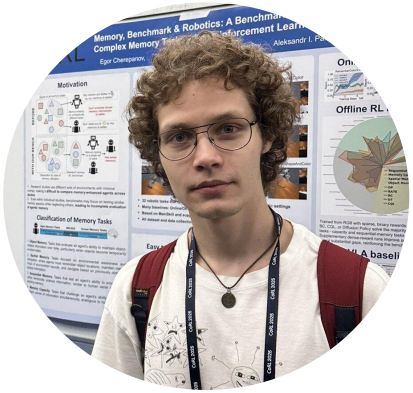
Transformers in Deep Learning

Lecture 1

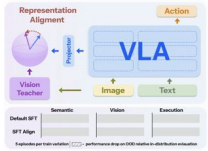
About me:



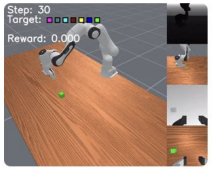
Nikita Kachaev
AIRI, Research Engineer
(ex Tinkoff, VK)

- Multimodal models, RL, Robotics, Foundation models

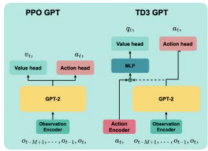
https://tttonyalpha.github.io/



Don't Blind Your VLA: Aligning Visual Representations for OOD Generalization
Nikita Kachaev, Mikhail Kolosov, Daniil Zelezetsky, Alexey K. Kovalev, Aleksandr I. Panov
A conference (TBA), Oral, 2025*
★ *Oral* ★
Project Page · PDF · Code

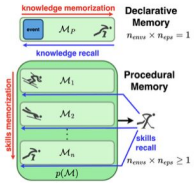


Memory, Benchmark & Robots: A Benchmark for Solving Complex Tasks with Reinforcement Learning
Egor Cherepanov, **Nikita Kachaev**, Alexey K. Kovalev, Aleksandr I. Panov
Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track, Spotlight, 2025
★ *Spotlight* ★
Project Page · PDF · Code



A New Perspective on Transformers in Online Reinforcement Learning for Continuous Control
Nikita Kachaev, Daniil Zelezetsky, Egor Cherepanov, Alexey K. Kovelev, Aleksandr I. Panov
The Thirteenth International Conference on Learning Representations (ICLR), 7th Robot Learning Workshop. Poster, 2025
PDF

Mind and Motion Aligned: A Joint Evaluation IsaacSim Benchmark for Task Planning and Low-Level Policies in Mobile Manipulation
Nikita Kachaev, Andrei Spiridonov, Andrey Gorodetsky, Kirill Muravyev, Nikita Oskolkov, Aditya Narendra, Vlad Shakhuro, Dmitry Makarov, Aleksandr I. Panov, Polina Fedotova, Alexey K. Kovalev
Reports of the RAS: Mathematics, Informatics, and Control Processes, paper, 2025
PDF



Unraveling the Complexity of Memory in RL Agents: an Approach for Classification and Evaluation
Egor Cherepanov, **Nikita Kachaev**, Artem Zholus, Alexey K. Kovalev, Aleksandr I. Panov
Conference on Robot Learnin (CORL), RemembeRL Workshop, Poster, 2025
PDF

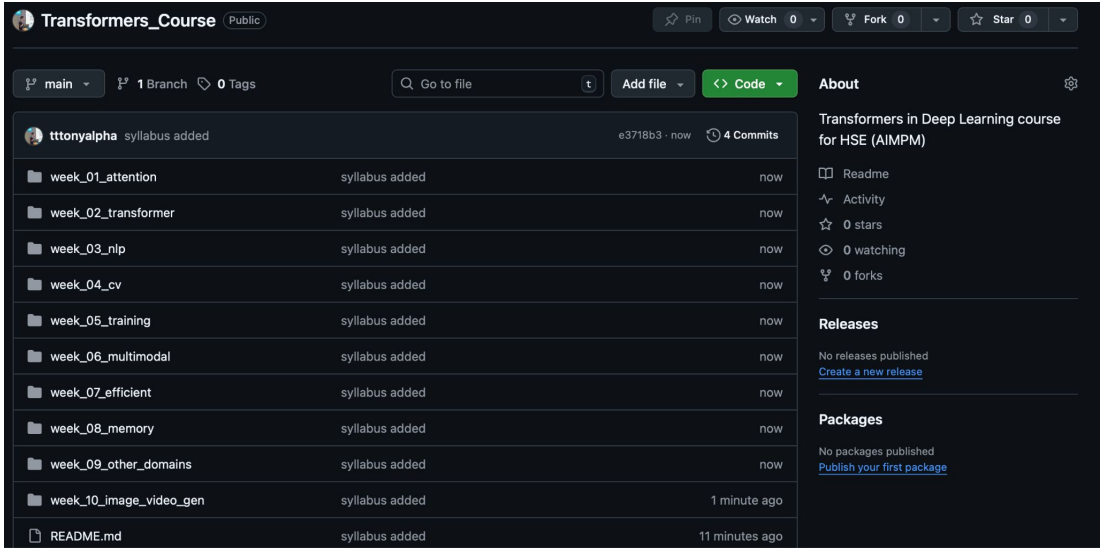
You will learn:

- How Transformers work
- Applications beyond NLP
- Modern Transformers best practices

Course:

Final grade =

0.7 × Homework + 0.3 × Exam



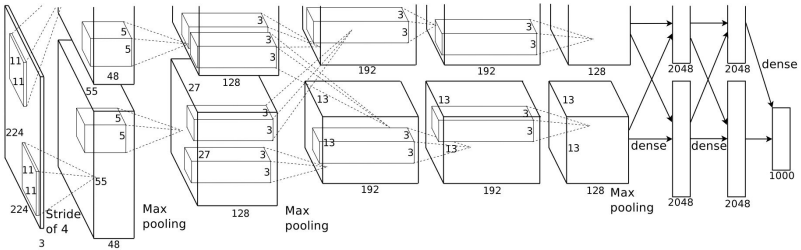
Syllabus

- **week01** Attention Intro
  - Lecture: Deep learning recap, embeddings, rnn bottelneck, from reccurence to attention
  - Seminar: Torch basics recap, Implementing Bahdanau attention
  - Homework 1
- **week02** Transformers basics
  - Lecture: QKW attention, FFN, residual connection, layer norm, positional encoding, dropout, tokenization, sampling.
  - Seminar: Transformer implementation from scratch
  - Homework 2
- **week03** Transformers in NLP
  - Lecture: Transfer learning, BERT (BART, RoBerta and etc.), GPT, T5.
  - Seminar: HF Transformers
  - Homework 3
- **week04** Transformers in CV
  - Lecture: DETR, ViT, Swin.
  - Seminar: ViT deep dive
  - Homework 3

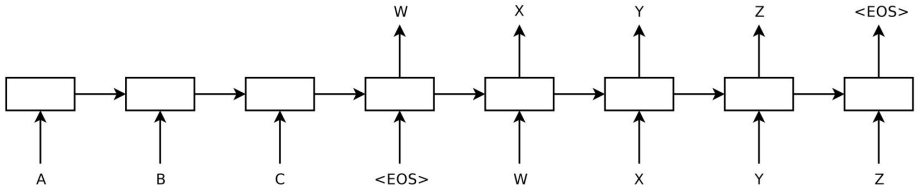
https://github.com/tttonyalpha/Transformers_Course

Deep learning before transformers:

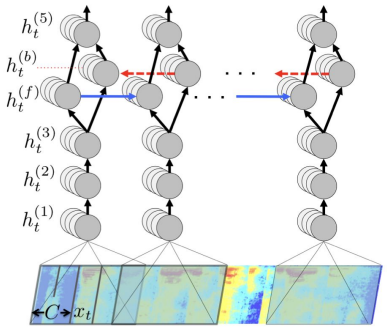
CV (AlexNet)



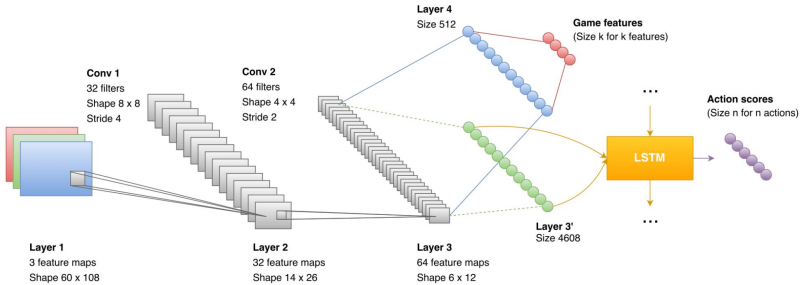
NLP (RNNs)



Speech (DeepSpeach)



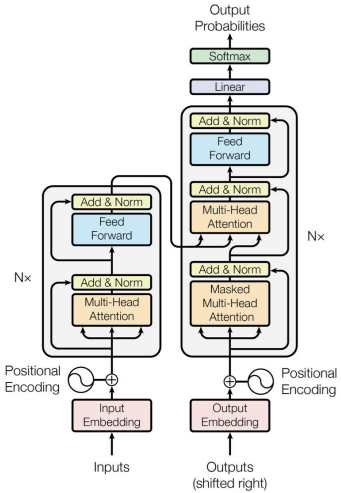
Robotics/RL (DRQN)



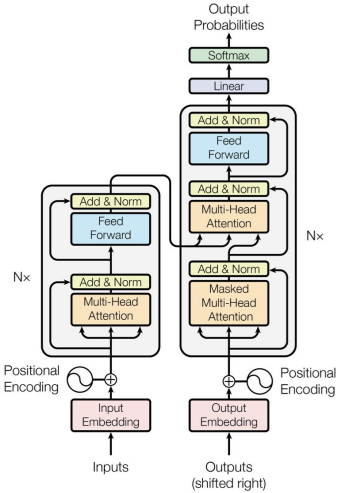
https://arxiv.org/pdf/1409.3215, https://arxiv.org/abs/1609.05521, https://arxiv.org/pdf/1412.5567

Deep learning after transformers:

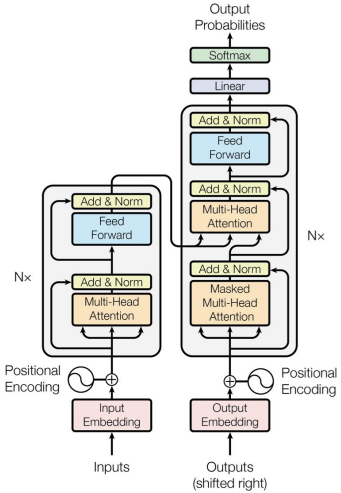
CV



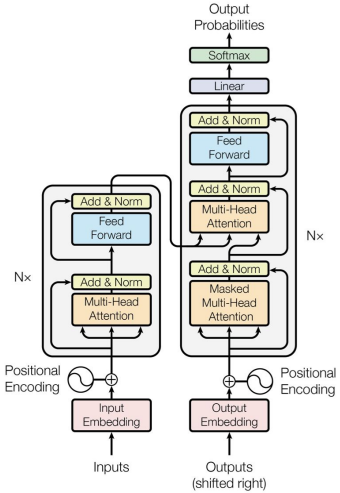
NLP



Speech

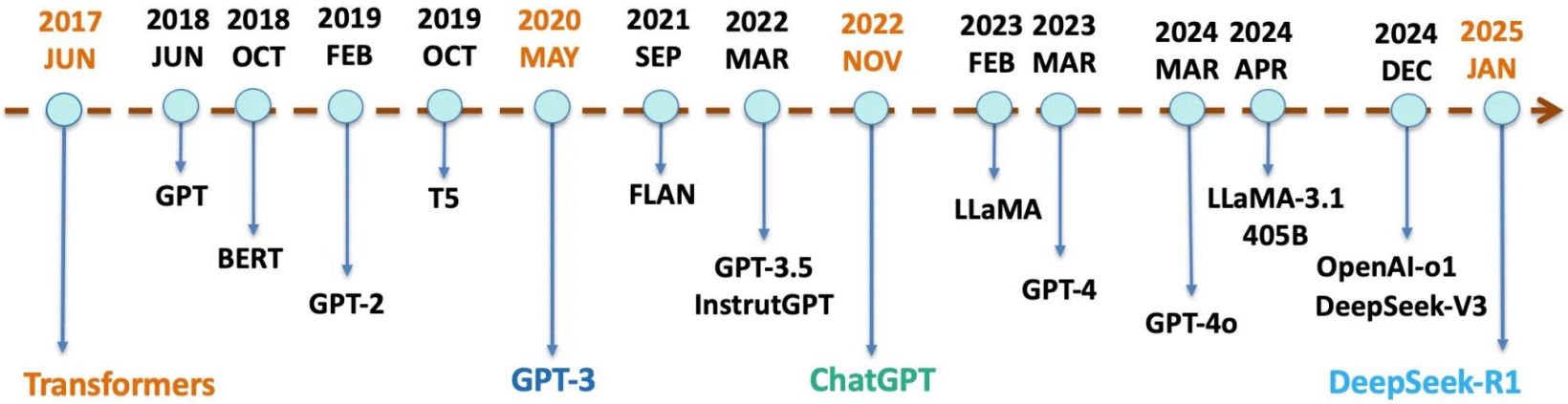


Robotics/RL



https://arxiv.org/abs/1706.03762

Transformers timeline:



https://medium.com/@lmpo/a-brief-history-of-lmms-from-transformers-2017-to-deepseek-r1-2025-dae75dd3f59a

Transformers: why do people use them so much?

Andrej Karpathy ✓
@karpathy

The Transformer is a magnificient neural network architecture because it is a general-purpose differentiable computer. It is simultaneously:
1) expressive (in the forward pass)
2) optimizable (via backpropagation+gradient descent)
3) efficient (high parallelism compute graph)

9:54 PM · Oct 19, 2022

Andrej Karpathy ✓ @karpathy · Oct 19, 2022
(1) because its message-passing-like architecture is general (i.e. completeness) and powerful (i.e. efficiency), able to cover many real-world algorithms and in a small number of compute steps; an an empirical finding.

4 8 292

Andrej Karpathy ✓ @karpathy · Oct 19, 2022
(2) because of residual connections, layer normalizations, and softmax attention. Absence of any flat tails. Residual connections support a kind of ability to learn short algorithms (think low LOC) fast and first, then gradually extend them longer during training.

10 10 257

https://x.com/karpathy/status/1582807367988654081?s=20

Andrej Karpathy ✓ @karpathy · Oct 19, 2022
(3) because the compute graph is shallow and wide, mapping significantly better to our high-parallelism compute architectures (think GPUs). An earlier attempt that understood the significance and optimized for this property was the Neural GPU paper (arxiv.org/abs/1511.08228)

2 10 258

Andrej Karpathy ✓ @karpathy · Oct 19, 2022
Its success lies in a single architecture that simultaneously satisfies all of these properties. The original Attention Is All You Need paper is a bit haphazard and undersells the magnitude of these insights, their history and motivations. But there's a lot going on :)

10 9 317

Andrej Karpathy ✓ @karpathy · Oct 19, 2022
So I probably would have called the paper something like "Transformer: A general-purpose, efficient, optimizable computer" and presented it alongside the Neural Turing Machine, NeuralGPU and friends, then applied it to translation as an example. Something like that, but ok :)

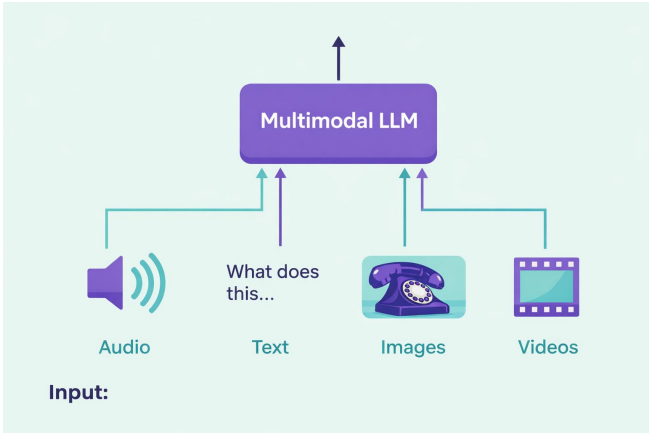
20 25 443

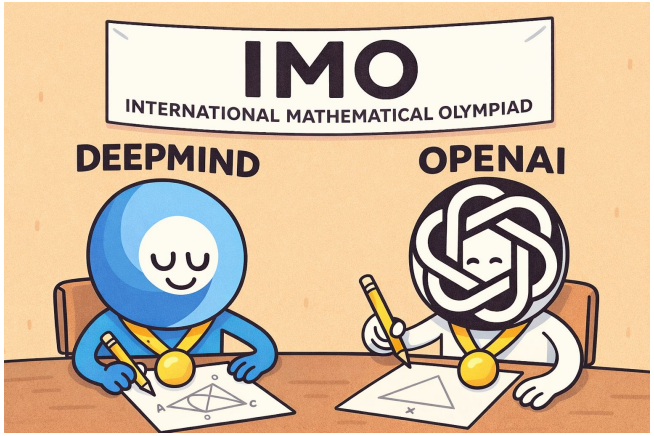
Andrej Karpathy ✓ @karpathy · Oct 19, 2022
A few people have (correctly) pointed out the hindsight here, which is fair. I don't suspect the authors would have known that 5 years later that architecture will have taken over most of AI ~unchanged, except for a re-shuffling of layernorms. Calls for a followup paper :)

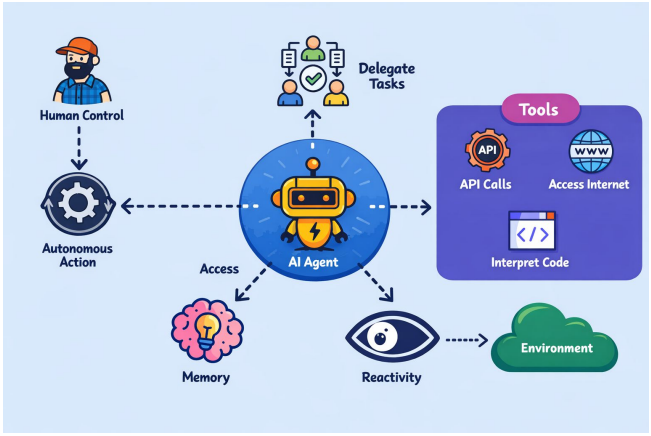
12 16 375

Where we are (2026)

- **Multimodality** (text + image + audio + video).
- **Long-context working memory.**
- **Reasoning improvements** are increasingly driven by test-time scaling.
- **Agentic systems.**
- **Alignment & safety** moved “beyond classic RLHF”.



