Language modeling Seq2seq

NLP

lecturer: Mollaev D. E.
Sber AI Lab

Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Evaluation LMs
- Seq2seq framework
- Attention

Lecture plan

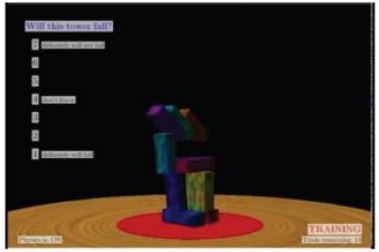
- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Train Models

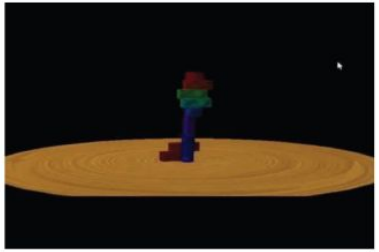
- have some properties of trains (look like ones)
- can behave similarly to trains
- good models have more of the above

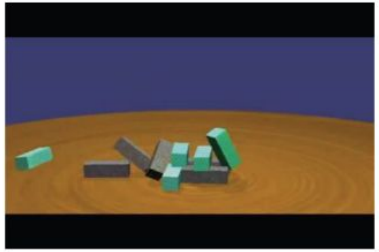
Models of Physical World

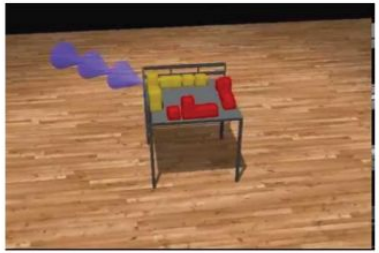
- understand which events are in better agreement with the world, which are more likely
- can predict what happens given some “context”



Will it fall?


In which direction?

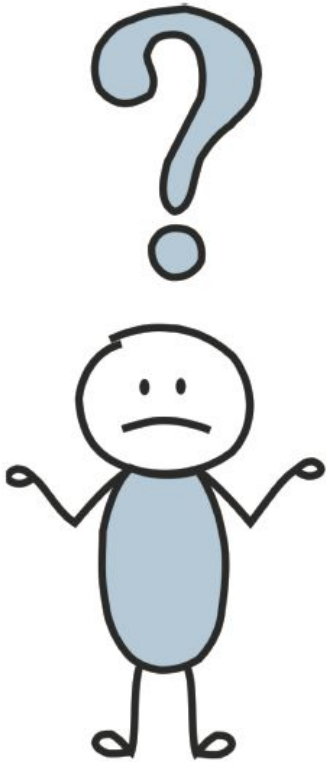

Different masses


Complex scenes


Infer the mass


Predict fluids

Language Models



Language Models

The intuition is exactly the same!

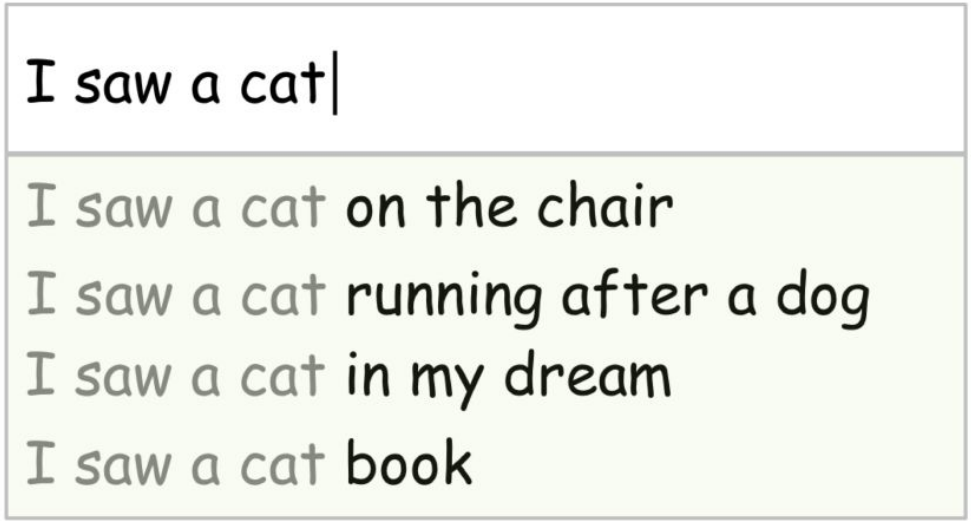
What is different, is the notion of an event: for language, an event is a linguistic unit (text, sentence, token, symbol).

> Language Models (LMs) estimate the probability of different linguistic units: symbols, tokens, token sequences.

We do we need it?

We deal with Language Models every day!

Web search engine / ...



We do we need it?

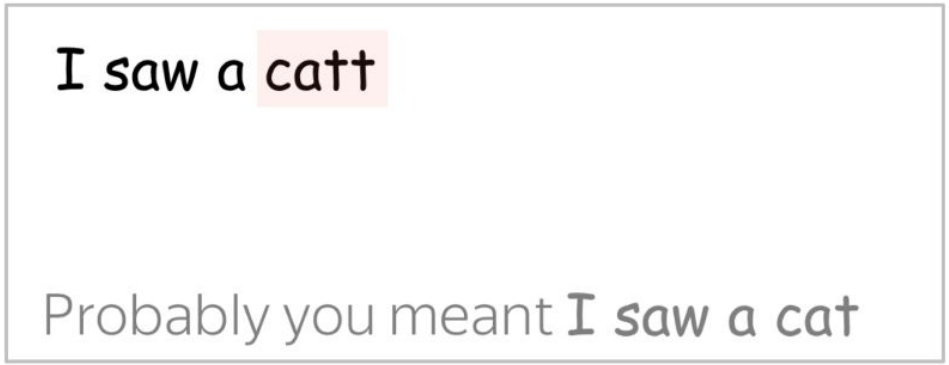
We deal with Language Models every day!

Translation service / mail agent / ...

We do we need it?

We deal with Language Models every day!

Translation service / mail agent / ...

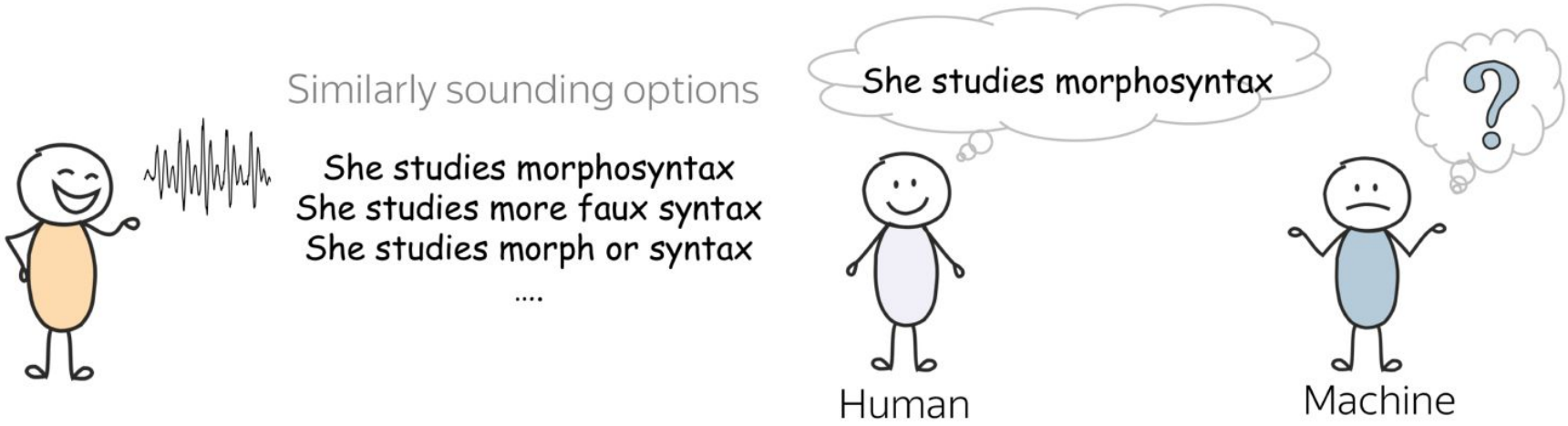


We do we need it?

