Introduction to NLP

Credits

- YSDA NLP course (Elena Voita et al.):

  https://github.com/yandexdataschool/nlp_course/tree/master
- CS224N: Stanford NLP course (Cristopher Manning, Abigail See et al.):

  https://web.stanford.edu/class/cs224n
- https://github.com/bentrevett/pytorch-sentiment-analysis
- https://github.com/bentrevett/pytorch-seq2seq

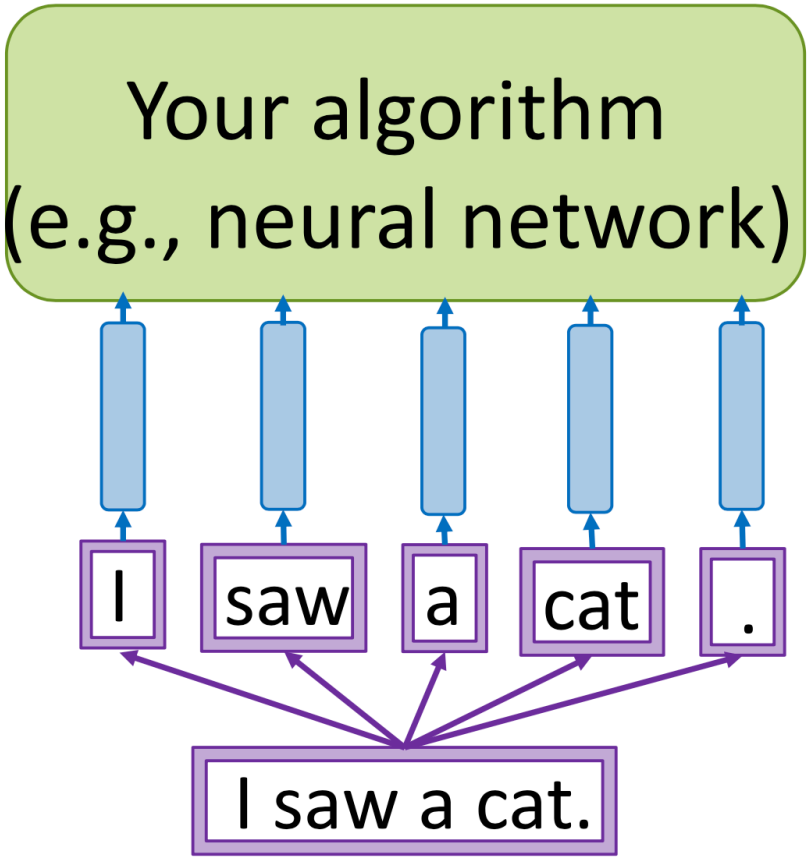
NLP Problems

- Text classification
- Named Entity Recognition
- Language models
- Text Summarization
- Machine Translation
- Speech-to-Text
- Question Answering
- Dialogue Systems
- and many more...

NLP Problems

- Text classification
- Named Entity Recognition
- Language models
- Text Summarization
- Machine Translation
- Speech-to-Text
- Question Answering
- Dialogue Systems
- and many more...

Word representations



Any algorithm for solving any task

Word representation - vector
(word embedding)

Sequence of tokens

Text

Representing words as discrete symbols

- We regard words as discrete symbols: hotel, conference, motel
- Words can be represented by **one-hot vectors**:
  - motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
  - hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
- Vector dimension = number of words in vocabulary (e.g., 500,000)

What is word meaning?

What is the meaning of the word “**bardiwac**”?

What is word meaning?

What is the meaning of the word “**bardiwac**”?

- "He handed her her glass of bardiwac”
- "Beef dishes are made to complement the bardiwacs"
- "Nigel staggered to his feet, face flushed from too much bardiwac”
- "Malbec, one of the lesser–known bardiwac grapes, responds well to Australia’s sunshine”
- "I dined off bread and cheese and this excellent bardiwac”
- "The drinks were delicious: blood–red bardiwac as well as light, sweet Rhenish”

What is word meaning?

What is the meaning of the word “**bardiwac**”?

- "He handed her her glass of bardiwac”
- "Beef dishes are made to complement the bardiwacs"
- "Nigel staggered to his feet, face flushed from too much bardiwac”
- "Malbec, one of the lesser-known bardiwac grapes, responds well to Australia’s sunshine”
- "I dined off bread and cheese and this excellent bardiwac”
- "The drinks were delicious: blood-red bardiwac as well as light, sweet Rhenish”

“You shall know a word by the company it keeps.” - Firth (1957)