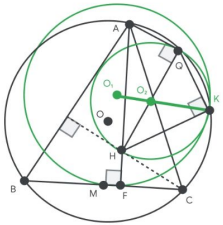




Alpha Geometry

IMO 2015 P3

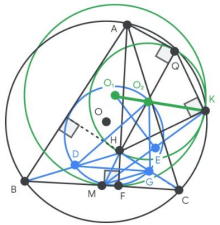
Let ABC be an acute triangle. Let (O) be its circumcircle, H its orthocenter, and F the foot of the altitude from A. Let M be the midpoint of BC. Let Q be the point on (O) such that QH ⊥ QA and let K be the point on (O) such that KH ⊥ KQ. Prove that the circumcircles (O_1) and (O_2) of triangles FKM and KQH are tangent to each other.



AlphaGeometry

Solution

```
[...]
Construct D: midpoint BH [a]
[a], O₂ midpoint HQ ⇒ BQ ∥ O₂D [20]
[...]
Construct G: midpoint HC [b]
∠GMD = ∠GO₂D ⇒ M O₂ G D cyclic [26]
[...]
[a],[b] ⇒ BC ∥ DG [30]
[...]
Construct E: midpoint MK [c]
[c] ⇒ ∠KFC = ∠KO₁E [104]
[...]
∠FKO₁ = ∠FKO₂ ⇒ KO₁ ∥ KO₂ [109]
[109] ⇒ O₁O₂K collinear ⇒ (O₁)(O₂) tangent
```



https://deepmind.google/blog/alphageometry-an-olympiad-level-ai-system-for-geometry/

Alpha Evolve

Analysis

$$\max_{-1/2\le t\le 1/2}\int_{\mathbb{R}} f_i(t-x)f_i(x)\,dx \ge C_i\left(\int_{-1/4}^{1/4} f_i(x)\,dx\right)^2$$

1.5098 → **1.5053**

$$\|f*f\|_2^2 \le C'\|f*f\|_1\|f*f\|_\infty$$

0.8892 → **0.8962**

$$\max_{-1/2\le t\le 1/2}\left|\int_{\mathbb{R}} f(t-x)f(x)\,dx\right| \ge C''\left(\int_{-1/4}^{1/4} f(x)\,dx\right)^2$$

1.4581 → **1.4557**

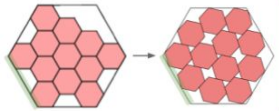
$$A(f)A(\hat{f}) \ge C'''$$

0.3523 → **0.3521**

Geometry

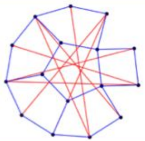
Hexagon outer edge

4.000 → **3.942**



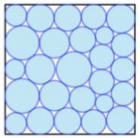
Max distance/min distance

12.890 → **12.889**

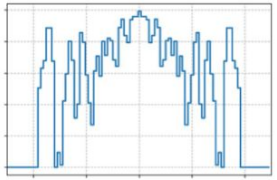


Sum of radii

2.6340 → **2.6358**



Combinatorics



$$\sup_{x\in[-2,2]}\int_{-1}^{1} f(t)g(x+t)\,dt \ge C$$

0.380926 → **0.380924**

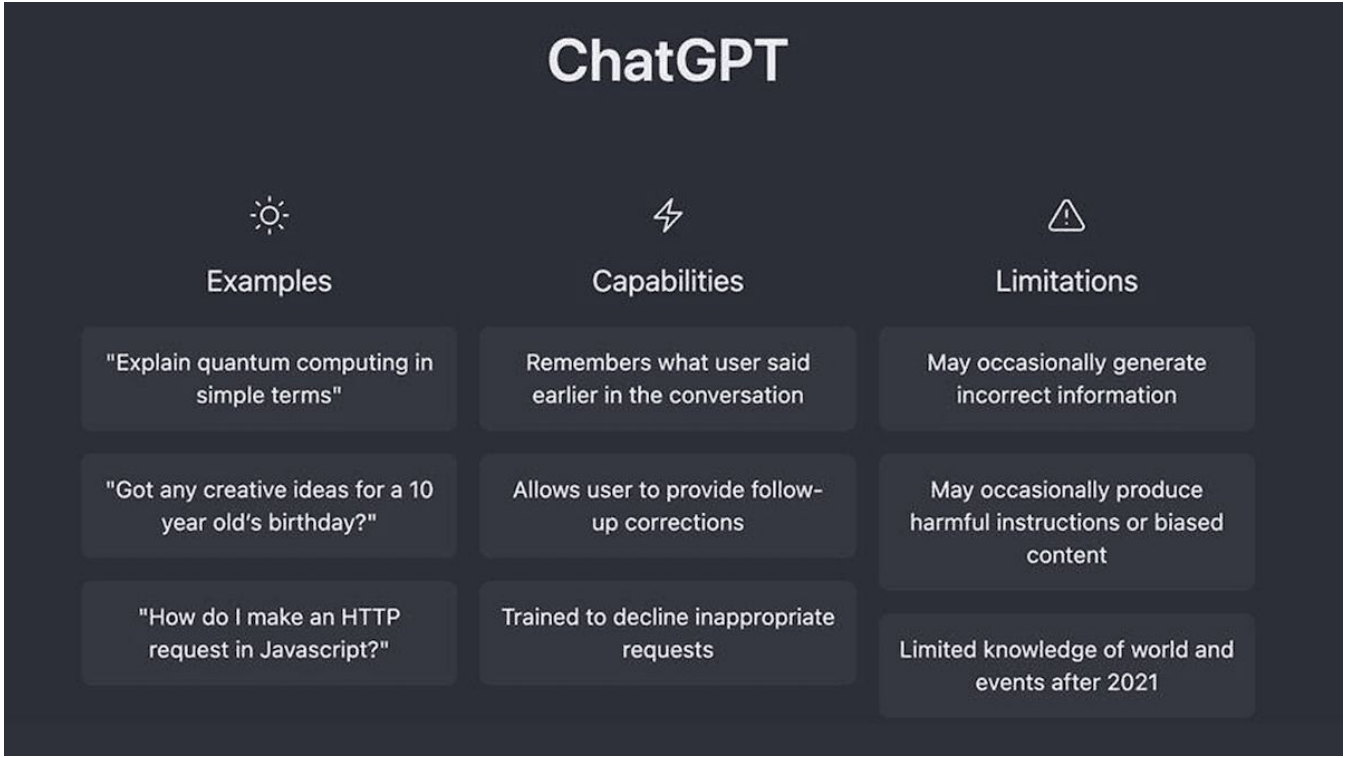
$$|A+B| \ll |A|$$

$$|A-B| \gg |A|^C$$

1.1446 → **1.1584**

https://arxiv.org/abs/2506.13131

Chat GPT


ChatGPT
Examples
"Explain quantum computing in simple terms"
"Got any creative ideas for a 10 year old's birthday?"
"How do I make an HTTP request in Javascript?"
Capabilities
Remembers what user said earlier in the conversation
Allows user to provide follow-up corrections
Trained to decline inappropriate requests
Limitations
May occasionally generate incorrect information
May occasionally produce harmful instructions or biased content
Limited knowledge of world and events after 2021

NVIDIA Cosmos-Reasong1

Embodied Reasoning Benchmark



Based on the ego vehicle's actions in the video, what is the most likely immediate action it will take next?

A: turn right, B: turn left, C: change to right lane, D: change to left lane

Wait, maybe the question is a trick. The video shows the road markings are double yellow lines, which means no passing, so changing lanes is prohibited. Therefore, the driver cannot change lanes, so options C and D are impossible. The only remaining options are turning right or left. But since the video doesn't show any turns, and the driver is maintaining their path, perhaps the answer is that none of the options are correct, but since we have to choose, I have no choice but to select none. Therefore, the answer is none.

https://arxiv.org/abs/2503.15558

Generative models

Kandinsky 5.0: A Family of Foundation Models for Image and Video Generation

Kandinsky Lab*

* A detailed list of the contributors can be found in the end of this paper.

https://arxiv.org/abs/2511.14993

The Future

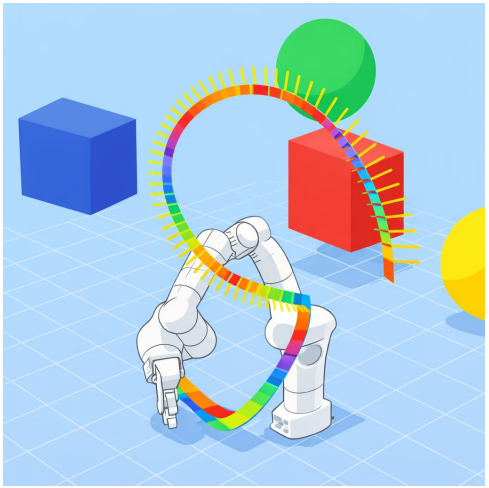
- **Reliable memory (not just long context).** Robust retrieval, grounded citations.
- **Always-on Multimodal Assistants.** Real-time voice + vision assistants for work, education, accessibility, and robotics.
- **Verification and truthfulness at scale.** Built-in self-checking, external validators. Knowing when it doesn't know, and escalating correctly.
- **Grounding to the physical world.** More robust world models.
- **Scientific & engineering acceleration.**

Key idea: embed everything

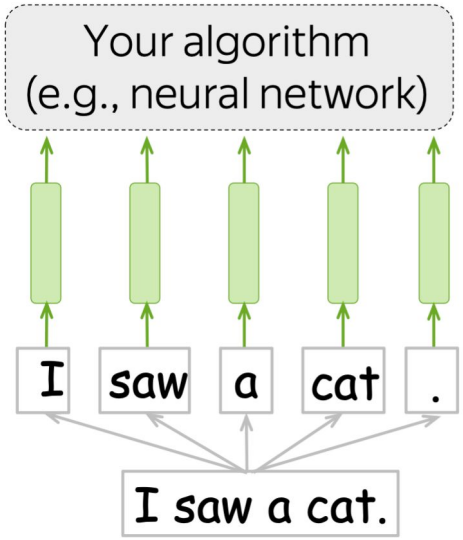
Text (token)

Image (patch)

Robot trajectory (timestep)



Word Embeddings



Any algorithm for solving a task

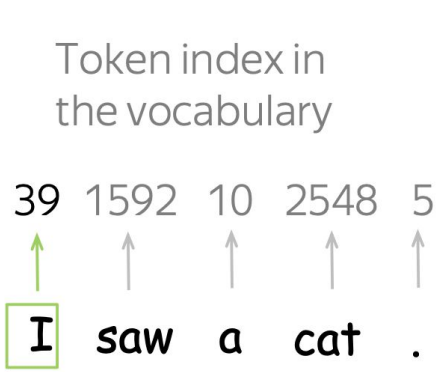
Word representation - vector
(input for your model/algorithm)

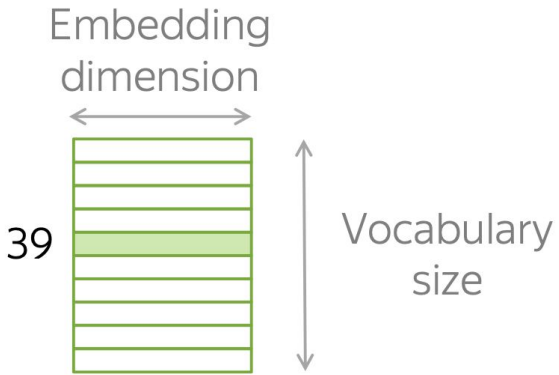
Sequence of tokens

Text (your input)

https://lena-voita.github.io/nlp_course.html

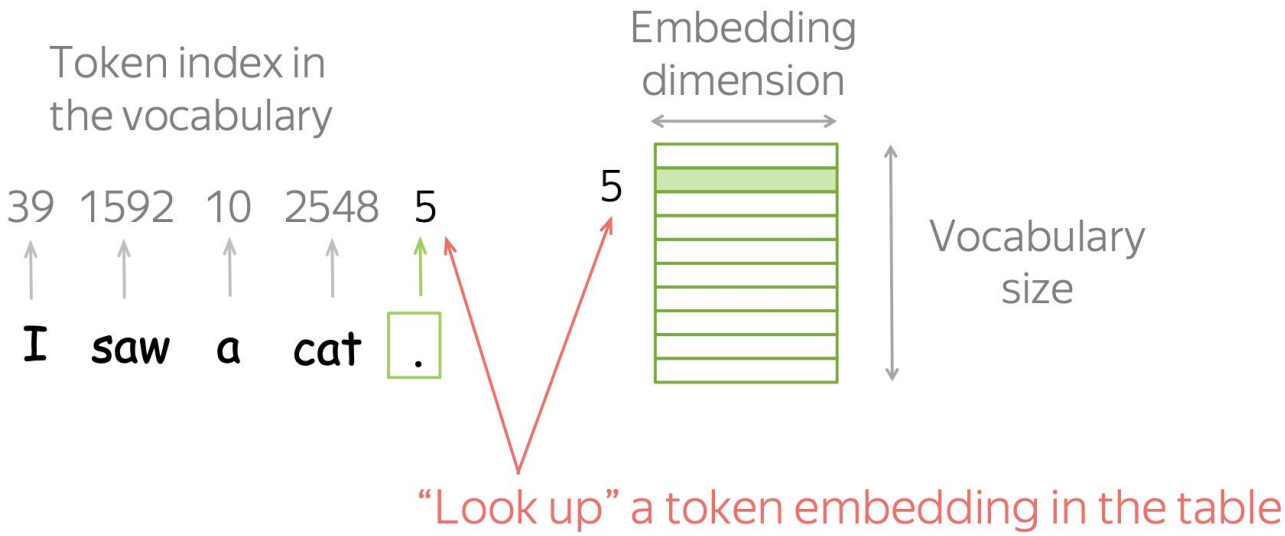
Word Embeddings





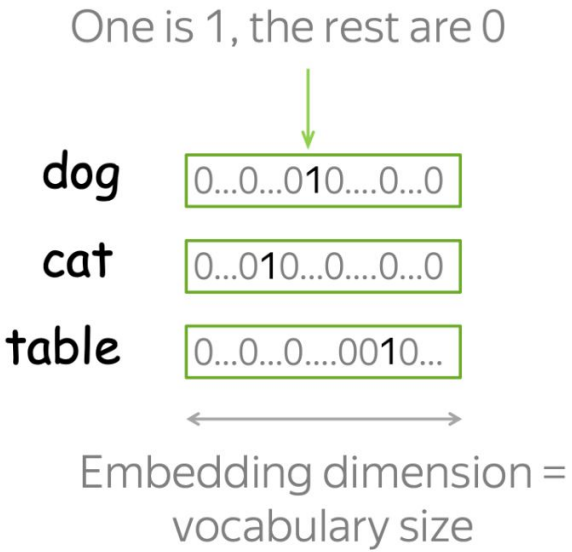
https://lena-voita.github.io/nlp_course.html

Word Embeddings



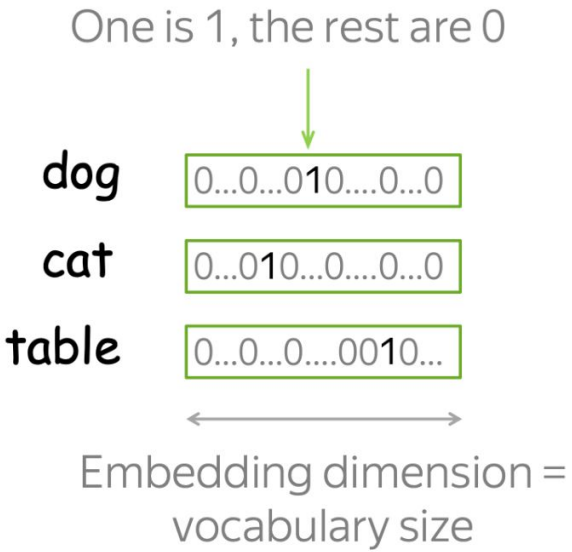
https://lena-voita.github.io/nlp_course.html

One-Hot Embeddings



https://lena-voita.github.io/nlp_course.html

One-Hot Embeddings



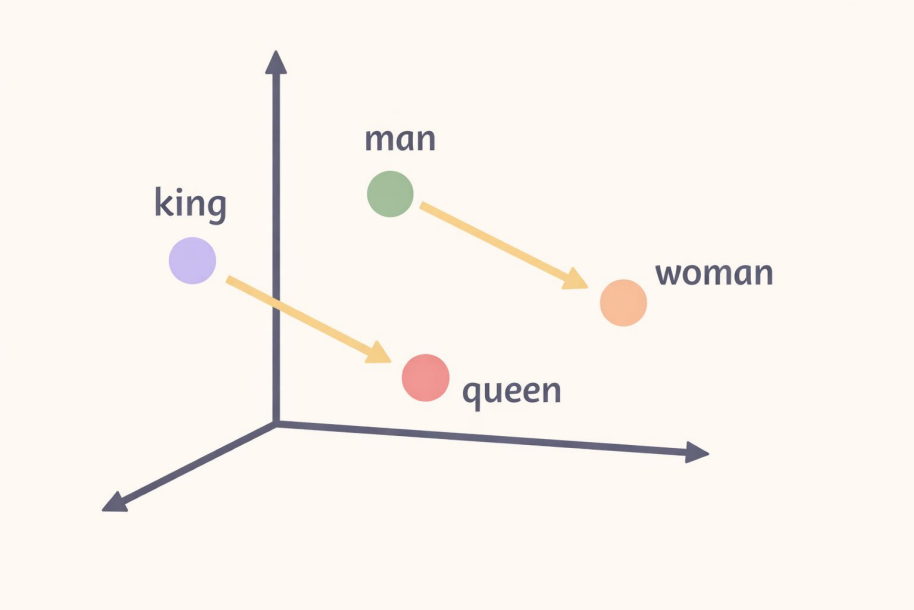
Problems:

- Vector size is too large
- Vectors know nothing about meaning

  e.g., **cat** is as close to **dog** as it is to **table**!

https://lena-voita.github.io/nlp_course.html

Word Embeddings: Semantics relations



Word2Vec Embeddings

> We have to put information about contexts into word vectors.