We deal with Language Models every day!

Keyboard / mail agent / ...

Ambiguity



The **morphosyntax** example is from the slides by Alex Lascarides and Sharon Goldwater, Foundations of Natural Language Processing course at the University of Edinburgh.

Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Modeling

What is the probability to pick a green ball?

Modeling

What is the probability to pick a green ball?

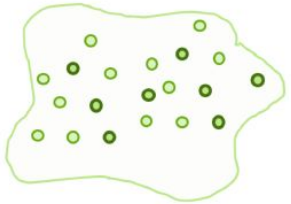
$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

Modeling

What is the probability to pick a green ball?

$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

Can we do the same for sentences?



Text corpus

$$P(\text{the mut is tinming the tebn}) = \frac{0}{|\text{corpus}|} = 0$$

$$P(\text{mut the tinming tebn is the}) = \frac{0}{|\text{corpus}|} = 0$$

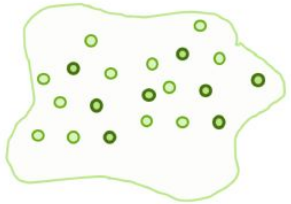
With this approach, sentences that never occurred in the corpus will receive zero probability

Modeling

What is the probability to pick a green ball?

$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

Can we do the same for sentences?



Text corpus

$$P(\text{the mut is tinming the tebn}) = \frac{0}{|\text{corpus}|} = 0$$

$$P(\text{mut the tinming tebn is the}) = \frac{0}{|\text{corpus}|} = 0$$

With this approach, sentences that never occurred in the corpus will receive zero probability

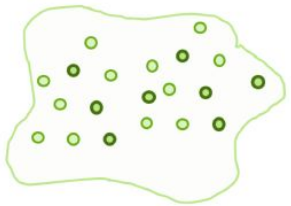
But the first sentence is “more likely” than the second!
This method is not good!

Modeling

What is the probability to pick a green ball?

$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

Can we do the same for sentences?



Text corpus

$$P(\text{the mut is tinming the tebn}) = \frac{0}{|\text{corpus}|} = 0$$

$$P(\text{mut the tinming tebn is the}) = \frac{0}{|\text{corpus}|} = 0$$

With this approach, sentences that never occurred in the corpus will receive zero probability

But the first sentence is “more likely” than the second! This method is not good!

We can not estimate sentence probabilities reliably if we treat them as atomic units!

Modeling

Image we

- read the sentence **I saw a cat on a mat** word by word,
- update probability every time we see a new token

$P(\mathbf{I}) =$

$P(\mathbf{I})$

Probability of **I**

Modeling

Formally,

- $(y_1, y_2, \ldots, y_n)$ is a sequence of tokens,
- $P(y_1, y_2, \ldots, y_n)$ - probability to see these tokens (in this order)

Using the product rule of probability (aka “chain rule”), we get:

$$P(y_1, y_2, \ldots, y_n) = P(y_1) \cdot P(y_2|y_1) \cdot P(y_3|y_1, y_2) \cdot \cdots \cdot P(y_n|y_1, \ldots, y_{n-1}) = \prod_{t=1}^{n} P(y_t|y_{<t})$$

Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Neural LMs

$$P(y_1, y_2, \ldots, y_n) = P(y_1) \cdot P(y_2|y_1) \cdot P(y_3|y_1, y_2) \cdot \cdots \cdot P(y_n|y_1, \ldots, y_{n-1}) = \prod_{t=1}^{n} P(y_t|y_{<t}).$$

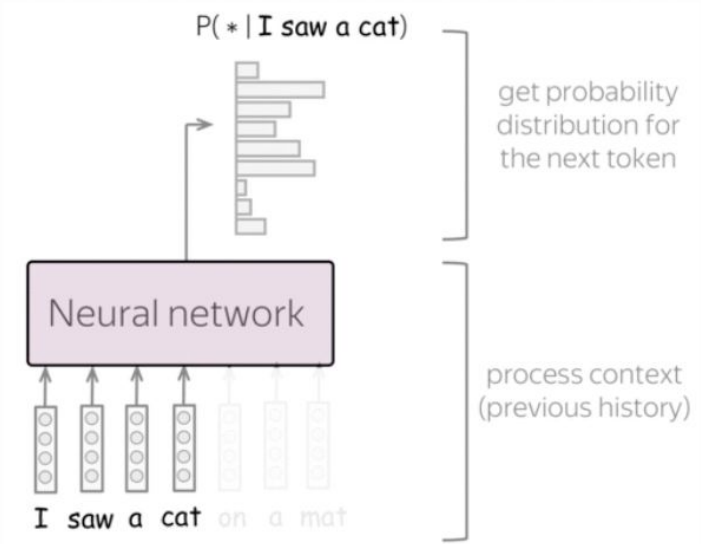
We need to: define how to compute the conditional probabilities $P(y_t|y_1, \ldots, y_{t-1})$.

How: Train a neural network to predict them.

Neural LMs

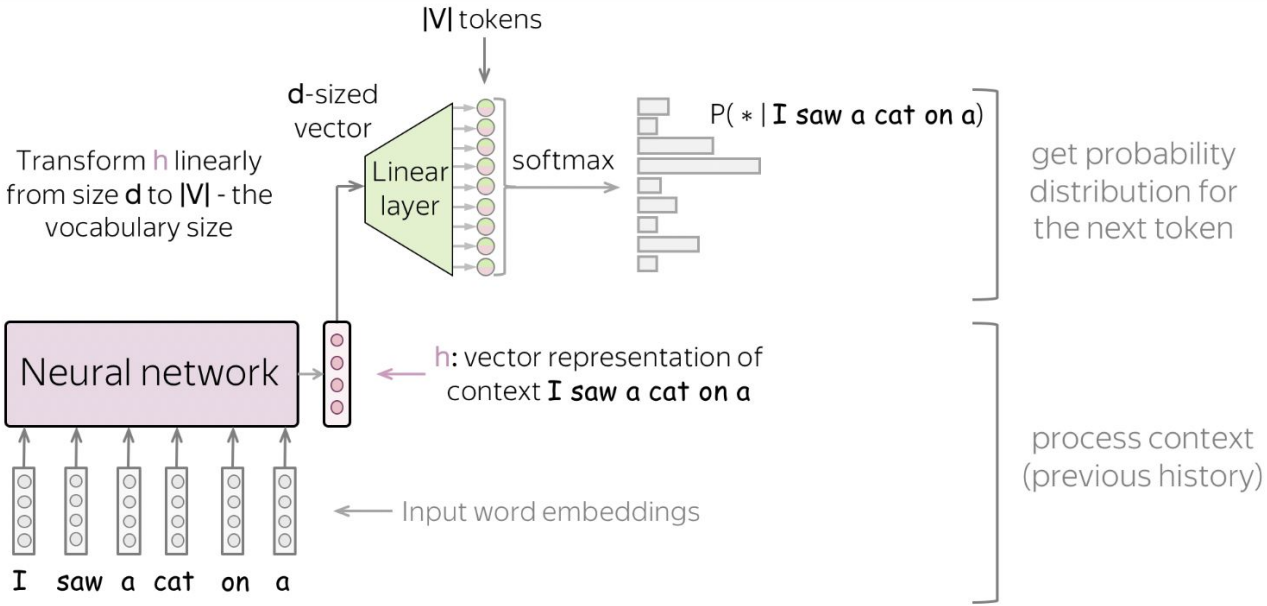
Intuitively, neural Language Models do two things:

- **process context → model-specific**
  The main idea here is to get a vector representation for the previous context. Using this representation, a model predicts a probability distribution for the next token. This part could be different depending on model architecture (e.g., RNN, CNN, whatever you want), but the main point is the same - to encode context.
- **generate a probability distribution for the next token → model-agnostic**
  Once a context has been encoded, usually the probability distribution is generated in the same way - see below.



