, like removing `A photo of ...` or `A black and white picture of ...`

I didn't specify any particular hyperparameters, but the init. LR was 3e-5.

Regarding the metric, this is what I had

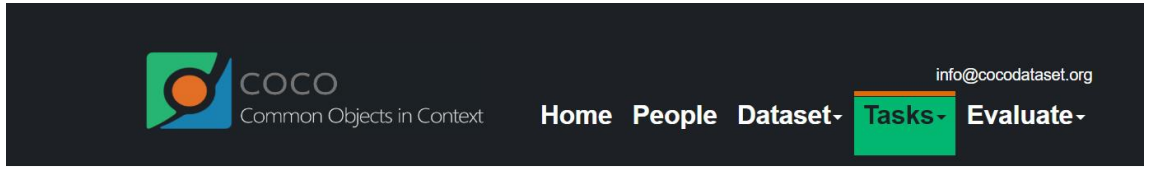
Epoch... (5/30 | Step: 11535 | Eval Loss: 1.8884644508361816 | Eval rouge1: 41.1904 | Eval rouge2: 15.6084 | Eval rougeL: 37.3854 | Eval rougeLsum: 37.3791 | Eval gen_len: 11.4548 |)

which should be the default metrics when you run HF (image to text) summarization task training example script.

Best Regards,
Yih-Dar

Mohammad Taghizadeh <taghizadeh.mhmd@gmail.com> 於 2022年11月1日 週二 凌晨12:23寫道:

Validation Rouge-L Metric = 37.38



COCO 2015 Image Captioning Task

The man at bat readies to swing at the pitch while the umpire looks on.

A large bus sitting next to a very tall building.

1. Overview

Update: Challenge winners were announced at CVPR 2015. The COCO caption evaluation server remains open. Please submit new results to compare to state-of-the-art methods using several automatic evaluation metrics. The COCO 2015 Captioning Challenge is now, however, complete. Results were presented as part of the CVPR 2015 Large-scale Scene Understanding (LSUN) workshop and are available to view on the leaderboard.

This captioning challenge is part of the Large-scale Scene Understanding (LSUN) CVPR 2015 workshop organized by Princeton University. For further details please visit the LSUN website.

The COCO Captioning Challenge is designed to spur the development of algorithms producing image captions that are informative and accurate. Teams will be competing by training their algorithms on the COCO 2014 dataset and having their results scored by **human judges**.

2. Dates

April 1, 2015	Training and testing data, and evaluation software released
May 29, 2015	Submission deadline at 11:59 PST
June 5, 2015	Challenge results (human judgment) released
June 12, 2015	Winner presents at the LSUN Workshop at CVPR 2015

Vision Transformer implementation on Hugging Face and Results

```
print(human_caption[200])
print(vit_caption[200])
```

```
['Several books are stacked on a table. ', 'A set of books sitting on top of a shelf.', 'A few books lined up beside each other on the shelf. ', 'The books are stacked on the shelve on the rack ', 'A shelf with many different types of books.']
a stack of books on top of a shelf
```

```
len(human_caption), len(vit_caption), len(results)
```

```
(1678, 1678, 1678)
```

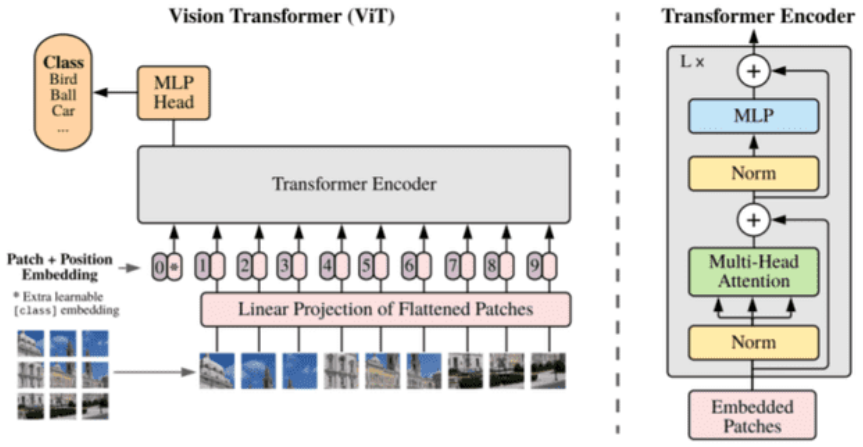
Calculate BLEU Score using NLTK (sentence_bleu)

```
from nltk.translate.bleu_score import sentence_bleu, SmoothingFunction

sum_blue = 0
for i in range(len(results)):
    reference = []
    for item in human_caption[i]: reference.append(item.split())
    candidate = vit_caption[i].split()
    sum_blue += sentence_bleu(reference, candidate, smoothing_function=SmoothingFunction().method4)
print(f"Average BLEU score -> {sum_blue/len(results)}")
```

```
Average BLEU score -> 0.25589140693836127
```