> How: **Learn** word vectors by teaching them to **predict contexts**.

https://lena-voita.github.io/nlp_course.html

Word2Vec Embeddings

… I saw a cute grey cat playing in the garden …

https://lena-voita.github.io/nlp_course.html

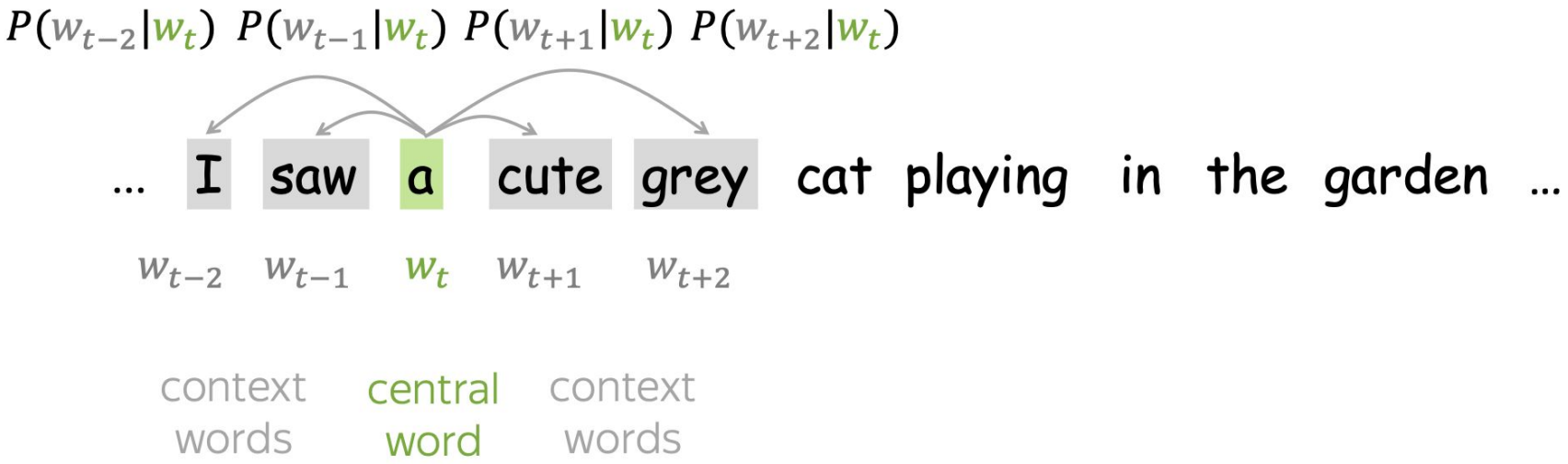
Word2Vec Embeddings

https://lena-voita.github.io/nlp_course.html

Word2Vec Embeddings

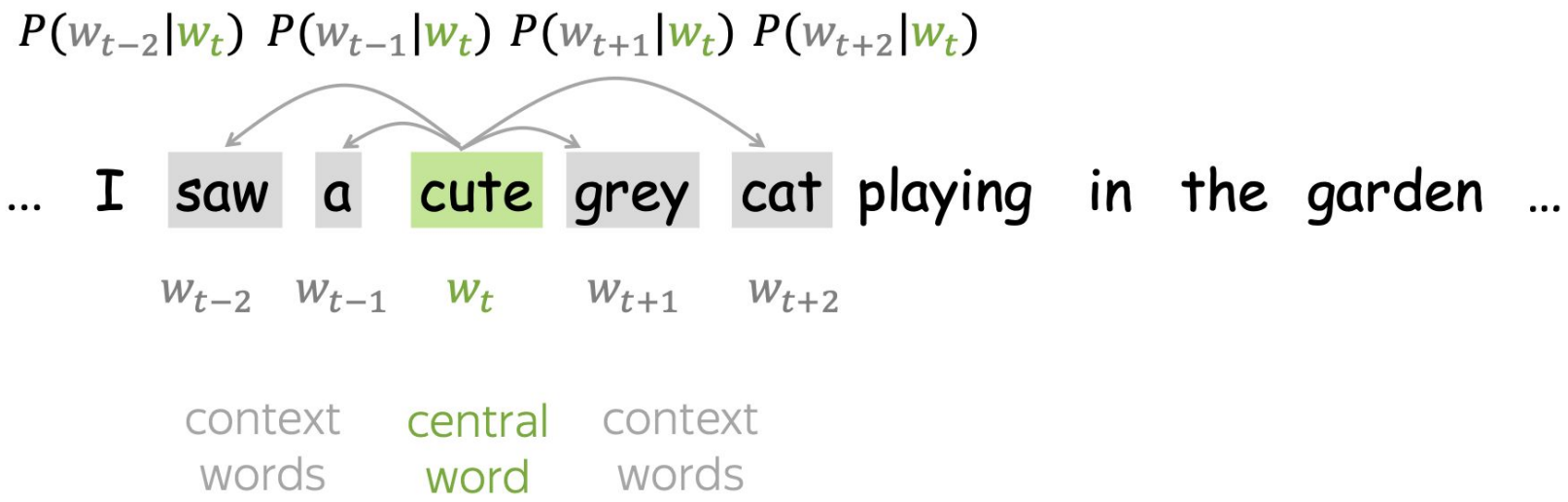
https://lena-voita.github.io/nlp_course.html

Word2Vec Embeddings



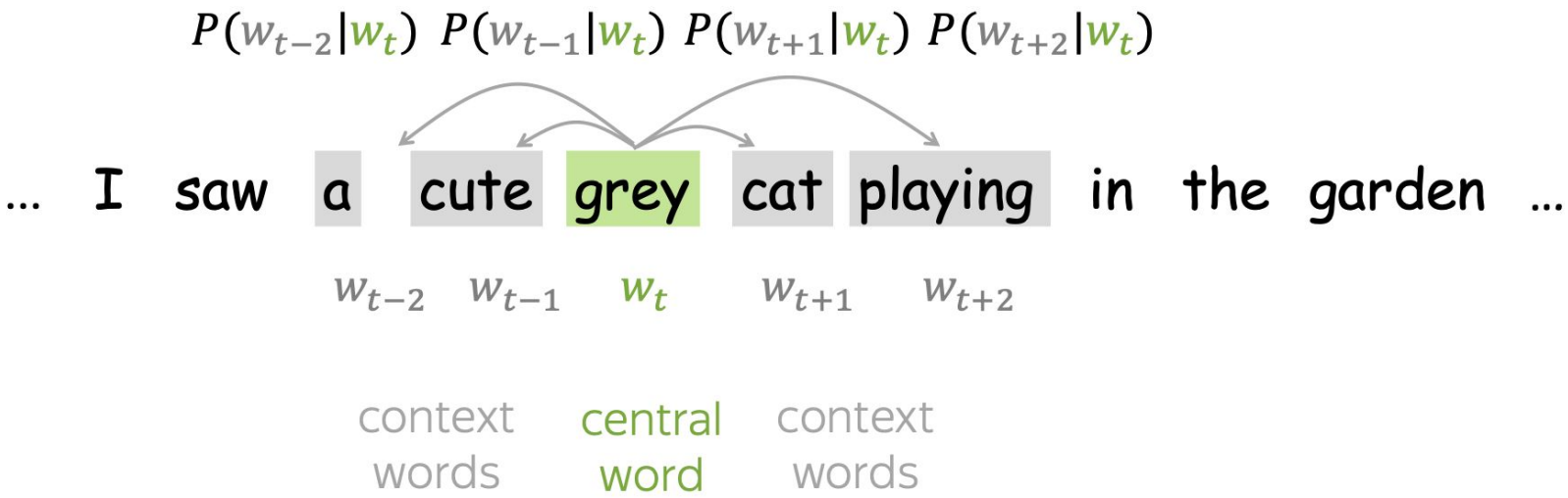
https://lena-voita.github.io/nlp_course.html

Word2Vec Embeddings



https://lena-voita.github.io/nlp_course.html

Word2Vec Embeddings



https://lena-voita.github.io/nlp_course.html

Likelihood of all text corpus

We want our model to think that the training data is "likely"

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m, \\ j \neq 0}} P(w_{t+j} | w_t, \theta)$$

https://lena-voita.github.io/nlp_course.html

Likelihood of all text corpus

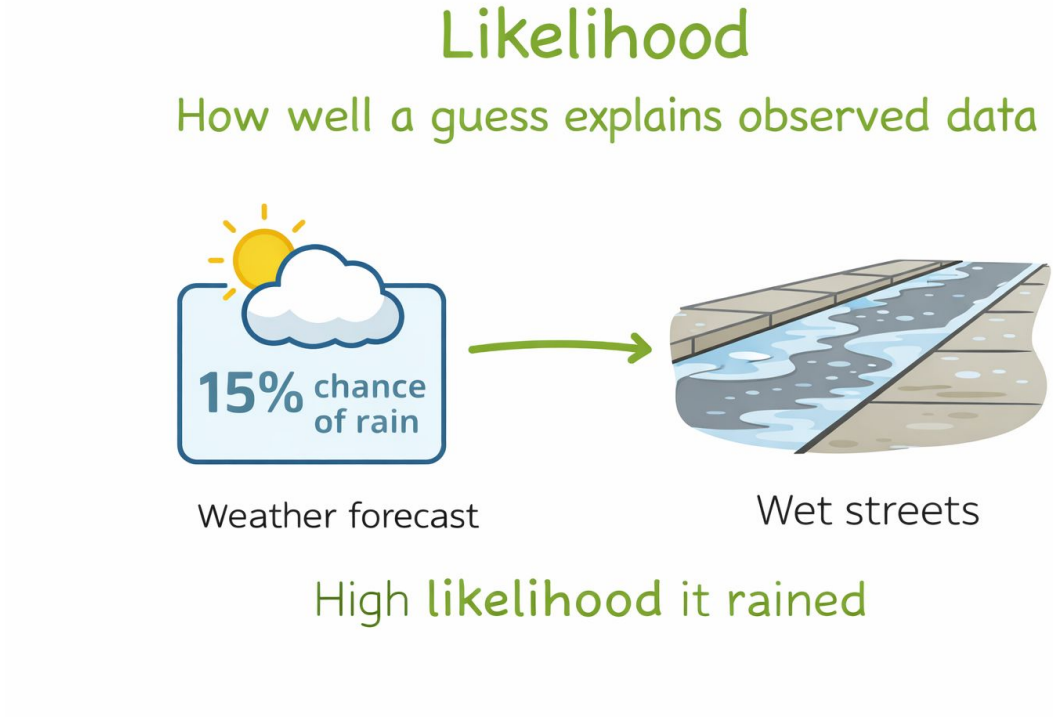
We want our model to think that the training data is "likely"

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m, \\ j \neq 0}} P(w_{t+j} | w_t, \theta)$$

θ are all variables to optimize

https://lena-voita.github.io/nlp_course.html

Likelihood


Likelihood
How well a guess explains observed data
15% chance of rain
Weather forecast
Wet streets
High likelihood it rained

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta)$$

https://lena-voita.github.io/nlp_course.html

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}} \log P(w_{t+j}|w_t,\theta)$$

https://lena-voita.github.io/nlp_course.html

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}} \log P(w_{t+j}|w_t,\theta)$$

go over text

https://lena-voita.github.io/nlp_course.html

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}} \log P(w_{t+j}|w_t,\theta)$$

go over text

with a sliding window

https://lena-voita.github.io/nlp_course.html

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}} \log P(w_{t+j}|w_t,\theta)$$

go over text

with a sliding window

compute probability of the context word given the central

https://lena-voita.github.io/nlp_course.html

Word2Vec Loss

$$\text{Loss} = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}}\log P(w_{t+j}|w_t,\theta)$$

How to compute this?

go over text

with a sliding window

compute probability of the context word given the central

https://lena-voita.github.io/nlp_course.html

Softmax function:

Softmax function $\mathbb{R}^n \rightarrow \mathbb{R}^n$:

$$softmax(\boldsymbol{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)} = p_i$$

- maps arbitrary values x_i to a probability distribution p_i
- "max" because amplifies probability of largest x_i
- "soft" because still assigns some probability to smaller x_i

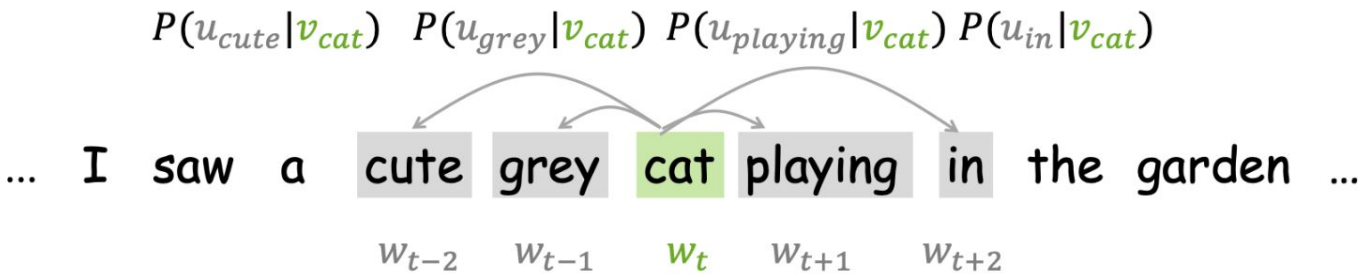
https://lena-voita.github.io/nlp_course.html

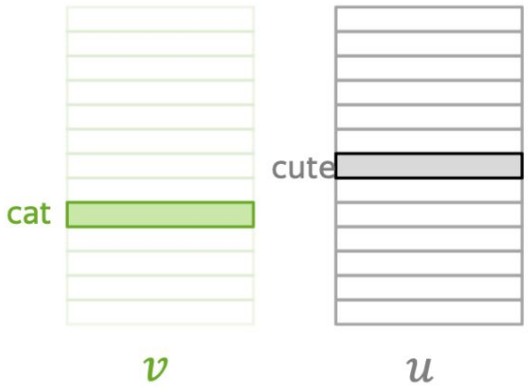
Probability of context word

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

https://lena-voita.github.io/nlp_course.html

One Training step





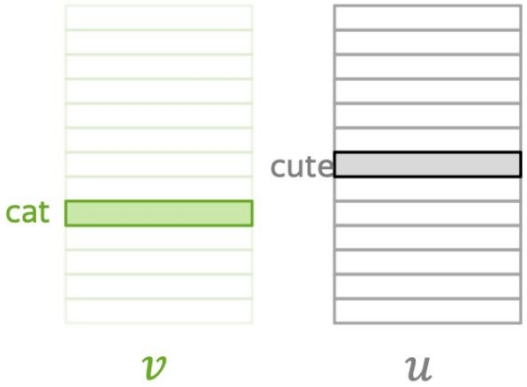
Look at the loss component for this step:

$$-\log P(cute|cat) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_{w \in V} \exp(u_w^T v_{cat})}$$

https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_{w \in V} \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$

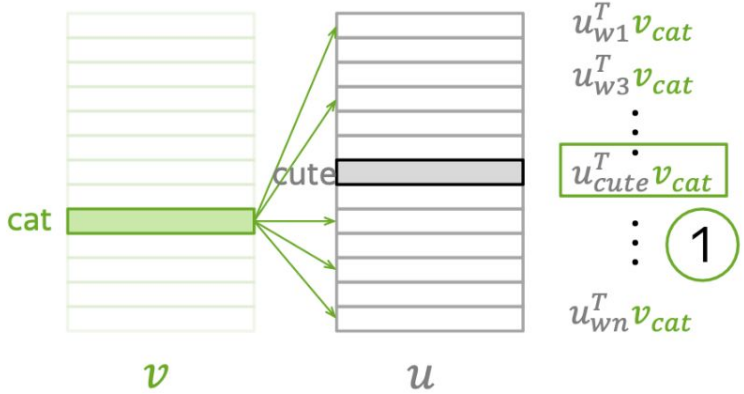


https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_{w \in V} \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$

1. Take dot product of v_{cat} with all u



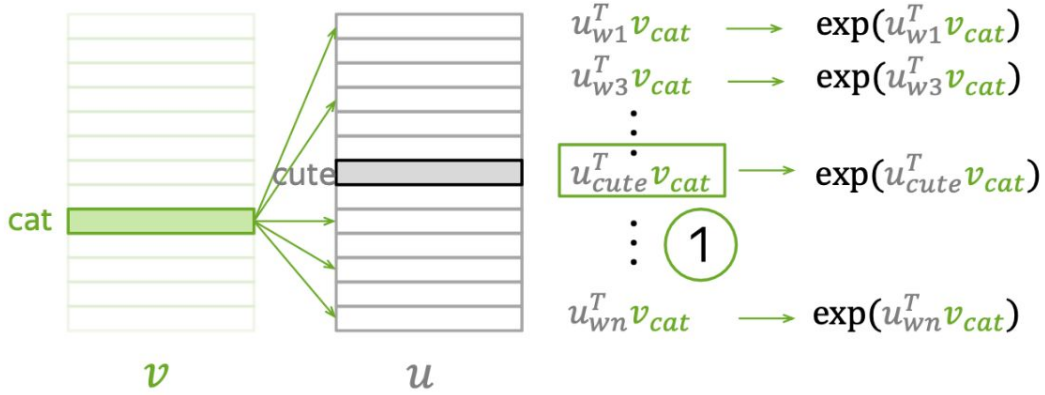
https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_{w \in V} \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$

1. Take dot product of v_{cat} with all u

2. exp

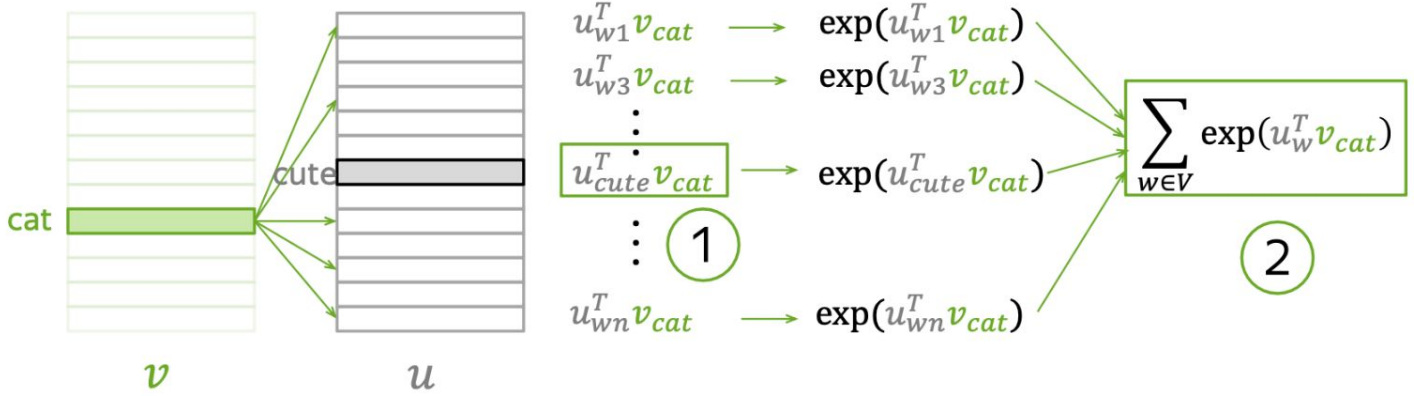


https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_{w \in V} \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$