Neural LMs: High-level pipeline

- feed word embedding for previous (context) words into a network;
- get vector representation of context from the network;
- from this vector representation, predict a probability distribution for the next token.



Neural LMs: Training

Neural LMs: Training

Formally, let's assume we have a training instance with the source $x = (x_1, \ldots, x_m)$ and the target $y = (y_1, \ldots, y_n)$. Then at the timestep t, a model predicts a probability distribution $p^{(t)} = p(*|y_1, \ldots, y_{t-1}, x_1, \ldots, x_m)$. The target at this step is $p^* = \textbf{one-hot}(y_t)$, i.e., we want a model to assign probability 1 to the correct token, y_t, and zero to the rest.

The standard loss function is the cross-entropy loss. Cross-entropy loss for the target distribution p^* and the predicted distribution p is

$$Loss(p^*, p) = -p^* \log(p) = -\sum_{i=1}^{|V|} p_i^* \log(p_i).$$

Since only one of p_i^* is non-zero (for the correct token y_t), we will get

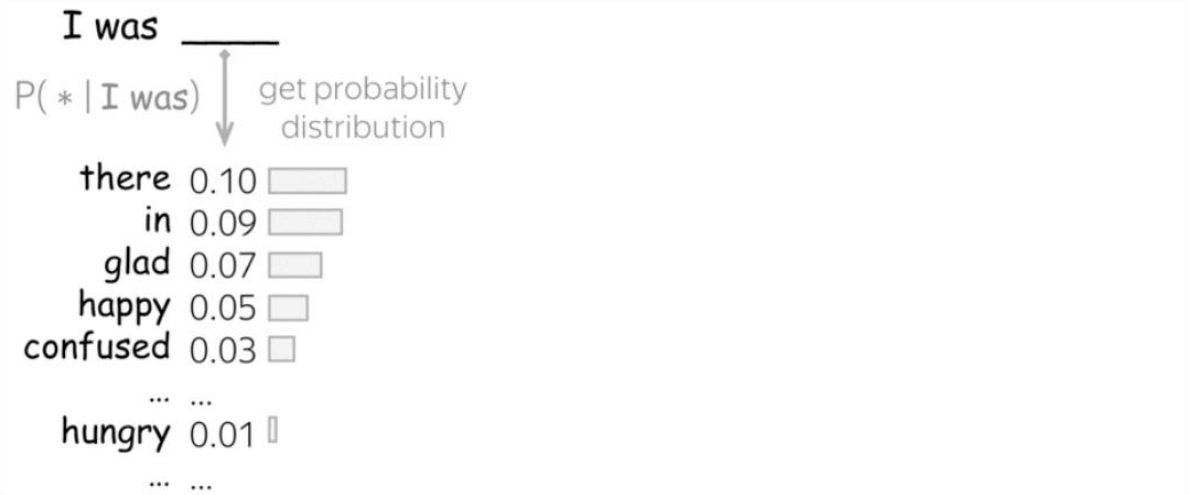
$$Loss(p^*, p) = -\log(p_{y_t}) = -\log(p(y_t|y_{<t}, x)).$$

Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Generating strategy

As we saw before, to generate a text using a language model you just need to sample tokens from the probability distribution predicted by a model.



Generating strategy

Coherence and Diversity

You can modify the distributions predicted by a model in different ways to generate texts with some properties. While the specific desired text properties may depend on the task you care about (as always), usually you would want the generated texts to be:

- **coherent** - the generated text has to make sense;
- **diverse** - the model has to be able to produce very different samples.

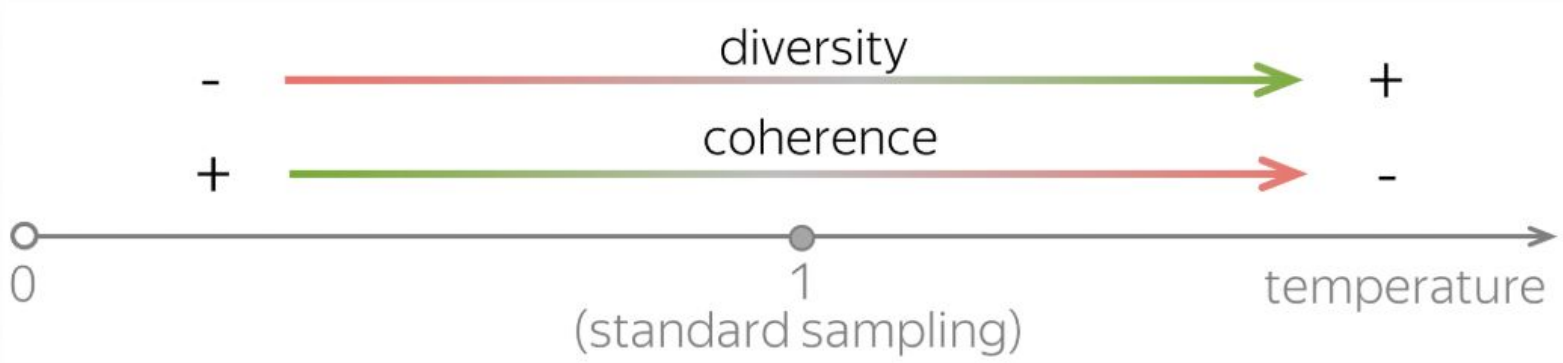
Generating strategy

- Standard Sampling
- Sampling with temperature
  - Top-k sampling
  - Nucleus Sampling

Sampling with temperature

$$\frac{\exp(h^T w)}{\sum_{w_i \in V} \exp(h^T w_i)} \rightarrow \frac{\exp\left(\frac{h^T w}{\tau}\right)}{\sum_{w_i \in V} \exp\left(\frac{h^T w_i}{\tau}\right)}$$

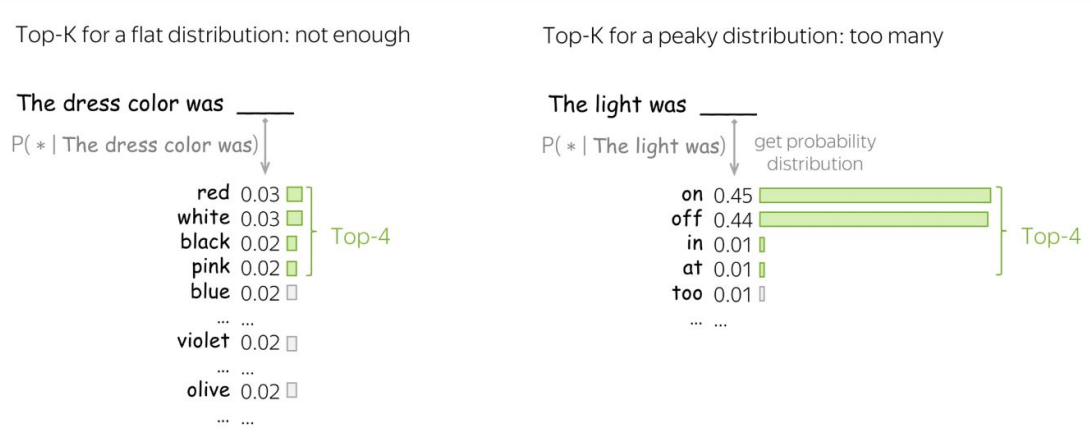
τ - softmax temperature



Top-k sampling

Fixed K is not always good

While usually top-K sampling is much more effective than changing the softmax temperature alone, the fixed value of K is surely not optimal. Look at the illustration below.



