AI in Mathematics
Lecture 6
Deep Learning in Mathematics

Bar-Ilan University
Nebius Academy | Stevens Institute of Technology

April 29, 2025

About This Course

~~1 week: Intro~~

~~2 weeks: Classic ML~~

~~2 weeks: Deep Learning in Mathematics~~

3 weeks: Math as an NLP problem (LLMs etc.)

3 weeks: Reinforcement Learning (RL) in Math

1 week: Advanced AI topics

1 week: Project Presentations

Lyapunov Functions

A Lyapunov function is a function associated with an ordinary differential equation (ODE):

$$\dot{x} = g(x), \quad x \in \mathbb{R}^n.$$

Definition: A function $V : \mathbb{R}^n \to \mathbb{R}$ is called a **Lyapunov function** for the system if:

- $V(x) > 0$ for all $x \neq 0$,
- $V(0) = 0$,
- $\dot{V}(x) = \langle \nabla V(x), \dot{x} \rangle \leq 0.$

Why are they important?

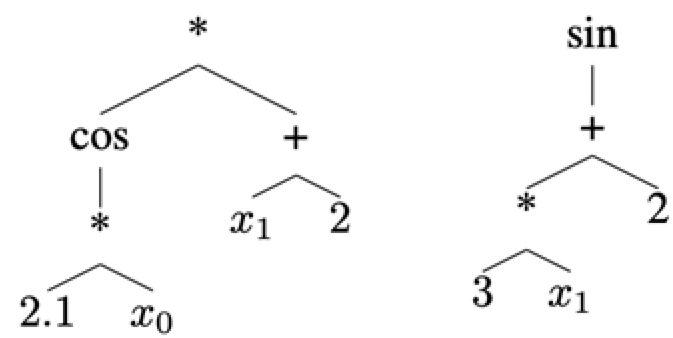
Lyapunov function $\Leftrightarrow$ Stable system

Lyapunov Functions

The problem of predicting Lyapunov function naturally states the question: How to represent functions as features or provide them as an answer?

$$\begin{cases} \dot{x}_0 = \cos(2.1x_0)(x_1 + 2) \\ \dot{x}_1 = \sin(3x_1 + 2) \end{cases}$$

is represented as



Lyapunov Functions

Previously our methods depended on fixed size of features and output was either a class or a number

We will need a new functionality to cover such a problem.

Lyapunov functions

We train transformers with 8 layers, 10 attention heads and an embedding dimension of 640 (ablation studies on different model sizes can be found in Appendix C), on batches of 16 examples, using the Adam optimizer [Kingma and Ba, 2014] with a learning rate of 10–4, an initial linear warm-up phase of 10,000 optimization steps, and inverse square root scheduling.

Inverse Square Root Schedule

It is a learning rate schedule (for gradient descent)

$$\frac{1}{\sqrt{\max(n,k)}}$$

where n is the current training iteration and k is the number of warm-up steps. This sets a constant learning rate for first k steps and then decays the learning rate until pre-training is over.

Lyapunov functions

We train transformers with 8 layers, 10 attention heads and an embedding dimension of 640 (ablation studies on different model sizes can be found in Appendix C), on batches of 16 examples, using the Adam optimizer [Kingma and Ba, 2014] with a learning rate of 10–4, an initial linear warm-up phase of 10,000 optimization steps, and inverse square root scheduling.

Transformers

Will be next time!

Sequential data

General idea:
We want to be able to process sequential data:

0, 1, 1, 2, 3, 5, 8, 13, 21…

В прямоугольном треугольнике квадрат гипотенузы равен сумме квадратов двух других сторон.

במשולש ישר-זווית, ריבוע היתר שווה לסכום ריבועי שתי הצלעות האחרות.

In a right-angled triangle, the square of the hypotenuse is equal to the sum of the squares of the other two sides.

$\sin(x) + \frac{\cos(x)}{2x}$ - 17x + 4

Types of problems

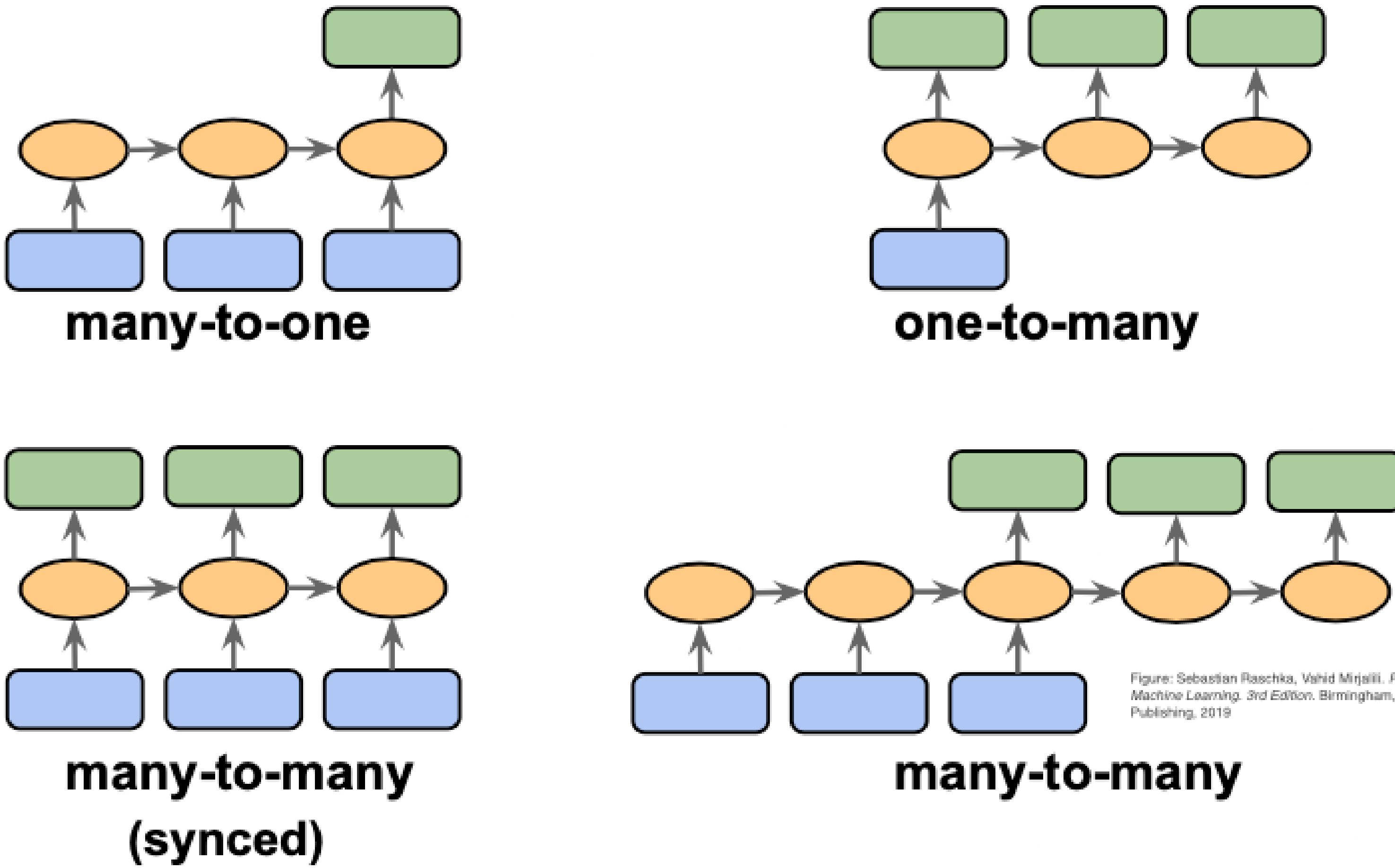
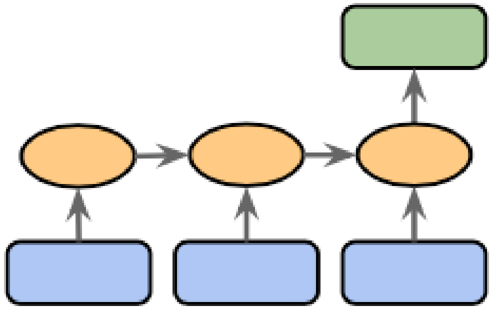
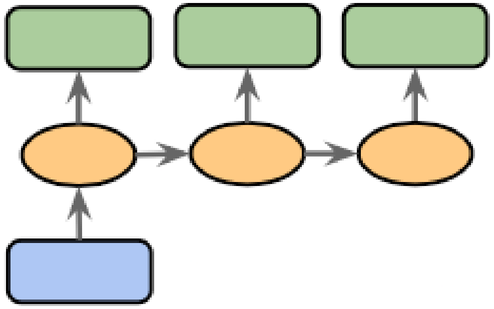