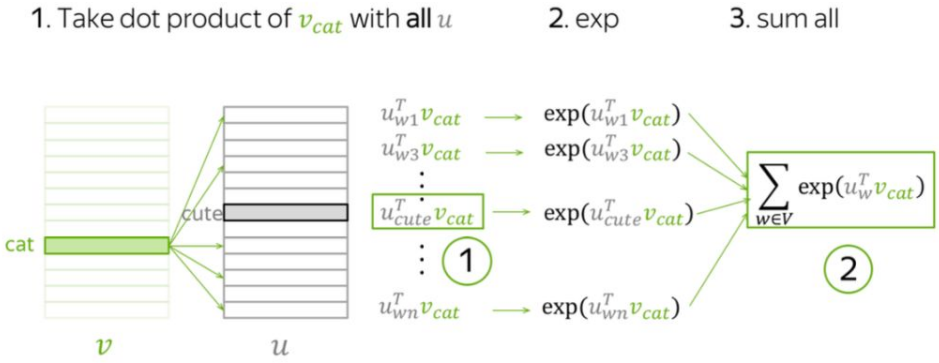
1. Take dot product of v_{cat} with all u 2. exp 3. sum all



https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat)$$
$$= -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$



4. get loss (for this one step)

$$J_{t,j}(\theta) = \underbrace{-u_{cute}^T v_{cat}}_{1} + \underbrace{\log \sum_{w \in V} \exp(u_w^T v_{cat})}_{2}$$

5. evaluate the gradient, make an update

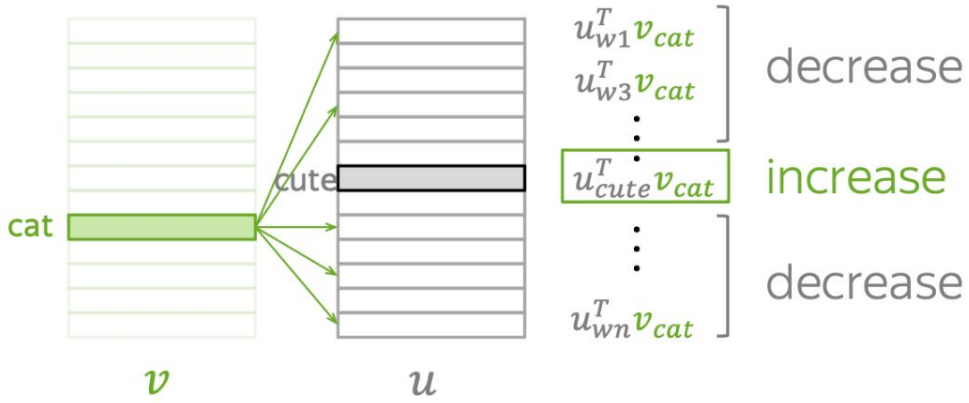
$$v_{cat} := v_{cat} - \alpha \frac{\partial J_{t,j}(\theta)}{\partial v_{cat}}$$

$$u_w := u_w - \alpha \frac{\partial J_{t,j}(\theta)}{\partial u_w} \; \forall \, w \in V$$

https://lena-voita.github.io/nlp_course.html

One Training step

$$-\log P(cute|cat) = -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$

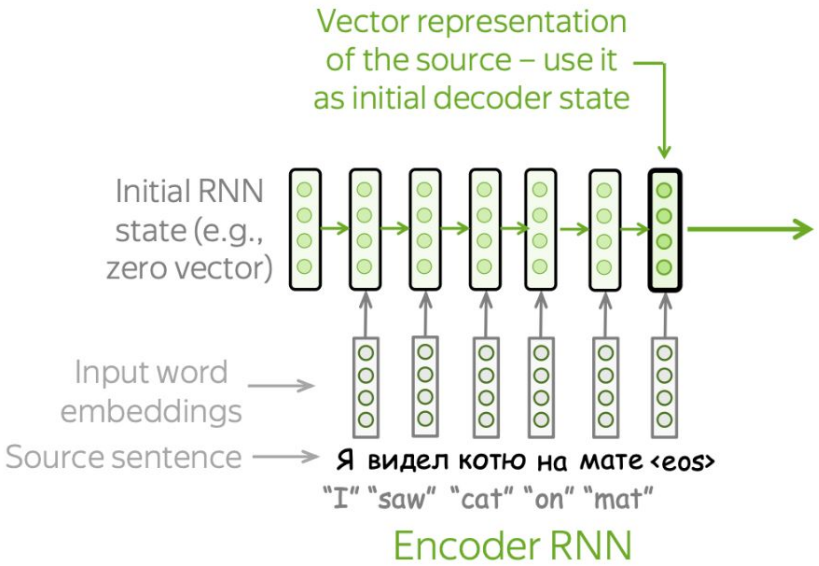


https://lena-voita.github.io/nlp_course.html

Machine Translation before transformers

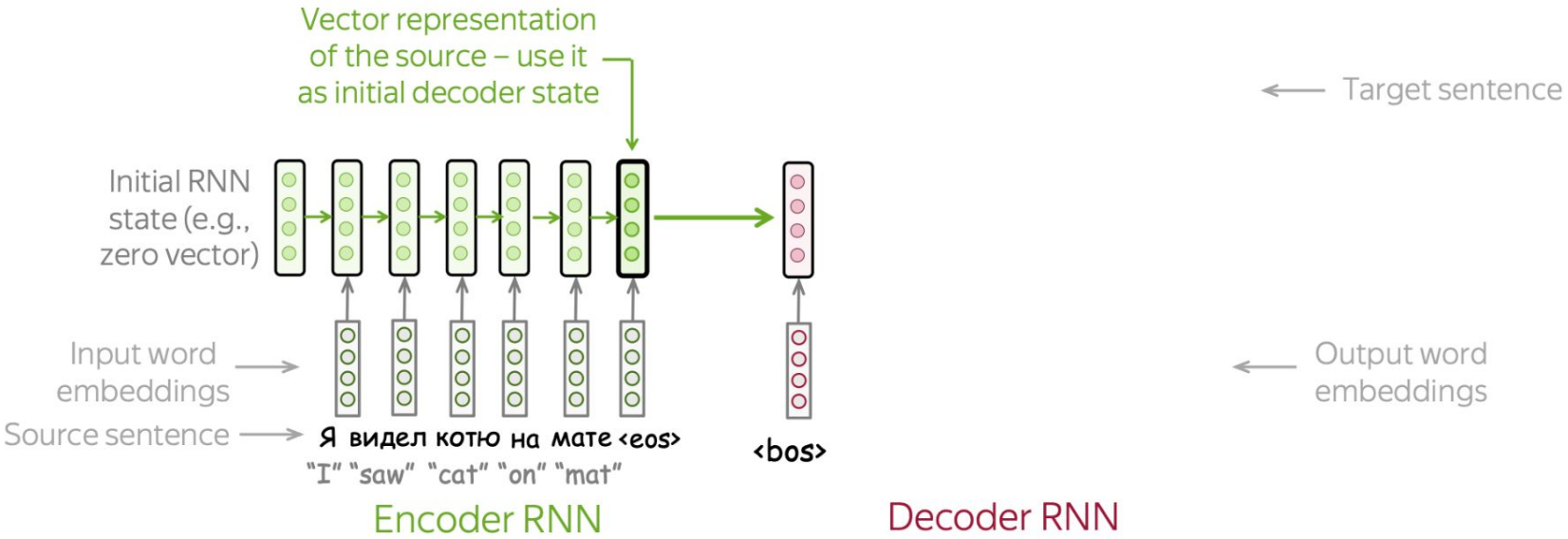
GTA San Andreas (2005)

Seq2Seq models: RNN Encoder-Decoder



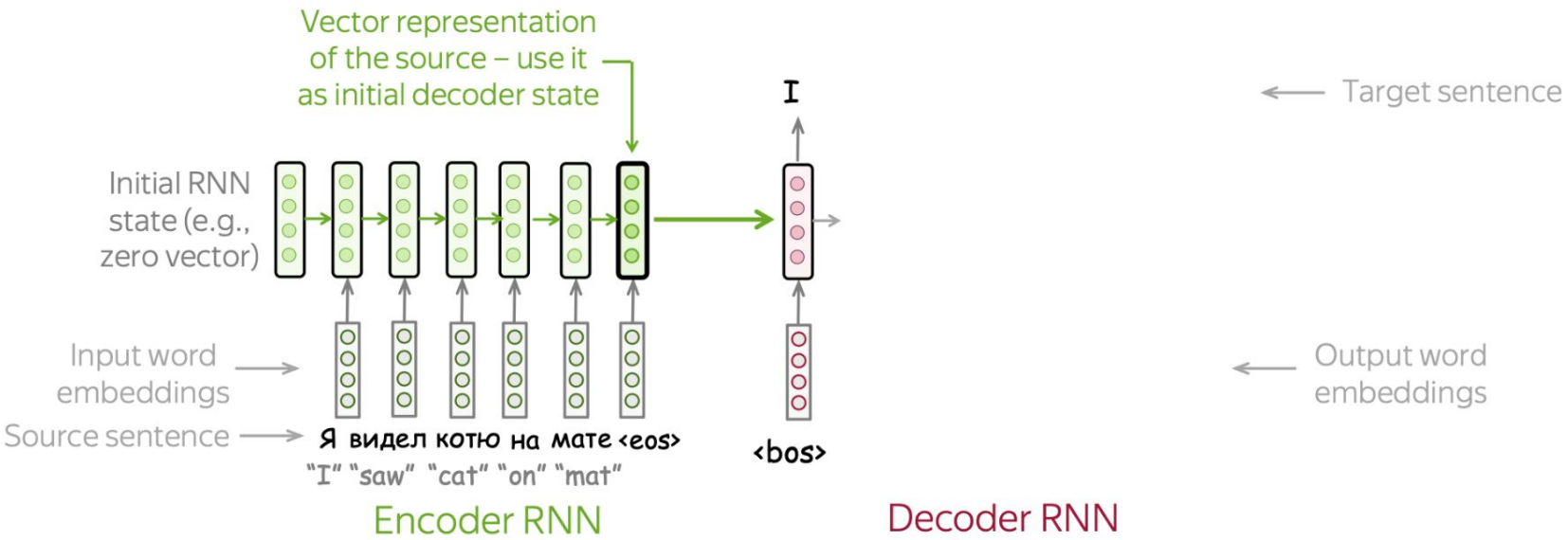
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: RNN Encoder-Decoder



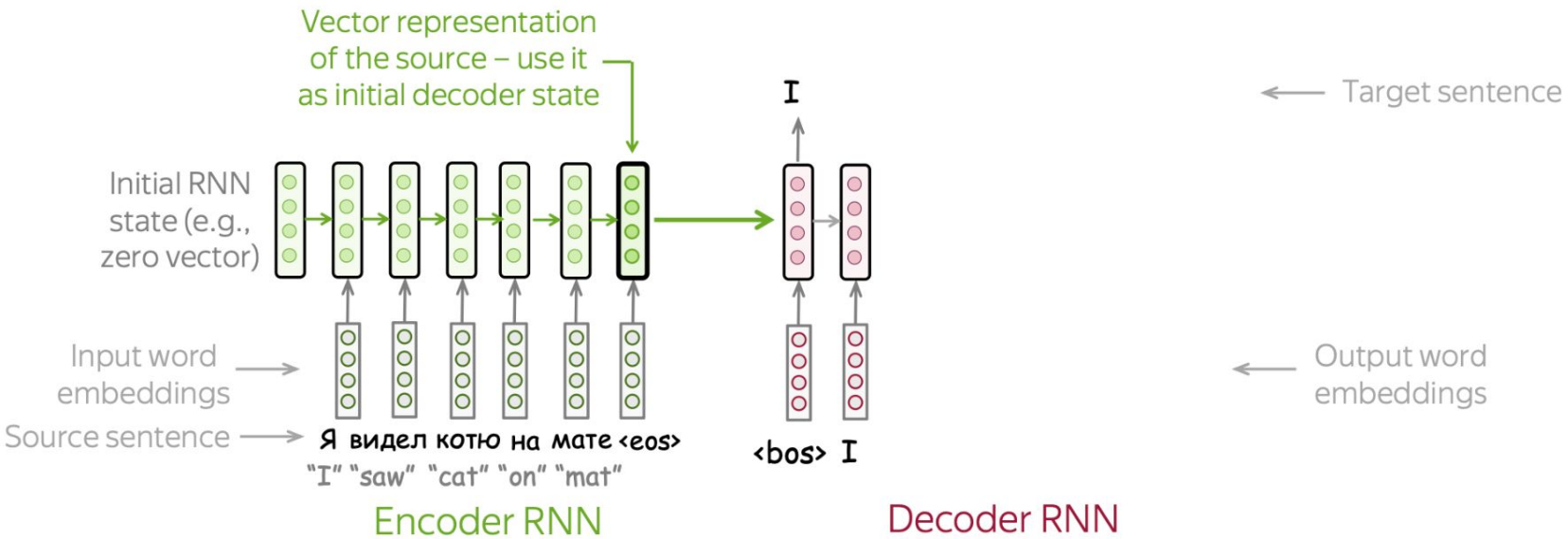
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: RNN Encoder-Decoder



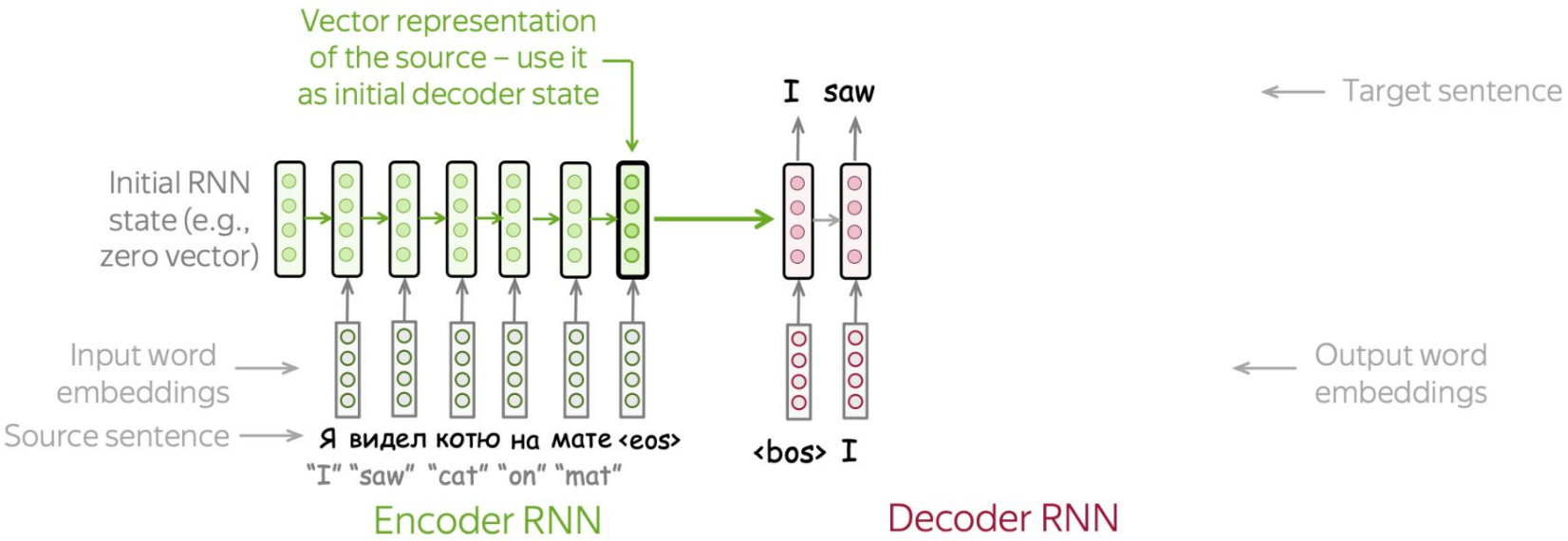
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: RNN Encoder-Decoder



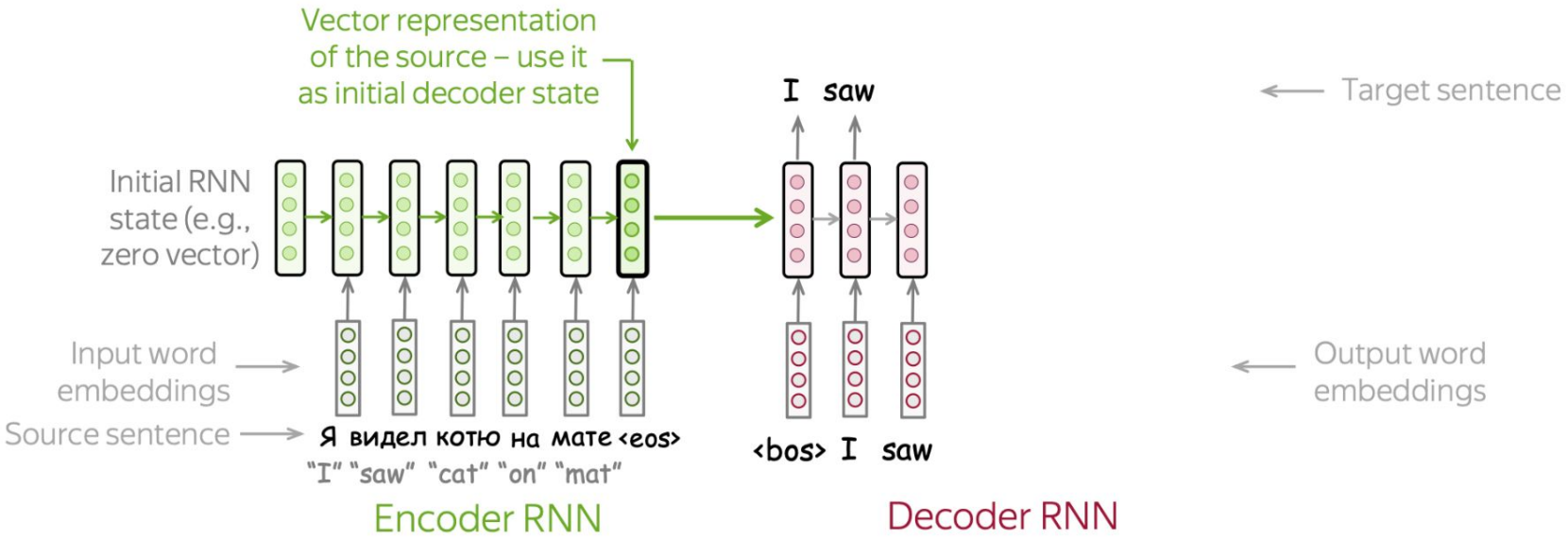
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: RNN Encoder-Decoder



