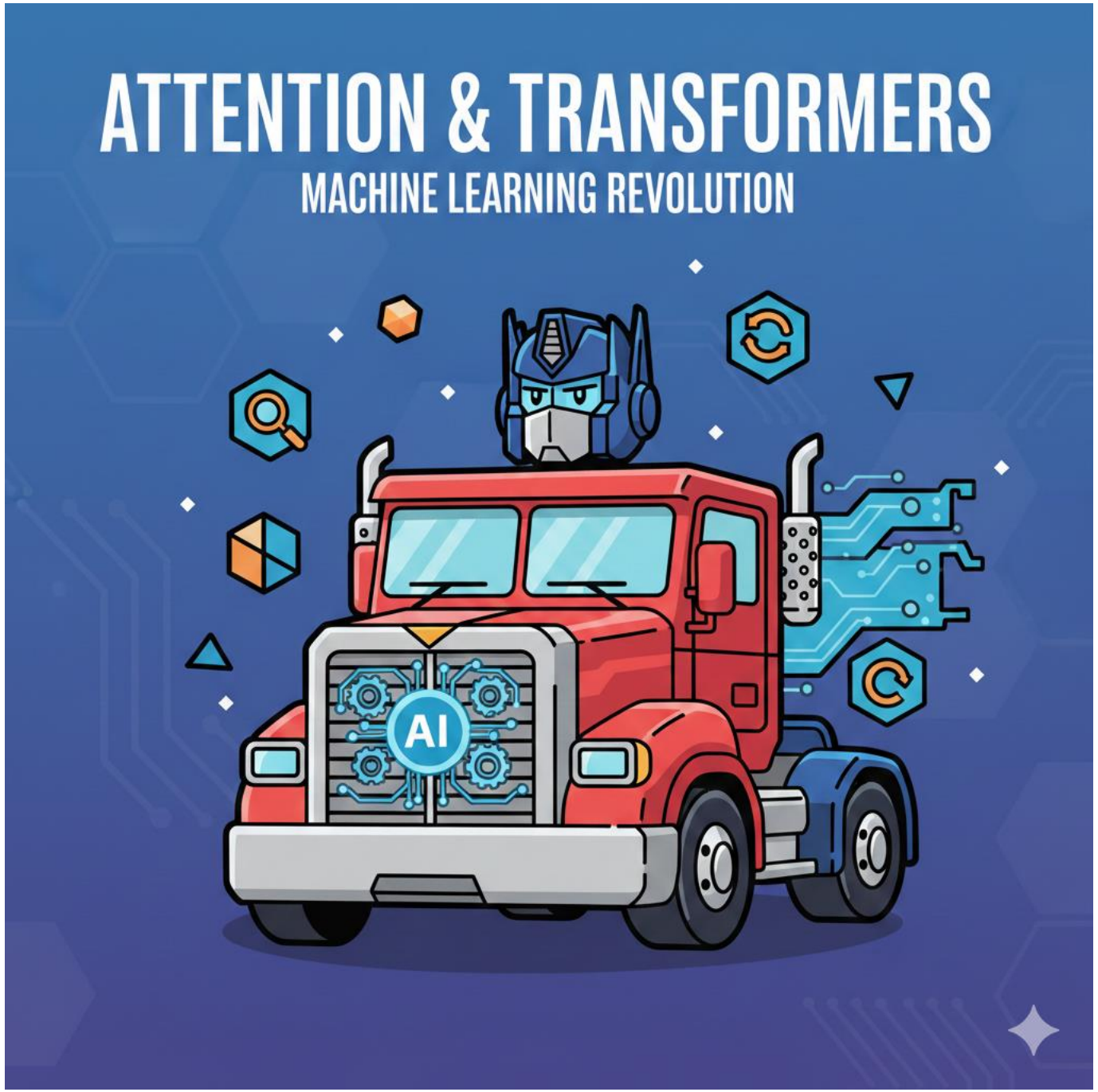


INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

IIIT Hyderabad

Makarand Tapaswi
https://makarandtapaswi.github.io/

AlphaGrep Winter School

11th December 2025

This talk is a historical perspective

- On modern Transformers developments, check article:
- https://www.gleech.org/tplus

The Transformer++

Let the "Transformer++" be a Transformer with

- A fused attention implementation (the scaled dot-product backend -> FlashAttention). Subquadratic memory complexity in input sequence length. Practically: can double GPU utilization and so halve training time. Also enables longer contexts and speeds up inference on long context input.
- Rotary position embedding (sinusoidal -> learned APE -> RoPE)
- Removing attention's redundant key heads and value heads (vanilla MHA -> MQA -> GQA)
- Regularized / preconditioned optimizer (Adam -> AdamW -> SOAP)
- Normalise before each layer (post LayerNorm -> pre LayerNorm)
- When doing layer normalization: just rescale, don't centre (LayerNorm -> RMSNorm)
- Divine activation function for the MLP (GeLU -> … -> SwiGLU or GeGLU)
- Tied embeddings. An oldie but goodie.
- Fix logit drift (query/key normalization)
- Fixing that one softmax off-by-one (fixed in some places around 2021)

Attention is All You Need!

- Almost there ... but, before that

Neural Machine Translation by Jointly Learning to Align and Translate

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio

ICLR 2015 Oral

https://arxiv.org/abs/1409.0473

Sequence-to-Sequence Models

Recurrent Neural Networks

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

- Why recurrent?

- Different variants of function f (GRU, LSTM, etc.)

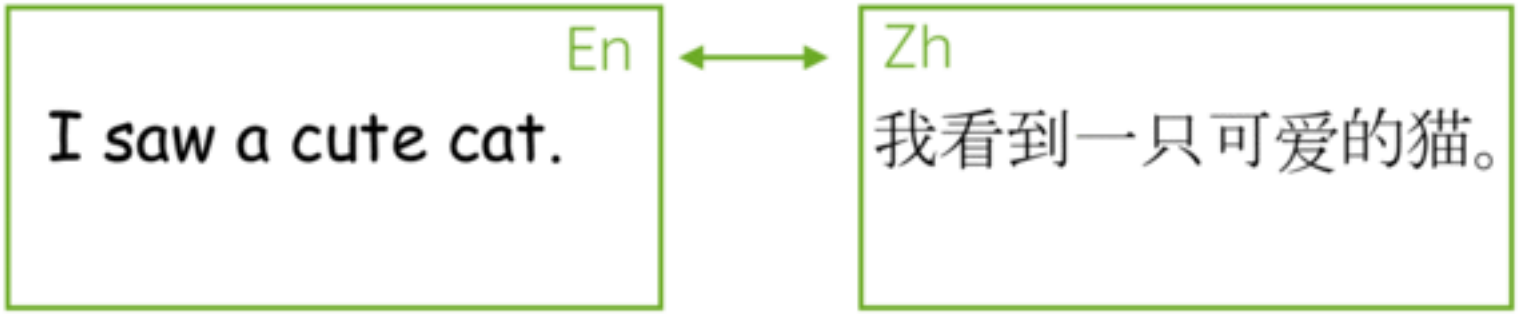
- Long Short-Term Memory Units
- Gated Recurrent Units

Examples of Sequence-to-Sequence tasks?

- Image Captioning
  - (strictly: image input to sequence)
- Sentiment Classification
  - (strictly: sequence to single output)
- Machine Translation
  - yes! sequence to sequence

Very nice material that we will follow!

- https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html#main_content



Formulating machine translation

Human Translation

$$y^* = \arg\max_y p(y|x)$$

The "probability" is intuitive and is given by a human translator's expertise

Machine Translation

model parameters

$$y' = \arg\max_y p(y|x, \theta)$$

Questions we need to answer

- modeling

How does the model for $p(y|x, \theta)$ look like?

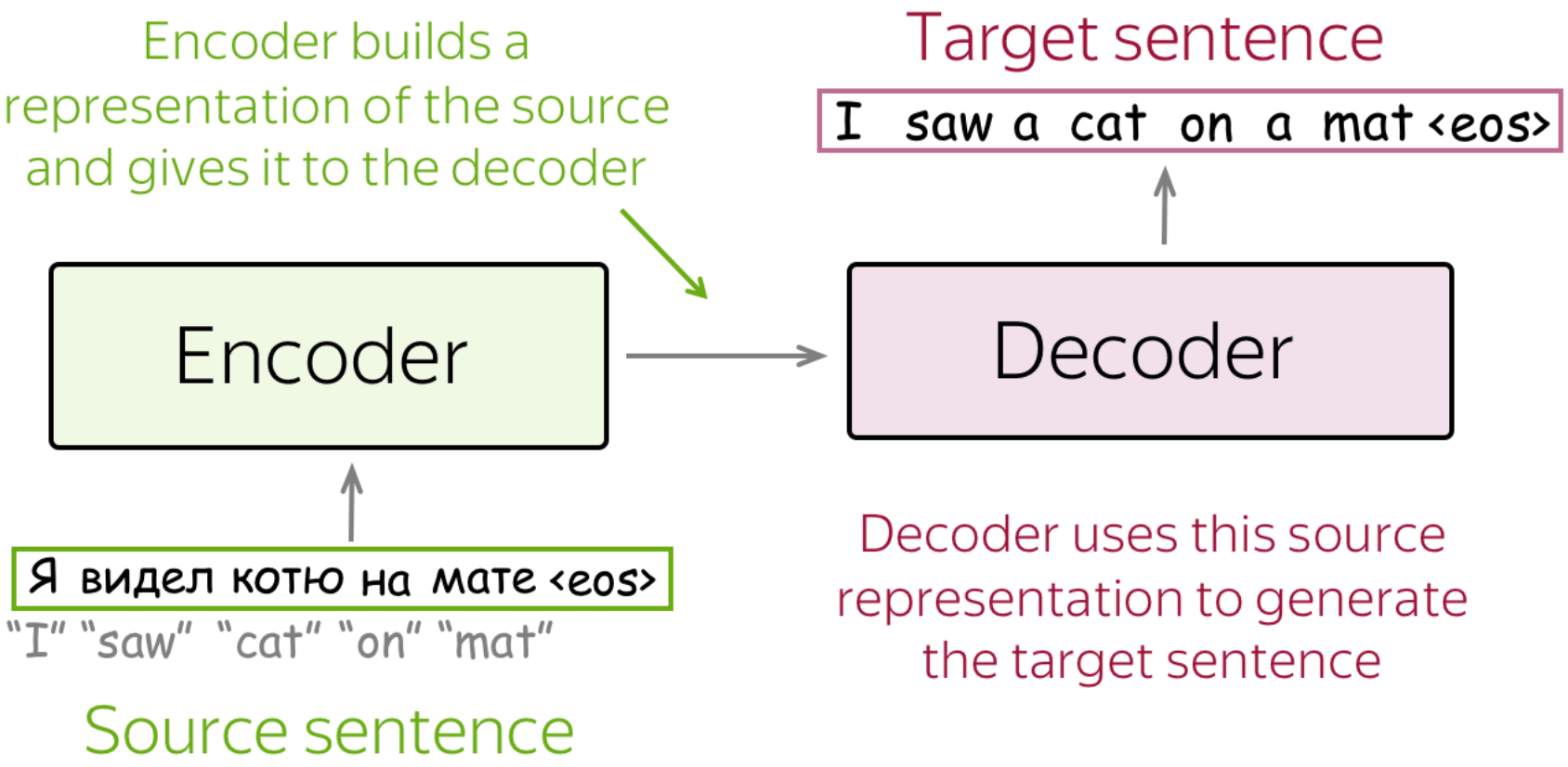
- learning

How to find θ?

- search

How to find the argmax?