Figure: Sebastian Raschka, Vahid Mirjalili. *Python Machine Learning. 3rd Edition*. Birmingham, UK: Packt Publishing, 2019

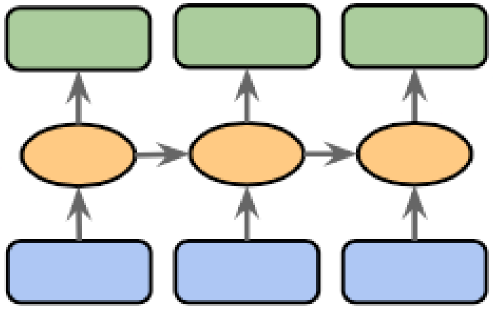
Types of problems



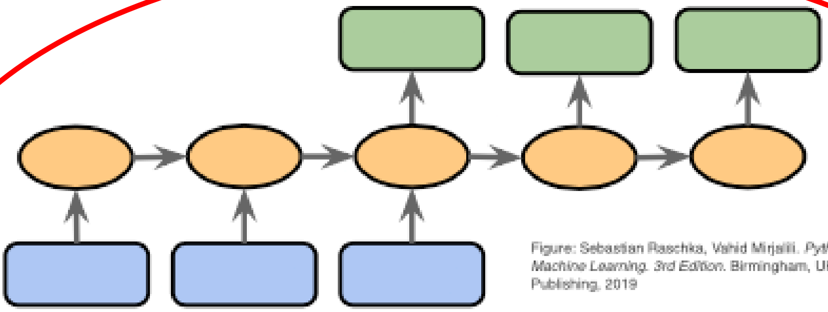
many-to-one



one-to-many



many-to-many
(synced)



many-to-many

Figure: Sebastian Raschka, Vahid Mirjalili. *Python Machine Learning. 3rd Edition*. Birmingham, UK: Packt Publishing, 2019

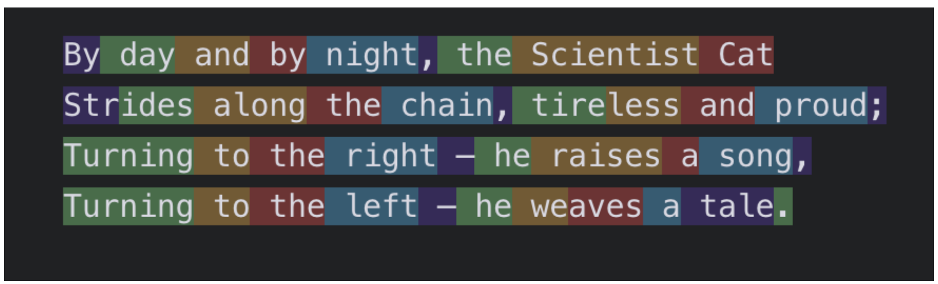
Approach

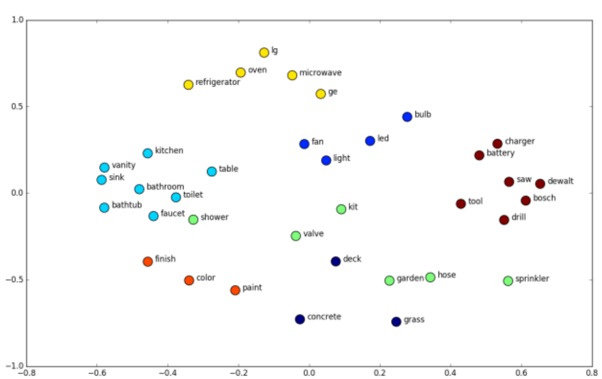
Tokenization + Embedding

Breaking sequential data into discrete tokens and mapping each token to a vector representation.

Dividing continuous sequences into meaningful, learnable segments.(tokens)

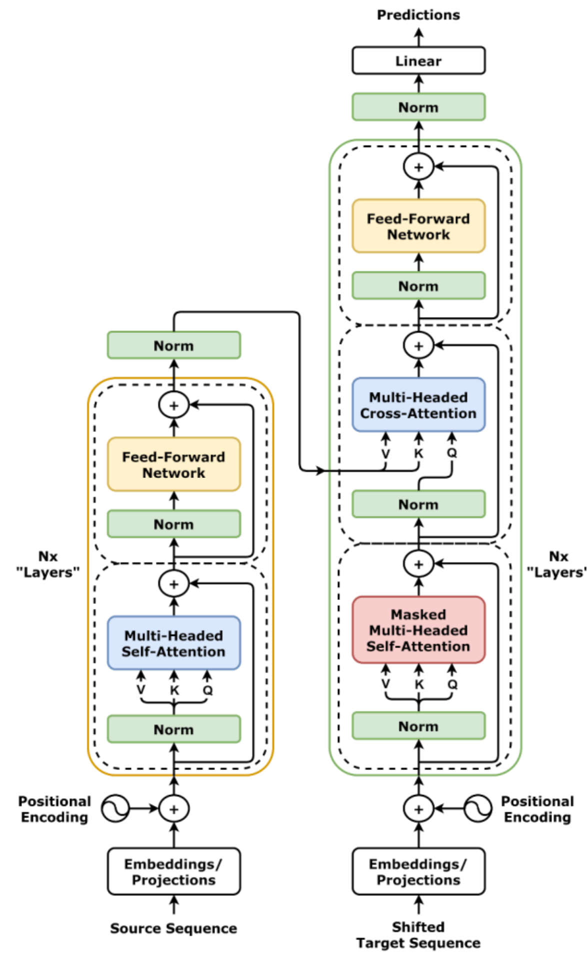
Representing each token or segment as a fixed-dimensional vector in an embedding space.