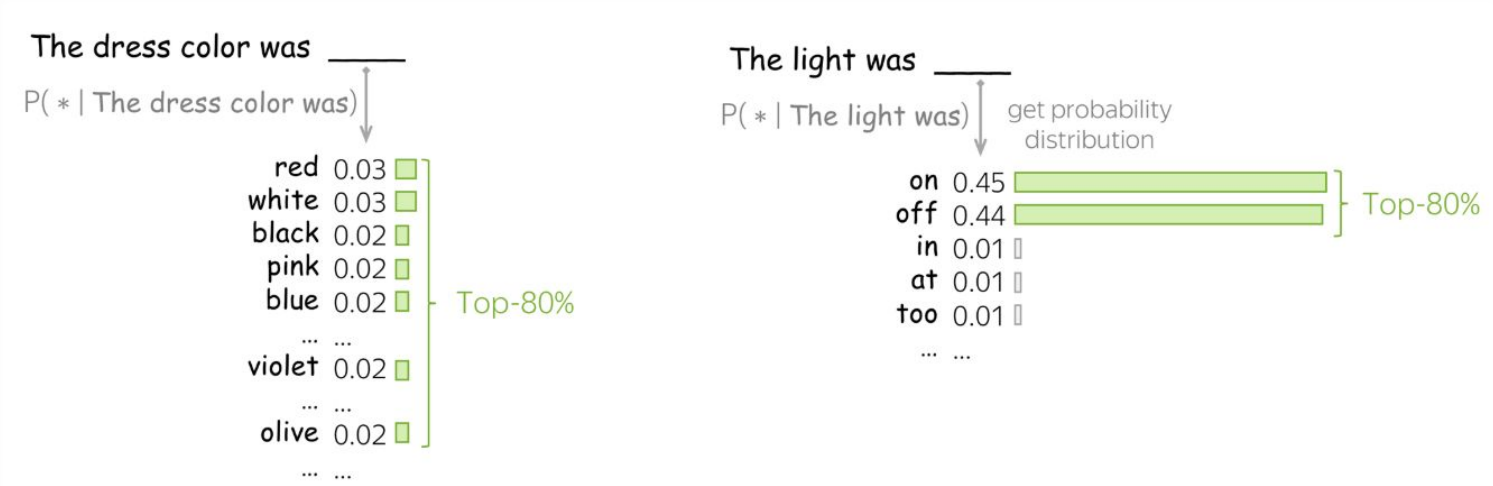
The fixed value of K in the top-K sampling is not good because top-K most probable tokens may

- cover very small part of the total probability mass (in flat distributions);
- contain very unlikely tokens (in peaky distributions).

Nucleus Sampling

A more reasonable strategy is to consider not top-K most probable tokens, but top-p% of the probability mass: this solution is called Nucleus sampling.

Look at the illustration: with nucleus sampling, the number of tokens we sample from is dynamic and depends on the properties of the distribution.



Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Evaluation LMs
- Seq2seq framework
- Attention

Perplexity and Text Log- Likelihood

$$L(y_{1:M}) = L(y_1, y_2, \dots, y_M) = \sum_{t=1}^{M} \log_2 p(y_t|y_{<t})$$

Log-likelihood of the text

$$Loss(y_{1:M}) = -\sum_{t=1}^{M} \log p(y_t|y_{<t})$$

Note: cross-entropy (our loss) is negative log-likelihood

It is more common to report its transformation called perplexity:

$$Perplexity(y_{1:M}) = 2^{-\frac{1}{M}L(y_{1:M})}$$

Low perplexity -> high data likelihood -> good!

Perplexity: Which Values Does it Take?

$$Perplexity(y_{1:M}) = 2^{-\frac{1}{M}L(y_{1:M})}$$

- The **best** perplexity is 1

  If the model is perfect and assigns probability 1 to correct tokens, then the log-probabilities are zero

- The **worst** perplexity is |V|

  If the model knows nothing about the data, it assigns probability 1/|V| to all tokens, regardless of context. Then:

$$Perplexity(y_{1:M}) = 2^{-\frac{1}{M}L(y_{1:M})} = 2^{-\frac{1}{M}\sum_{t=1}^{M}\log_2 p(y_t|y_{1:t-1})} = 2^{-\frac{1}{M}\cdot M\cdot\log_2\frac{1}{|V|}} = 2^{\log_2|V|} = |V|$$

Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Machine Translation

- Translation between natural languages

- More generally, translation between any sequences

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

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

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

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?

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?

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?

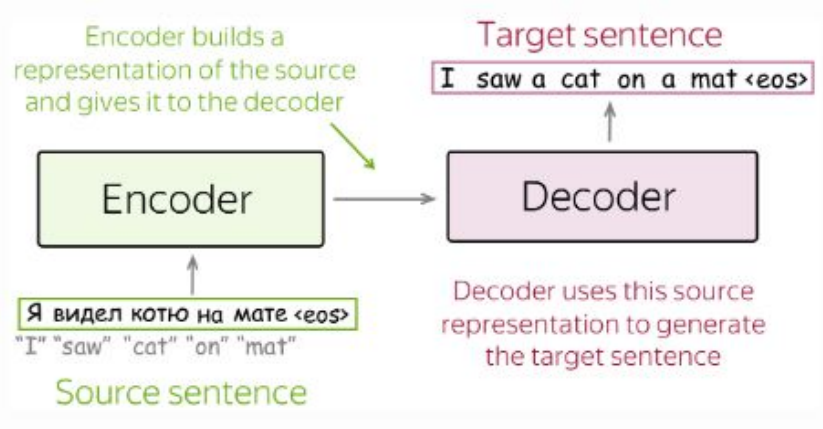
- **search**

  How to find the argmax?

Encoder-Decoder Framework

The standard modeling paradigm:

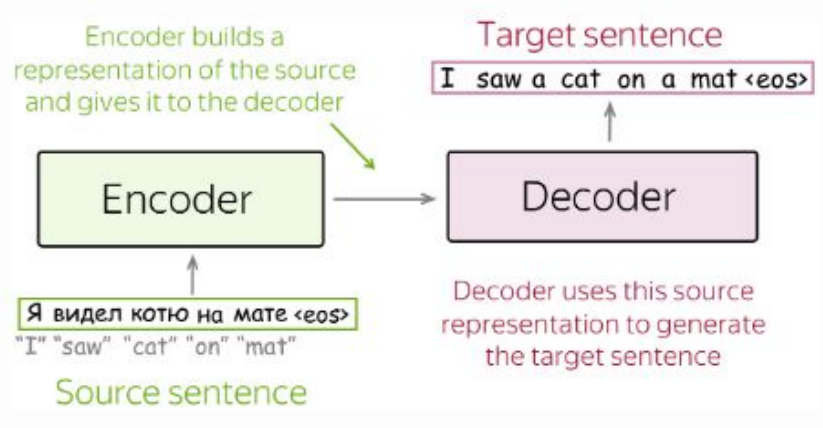
- Encoder – reads the source sentence and produces its representation



Encoder-Decoder Framework

The standard modeling paradigm:

- Encoder – reads the source sentence and produces its representation
- Decoder - uses source representation from the encoder to generate the target sequence.