Encoder Decoder



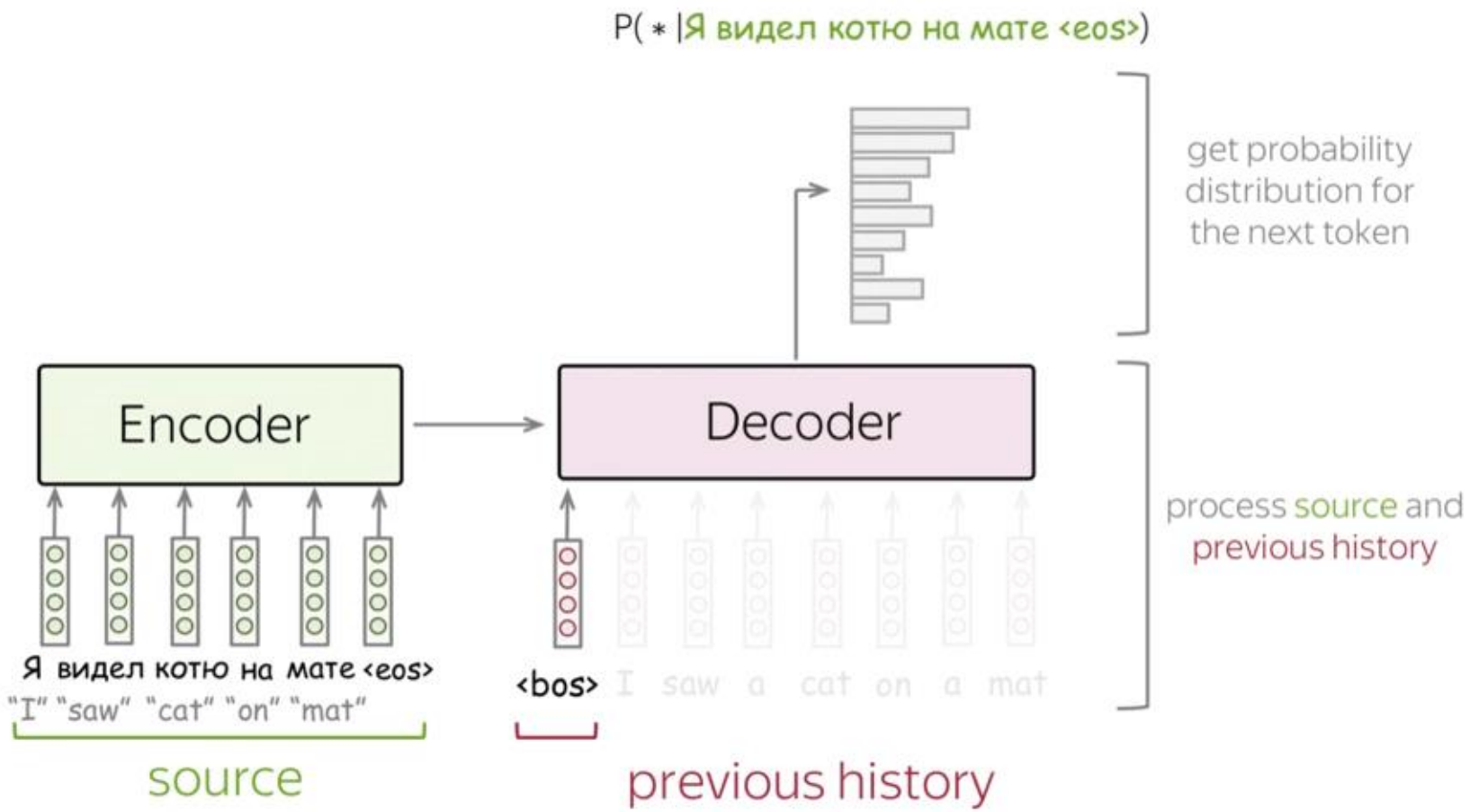
Conditional Language Models

Language Models: $P(y_1, y_2, \ldots, y_n) = \prod_{t=1}^{n} p(y_t | y_{<t})$

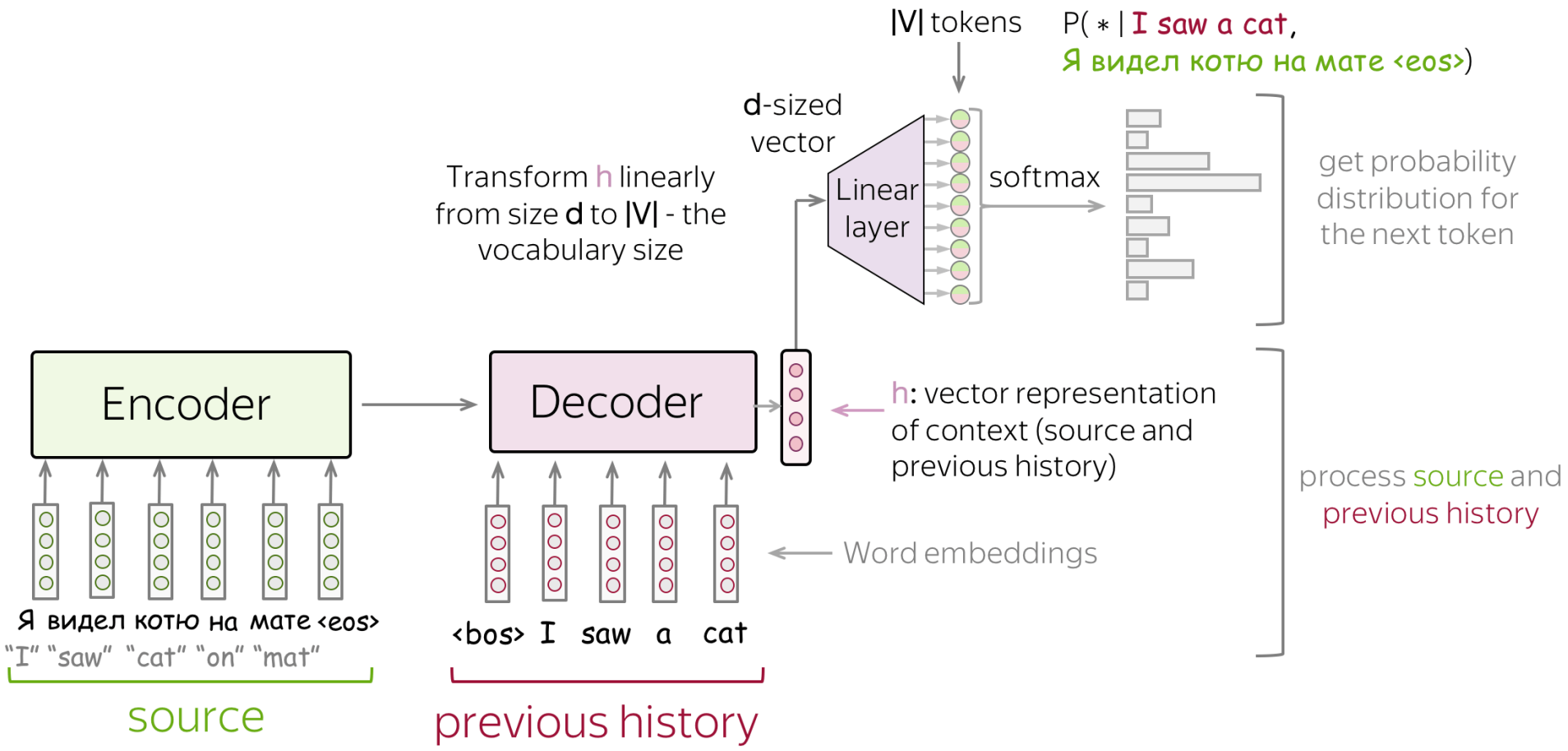
Conditional Language Models: $P(y_1, y_2, \ldots, y_n, | x) = \prod_{t=1}^{n} p(y_t | y_{<t}, x)$