Modeling

Using embeddings as input to learn patterns and predict outcomes.



Preprocessing

Reverse Polish Notation:

$$\sin(x) + \frac{\cos(x)}{2x} - 17x + 4 \rightarrow x \ \sin \ x \ \cos x \, 2 \ * / + x \, 17 \ * -4 \ +$$

We want to process the data that comes to us as a flow of entities and we want to have a meaningfull structure.

Other examples:

"**bold**" → "bold”

"12,345.67" → "12345.67”

"hello@#" → "hello”

"Hello" → "hello"

Tokenization

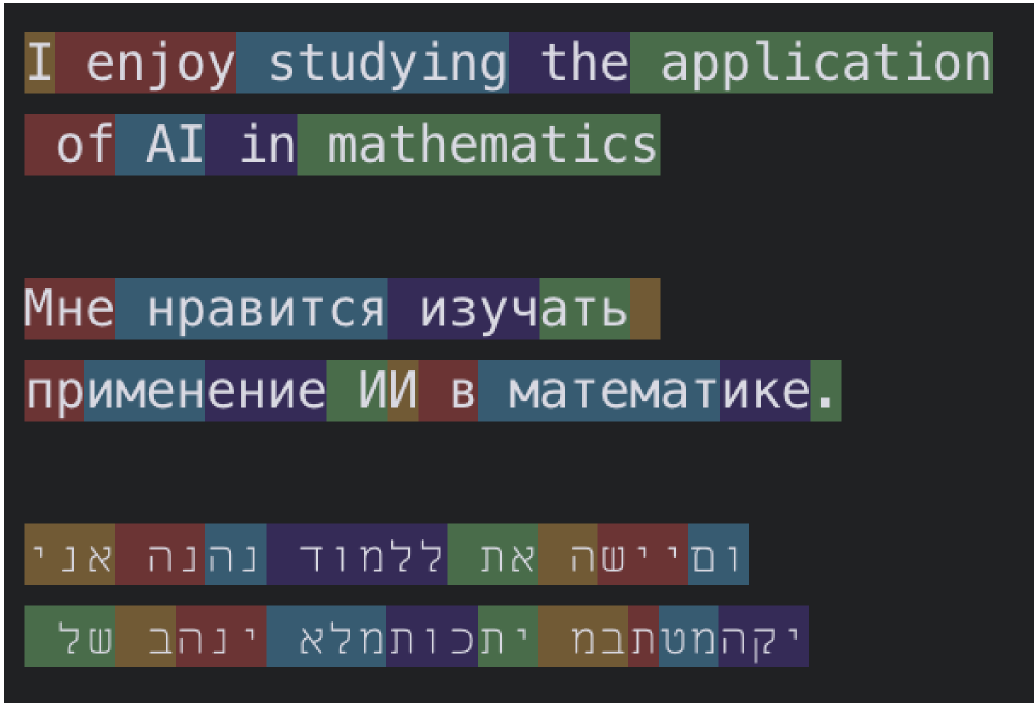
Rule based tokenization

Word tokenization:
I enjoy studying the application of AI in mathematics.

Letter tokenization:
I enjoy studying the application of AI in mathematics.

Grammar tokenization:
I enjoy studying the application of AI in mathematics.

Statistical methods



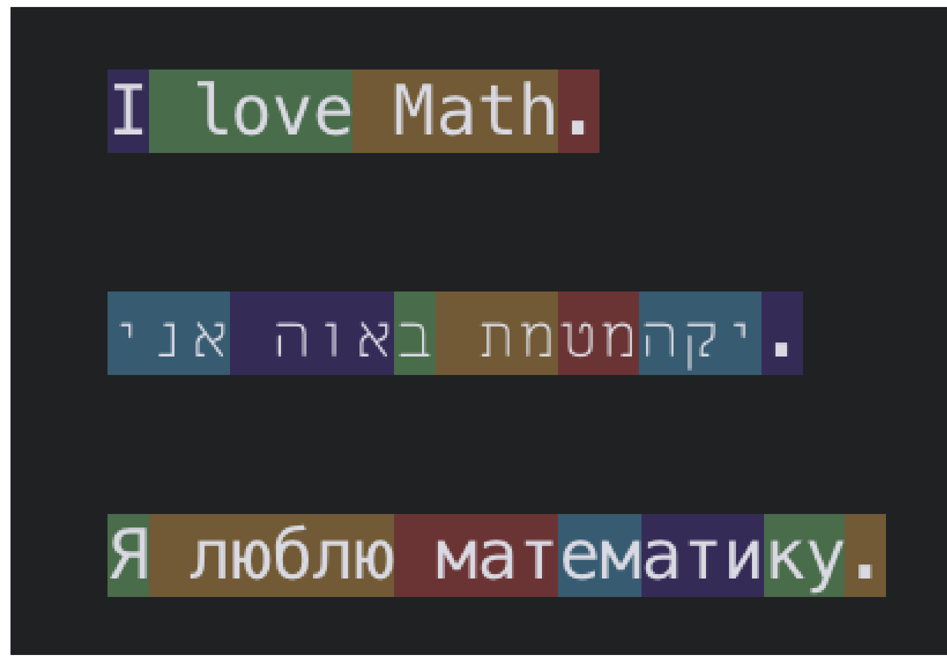
Byte Pair Encoding

aaabdaaabac → ZabdZabac → ZYdZYac → XdXac

Real life Chat GPT tokenizer:

English is better covered
Because it has more examples
in absolute values.

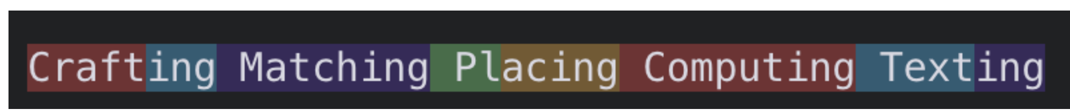
Hebrew tokens are placed in
the left to right direction as
Well because of how they are
processed.



Byte Pair Encoding

Something goes wrong with numbers
Why would we expect it?