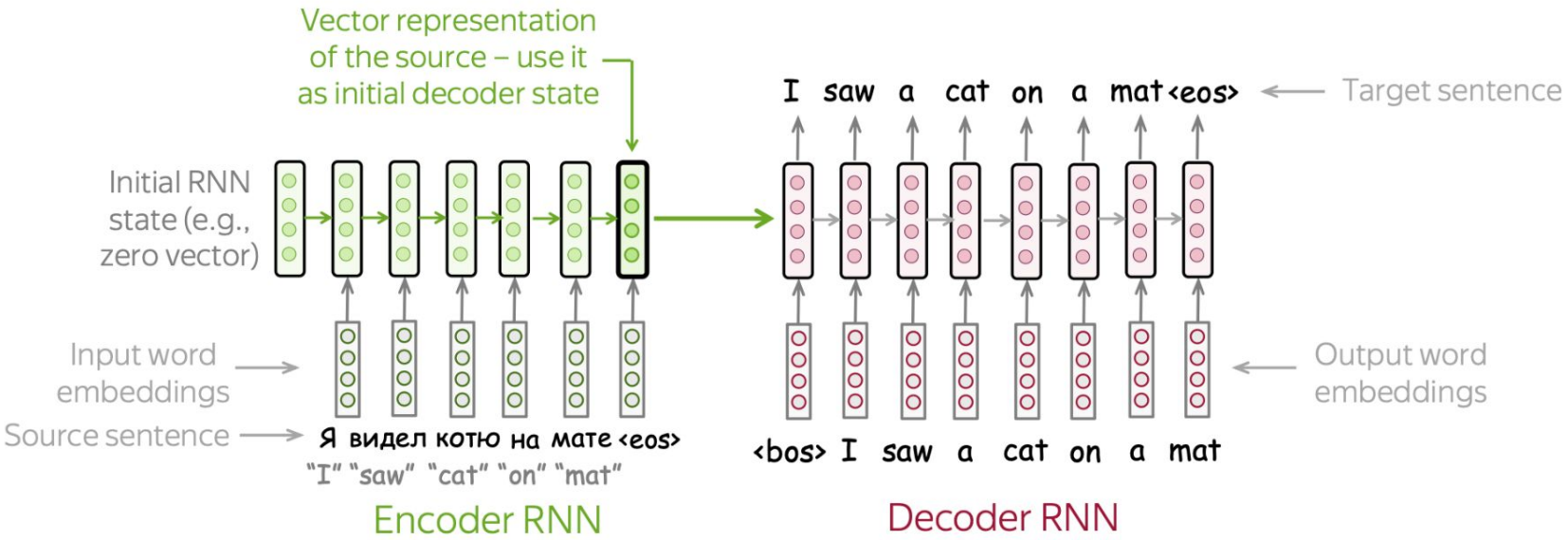
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: RNN Encoder-Decoder



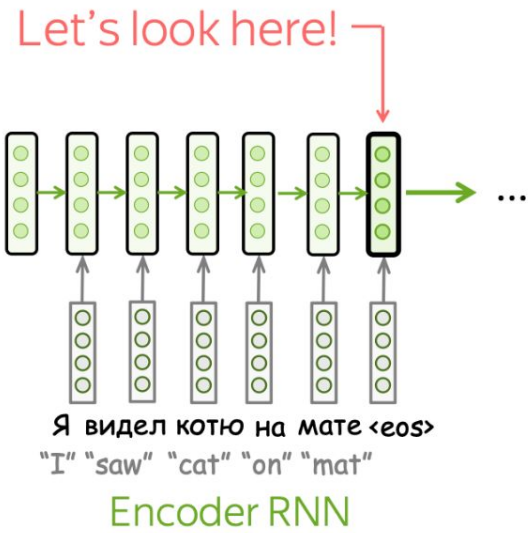
https://lena-voita.github.io/nlp_course.html

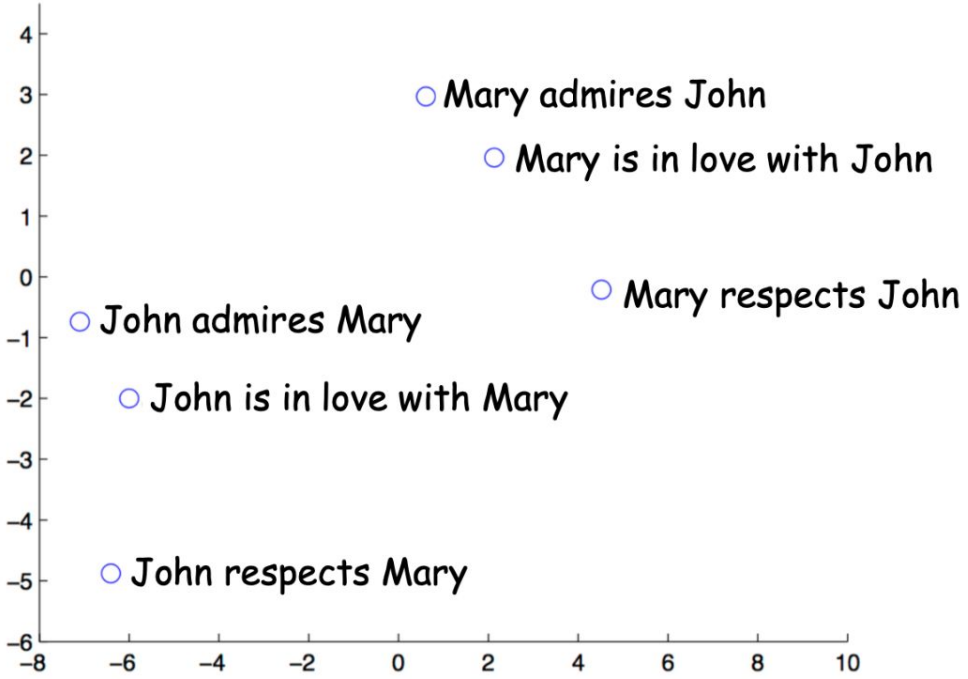
Seq2Seq models: RNN Encoder-Decoder



https://lena-voita.github.io/nlp_course.html

What does final encoder state represent?



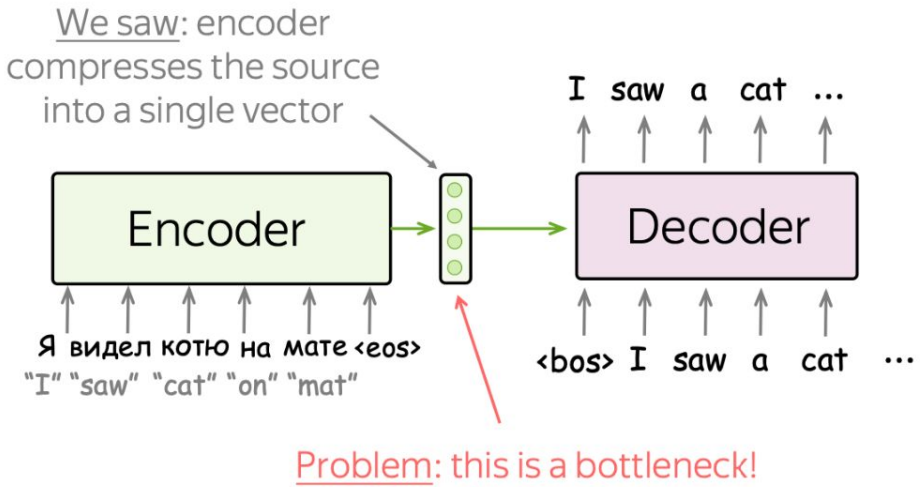


https://arxiv.org/abs/1409.3215

Seq2Seq models: RNN bottelneck

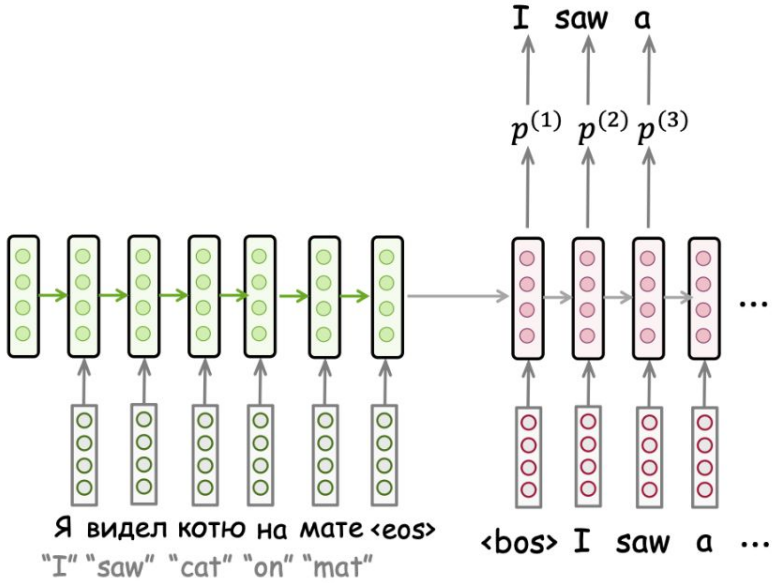
Fixed source representation is bad:

- for encoder, it is hard to compress a sentence
- for decoder, different information may be needed at different steps



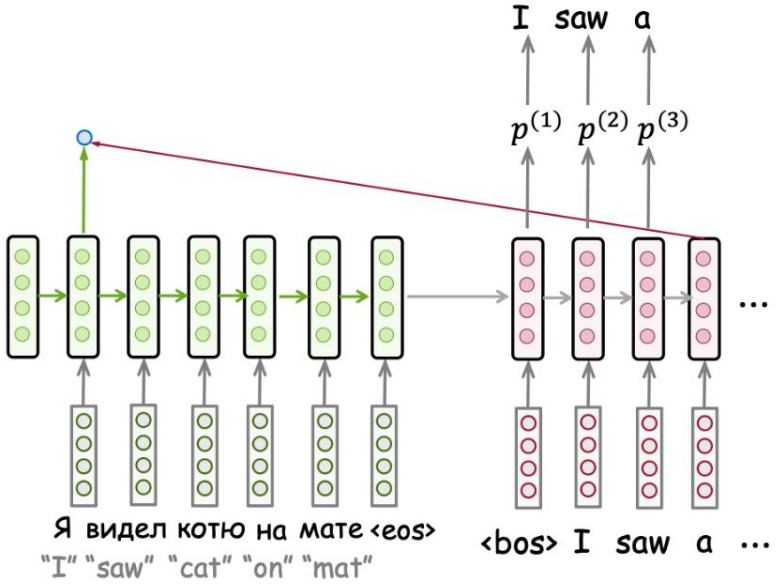
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



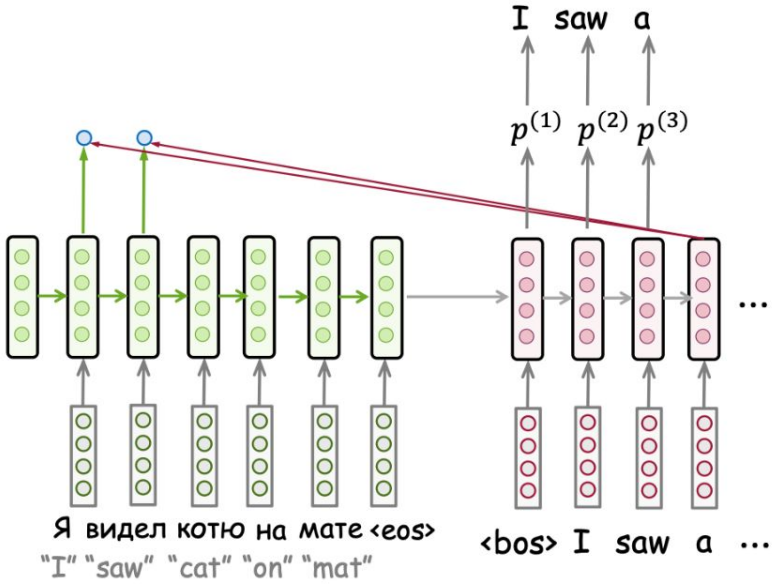
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



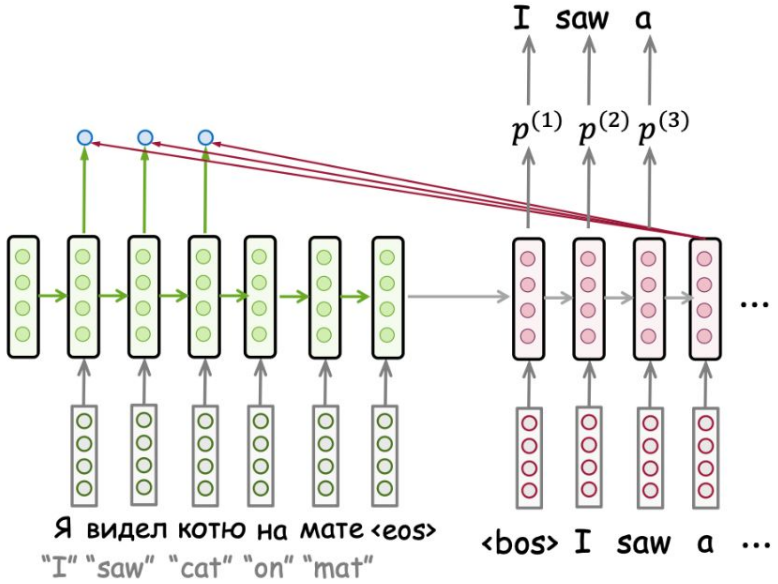
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



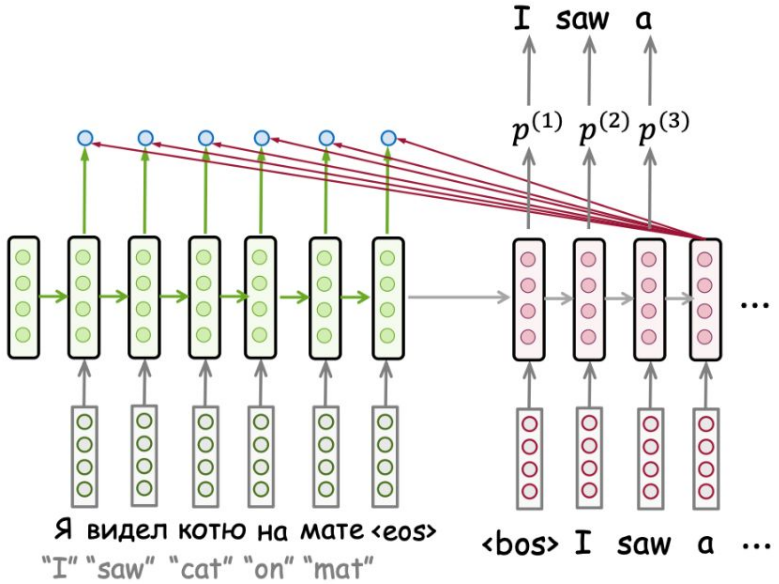
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



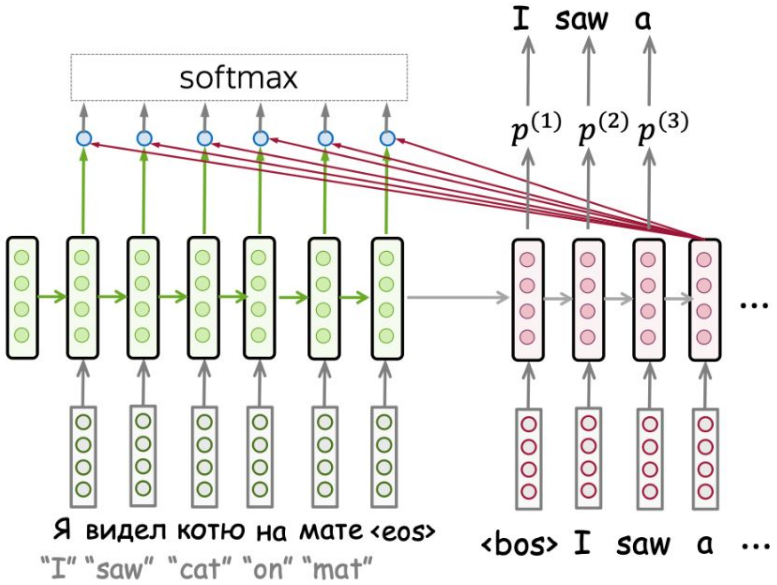
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



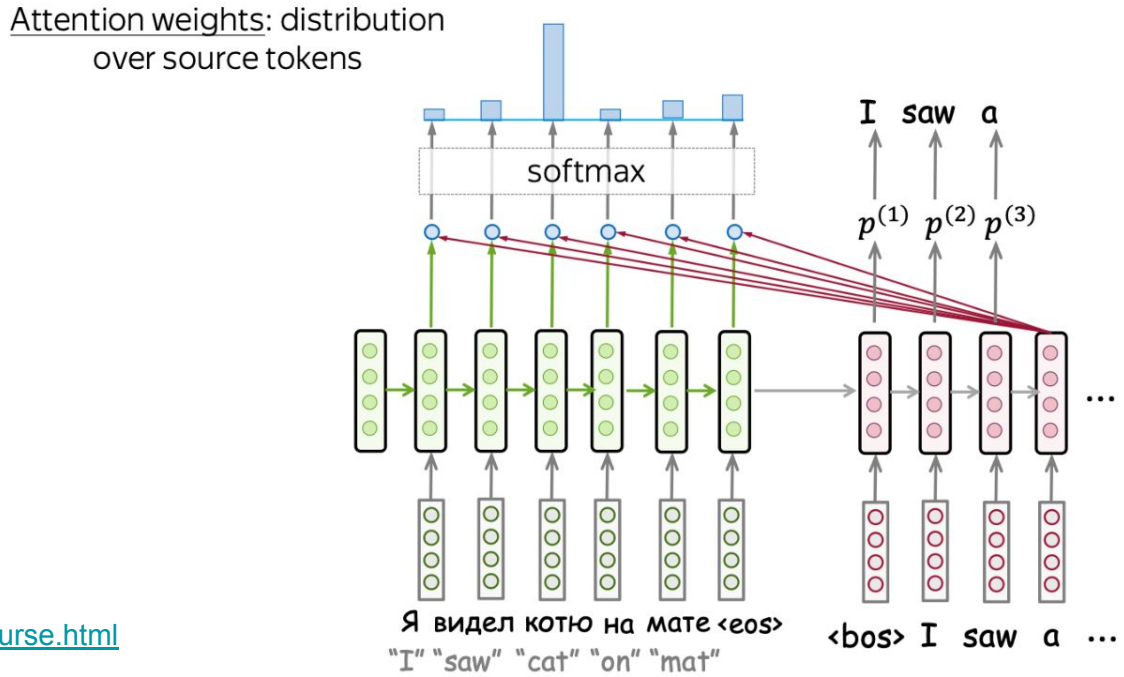
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