Conditional Language Models

Language Models: (left-to-right)

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

Conditional Language Models

Language Models: (left-to-right)

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

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

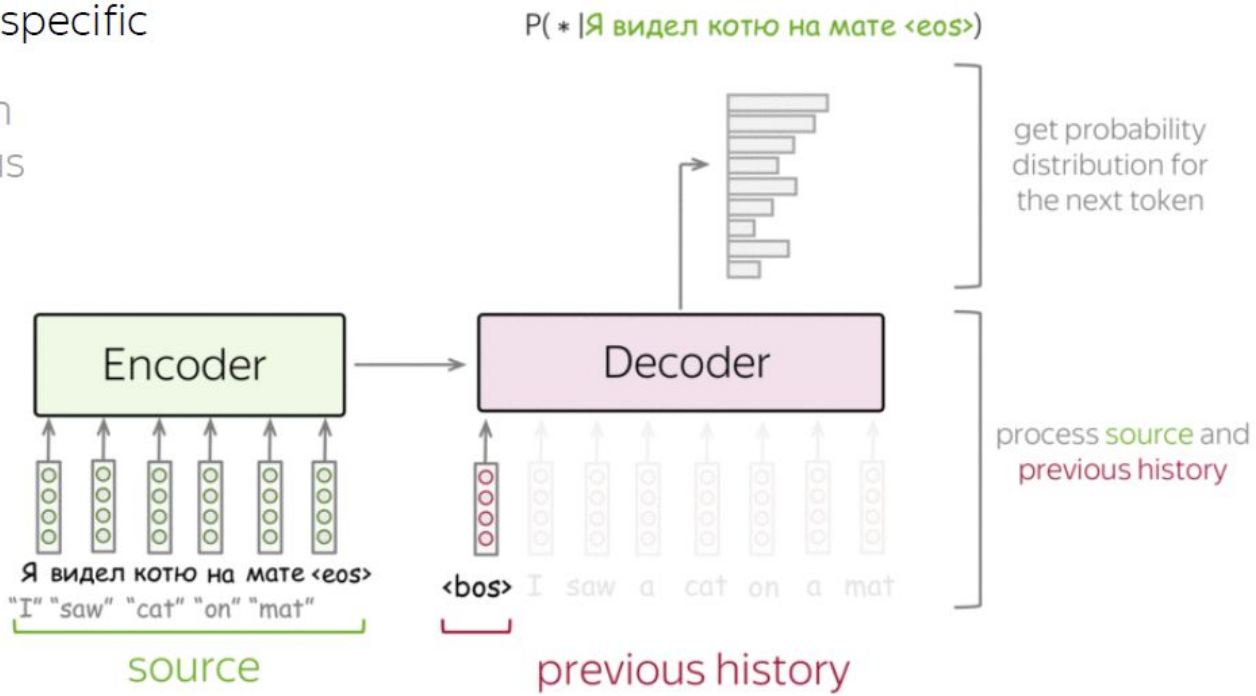
General View

- process context – model-specific

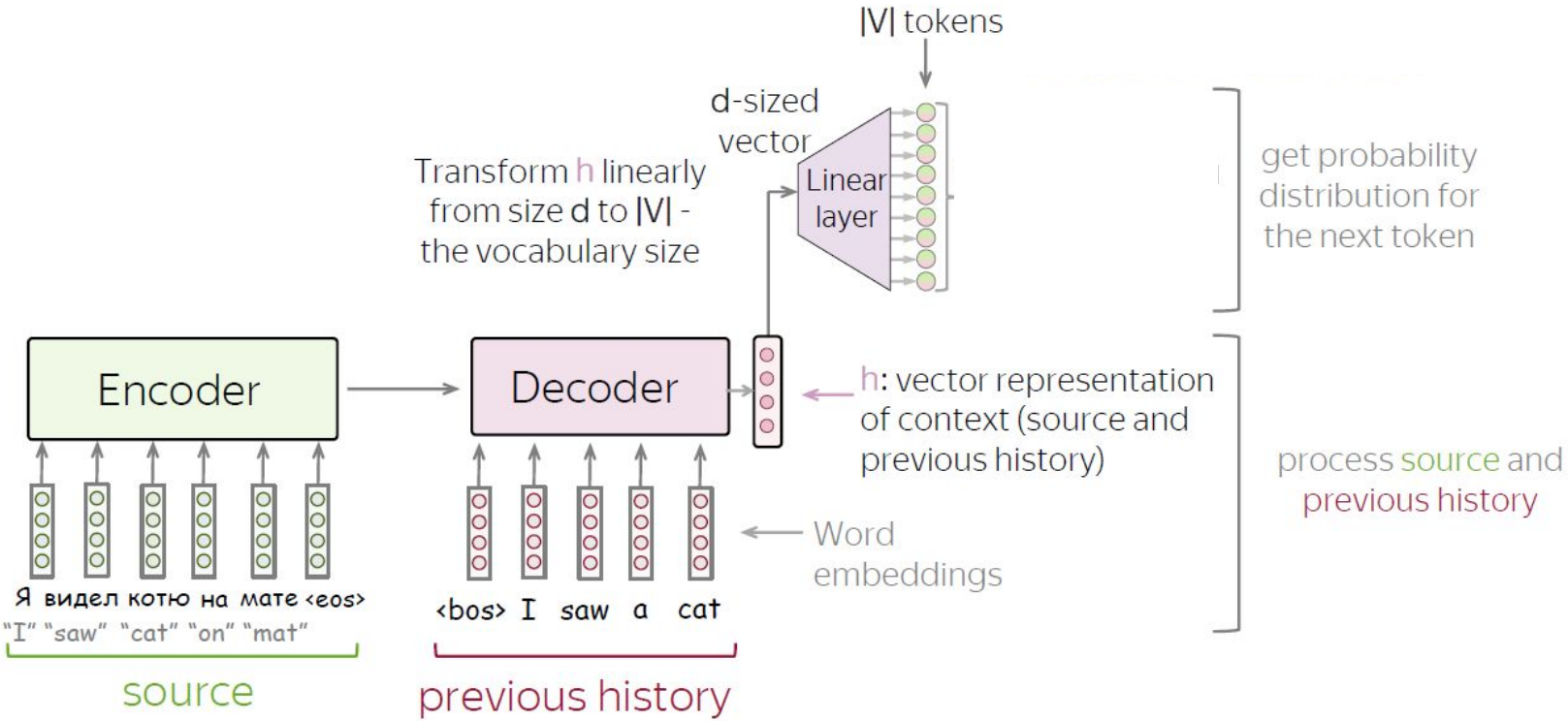
  Get vector representation of the source and previous target tokens