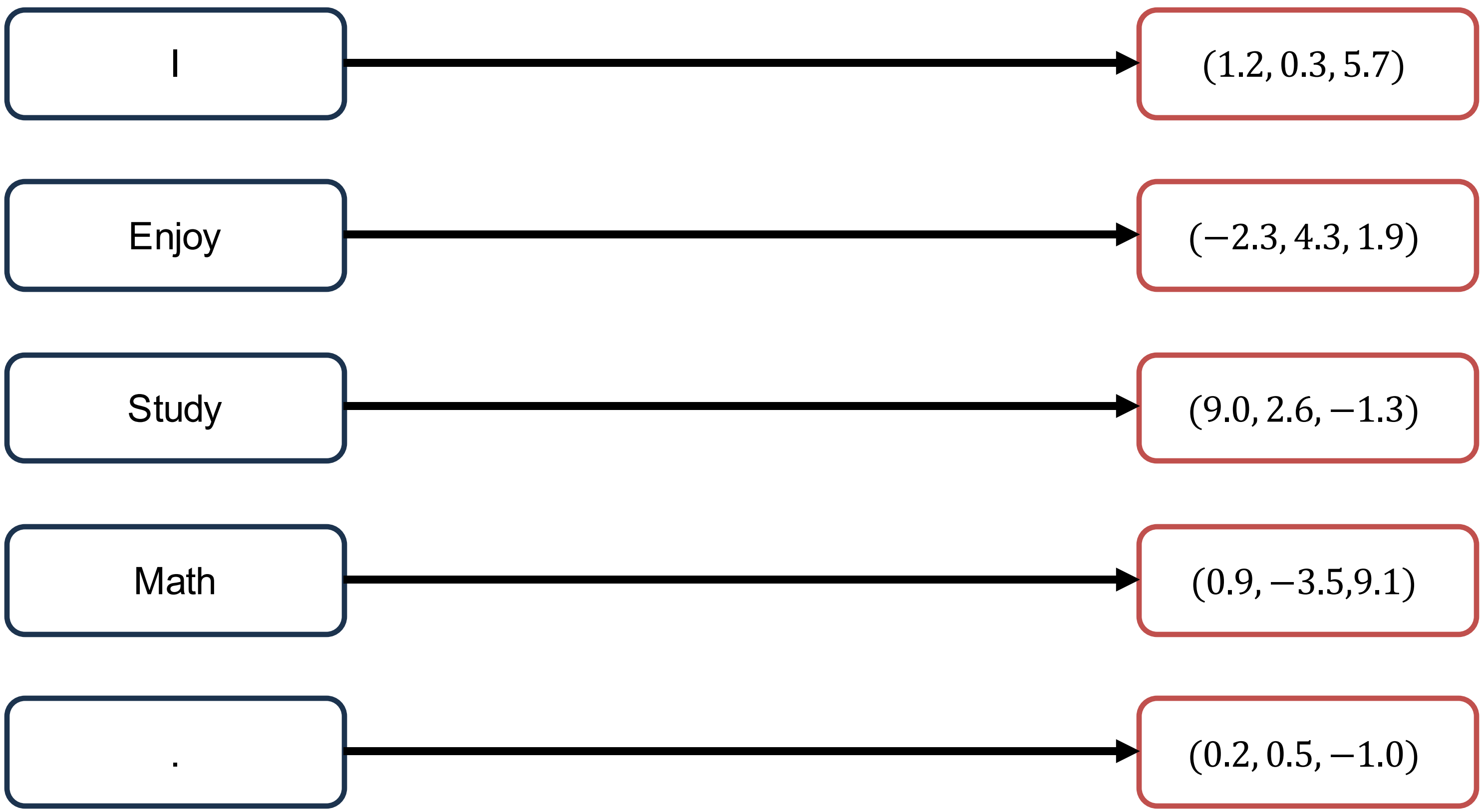
Sometimes it can detect morphemes, but generally it can do something strange



Embeddings

Embeddings that we want for machine learning algorithms:

$$f\colon \mathcal{X} \to \mathbb{R}^d$$



One Hot Encoding

id	token
13	I
24	enjoy
35	syudy
14	math
5	.

5	…	13	14	…	24	…	35	…
0		1	0		0		0	
0		0	0		1		0	
0		0	0		0		1	
0		0	1		0		0	
1		0	0		0		0	

What can be the problem here?

Embeddings

У лукоморья дуб зелёный;
Златая цепь на дубе том:
И днём, и ночью кот учёный
Всё ходит по цепи кругом;
Идёт направо – песнь заводит,
Налево – сказку говорит.

Embedding

У лукоморья дуб зелёный;
Златая цепь на дубе том:
И днём и ночью **кот учёный**
Всё ходит по цепи кругом;
Идёт направо — песнь заводит,
Налево — сказку говорит.

Chat GPT (Poetic)

אצל מפרץ, אלון ירוק עומד,
עליו שרשרת זהב מתפתלת;
וביום ובלילה — **חתול מדען**
סובב על השרשרת בלי הפסקת זמן;
פונה ימינה — שירו מזמר,
פונה שמאלה — סיפור הוא מספר.

By the bay a green oak rises high,
A golden chain upon it lies;
And day and night, **the Scientist Cat**
Paces the chain in a ceaseless track;
When he turns right — he sings a song,
When he turns left — he spins a tale along.

Google Translate

יש עץ אלון ירוק ליד לוקומוריה;
שרשרת זהב על עץ האלון ההוא:
החתול הוא אדם **מלומד** יומם ולילה
הכל מסתובב במעגל;
הוא הולך ימינה ומתחיל לשיר,
הוא מספר סיפור -משמאל.

By the seashore there is a green oak;
A golden chain on that oak:
Day and night **the learned cat**
Walks around the chain;
He goes to the right - he starts a song,
He goes to the left - he tells a story.

Chat GPT generated

Embedding

We would expect (hope):
מדען and מלומד(Scientist and learned) which have a similar meaning would correspond to simillar vectors.

But how do we or model can understand the fact that the words are simillar?

Answer: From a context, right?
I like drinking olut with friends in the bar.
Olut is made of wheat.

You got what olut is by a context and understanding surrounding words.
Let’s see how we can create an embedding which will capture context.

Word2Vec

Learn embeddings by predicting the context words given a target word (Skip-Gram) or vice versa (CBOW).

Given a sequence $W = (w_1, \ldots, w_T)$ we want to maximize the likelihood of context words appearing near the target central word:

$$\Pr(w_{t-i}, \ldots, w_{t-1}, \_\_, w_{t+1}, \ldots, w_{t+i} \mid w_t)$$

Word2Vec

And day and night, the **Scientist** Cat
Paces the chain in a ceaseless track;

And day and night, the Scientist Cat
Paces the chain in a ceaseless track;

And day and night, the Scientist Cat
Paces the chain in a ceaseless track;

Word2Vec

And day and night, the **Scientist** Cat
Paces the chain in a ceaseless track;

And day and night, the Scientist Cat
Paces the chain in a ceaseless track;

And day and night, the Scientist Cat
Paces the chain in a ceaseless track; <UNK> <UNK> <UNK>

Word2Vec

Learn embeddings by predicting the context words given a target word (Skip-Gram) or vice versa (CBOW).