condition on source x

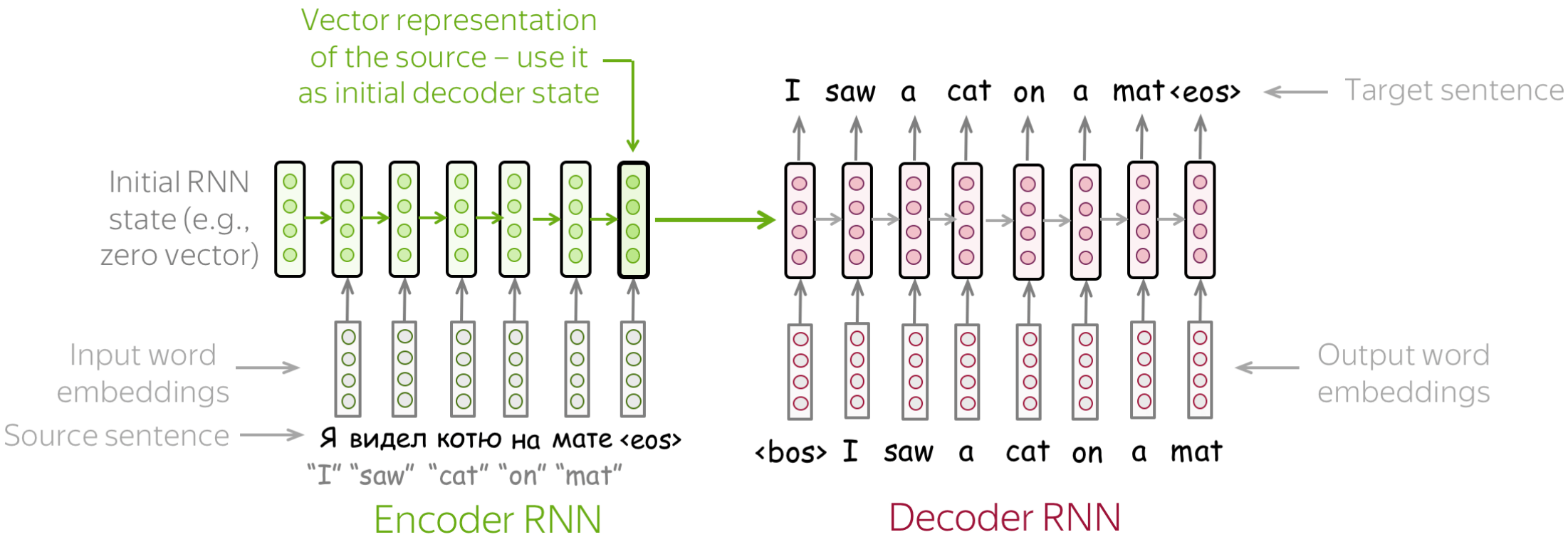
Encoder Decoder explained



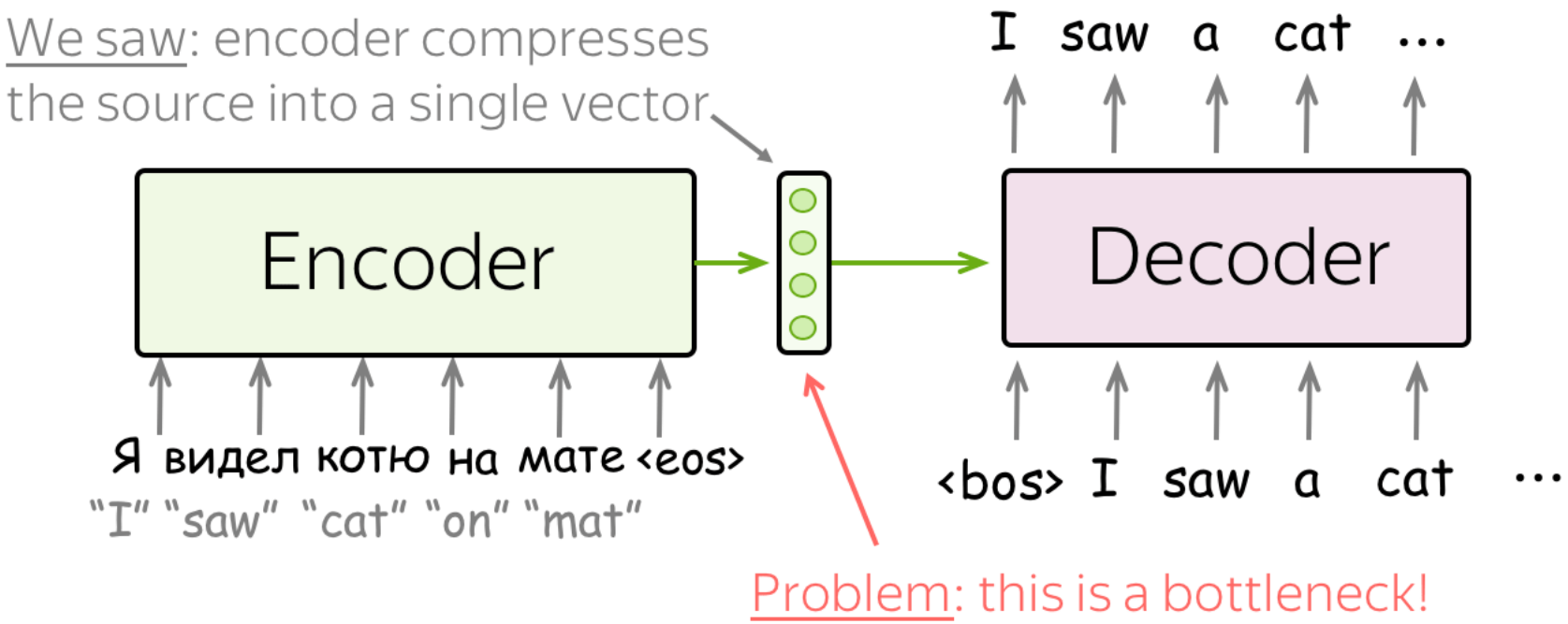
Using a decoder hidden representation



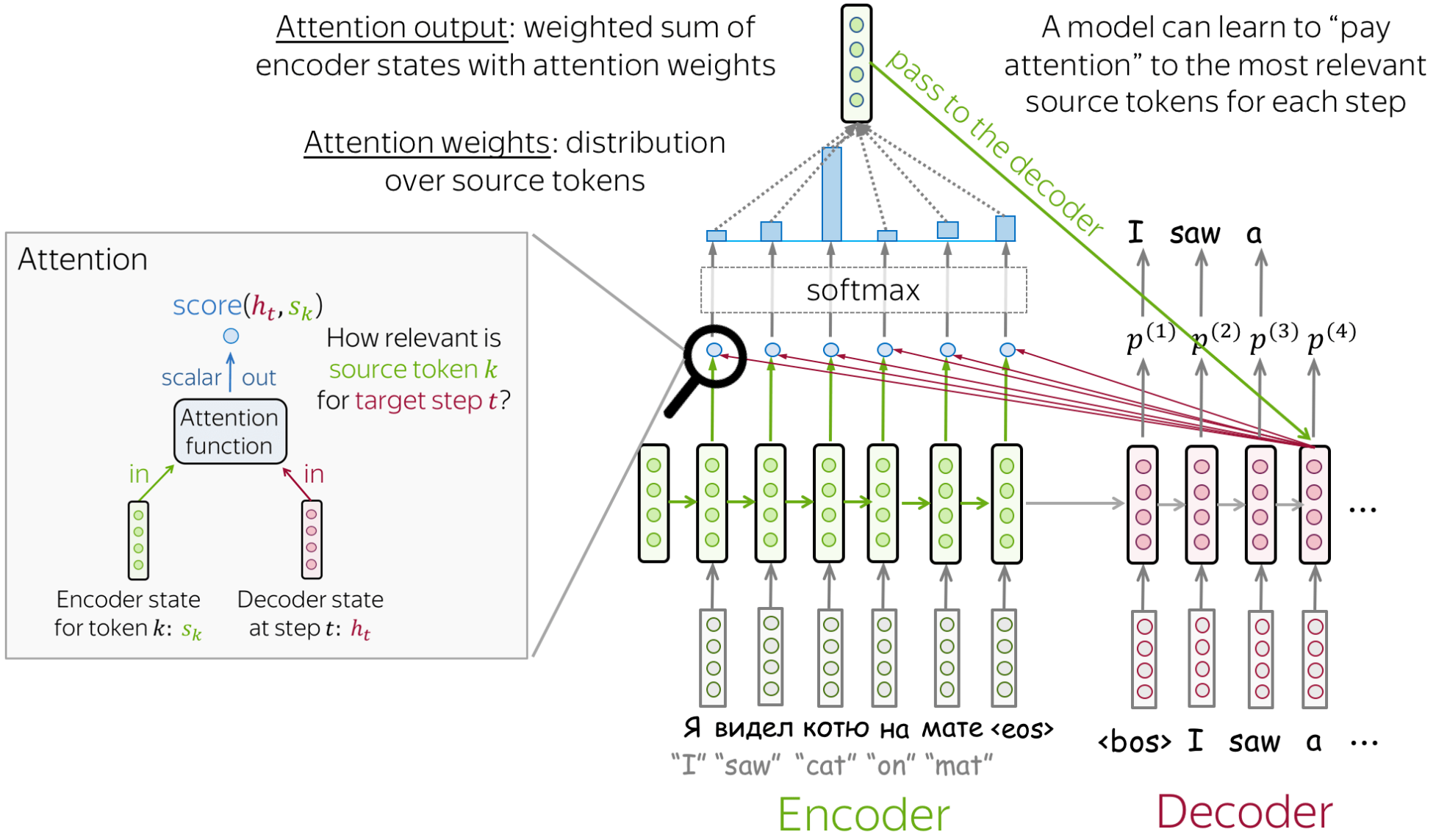
Seq2Seq RNNs



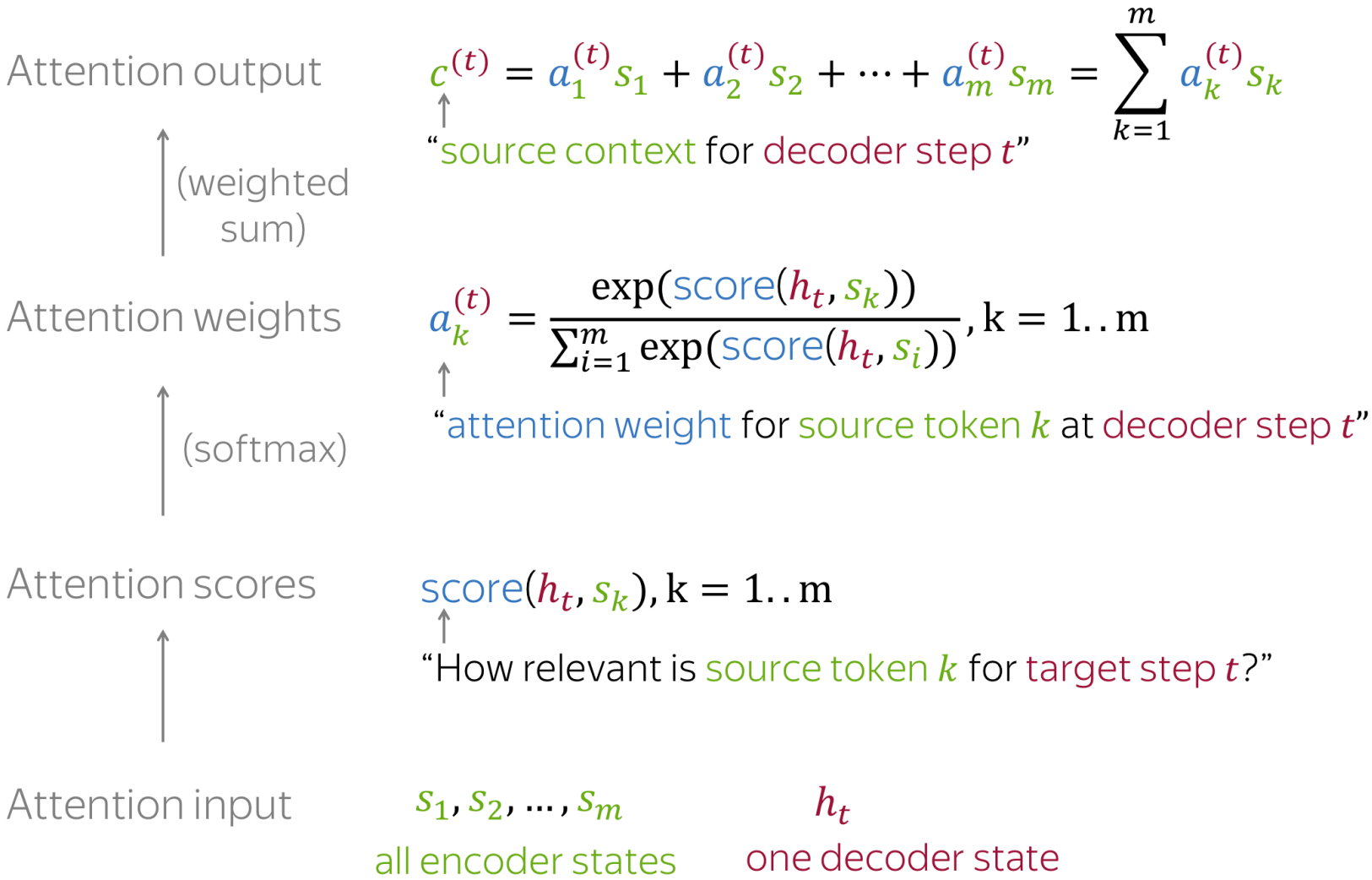
Fixed encoder representations are an issue



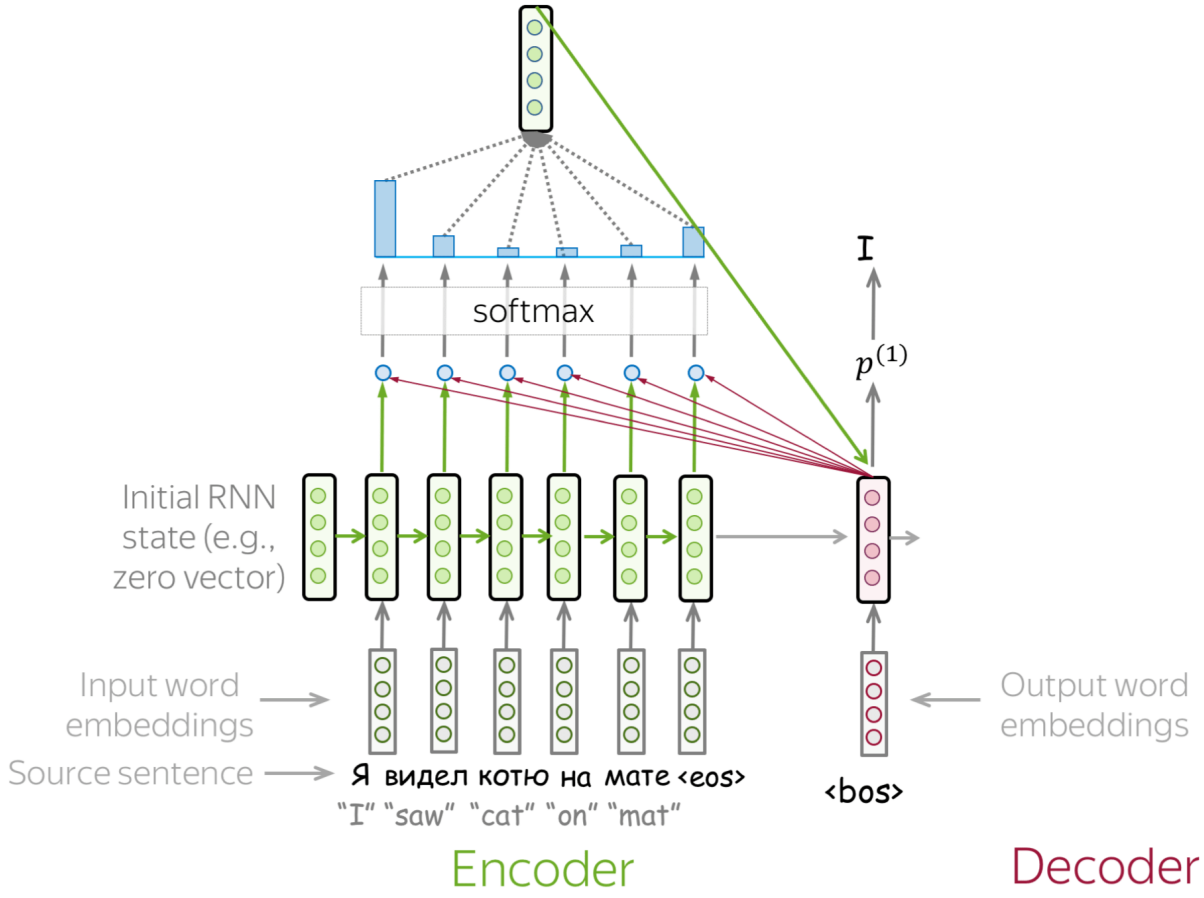
Attention


Attention output: weighted sum of encoder states with attention weights
Attention weights: distribution over source tokens
A model can learn to "pay attention" to the most relevant source tokens for each step
pass to the decoder
Attention
score(h_t, s_k)
scalar
out
Attention function
in
in
How relevant is source token k for target step t?
Encoder state for token k: s_k
Decoder state at step t: h_t
softmax
$p^{(1)}$ $p^{(2)}$ $p^{(3)}$ $p^{(4)}$
I saw a
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
<bos> I saw a ...
Encoder
Decoder

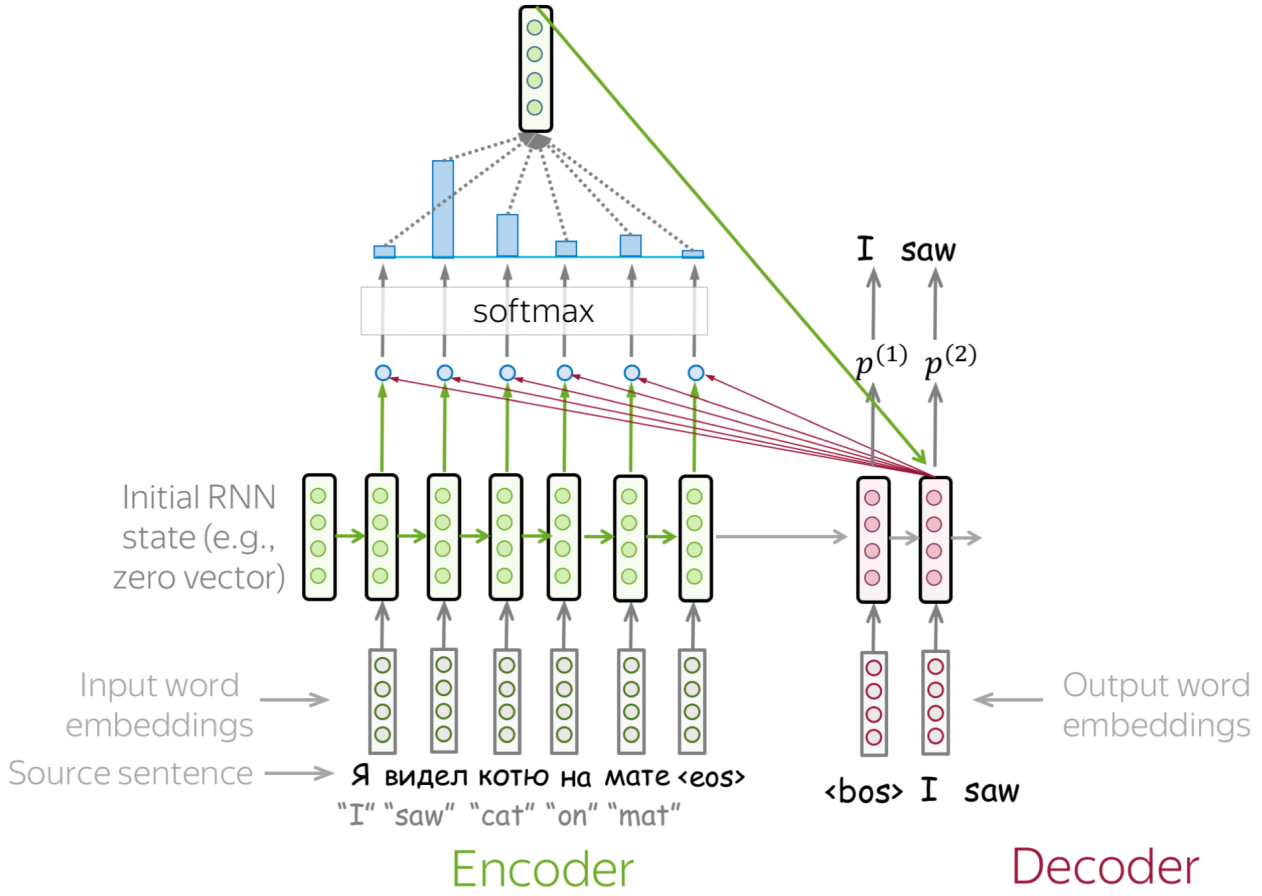
How to compute attention?