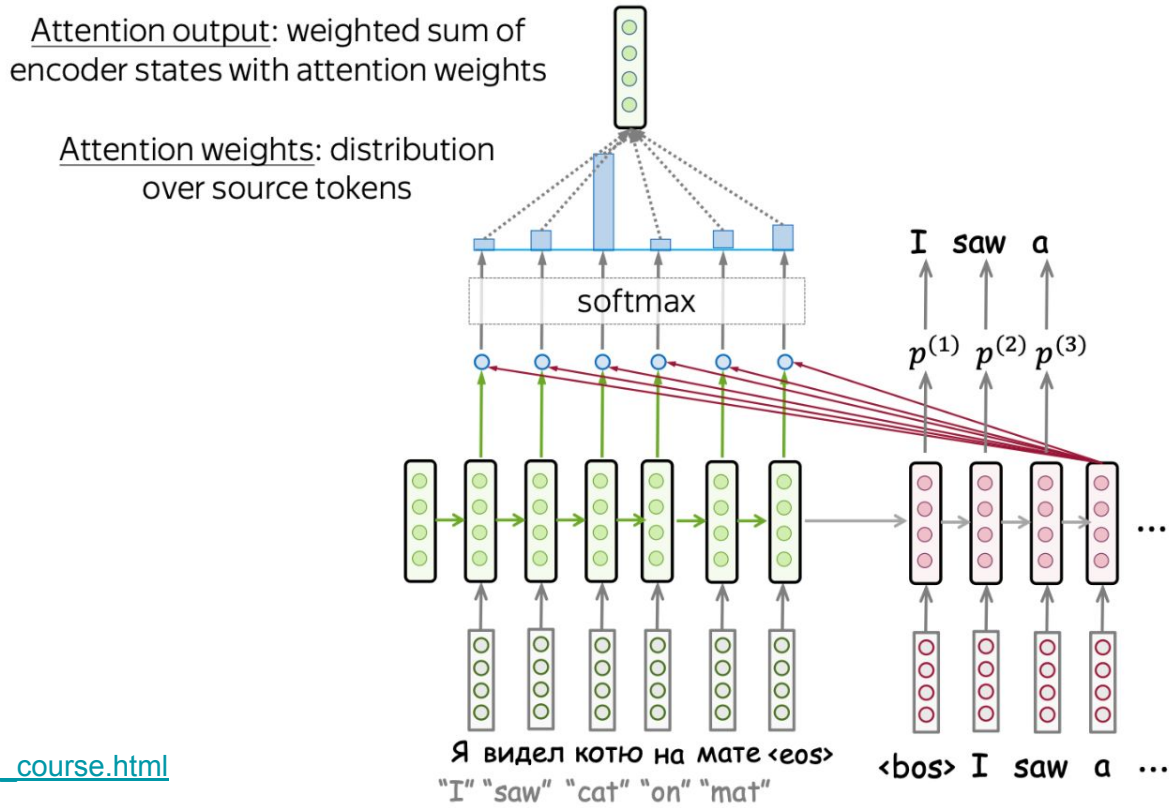
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



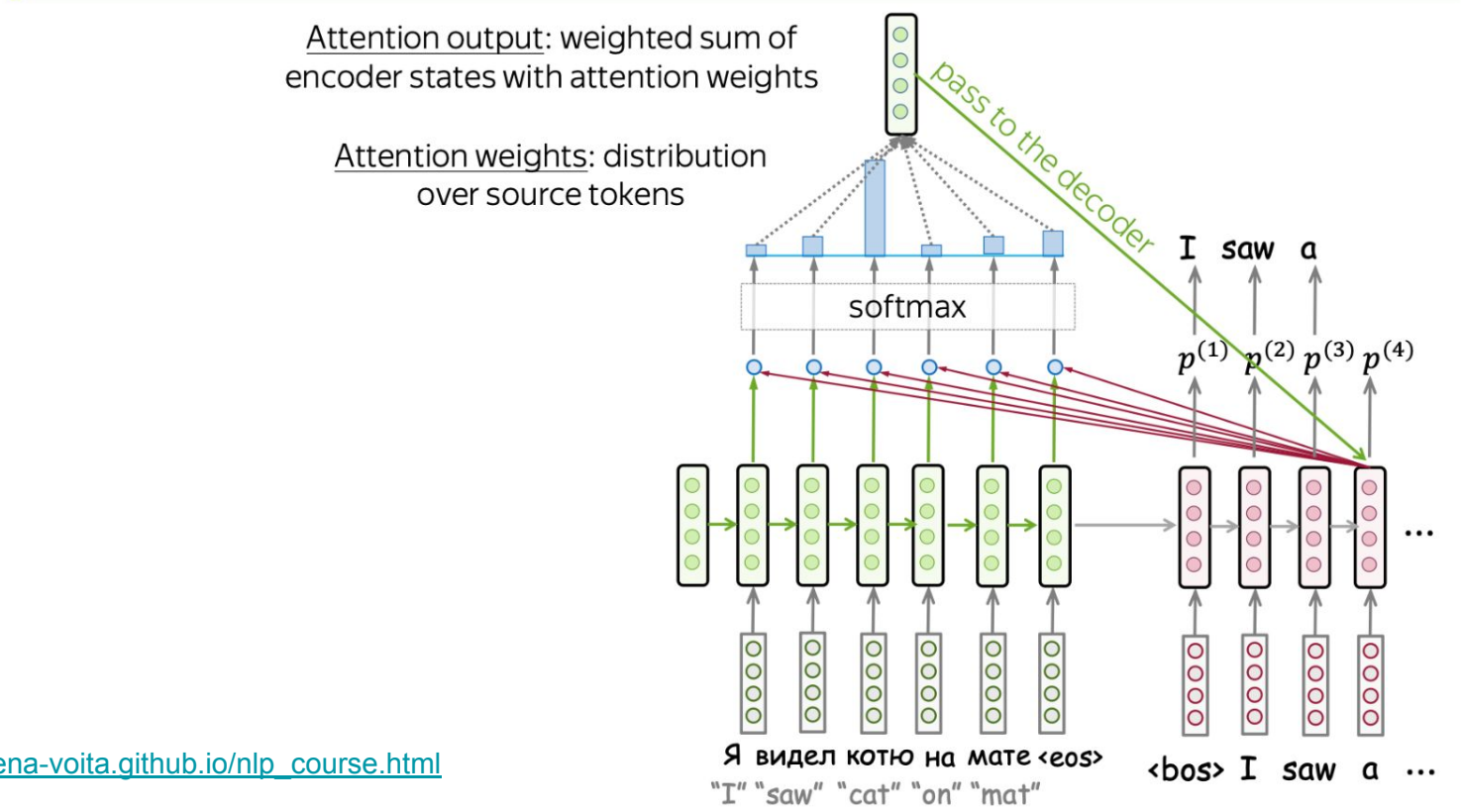
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



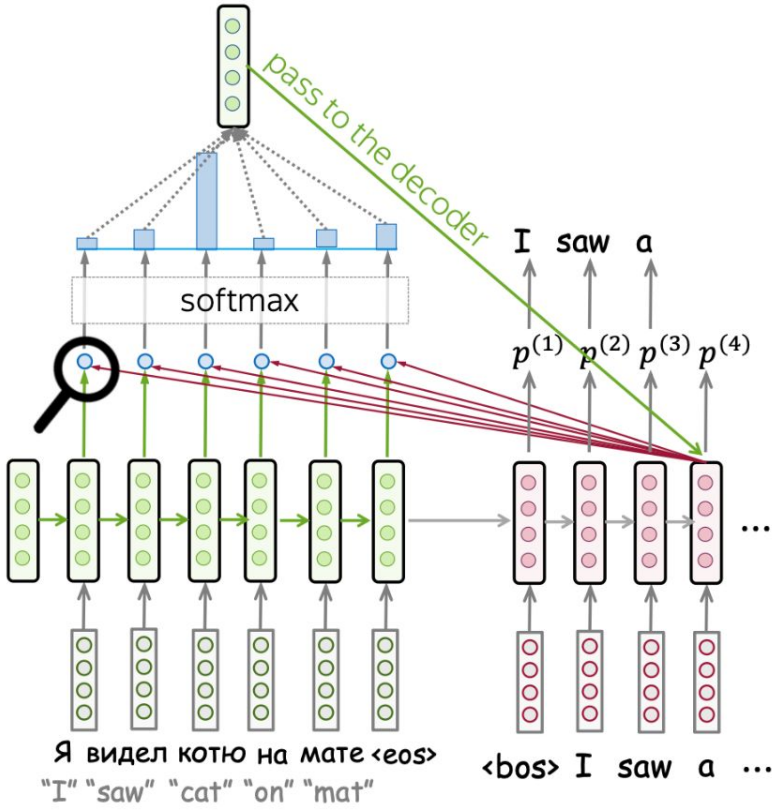
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



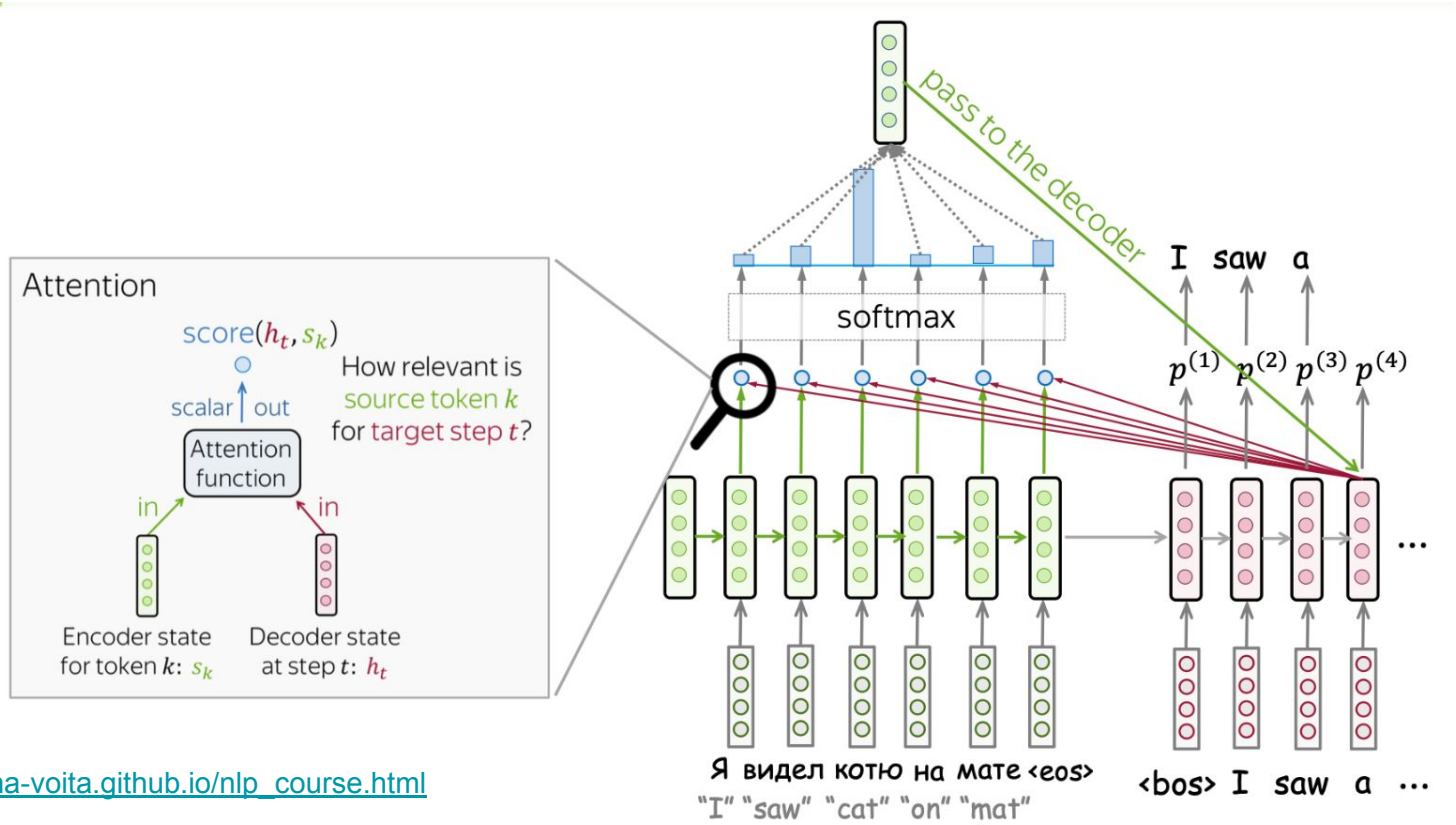
https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



https://lena-voita.github.io/nlp_course.html

Seq2Seq models: Encoder-Decoder Attention



https://lena-voita.github.io/nlp_course.html

Computation Pipeline

Attention input

$s_1, s_2, \ldots, s_m$

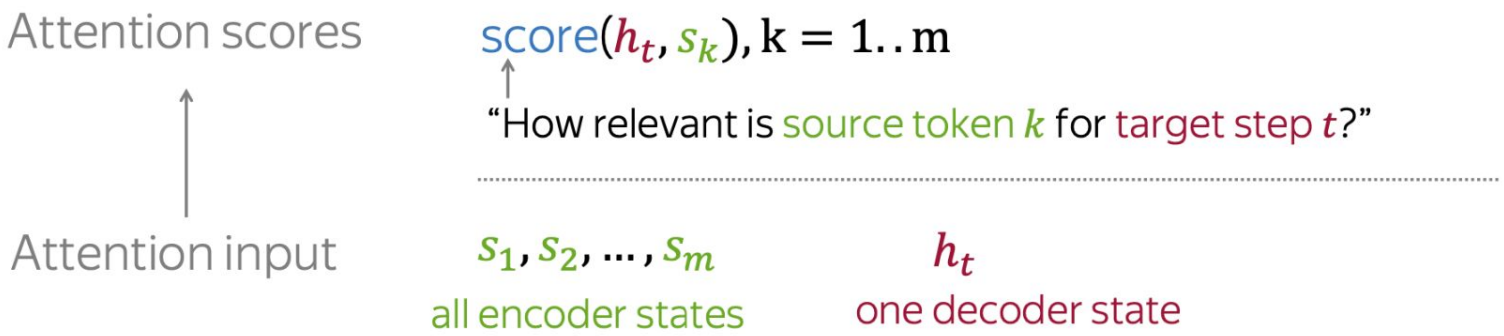
all encoder states

h_t

one decoder state

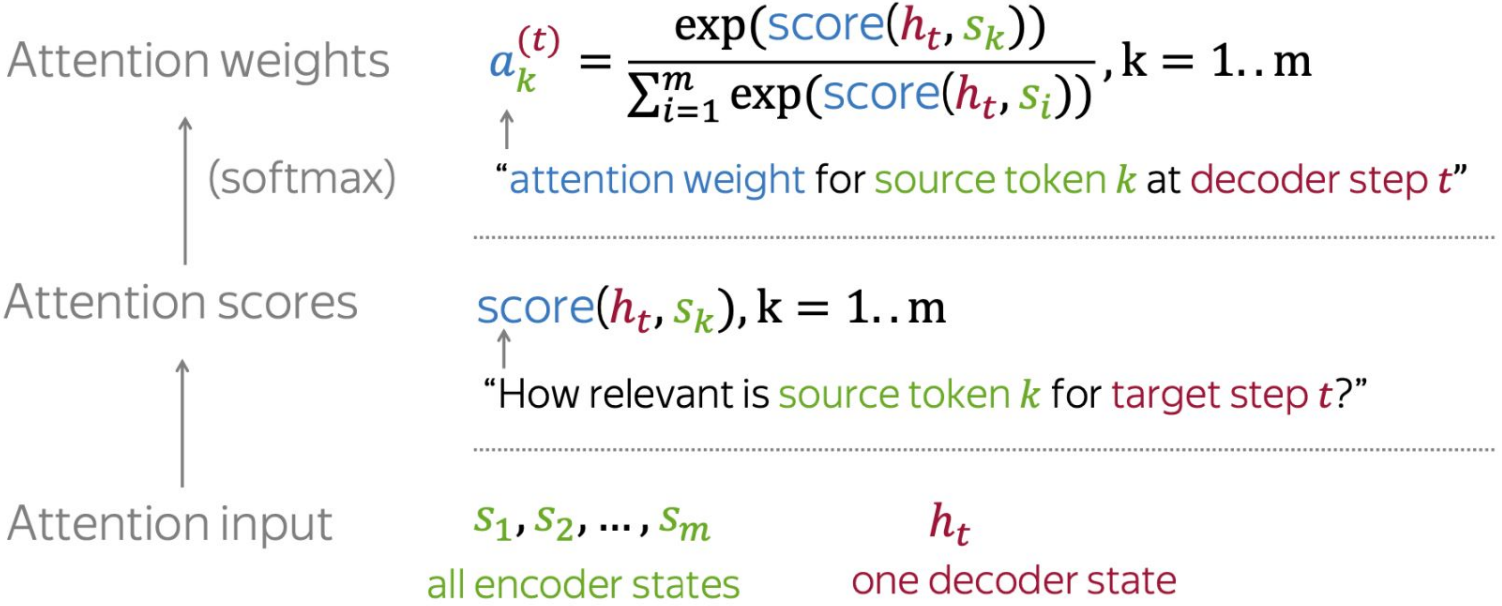
https://lena-voita.github.io/nlp_course.html