Given a sequence $W = (w_1, \dots, w_T)$ – one hot encodings of tokens, maximize:

$$\prod_{t=1}^{T} \prod_{-c \leq j \leq c, j \neq 0} p(w_{t+j} \mid w_t)$$

where c is the context window.

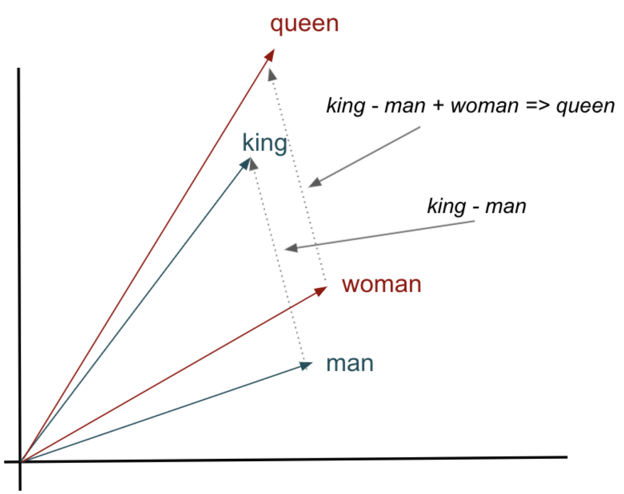
$$p(w_k \mid w_t) = \frac{\exp(A_{input}^T w_k \cdot A_{output}^T w_t)}{\sum_{w \in V} \exp(A_{input}^T w \cdot A_{output}^T w_t)}$$

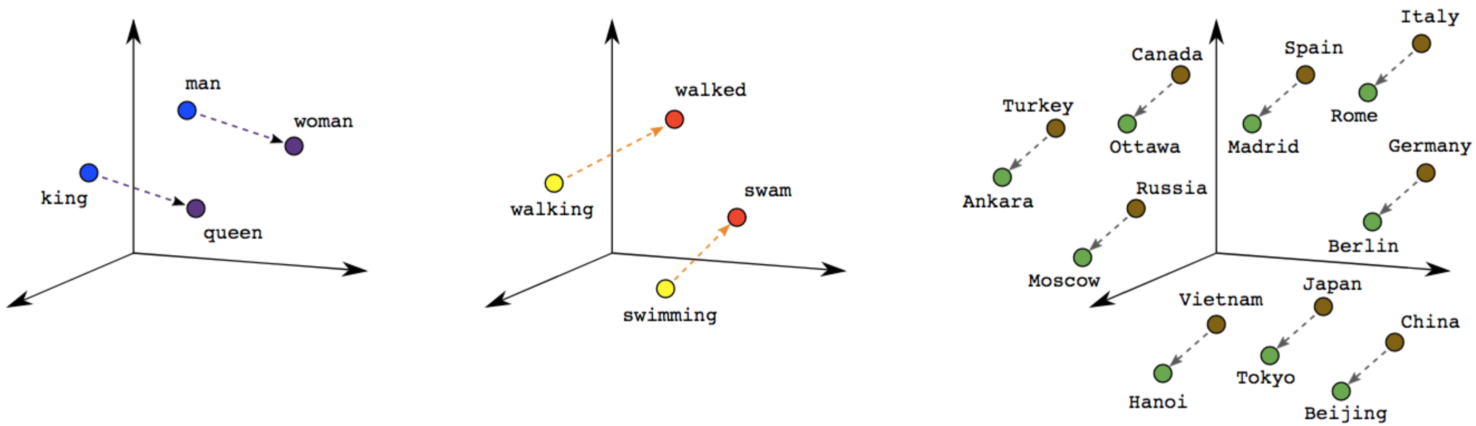
Word2Vec Arithmetics

Famous property:

$$king - man + woman \approx queen.$$

Nearby words by meaning become close vectors.





Male-Female Verb Tense Country-Capital

Word2Vec

Word2Vec (with some modifications) embeddings approximate Pointwise mutual information:

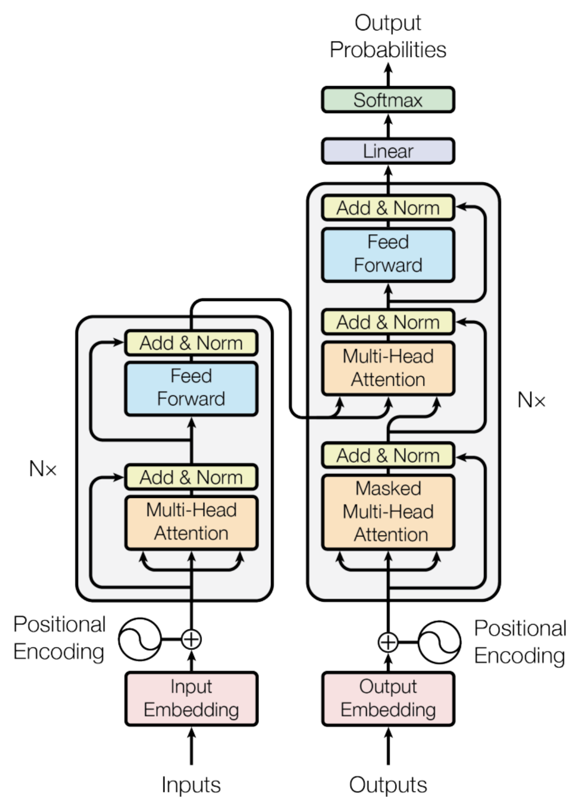
$$u_c^T v_w \sim PMI(w, c) = \log\left(\frac{N(w, c)N}{N(w)N(c)}\right)$$

This explains why similar words (that co-occur in similar contexts) end up **close together** in embedding space.