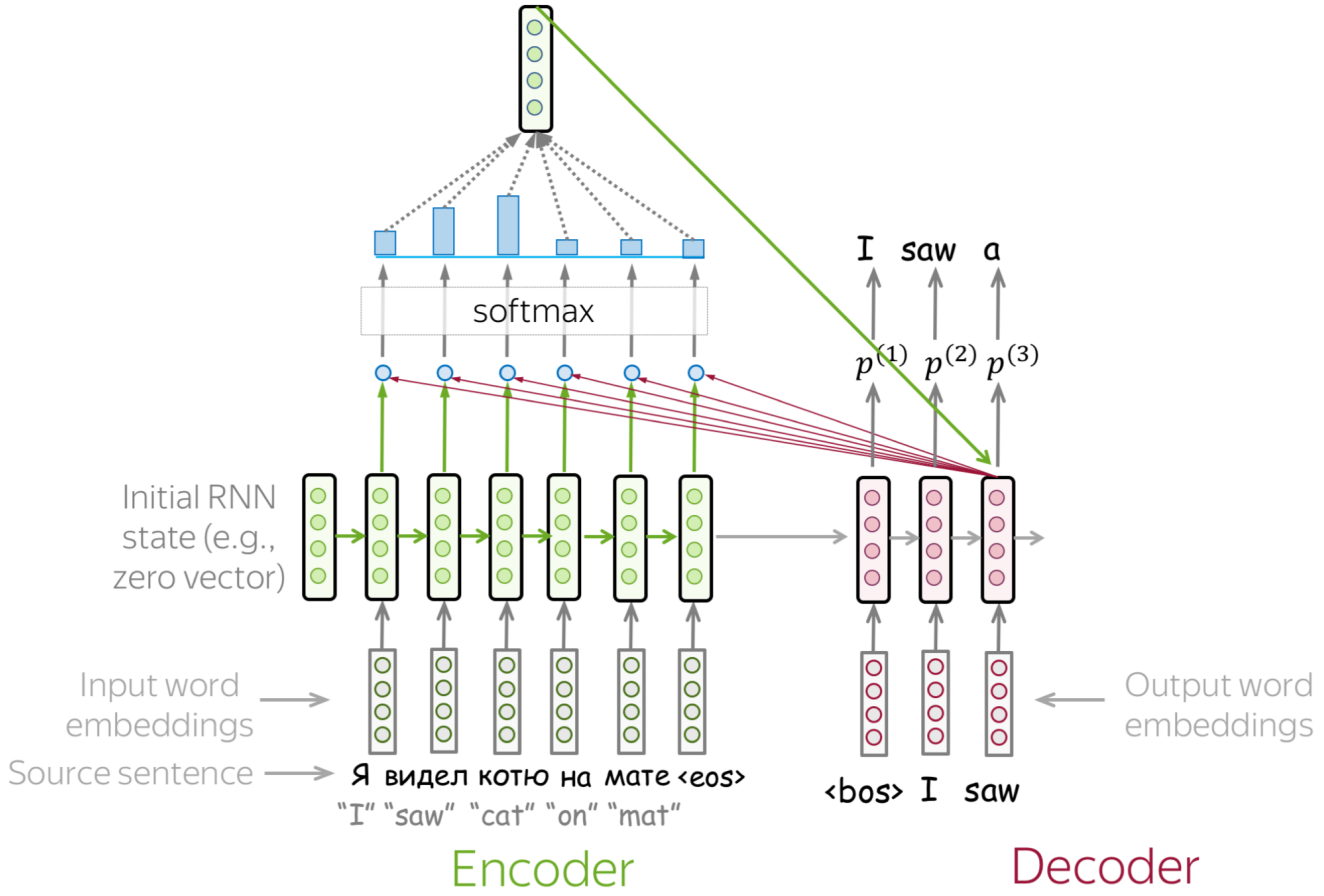
Decoding word 1



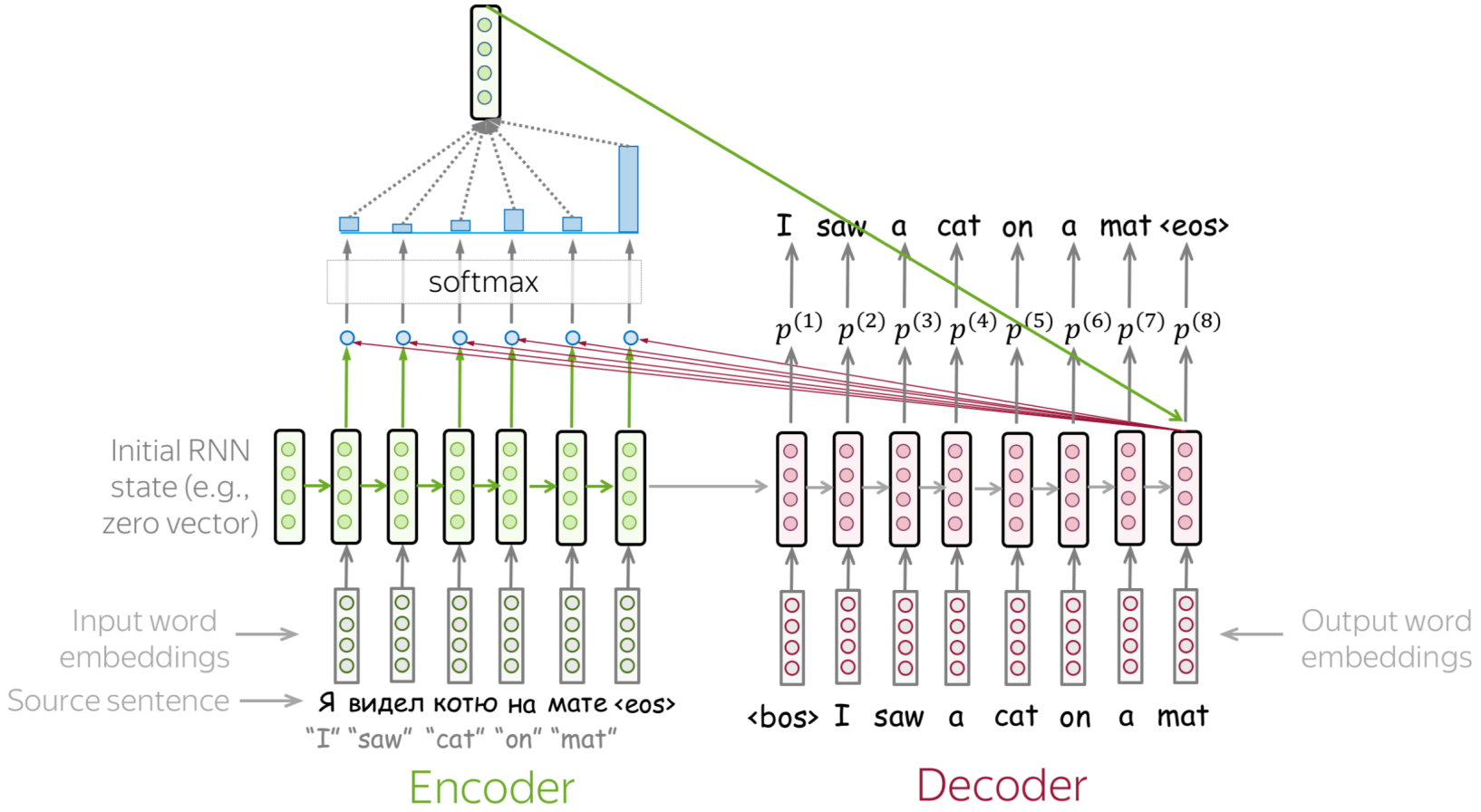
Decoding word 2



Decoding word 3

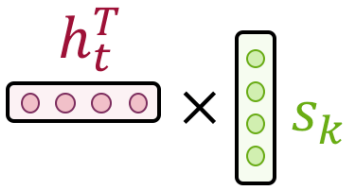


Decoding word N



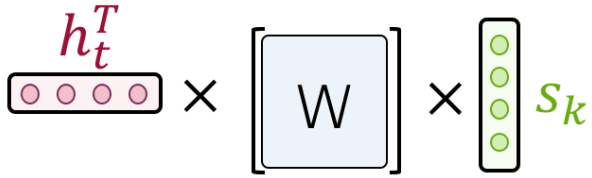
Computing Attention

Dot-product



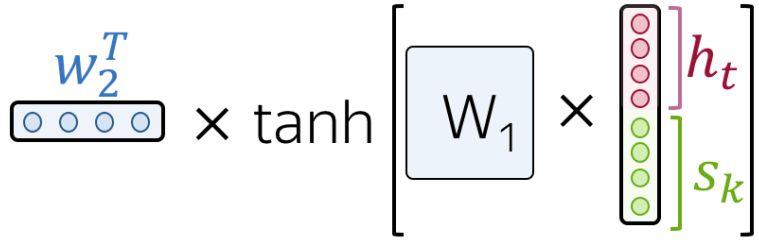
$\text{score}(h_t, s_k) = h_t^T \; s_k$

Bilinear



$\text{score}(h_t, s_k) = h_t^T \; W s_k$

Multi-Layer Perceptron



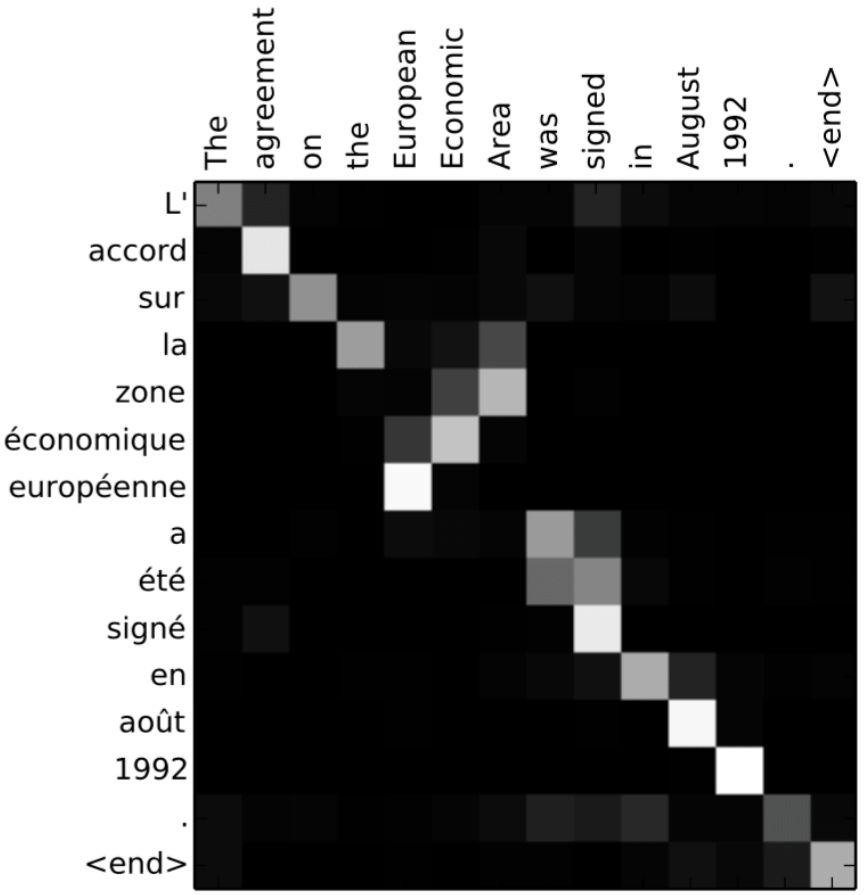
$\text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$

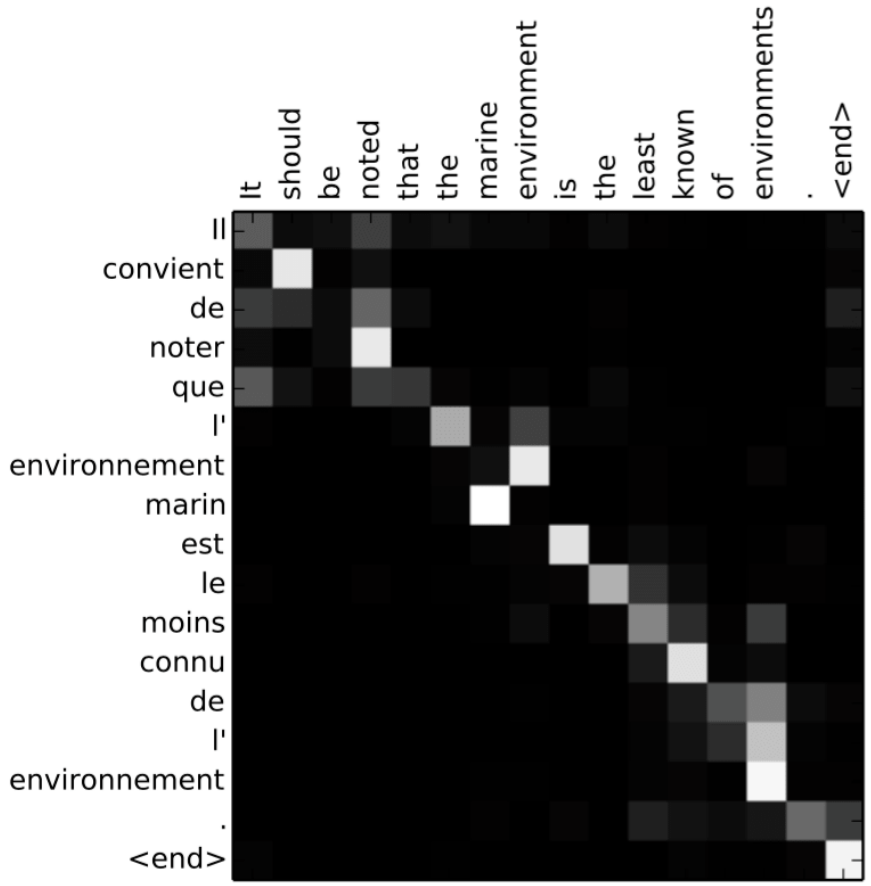
Softmax temperature

$$p_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}}$$

- T == 1: normal softmax
- T >> 1: p closer to uniform distribution (flat)
- T << 1: p closer to one-hot distribution (peaky)

Attention Learns Alignment!





Questions?