- evaluate probabilities – model-agnostic

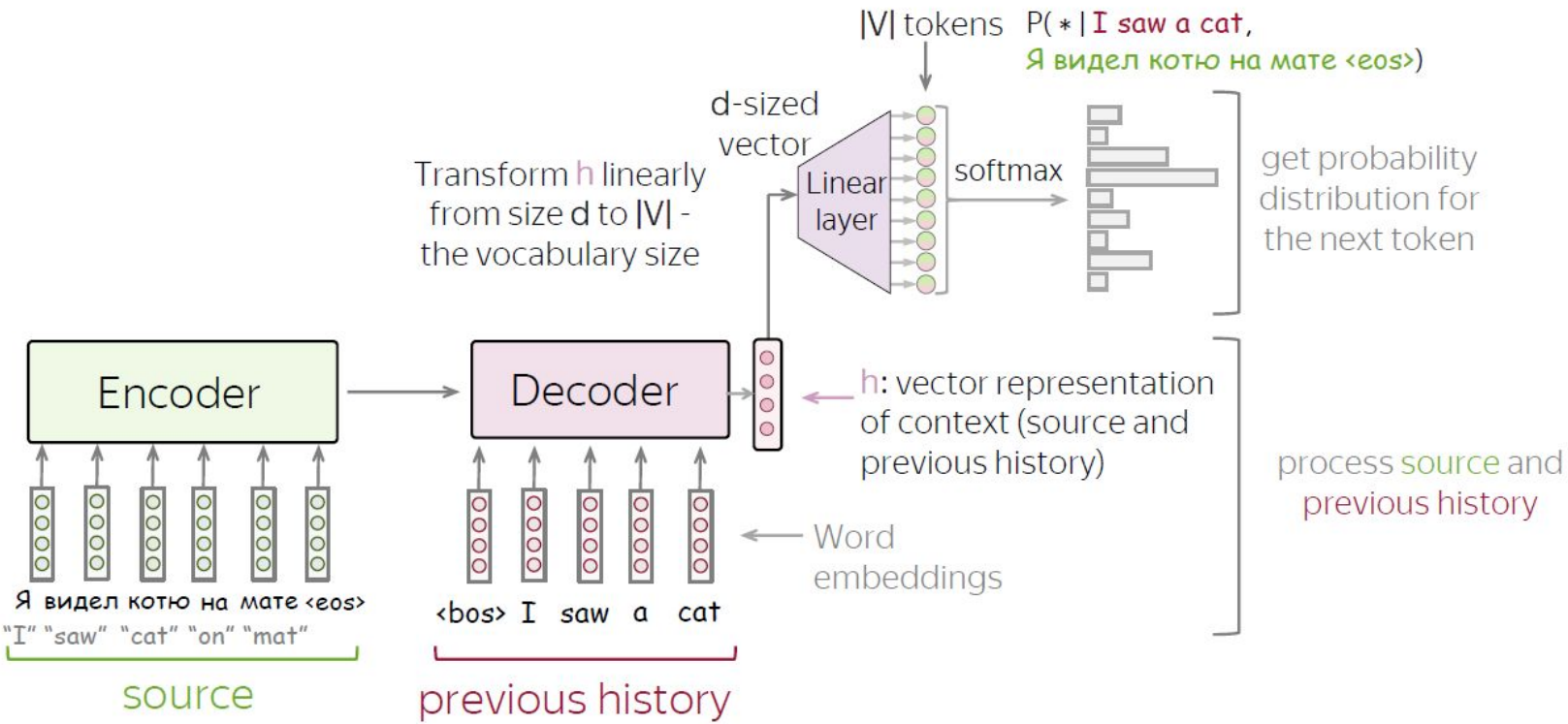
  Predict probability distribution for the next target token



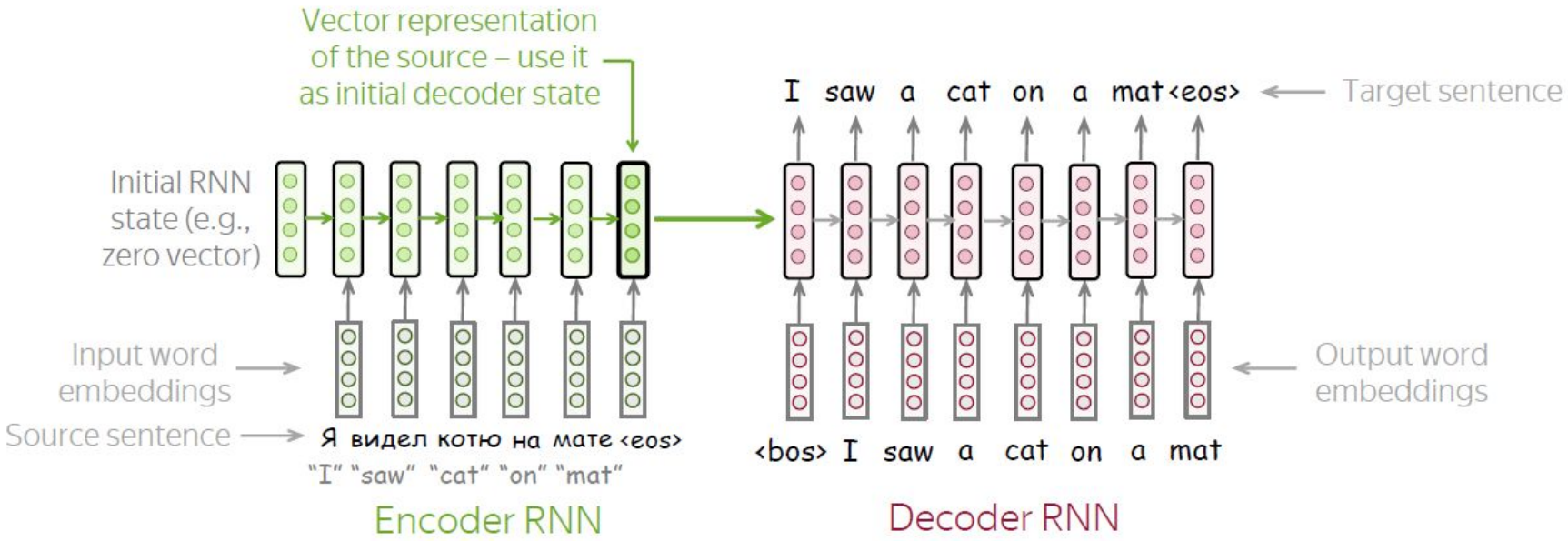
High-Level Pipeline



High-Level Pipeline



Two RNN Model



Generating

$$y' = \arg\max_y p(y|x) = \arg\max_y \prod_{t=1} p(y_t|y_{<t}, x)$$

Generating. Greed

$$y' = \arg\max_y p(y|x) = \arg\max_y \prod_{t=1} p(y_t|y_{<t}, x)$$

Straightforward:

- **greedy** - at each step, pick token with the highest probability

Generating. Greed Bad?

$$y' = \arg\max_y p(y|x) = \arg\max_y \prod_{t=1} p(y_t|y_{<t}, x)$$

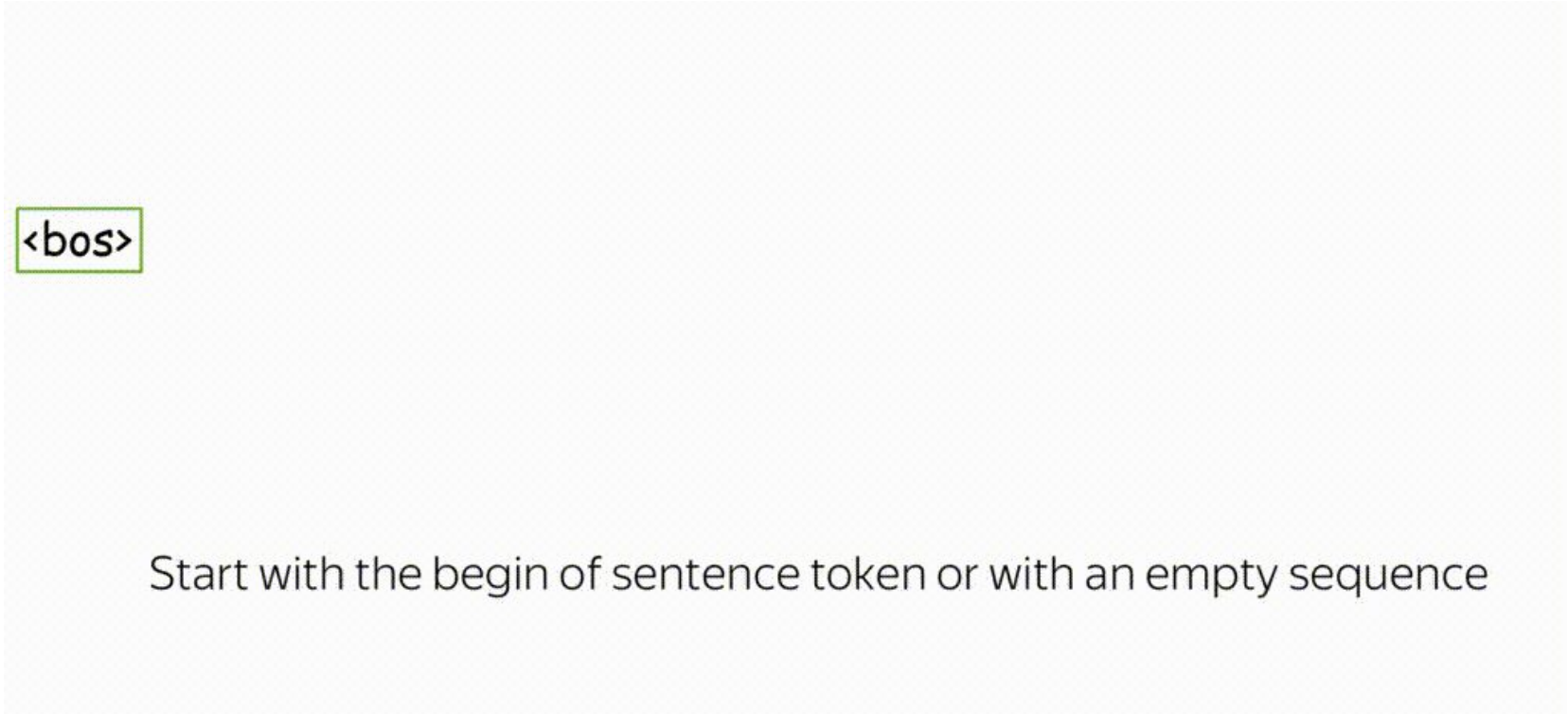
Straightforward:

- **greedy** - at each step, pick token with the highest probability

$$\arg\max_y \prod_{t=1}^{n} p(y_t|y_{<t}, x) \neq \prod_{t=1}^{n} \arg\max_{y_t} p(y_t|y_{<t}, x)$$

- this is bad!

Beam Search


<bos>
Start with the begin of sentence token or with an empty sequence

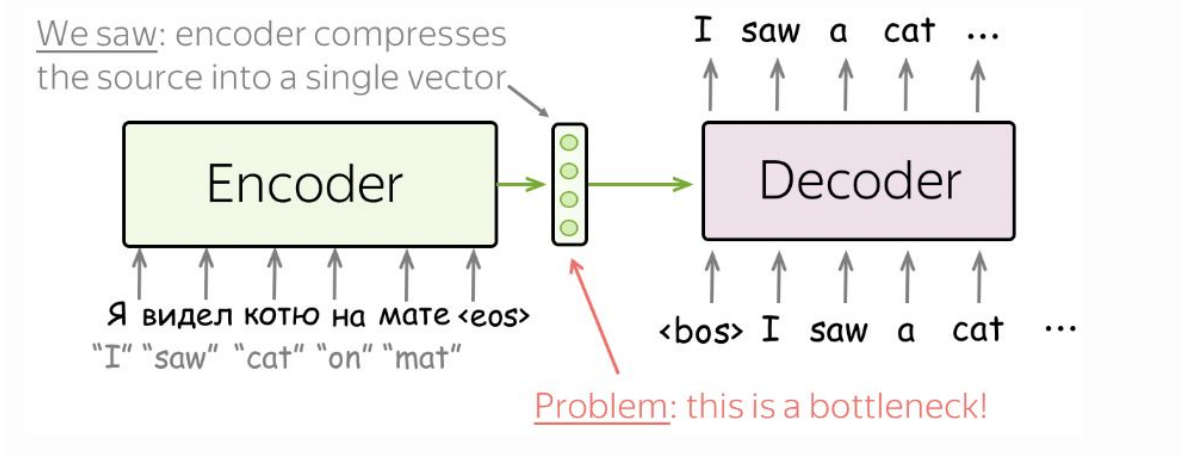
Lecture plan

- What is the language modeling?
- General framework
- Neural LMs
- Generating strategy
- Seq2seq framework
- Attention

Attention

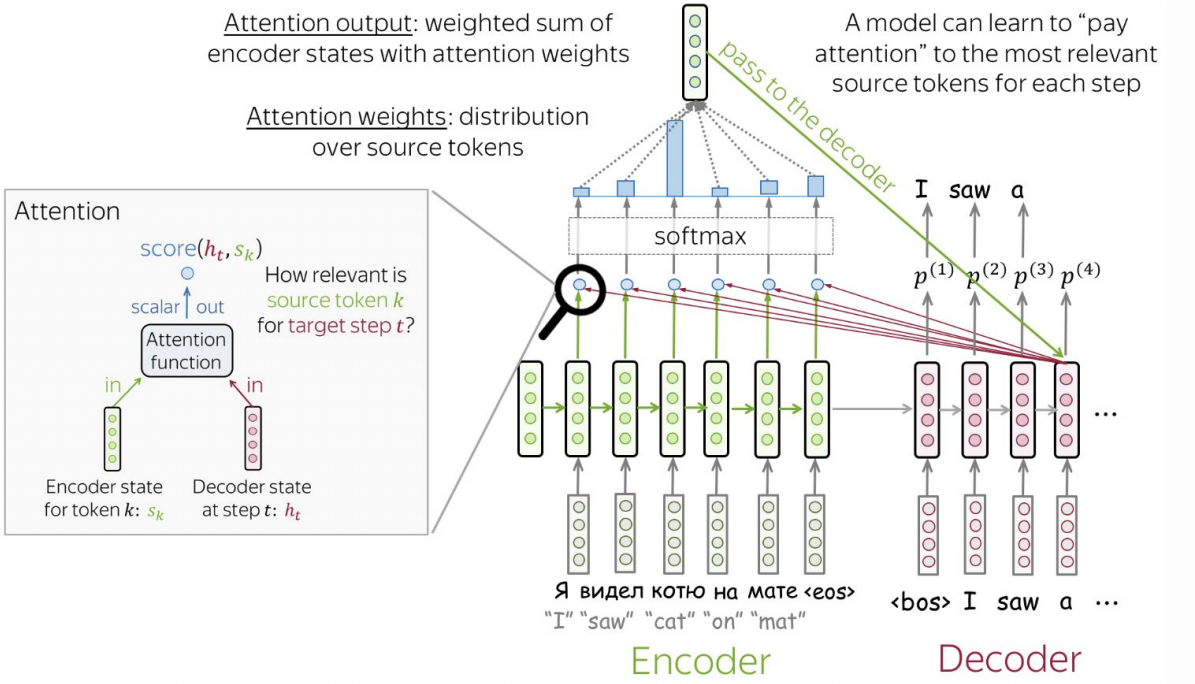
The Problem of Fixed Encoder Representation

> Problem: Fixed source representation is suboptimal: (i) for the encoder, it is hard to compress the sentence; (ii) for the decoder, different information may be relevant at different steps.