Modeling

BERT

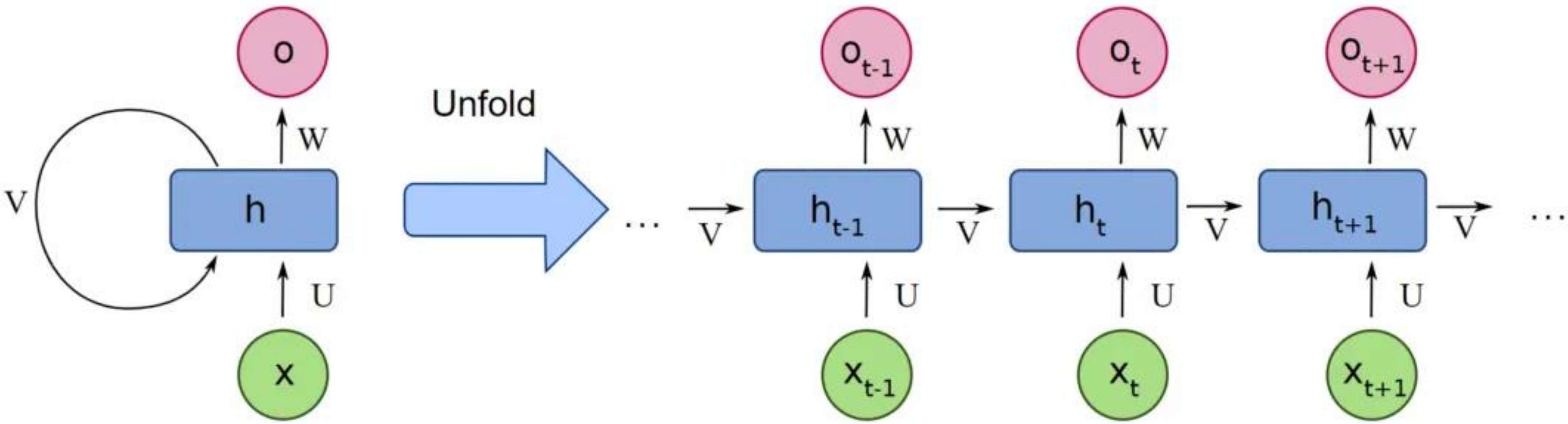
Encoder



GPT

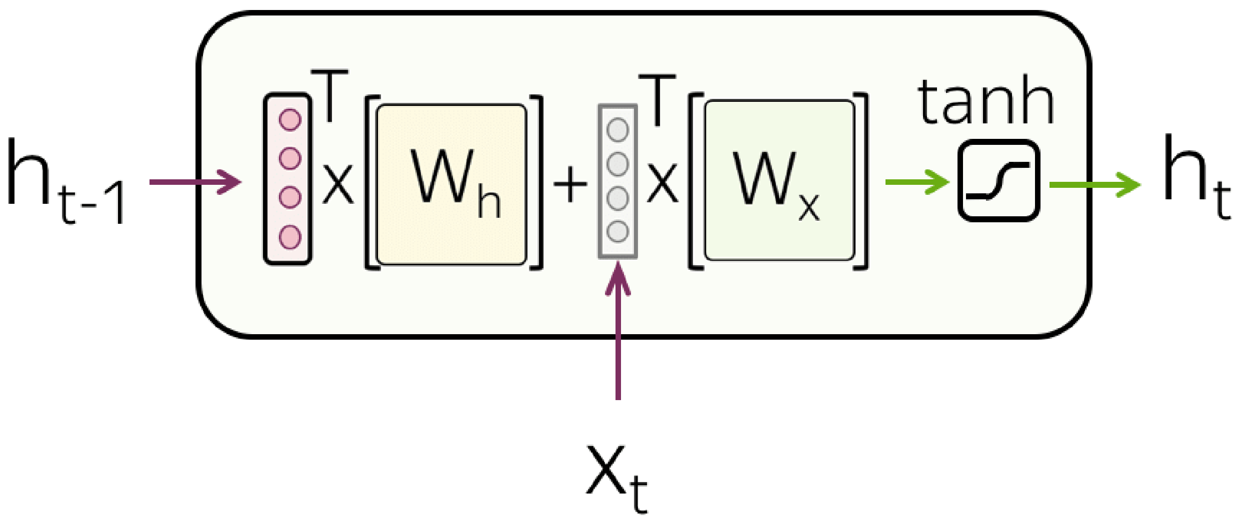
Decoder

Recurent Neural Network



Vanilla RNN

$$h_t = \tanh(h_{t-1} W_h + x_t W_x)$$



Pictures here and further from [NLP Course by Lena Voita]

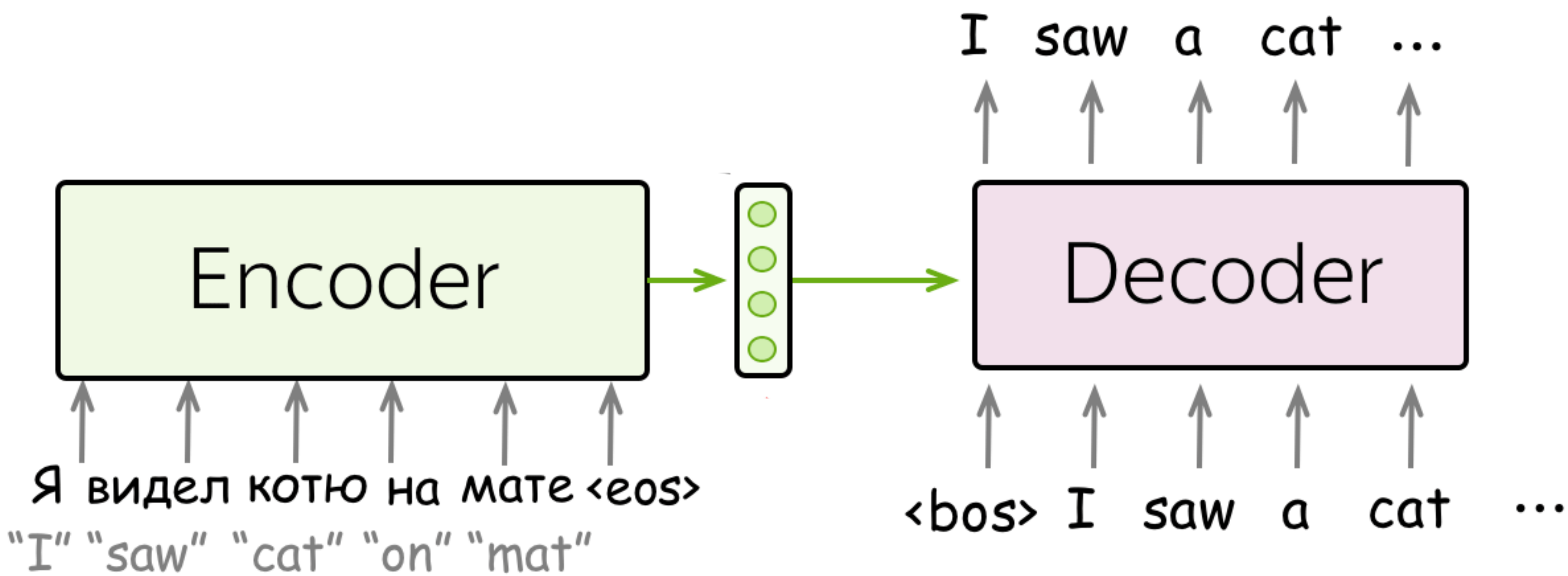
Loss function

- Model predicts sequence of distributions $(p_1, \ldots, p_T)$, where each p_t is a probability vector over vocabulary at time t,
- Ground-truth tokens are $(y_1, \ldots, y_T)$, where each y_t is an integer (index in vocabulary).