Image Captioning

Old ideas from 2015

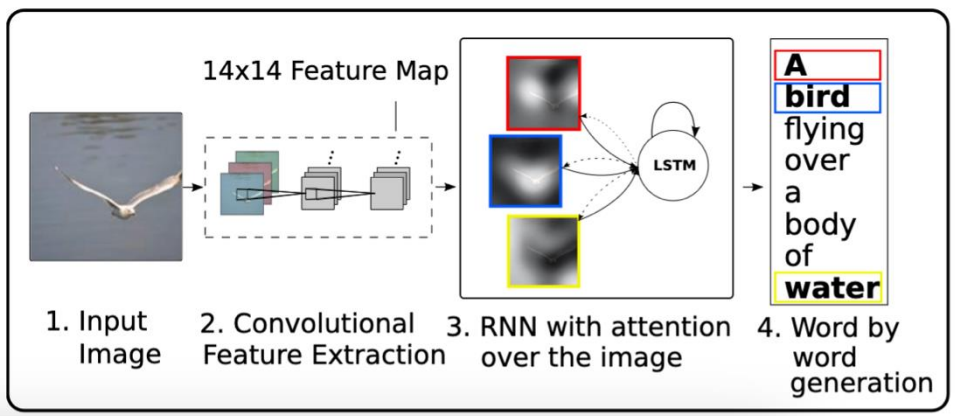
Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

Kelvin Xu KELVIN.XU@UMONTREAL.CA
Jimmy Lei Ba JIMMY@PSI.UTORONTO.CA
Ryan Kiros RKIROS@CS.TORONTO.EDU
Kyunghyun Cho KYUNGHYUN.CHO@UMONTREAL.CA
Aaron Courville AARON.COURVILLE@UMONTREAL.CA
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Richard S. Zemel ZEMEL@CS.TORONTO.EDU
Yoshua Bengio FIND-ME@THE.WEB

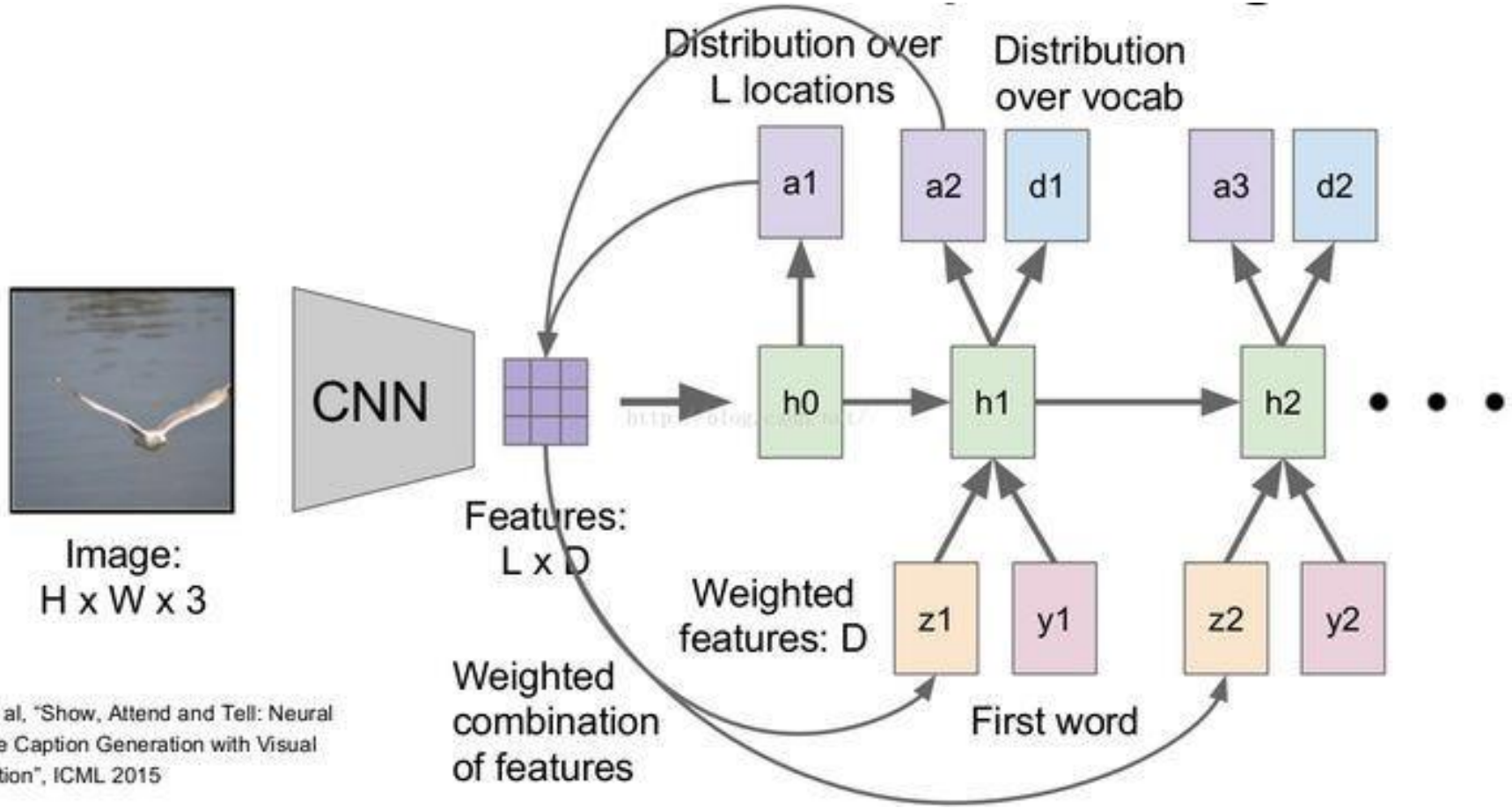
Abstract

Inspired by recent work in machine translation and object detection, we introduce an attention based model that automatically learns to describe the content of images. We describe how we can train this model in a deterministic manner using standard backpropagation techniques and stochastically by maximizing a variational lower bound. We also show through visualization how the model is able to automatically learn to fix its gaze on salient objects while generating the cor-

Figure 1. Our model learns a words/image alignment. The visualized attentional maps (3) are explained in section 3.1 & 5.4



Architecture Details



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

Show and Tell → Show, Attend, and Tell

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Understanding when things go wrong

Figure 5. Examples of mistakes where we can use attention to gain intuition into what the model saw.

A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and
a hat on a skateboard.

A person is standing on a beach
with a surfboard.

A woman is sitting at a table
with a large pizza.

A man is talking on his cell phone
while another man watches.

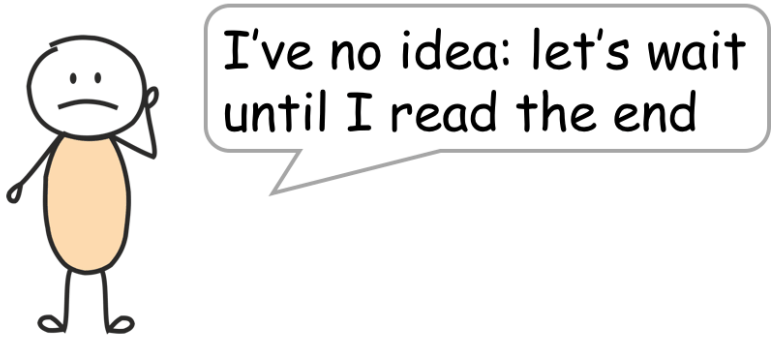
Questions?

Transformer Architecture

RNNs → Transformers

I arrived at the bank after crossing the … …street? …river?

What does **bank** mean in this sentence?



RNNs

O(N) steps to process a sentence with length N

Transformer

Constant number of steps to process any sentence

Attention is All You Need!

- ~~Almost there … but, before that~~

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

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

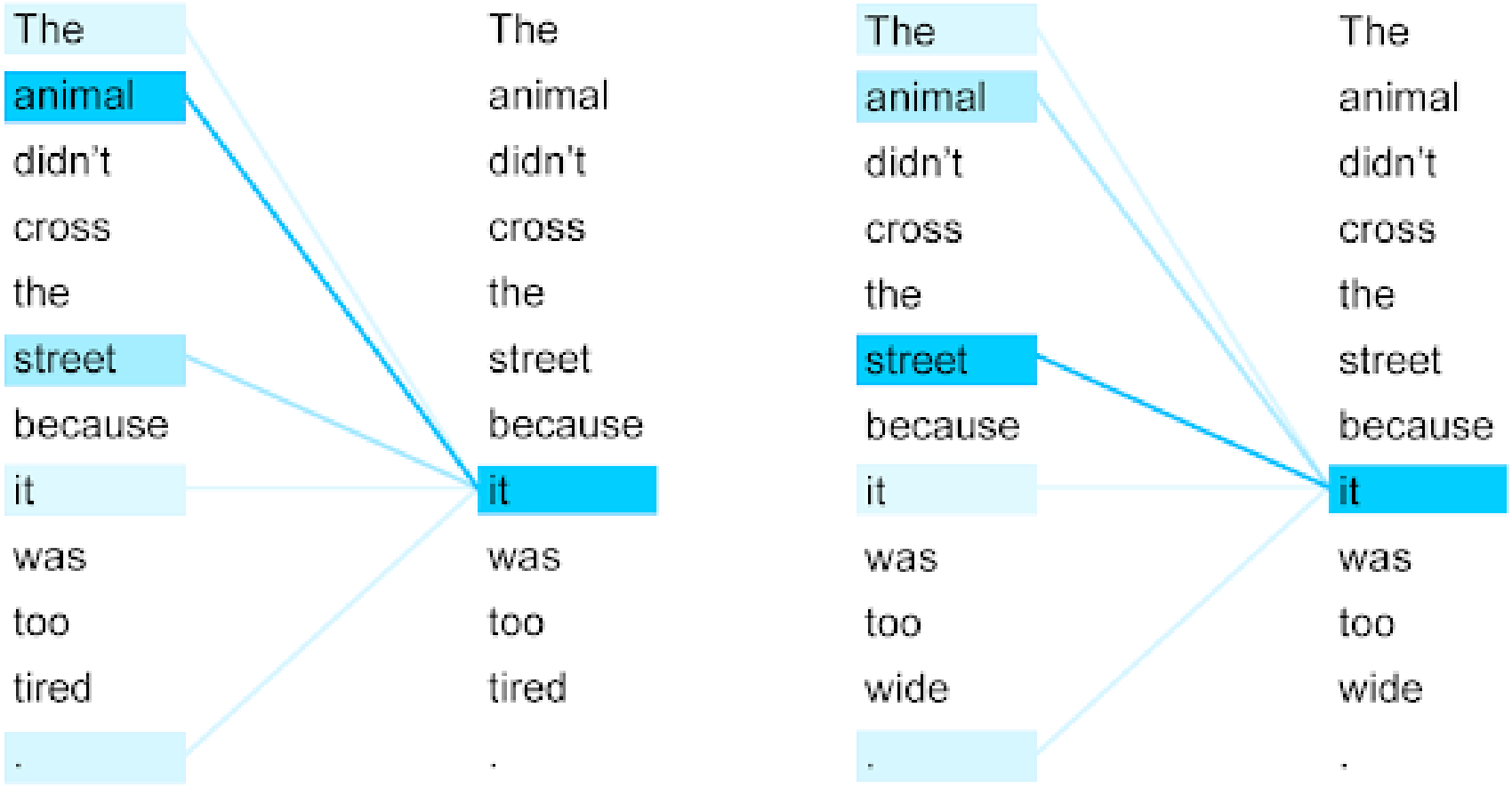
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions

What is “it”?



https://blog.research.google/2017/08/transformer-novel-neural-network.html

Information Flow

https://blog.research.google/2017/08/transformer-novel-neural-network.html

Auto-regressive decoding, Seq2Seq

Encoder

Who is doing:

- all source tokens

What they are doing:

- look at each other
- update representations

repeat N times

Decoder

Who is doing:

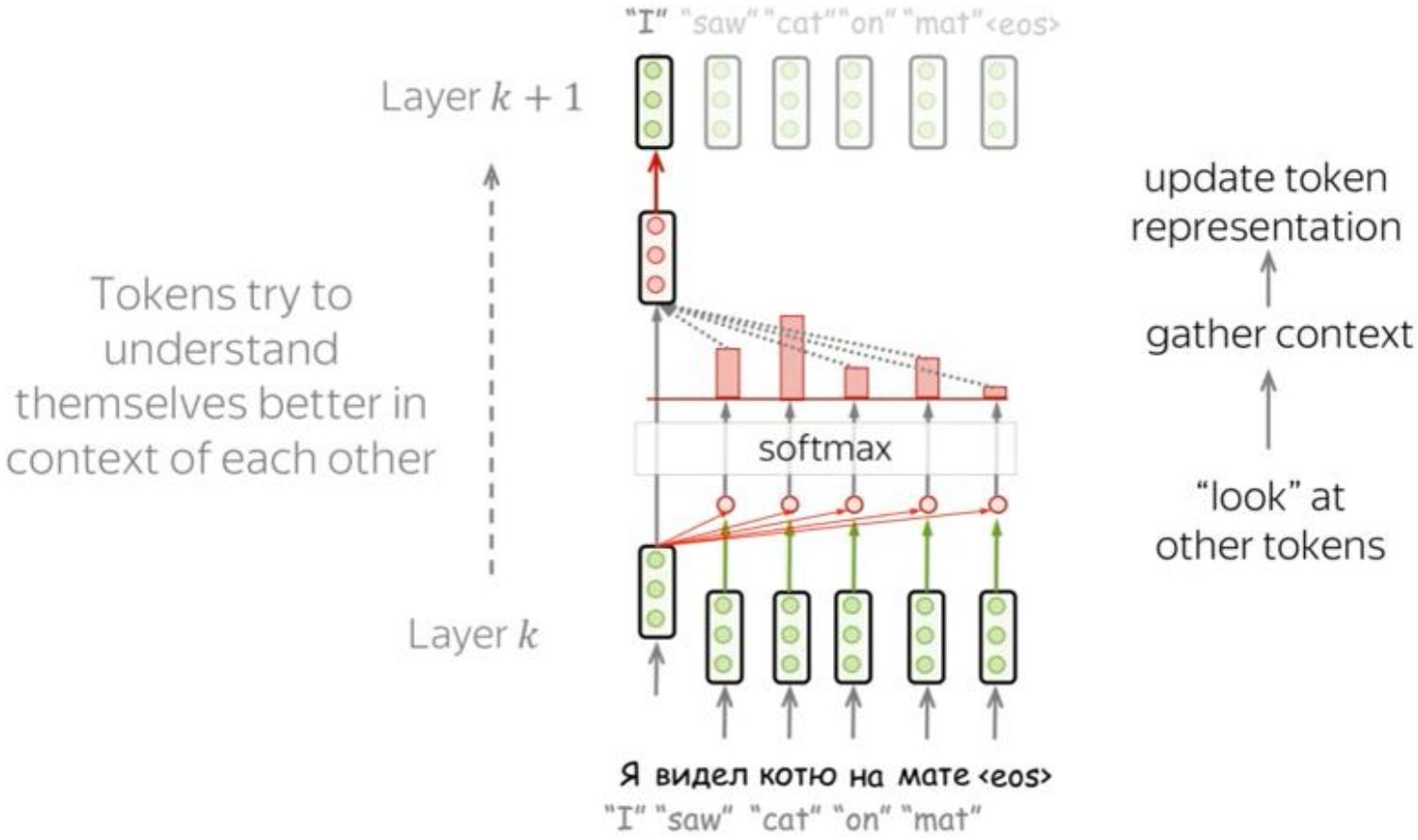
- target token at the current step

What they are doing:

- looks at previous target tokens
- looks at source representations
- update representation

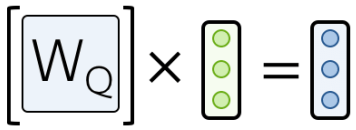
repeat N times

Encoder Self-attention



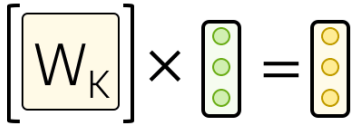
Self-Attention

Each vector receives three representations ("roles")



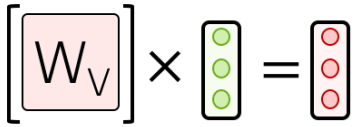
Query: vector **from** which the attention is looking

"Hey there, do you have this information?"