Transformers



https://huggingface.co/ydshieh/vit-gpt2-coco-en-ckpts/tree/main

Validation BLEU Metric = 0.26 on 1678 images
in COCO Caption Validation Dataset

Implementations and Consequence

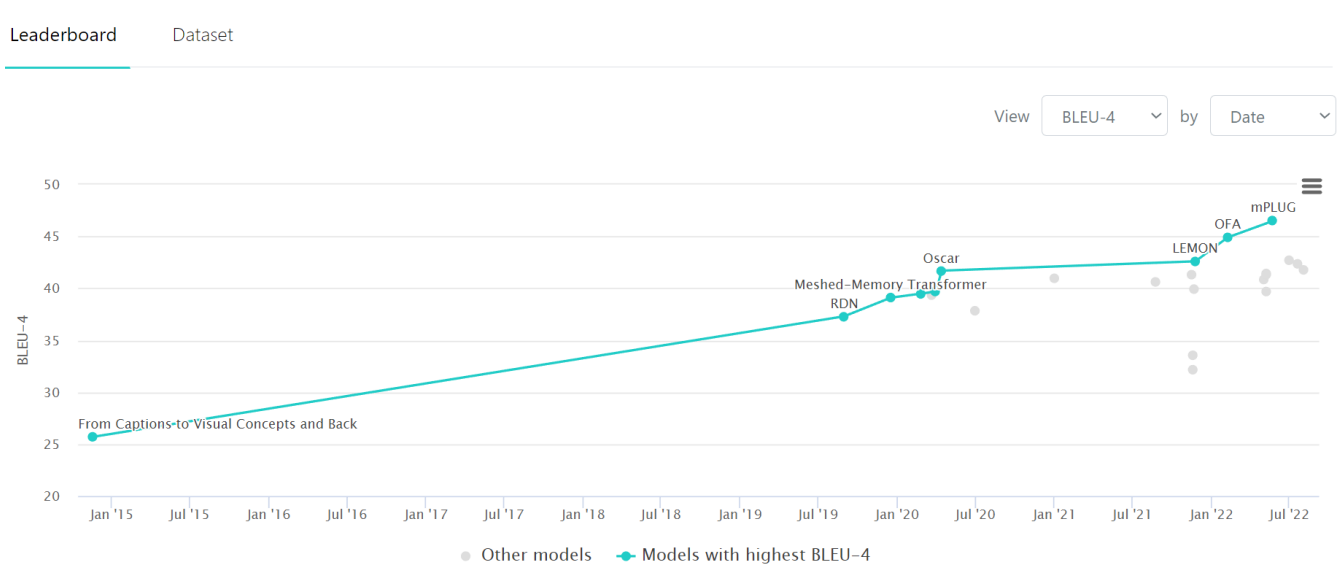
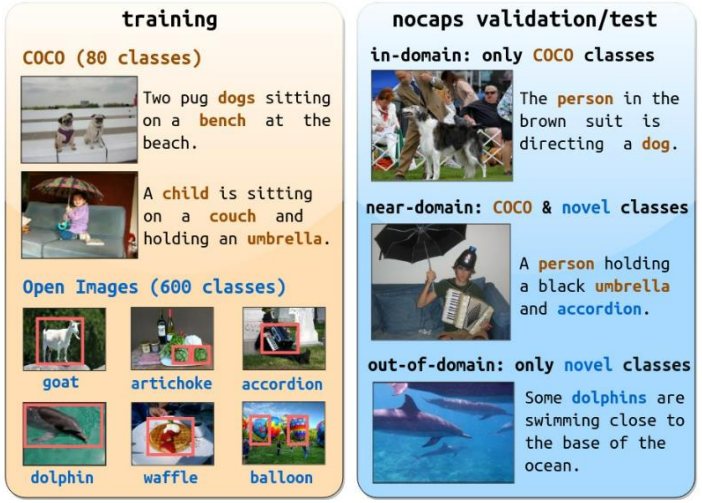


Figure 22: Image Captioning on COCO Captions. In this chart, you can see the ranking of Image captioning models measured on the COCO dataset. The measurement criterion is BLEU
https://paperswithcode.com/sota/image-captioning-on-coco-captions

Implementations link: https://github.com/M-Taghizadeh/BigData_Projects/blob/master/Image%20Captioning/Image_Captioning.ipynb

The number of generated caption words depends on the number of patch sizes of the ViT model.
In this implementation, we have about 16 words for 16x16 image patches based on base paper.