Then the **loss** is:

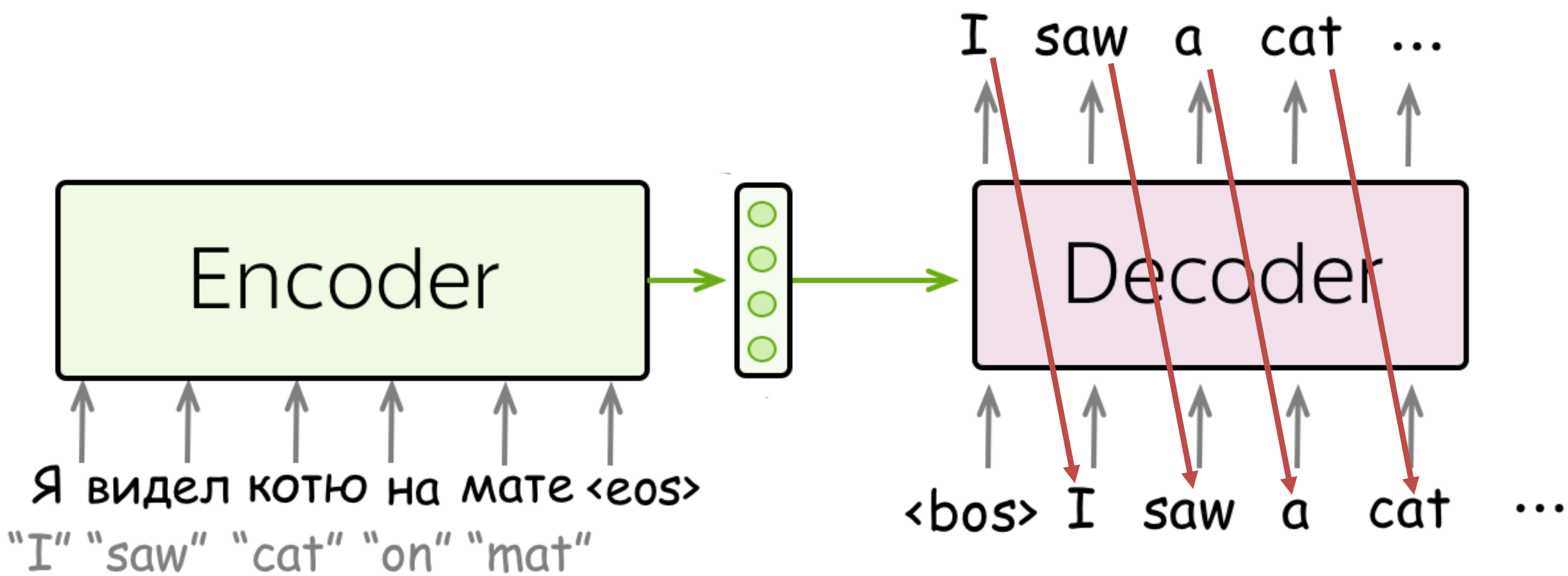
$$\mathcal{L} = -\sum_{t=1}^{T} \log p_t(y_t)$$

Seq2Seq



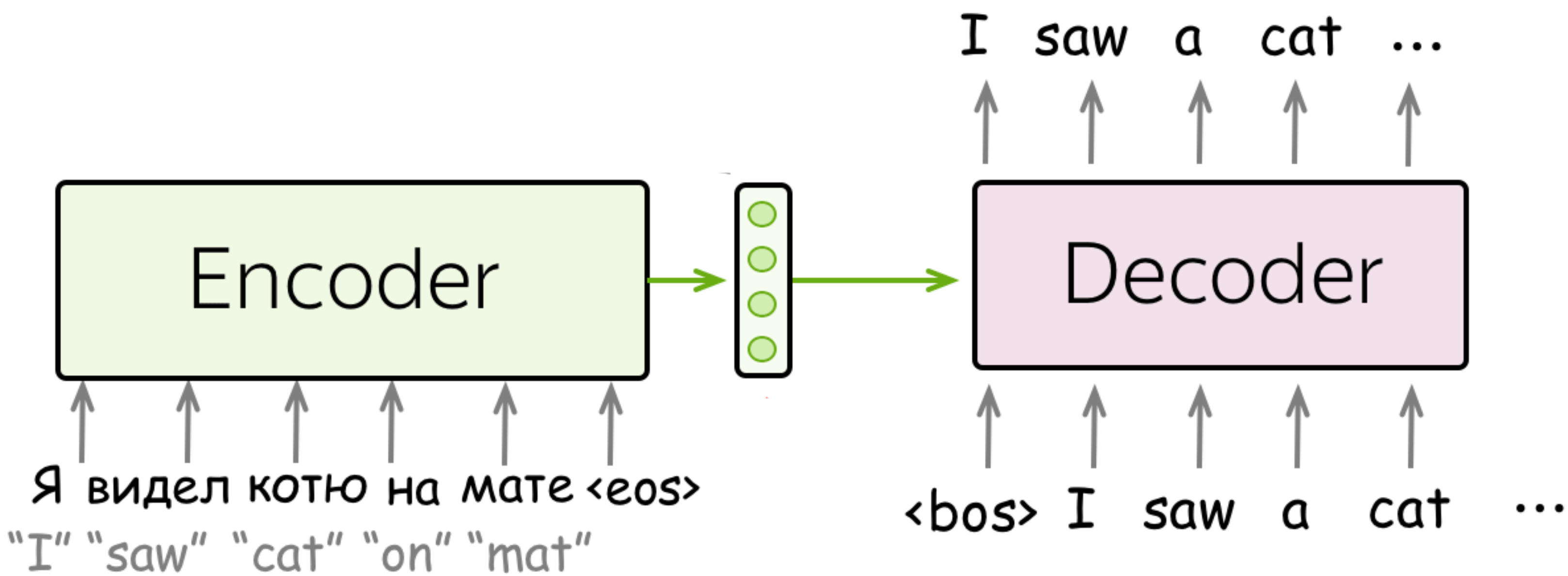
I saw a cat ...
Encoder
Decoder
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
<bos> I saw a cat ...

Seq2Seq



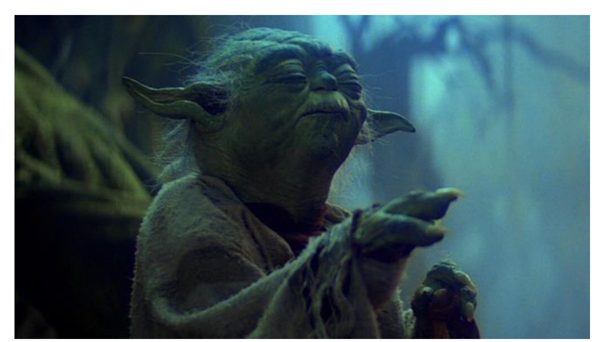
I saw a cat ...
Encoder
Decoder
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
<bos> I saw a cat ...