Computation Pipeline



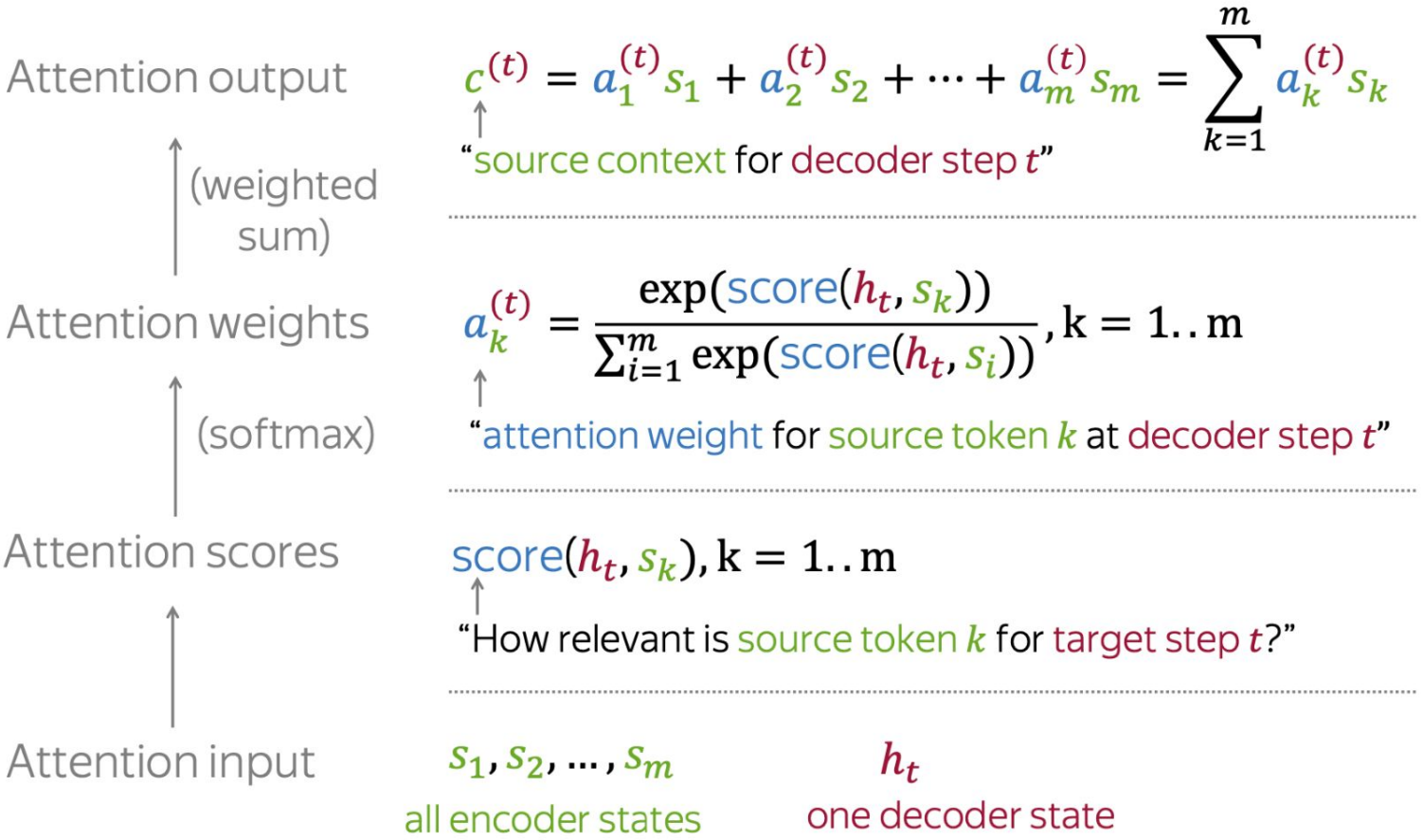
https://lena-voita.github.io/nlp_course.html

Computation Pipeline



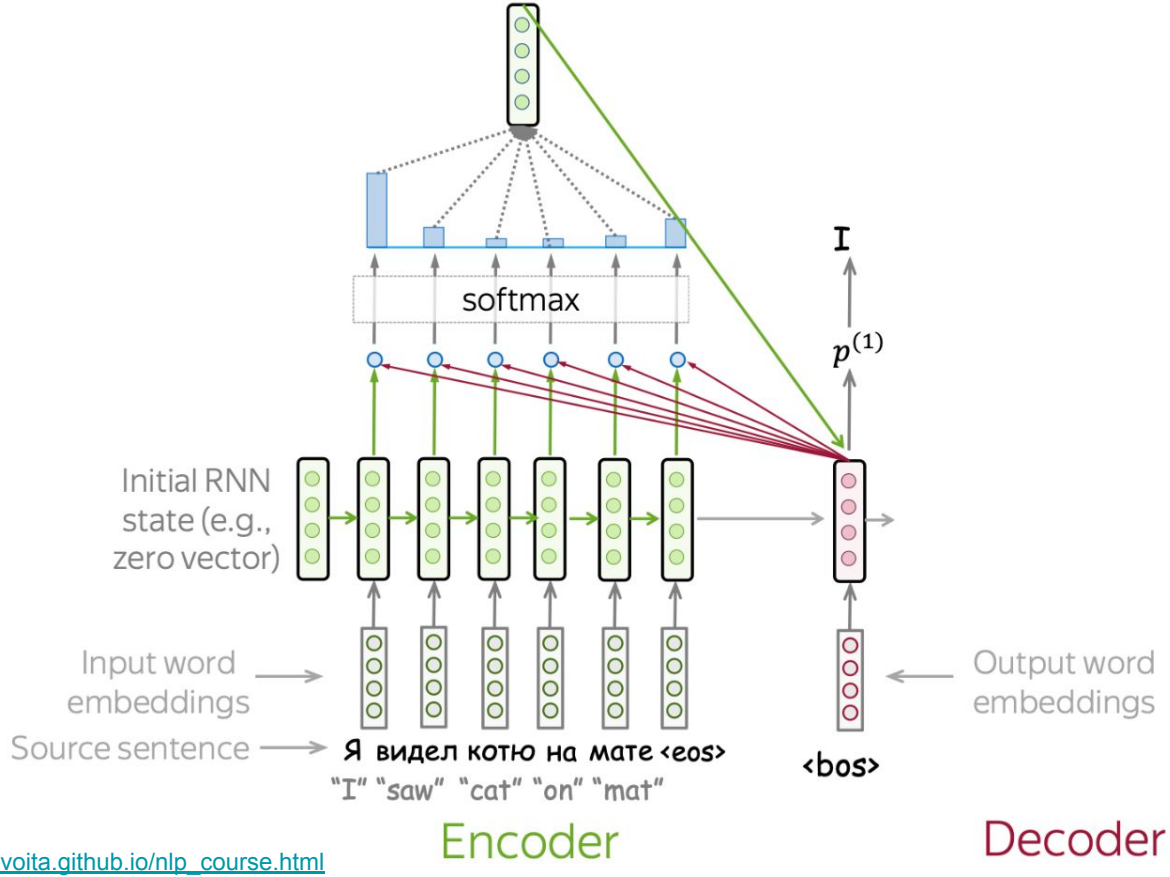
https://lena-voita.github.io/nlp_course.html

Computation Pipeline



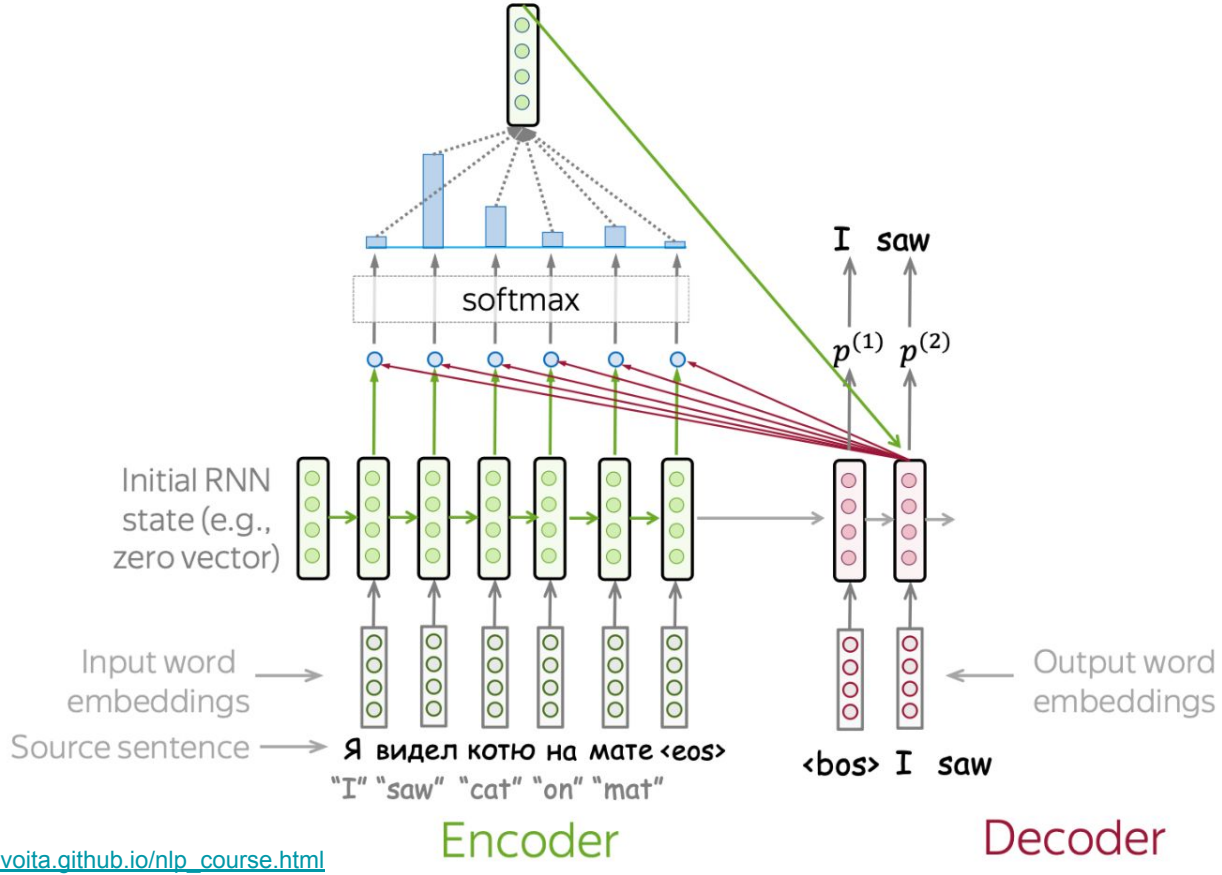
https://lena-voita.github.io/nlp_course.html

Model Learns to Pick Relevant Tokens



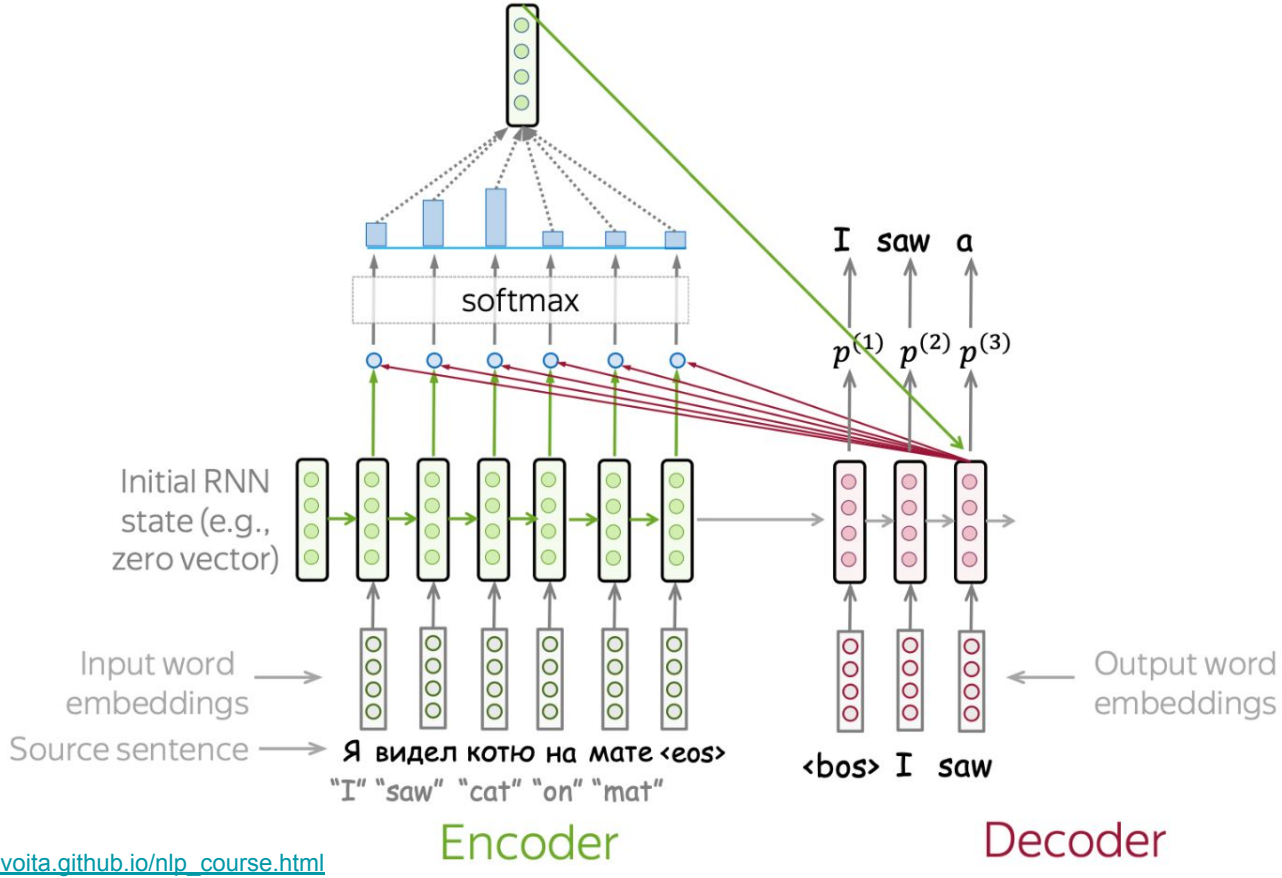
https://lena-voita.github.io/nlp_course.html

Model Learns to Pick Relevant Tokens



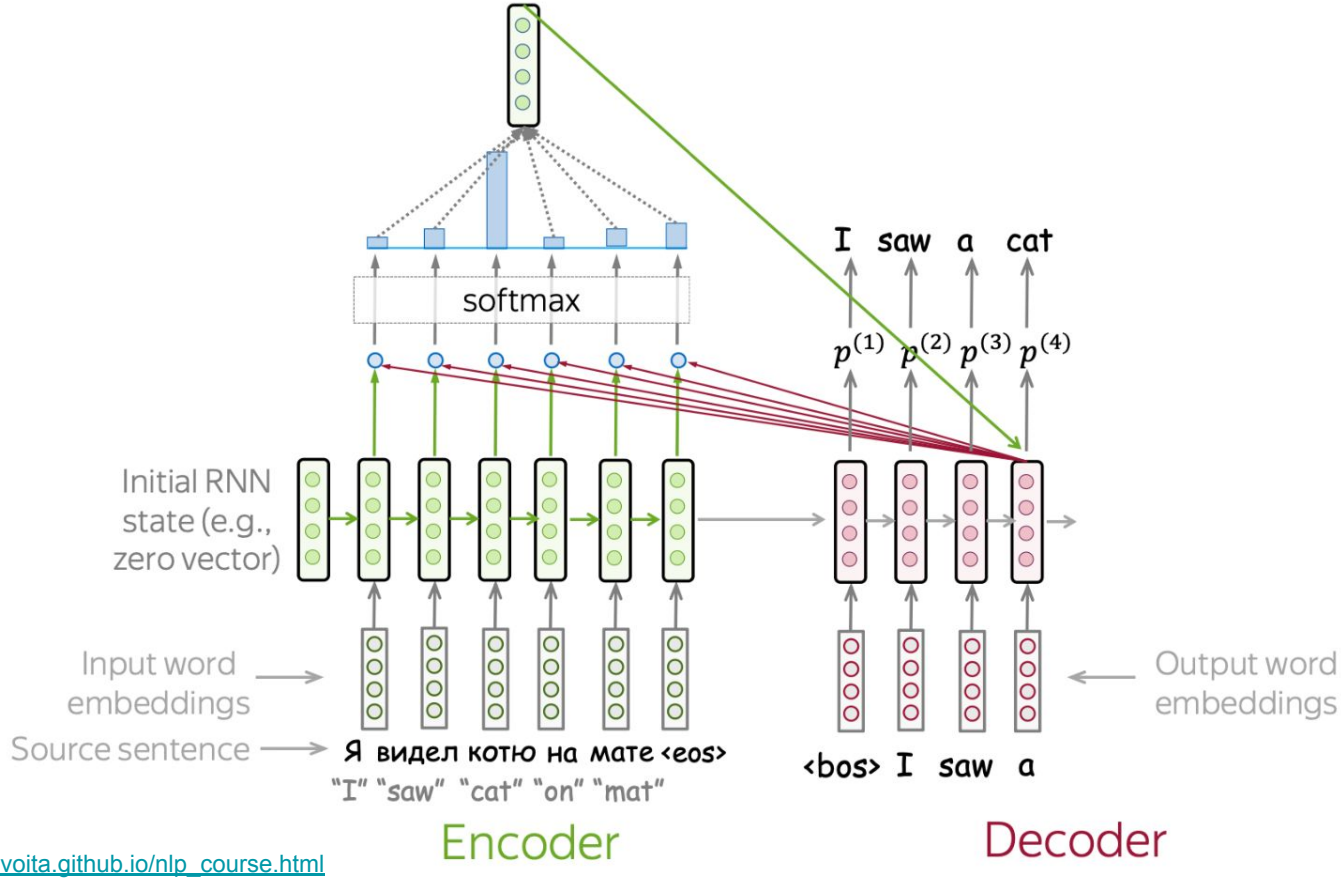
https://lena-voita.github.io/nlp_course.html

Model Learns to Pick Relevant Tokens



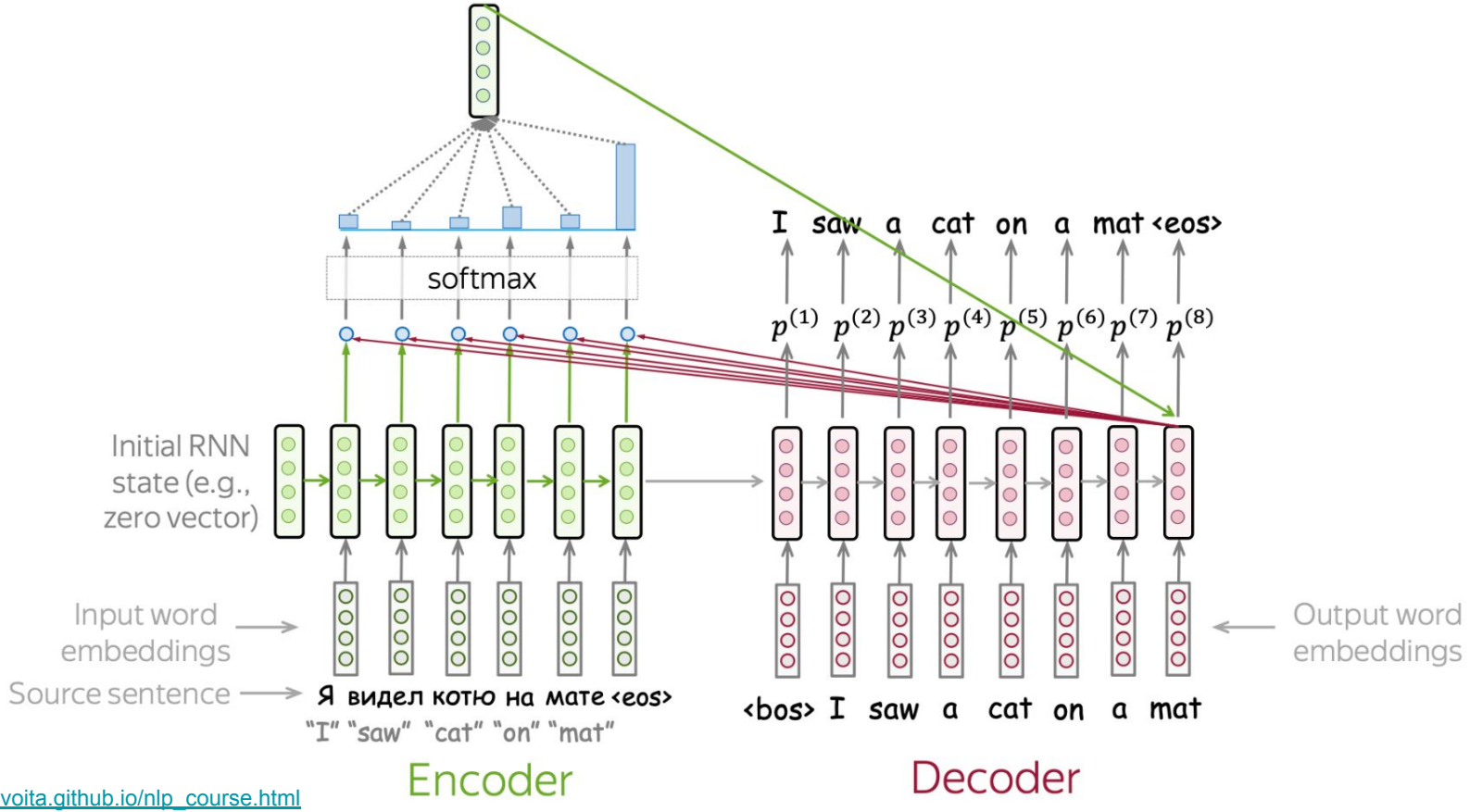
https://lena-voita.github.io/nlp_course.html