Future work: Nocaps vs COCO



The **nocaps** benchmark for novel object captioning (at scale).

Image captioning models have achieved impressive results on datasets containing limited visual concepts and large amounts of paired image-caption training data. However, if these models are to ever function in the wild, a much larger variety of visual concepts must be learned, ideally from less supervision. To encourage the development of image captioning models that can learn visual concepts from alternative data sources, such as object detection datasets, we present the first large-scale benchmark for this task. Dubbed **nocaps**, for novel object captioning at scale, our benchmark consists of 166,100 human-generated captions describing 15,100 images from the Open Images validation and test sets. The associated training data consists of COCO image-caption pairs, plus Open Images image-level labels and object bounding boxes. Since Open Images contains many more classes than COCO, nearly 400 object classes seen in test images have no or very few associated training captions (hence, **nocaps**). We extend existing novel object captioning models to establish strong baselines for this benchmark and provide analysis to guide future work.

Figure 23: The Nocaps benchmark for novel object captioning (at scale).

https://nocaps.org/download

Nocaps: novel object captioning at scale

1. In COCO, we have 5 captions for each image, but in this dataset, we have 10 captions for each image.
2. Nocaps has provided more object classes than COCO.

nocaps v0.1

Validation Set

- Sourced from Open Images validation set (v4).
- 4500 images, 10 captions per image.
  - 648 **in-domain** images.
  - 2938 **near-domain** images.
  - 914 **out-domain** images.

Download Validation Set Captions

Annotations Format

```
{
    "licenses": [],
    "info": {
        "url": "http://nocaps.org",
        "date_created": "2018/11/06",
        "version": "0.1",
        "description": "nocaps validation dataset",
        "contributor": "nocaps team",
        "year": 2018
    },
    "images": [
        {
            "id": 0,
            "open_images_id": "0013ea2087020901",
            "height": 1024,
            "width": 732,
            "coco_url": "https://s3.amazonaws.com/nocaps/val/0013ea2087020901.jpg",
            "file_name": "0013ea2087020901.jpg",
            "license": 0,
            "date_captured": "2018-11-06 11:04:33"
        },
    ]
    "annotations": [  // This field is absent in test set.
        {
            "image_id": 0,
            "id": 0,
            "caption": "A baby is standing in front of a house."
        }
    ]
}
```

References

[1] A. Vaswani et al.
Attention is All you Need
Advances in Neural Information Processing Systems, 2017

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby.
An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
9th International Conference on Learning Representations, {ICLR} 2021

[3] García Gilabert, Javier
Image Captioning using pre-trained GPT-2 models
universitat politècnica de valència, 2021

[4] Li, Minghao and Lv, Tengchao and Chen, Jingye and Cui, Lei and Lu, Yijuan and Florencio, Dinei and Zhang, Cha and Li, Zhoujun and Wei, Furu
TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models, 2021

[5] M. Kaur and A. Mohta.
A Review of Deep Learning with Recurrent Neural Network
International Conference on Smart Systems and Inventive Technology (ICSSIT), 2019

[6] S. Hochreiter and J. Schmidhuber.
Long Short-Term Memory
Neural Computation, 1997

References

[7] F. Karim, S. Majumdar, H. Darabi and S. Chen
LSTM Fully Convolutional Networks for Time Series Classification
IEEE Access, 2018

[8] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich.
A survey on long short-term memory networks for time series prediction
Procedia CIRP, 2021

[9] Luong, Thang, Hieu Pham and Christopher D. Manning.
Effective Approaches to Attention-based Neural Machine Translation
EMNLP, 2015

[10] M. Sundermeyer, H. Ney and R. Schlüter.
From Feedforward to Recurrent LSTM Neural Networks for Language Modeling
IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2015

[11] I. Goodfellow et al.
Generative Adversarial Nets
Advances in Neural Information Processing Systems, 2014

[12] https://github.com/M-Taghizadeh/BigData_Projects