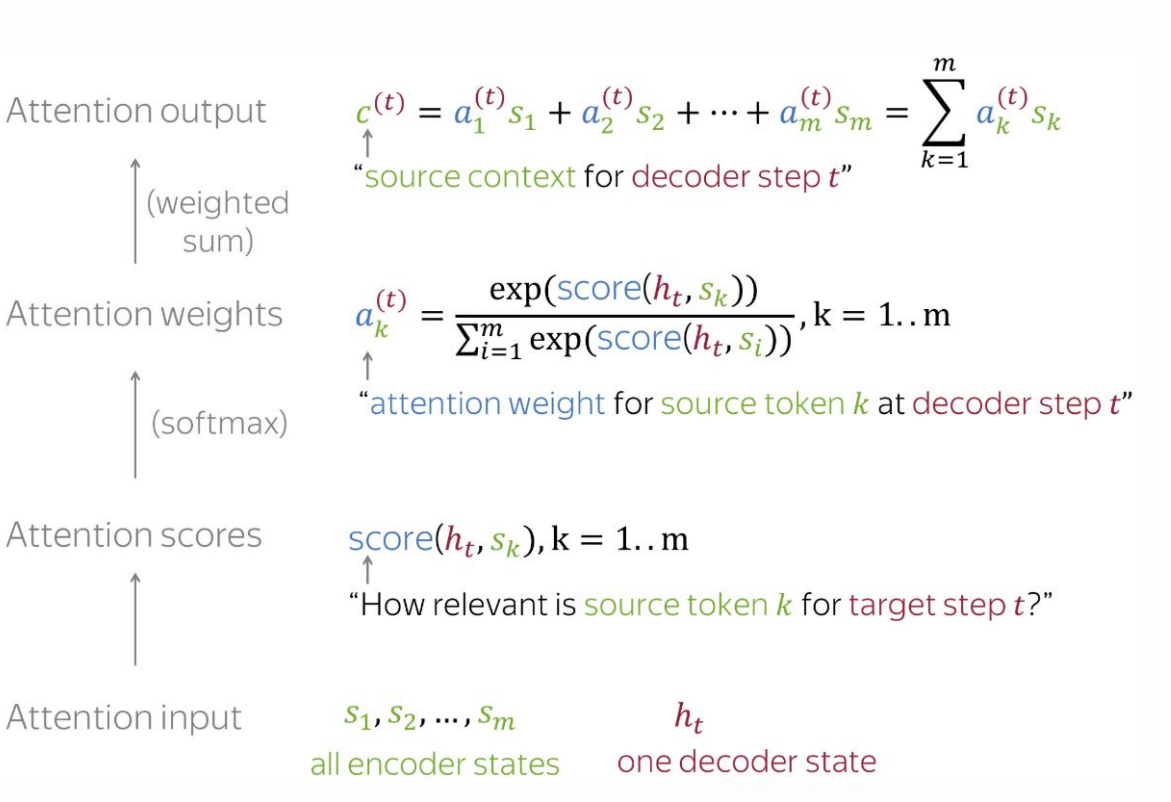


Attention: A High-Level View

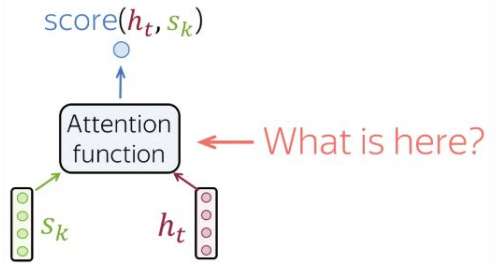
Attention: At different steps, let a model "focus" on different parts of the input.



Attention


Attention output
$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$
"source context for decoder step t"
(weighted sum)
Attention weights
$a_k^{(t)} = \frac{\exp(\text{score}(h_t, s_k))}{\sum_{i=1}^{m} \exp(\text{score}(h_t, s_i))}, \text{k} = 1..\text{m}$
"attention weight for source token k at decoder step t"
(softmax)
Attention scores
$\text{score}(h_t, s_k), \text{k} = 1..\text{m}$
"How relevant is source token k for target step t?"
Attention input
$s_1, s_2, \ldots, s_m$
all encoder states
h_t
one decoder state

How to Compute Attention Score?



Dot-product

$h_t^T \times s_k$

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

Bilinear

$h_t^T \times [W] \times s_k$

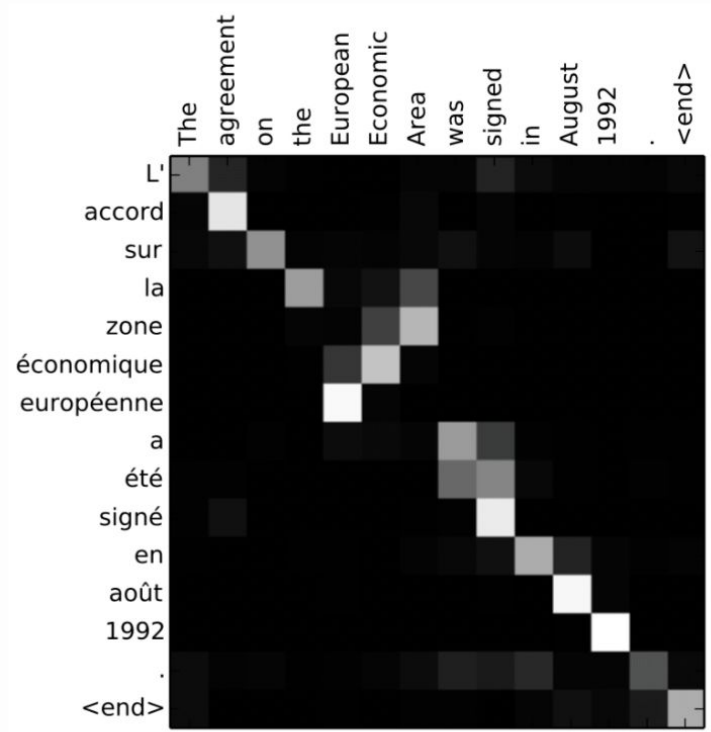
$$\text{score}(h_t, s_k) = h_t^T \, W \, s_k$$

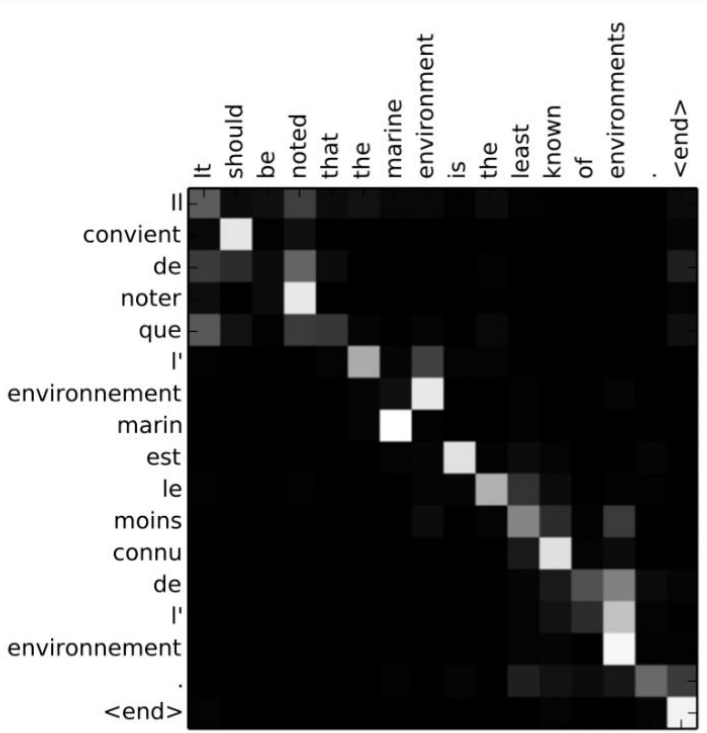
Multi-Layer Perceptron

$w_2^T \times \tanh\left[W_1 \times \begin{bmatrix} h_t \\ s_k \end{bmatrix}\right]$

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

Attention


The
agreement
on
the
European
Economic
Area
was
signed
in
August
1992
.
<end>
L'
accord
sur
la
zone
économique
européenne
a
été
signé
en
août
1992
.
<end>


It
should
be
noted
that
the
marine
environment
is
the
least
known
of
environments
.
<end>
Il
convient
de
noter
que
l'
environnement
marin
est
le
moins
connu
de
l'
environnement
.
<end>