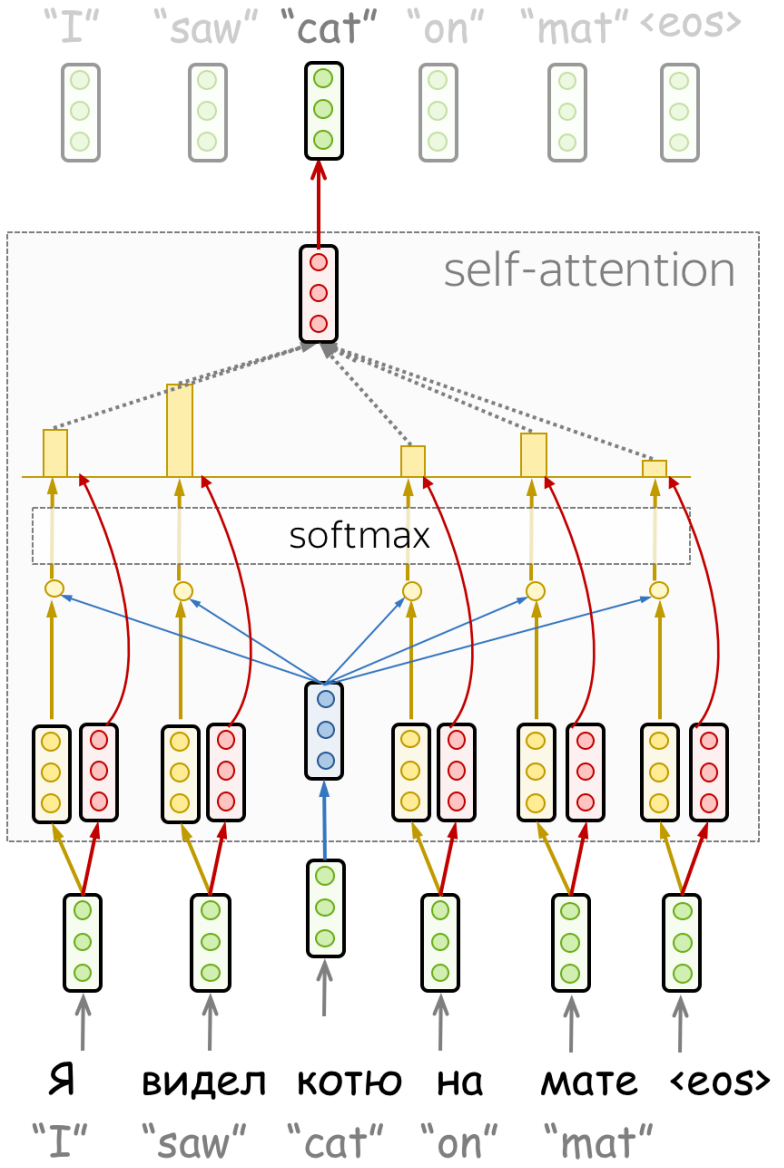
Key: vector **at** which the query looks to compute weights

"Hi, I have this information – give me a large weight!"



Value: their weighted sum is attention output

"Here's the information I have!"



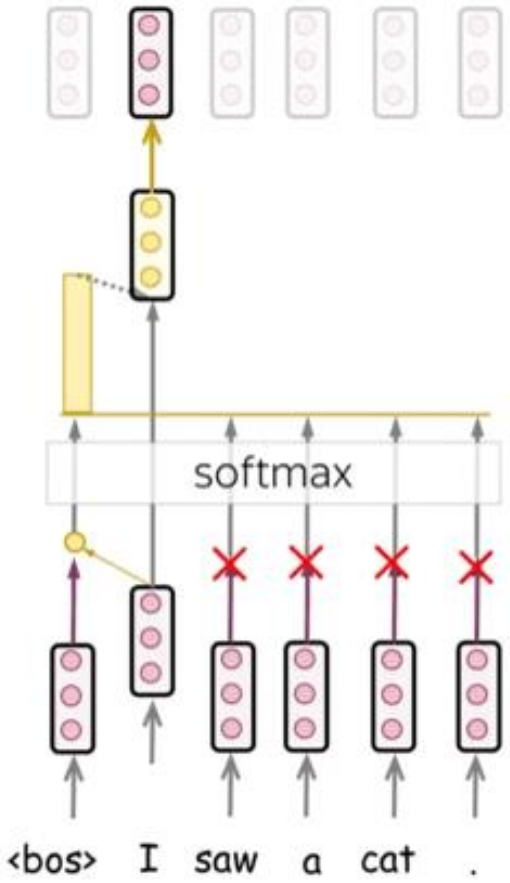
Attention computation

$$Attention(q, k, v) = \overbrace{softmax\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Attention weights}} v$$

from (q) to (k, v)

vector dimensionality of K, V

Masked self-attention

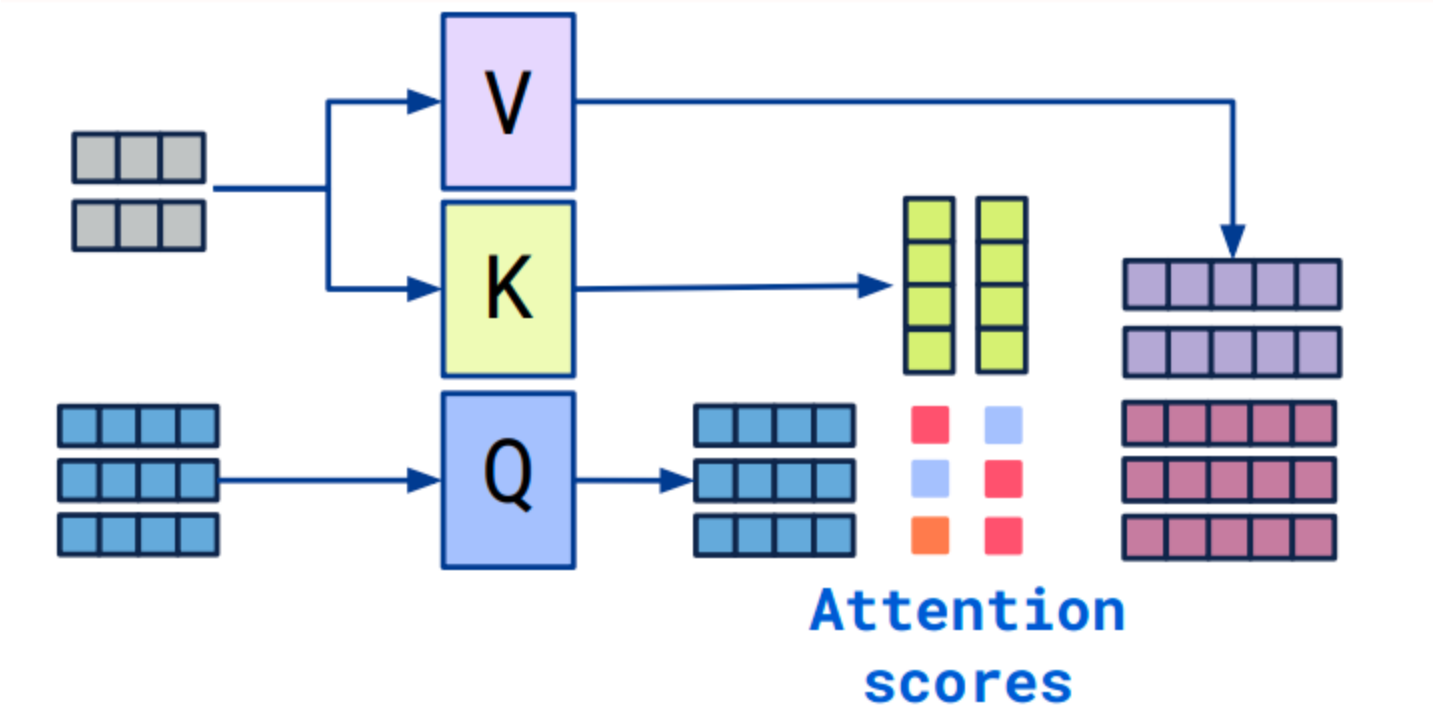


update token representation

gather context

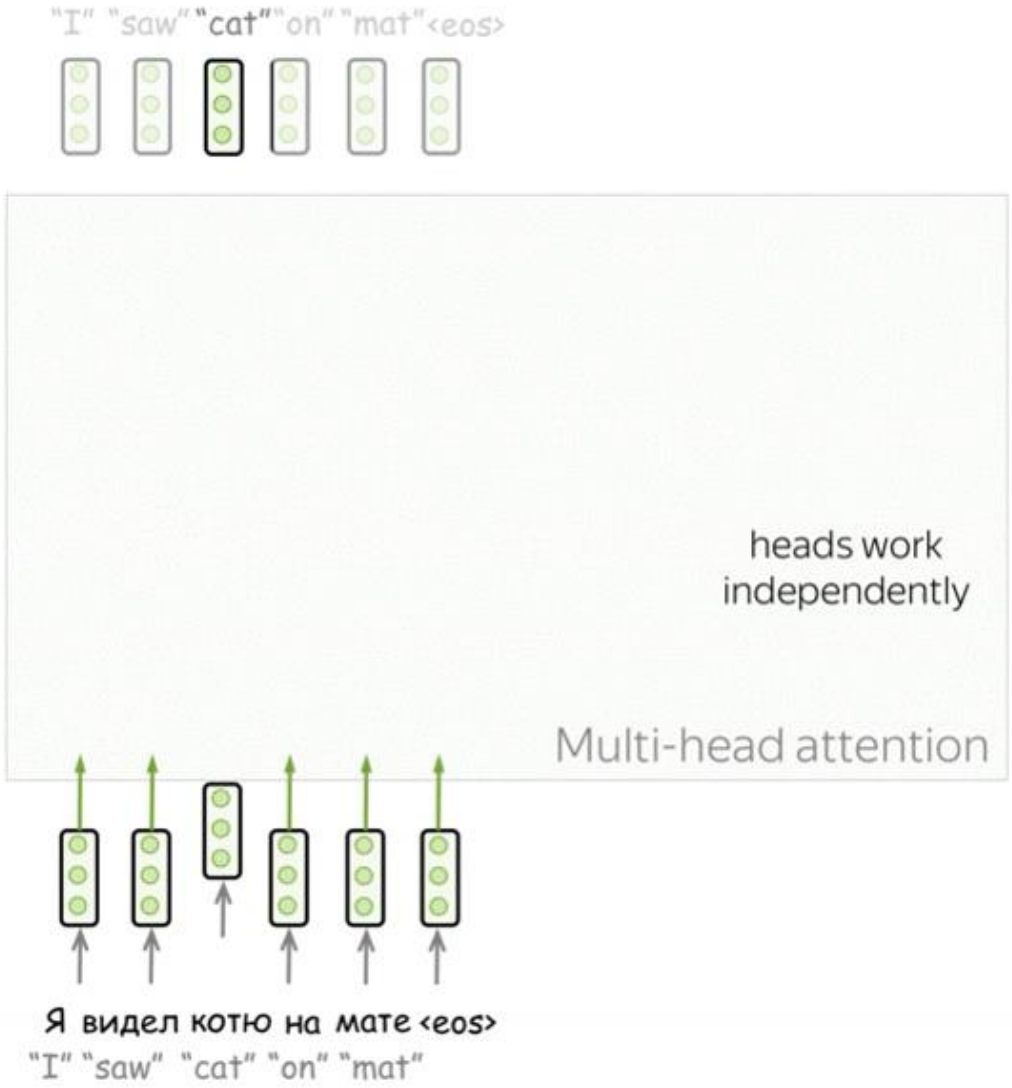
"look" at the previous tokens (future tokens are masked out)