Seq2Seq

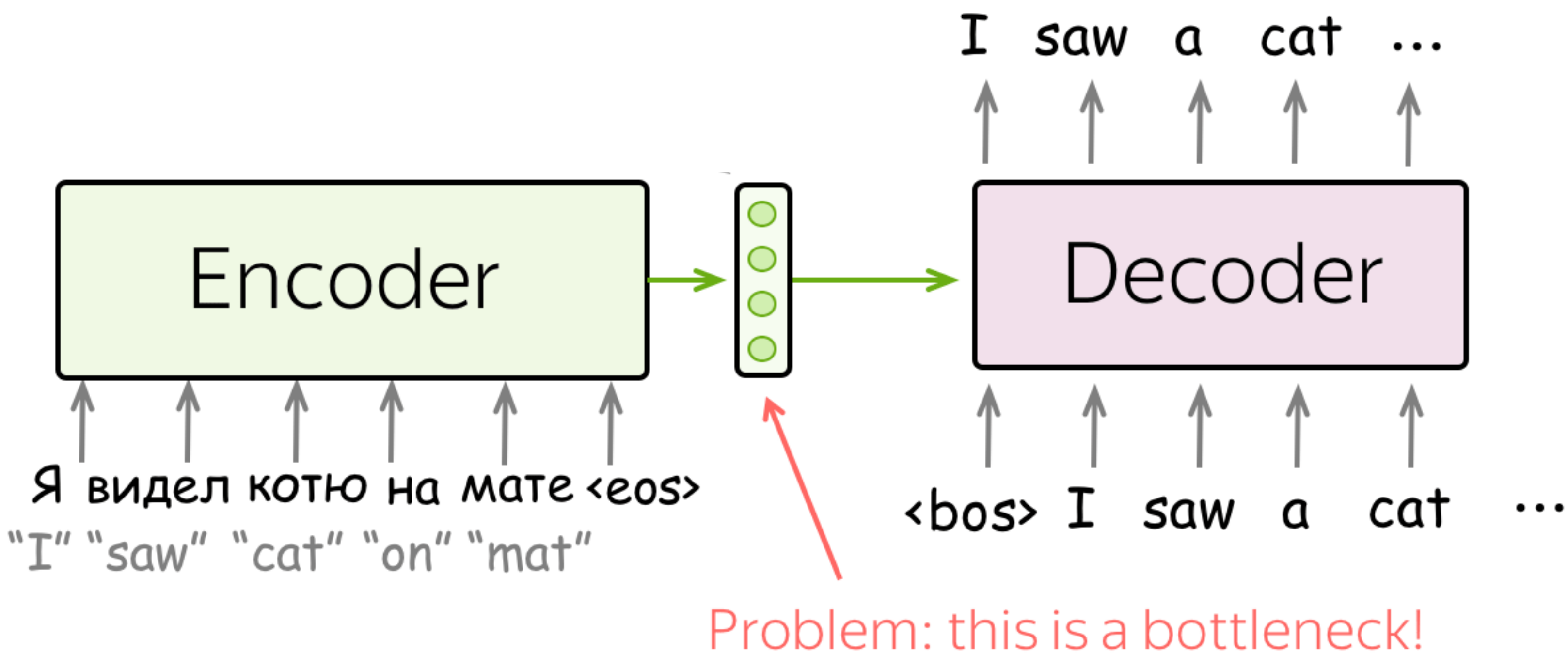


Teacher forcing during train:

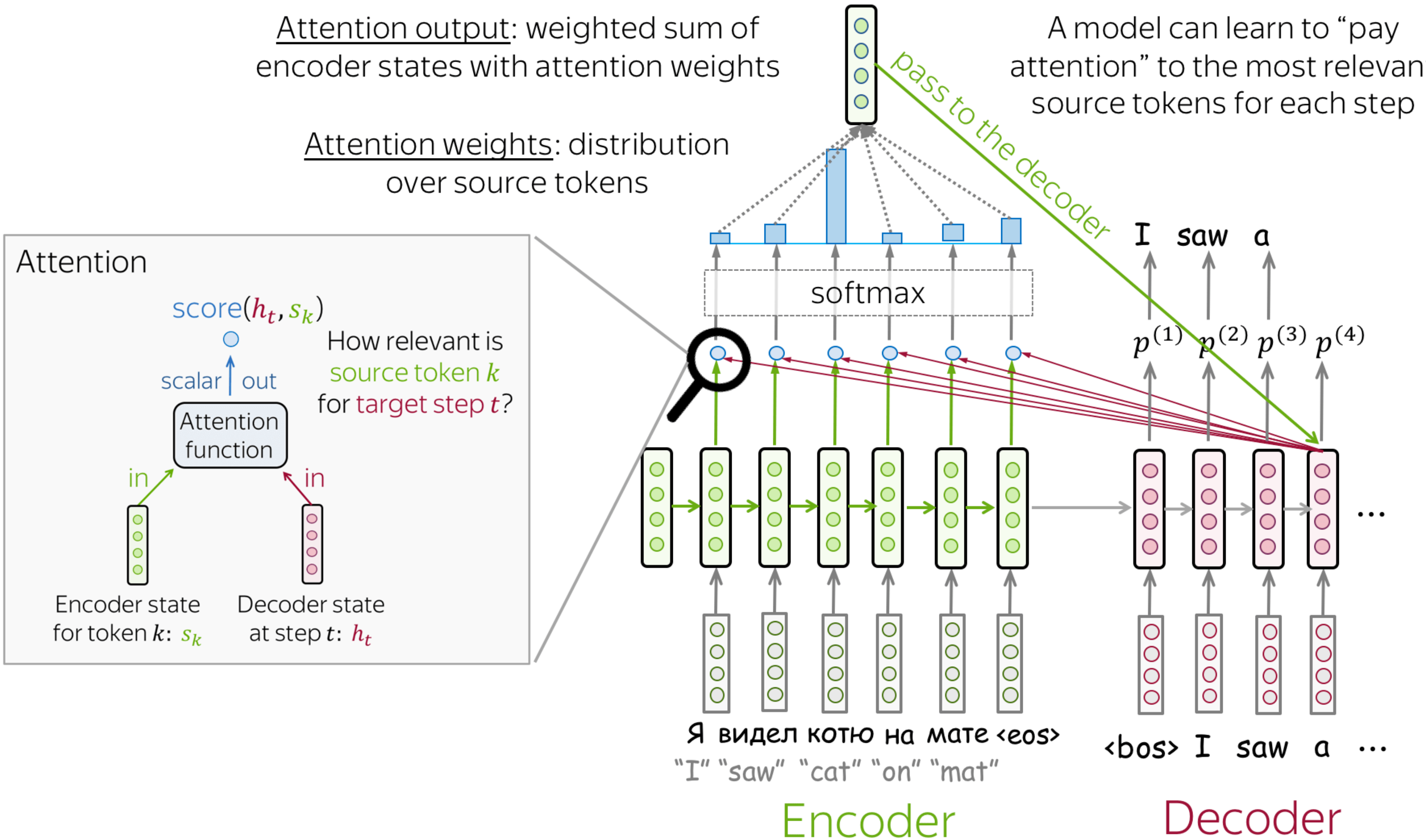
We provide ground-truth tokens on a train.



Seq2Seq


I saw a cat ...
Encoder
Decoder
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
<bos> I saw a cat ...
Problem: this is a bottleneck!

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
softmax
I saw a
$p^{(1)}$ $p^{(2)}$ $p^{(3)}$ $p^{(4)}$
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
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
<bos> I saw a …
…
Encoder
Decoder