Model Learns to Pick Relevant Tokens



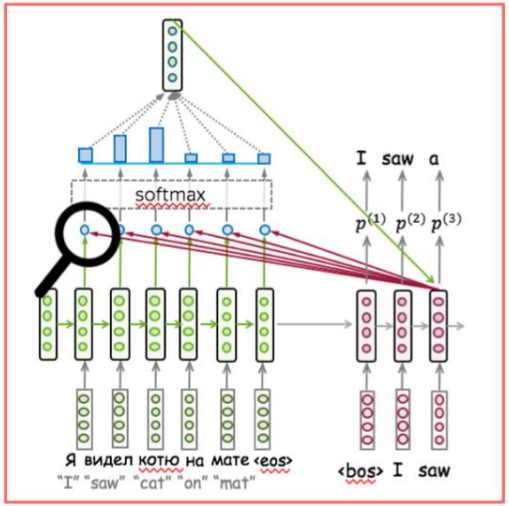
https://lena-voita.github.io/nlp_course.html

Model Learns to Pick Relevant Tokens

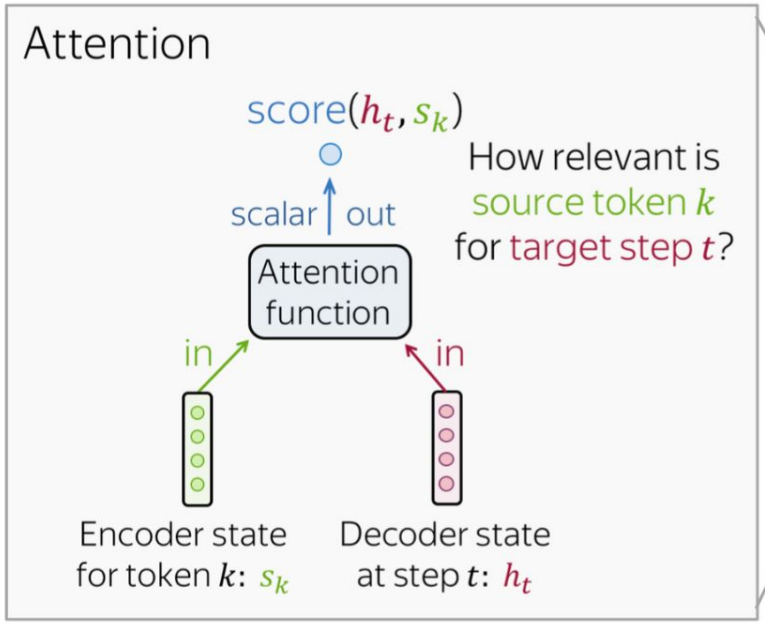


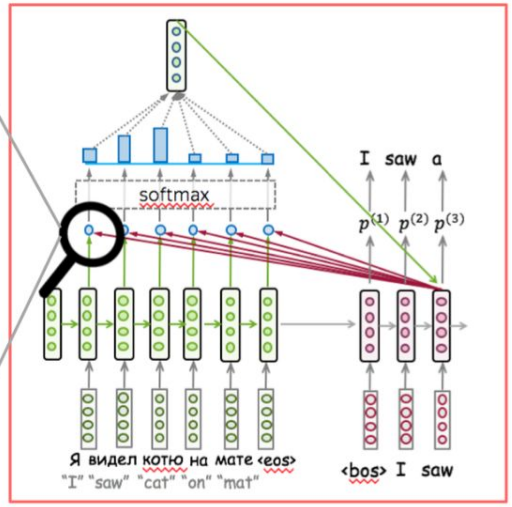
https://lena-voita.github.io/nlp_course.html

Attention Score Functions



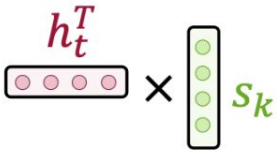
Attention Score Functions



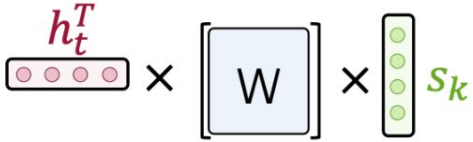


Attention Score Functions

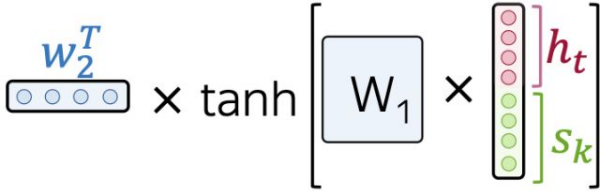
- Dot-product: $\text{score}(h_t, s_k) = h_t^T s_k$

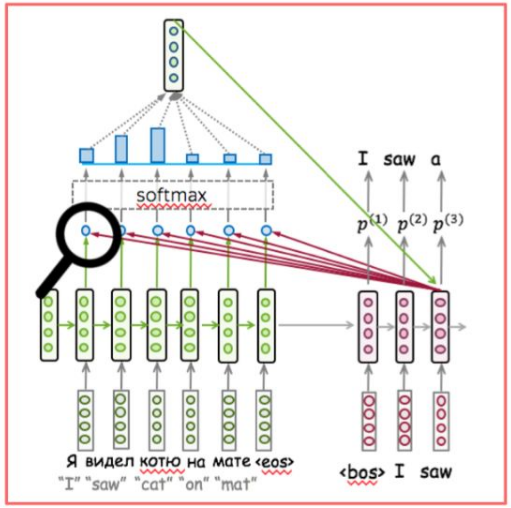


- Bilinear: $\text{score}(h_t, s_k) = h_t^T W s_k$



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





Bahdanau model (sep 2014)

Published as a conference paper at ICLR 2015

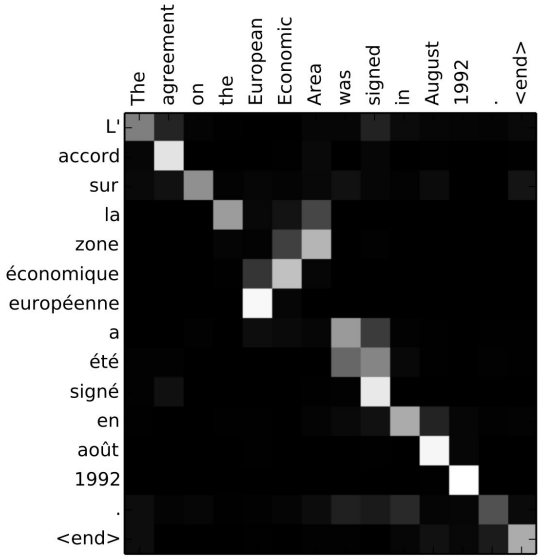
NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal

ABSTRACT

Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder–decoders and encode a source sentence into a fixed-length vector from which a decoder generates a translation. In this paper, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder–decoder architecture, and propose to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly. With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of English-to-French translation. Furthermore, qualitative analysis reveals that the (soft-)alignments found by the model agree well with our intuition.



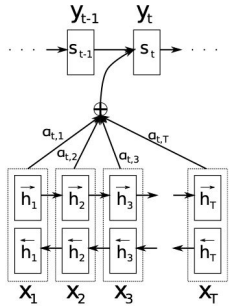


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \tag{6}$$

where

$$e_{ij} = a(s_{i-1}, h_j)$$

The probability α_{ij}, or its associated energy e_{ij}, reflects the importance of the annotation h_j with respect to the previous hidden state s_{i-1} in deciding the next state s_i and generating y_i. Intuitively, this implements a mechanism of attention in the decoder. The decoder decides parts of the source sentence to pay attention to. By letting the decoder have an attention mechanism, we relieve the encoder from the burden of having to encode all information in the source sentence into a fixed-length vector. With this new approach the information can be spread throughout the sequence of annotations, which can be selectively retrieved by the decoder accordingly.

https://arxiv.org/abs/1409.0473

Bahdanau model (sep 2014)

So I started thinking about how to avoid the bottleneck between encoder and decoder RNN. My first idea was to have a model with two “cursors”, one moving through the source sequence (encoded by a BiRNN) and another one moving through the target sequence. The cursor trajectories would be marginalized out using dynamic programming. KyungHyun Cho recognized this as an equivalent to Alex Graves’ RNN Transducer model. Following that, I may have also read Graves’ hand-writing recognition paper. The approach looked inappropriate for machine translation though.

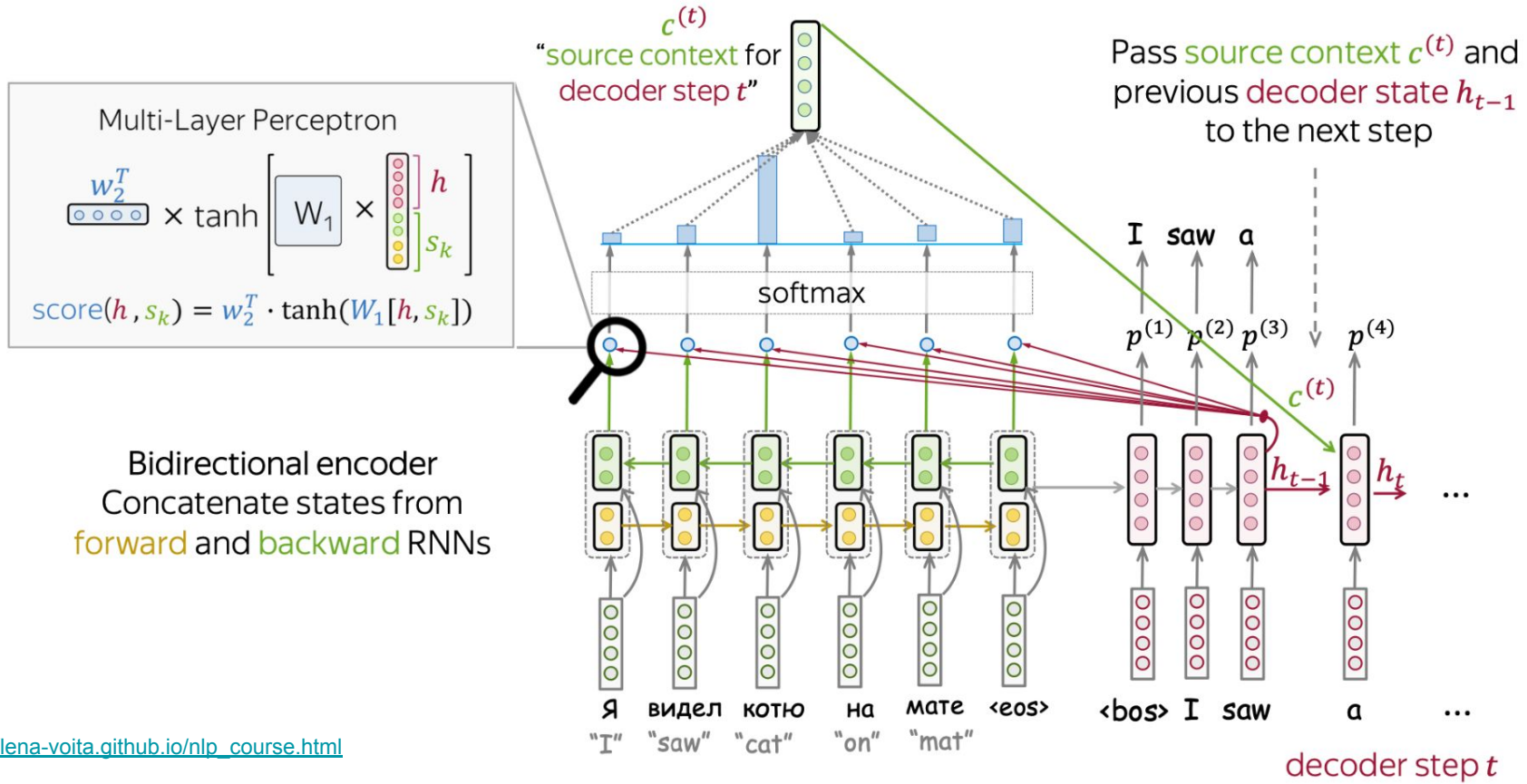
The above approach with cursors would be too hard to implement in the remaining 5 weeks of my internship. So I tried instead something simpler - two cursors moving at the same time synchronously (effectively hard-coded diagonal attention). That sort of worked, but the approach lacked elegance.

So one day I had this thought that it would be nice to enable the decoder RNN to learn to search where to put the cursor in the source sequence. This was sort of inspired by translation exercises that learning English in my middle school involved. Your gaze shifts back and forth between source and target sequence as you translate. I expressed the soft search as softmax and then weighted averaging of BiRNN states. It worked great from the very first try to my great excitement. I called the architecture RNNSearch, and we rushed to publish an ArXiV paper as we knew that Ilya and co at Google are somewhat ahead of us with their giant 8 GPU LSTM model (RNN Search still ran on 1 GPU).

As it later turned out, the name was not great. The better name (attention) was only added by Yoshua to the conclusion in one of the final passes.

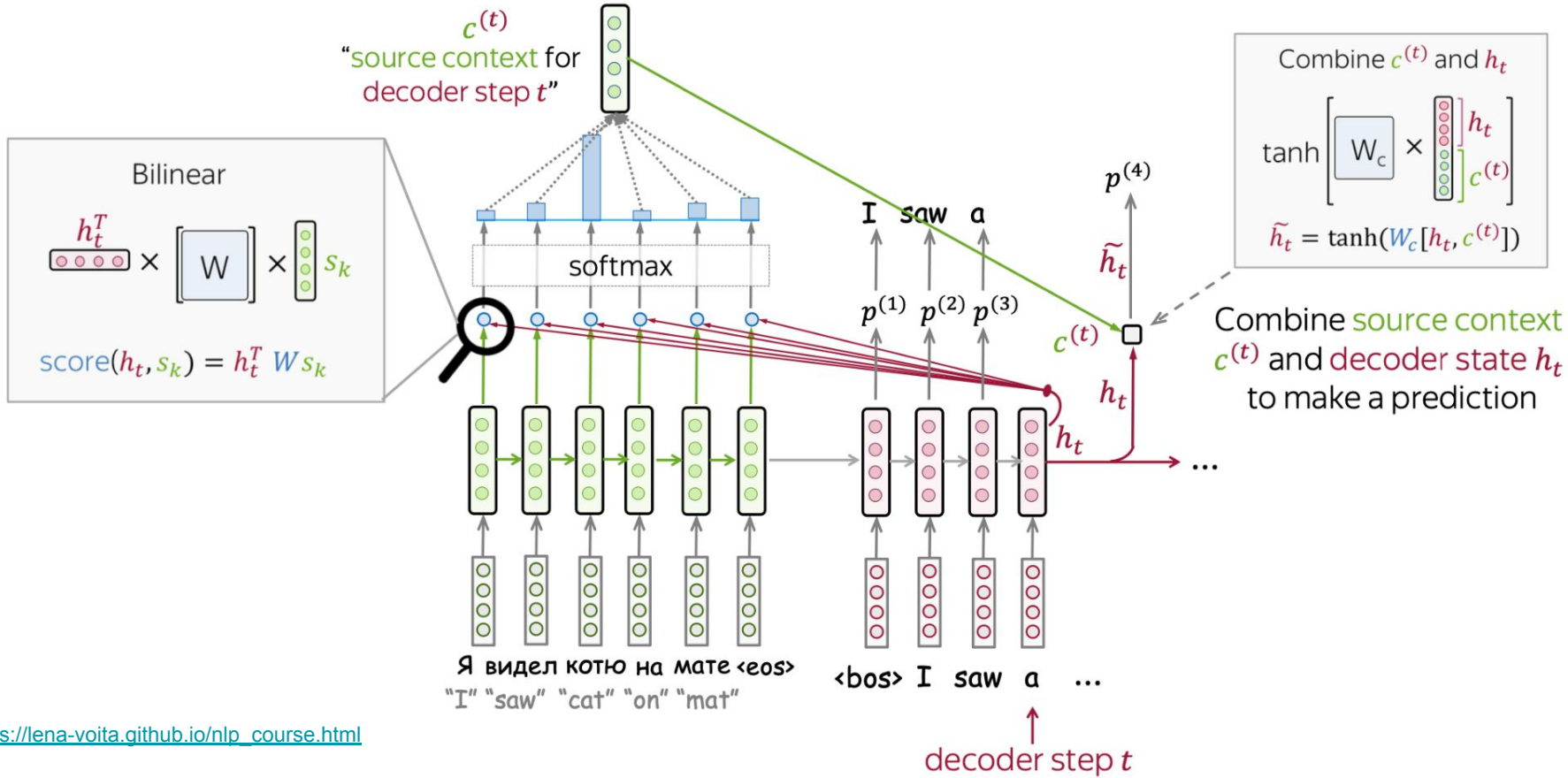
https://www.youtube.com/watch?v=XfpMkf4rD6E

Bahdanau model (sep 2014)



https://lena-voita.github.io/nlp_course.html

Luong model (aug 2015)



https://lena-voita.github.io/nlp_course.html