Cross-attention

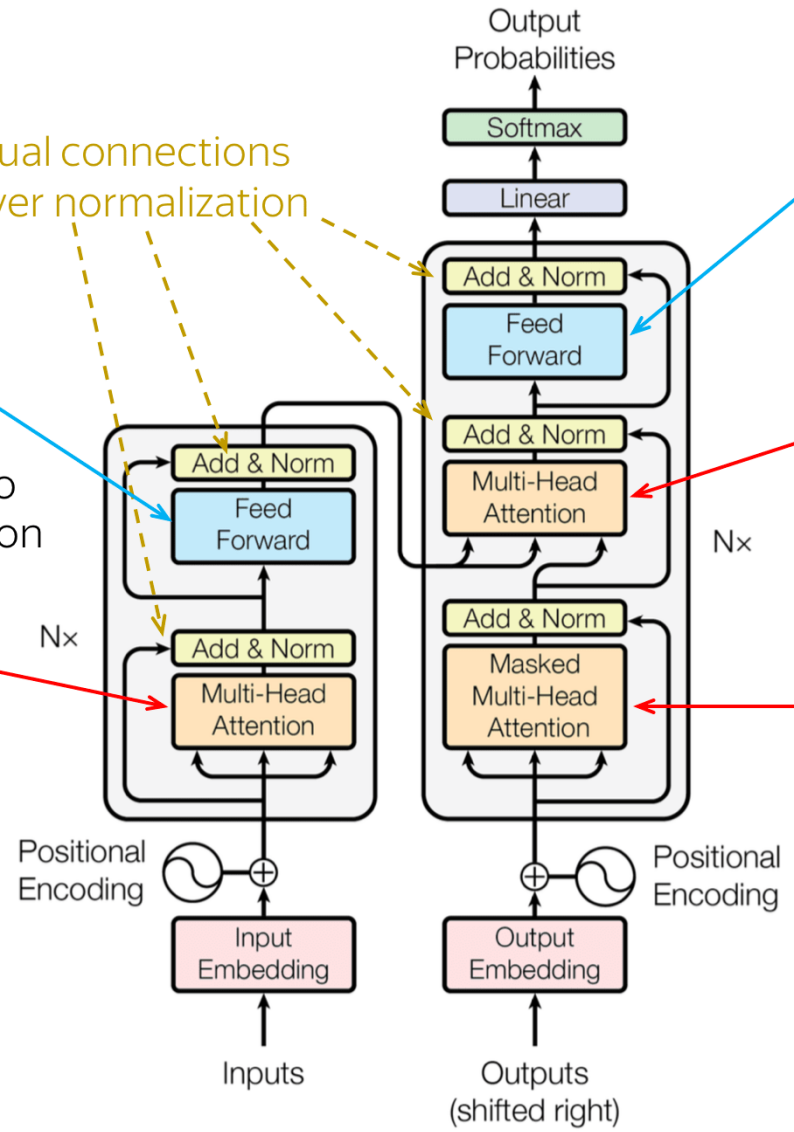


https://vaclavkosar.com/ml/cross-attention-in-transformer-architecture

Multi-head attention



Overall architecture



Residual connections and layer normalization

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Encoder self-attention: tokens look at each other

queries, keys, values are computed from encoder states

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

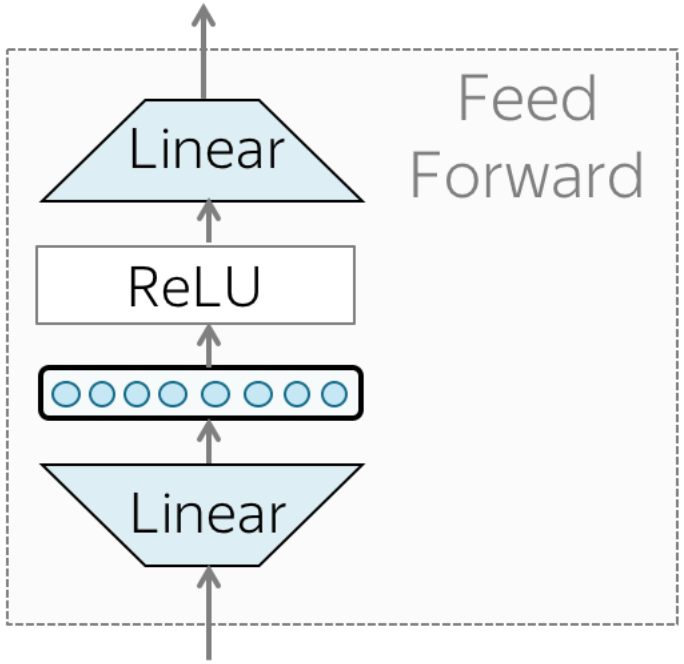
Decoder-encoder attention: target token looks at the source

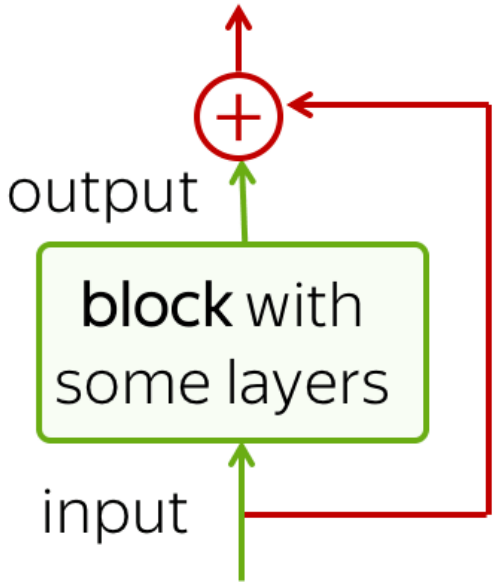
queries – from decoder states; keys and values from encoder states

Decoder self-attention (masked): tokens look at the previous tokens

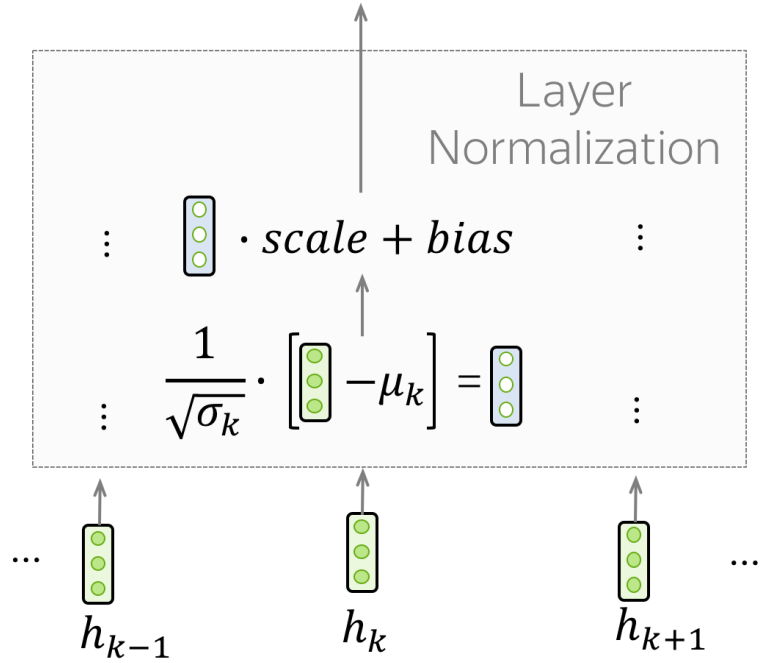
queries, keys, values are computed from decoder states

Blocks





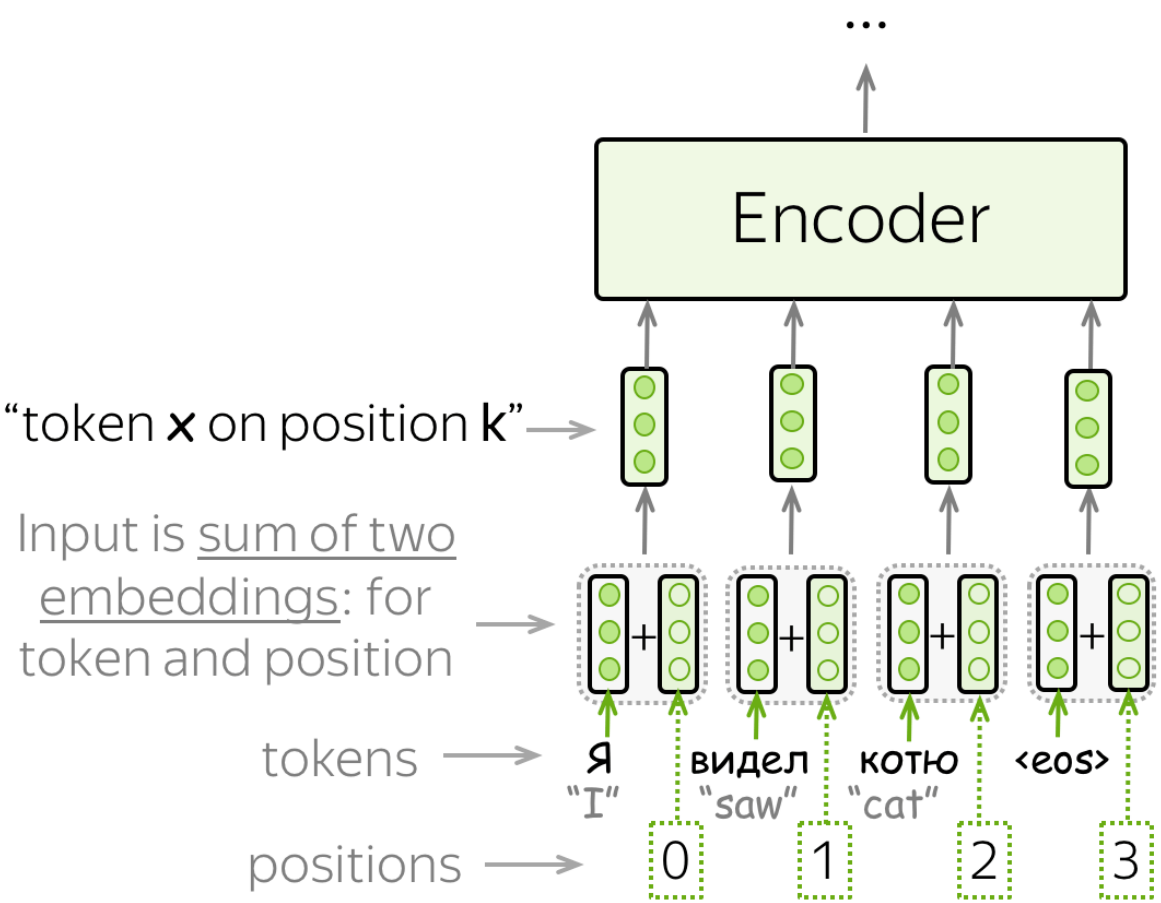
Residual connection:
add a block's input to
its output

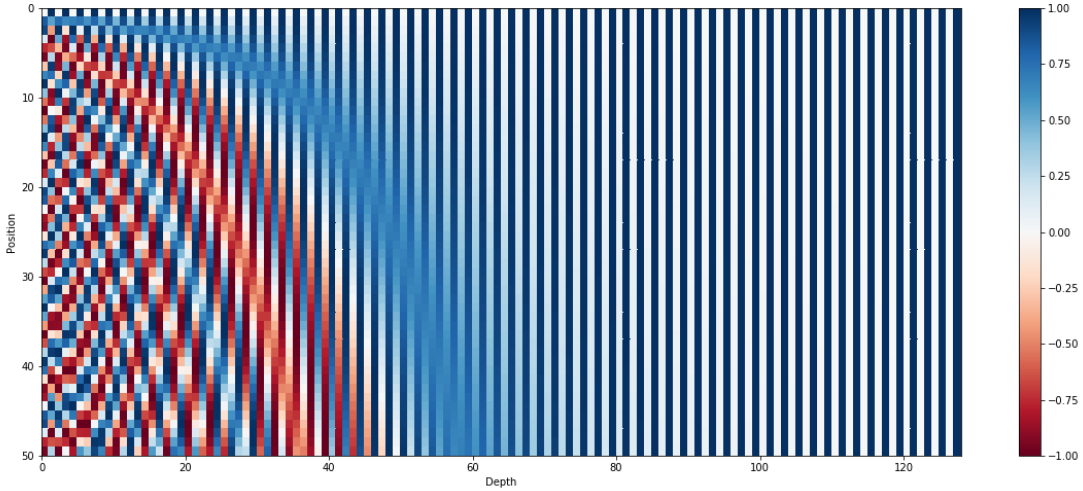


Permutations

- (encoder)
- Shuffling order of input tokens shuffles representations!
- Attention is permutation “invariant” to order

Position encoding / embedding



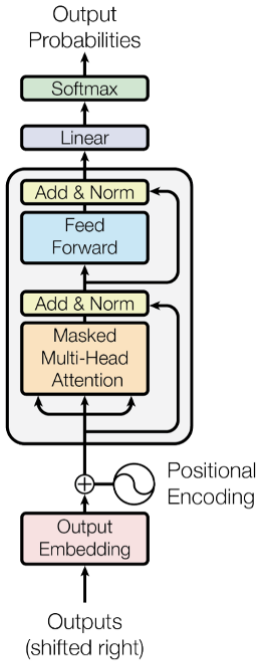


Practical Transformers: Sequence length

- Batch
- Padding
- Masking

Decoder-only GPT

[sat_]

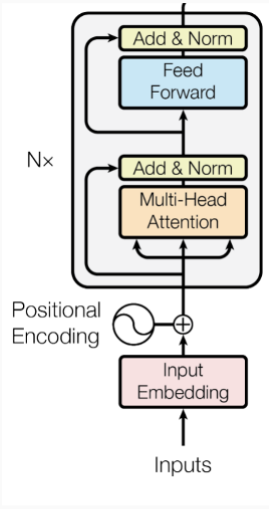


[START] [The_] [cat_]

Transformer image source: "Attention Is All You Need" paper

Encoder-only BERT

[*] [*] **[sat_]** [*] **[the_]** [*]



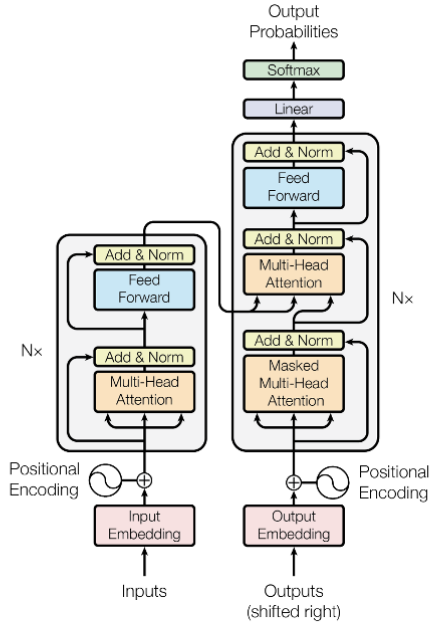
[The_] [cat_] **[MASK]** [on_] **[MASK]** [mat_]

Enc-Dec T5

Das ist gut.

A storm in Attala caused 6 victims.

This is not toxic.



Translate EN-DE: This is good.

Summarize: state authorities dispatched…

Is this toxic: You look beautiful today!

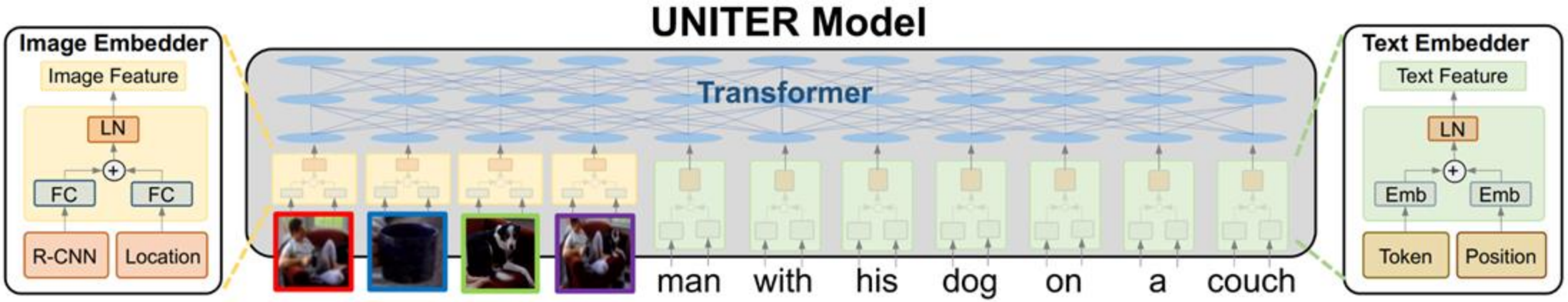
Questions?

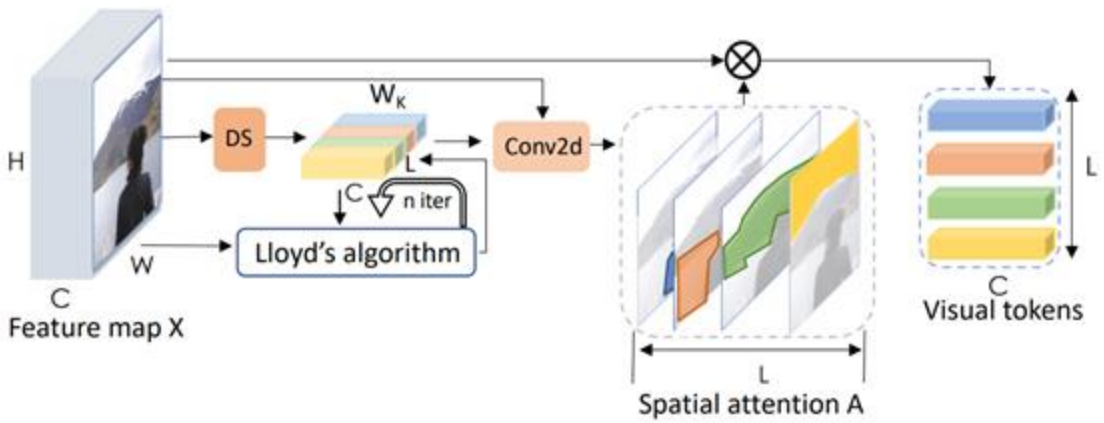
Vision Transformers

Vision Transformers

- Pixels as a sequence?
- Problem?
- So, what can we do?

UNITER





https://arxiv.org/abs/2010.11929

ViT: 16x16 patches is “all you need”!

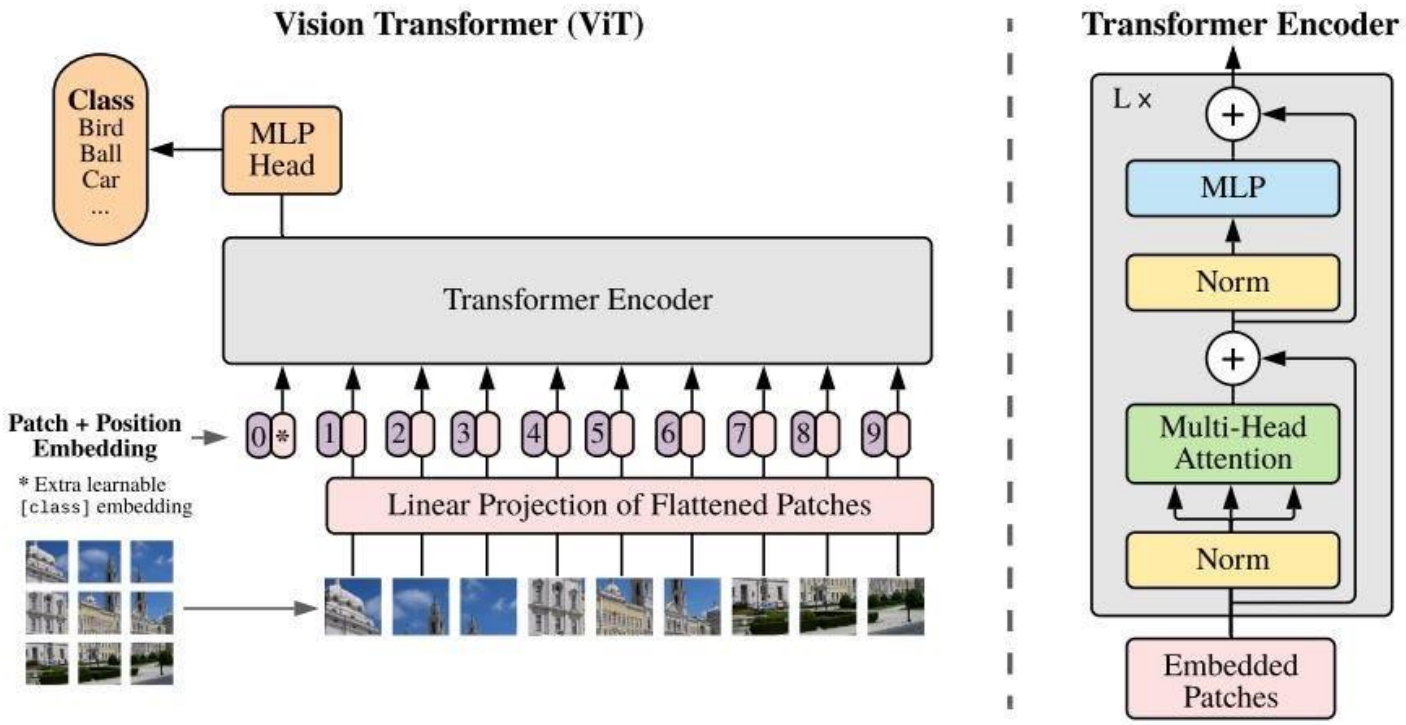


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable “classification token” to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Position encoding?

- 1d?
- 2d?
- none?

D.4 Positional Embedding

We ran ablations on different ways of encoding spatial information using positional embedding. We tried the following cases:

- Providing no positional information: Considering the inputs as a *bag of patches*.
- 1-dimensional positional embedding: Considering the inputs as a sequence of patches in the raster order (default across all other experiments in this paper).
- 2-dimensional positional embedding: Considering the inputs as a grid of patches in two dimensions. In this case, two sets of embeddings are learned, each for one of the axes, X-embedding, and Y-embedding, each with size $D/2$. Then, based on the coordinate on the path in the input, we concatenate the X and Y embedding to get the final positional embedding for that patch.
- Relative positional embeddings: Considering the relative distance between patches to encode the spatial information as instead of their absolute position. To do so, we use 1-dimensional Relative Attention, in which we define the relative distance all possible pairs of patches. Thus, for every given pair (one as query, and the other as key/value in the attention mechanism), we have an offset $p_q - p_k$, where each offset is associated with an embedding. Then, we simply run extra attention, where we use the original query (the content of query), but use relative positional embeddings as keys. We then use the logits from the relative attention as a bias term and add it to the logits of the main attention (content-based attention) before applying the softmax.

Torch hub!

VisionTransformer

The VisionTransformer model is based on the An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale paper.

Model builders

The following model builders can be used to instantiate a VisionTransformer model, with or without pre-trained weights. All the model builders internally rely on the `torchvision.models.vision_transformer.VisionTransformer` base class. Please refer to the source code for more details about this class.

`vit_b_16`(*[, weights, progress])	Constructs a vit_b_16 architecture from An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.
`vit_b_32`(*[, weights, progress])	Constructs a vit_b_32 architecture from An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.
`vit_l_16`(*[, weights, progress])	Constructs a vit_l_16 architecture from An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.
`vit_l_32`(*[, weights, progress])	Constructs a vit_l_32 architecture from An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.
`vit_h_14`(*[, weights, progress])	Constructs a vit_h_14 architecture from An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.

Dino, some beautiful properties!

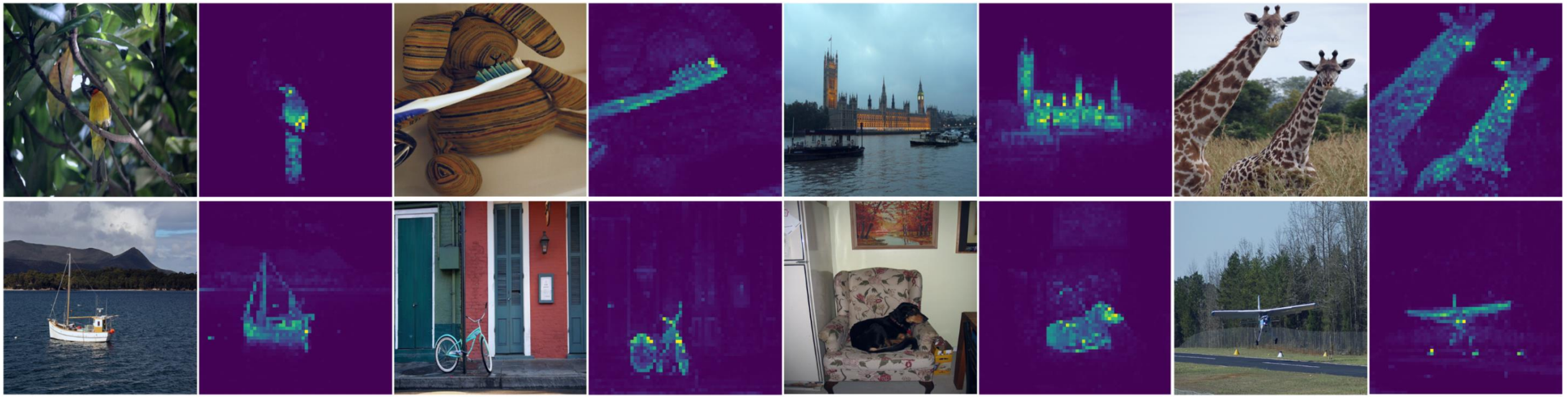


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

https://arxiv.org/abs/2104.14294

Dino, some beautiful properties!

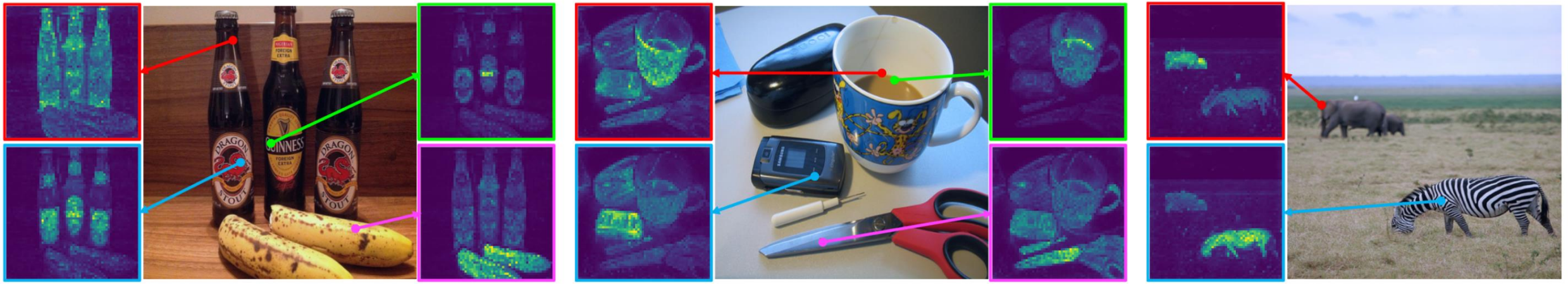


Figure 8: **Self-attention for a set of reference points.** We visualize the self-attention module from the last block of a ViT-S/8 trained with DINO. The network is able to separate objects, though it has been trained with no supervision at all.

https://arxiv.org/abs/2104.14294

ViT sizes

Table 2: Model architecture details.

Name	Width	Depth	MLP	Heads	Mio. Param	GFLOPs	
						224^2	384^2
s/28	256	6	1024	8	5.4	0.7	2.0
s/16	256	6	1024	8	5.0	2.2	7.8
S/32	384	12	1536	6	22	2.3	6.9
Ti/16	192	12	768	3	5.5	2.5	9.5
B/32	768	12	3072	12	87	8.7	26.0
S/16	384	12	1536	6	22	9.2	31.2
B/28	768	12	3072	12	87	11.3	30.5
B/16	768	12	3072	12	86	35.1	111.3
L/16	1024	24	4096	16	303	122.9	382.8
g/14	1408	40	6144	16	1011	533.1	1596.4
G/14	1664	48	8192	16	1843	965.3	2859.9

Questions?

Summary

Transformers

- Attention based models
- Versatile: encoder-only, decoder-only, encoder-decoder
- Scale amazingly with data and model size
  - (all LLMs around us are Transformers!)
- To use: think in terms of tokens and their relationships
  - applicable across language, vision, 3D point clouds, … [everything!]

Thank you!