Bardiwac is a heavy red alcoholic beverage made from grapes

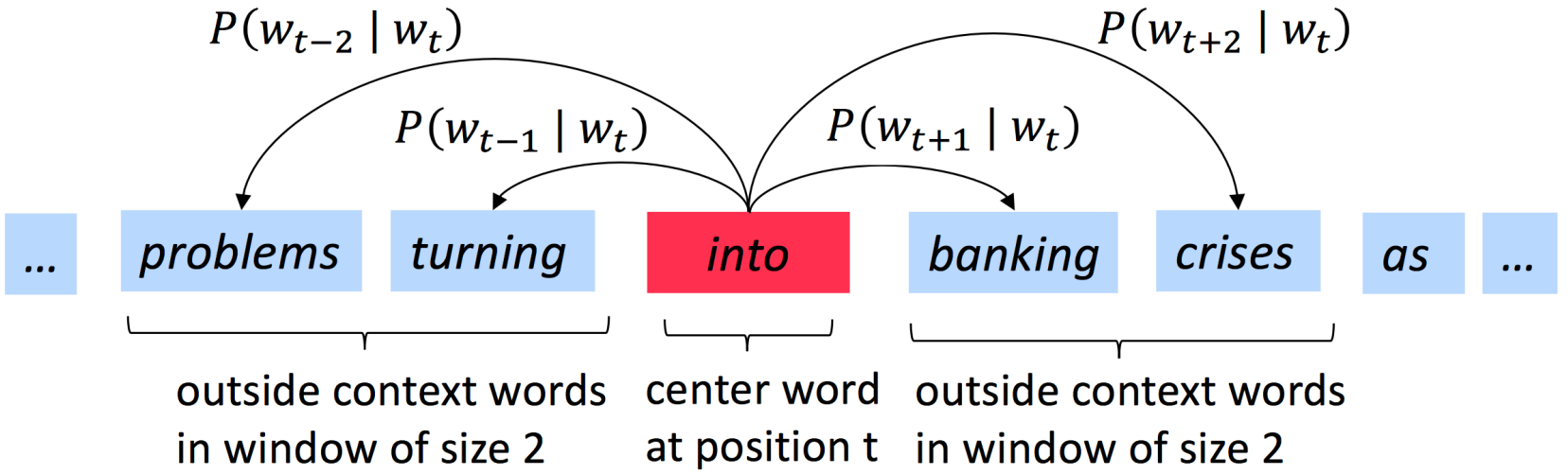
Word2Vec

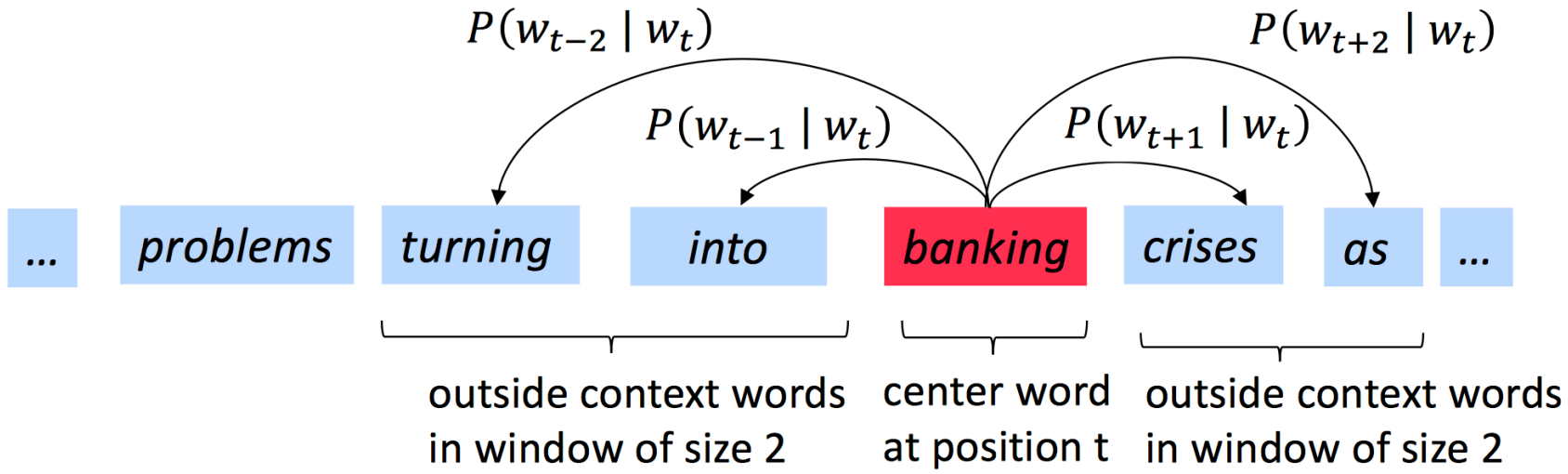
We will learn a dense vector for each word, chosen so that it is similar to vectors of words that appear in similar contexts.

- Large text corpus
- Every word in a fixed vocabulary is represented by a vector (an embedding-to-be)
- Go through each position *t* in the text, which has a center word *c* and context ("outside") words *o*
- Use the similarity of the word vectors for *c* and *o* to calculate the probability of *o* given *c* (or vice versa)
- Keep adjusting the word vectors to maximize this probability

Mikolov et al, 2013, https://arxiv.org/pdf/1310.4546.pdf

Word2Vec


$P(w_{t-2} \mid w_t)$
$P(w_{t-1} \mid w_t)$
$P(w_{t+1} \mid w_t)$
$P(w_{t+2} \mid w_t)$
...
problems
turning
into
banking
crises
as
...
outside context words in window of size 2
center word at position t
outside context words in window of size 2


$P(w_{t-2} \mid w_t)$
$P(w_{t-1} \mid w_t)$
$P(w_{t+1} \mid w_t)$
$P(w_{t+2} \mid w_t)$
...
problems
turning
into
banking
crises
as
...
outside context words in window of size 2
center word at position t
outside context words in window of size 2

Word2Vec Objective

For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_t.

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

θ *is all variables to be optimized*

The objective function is the (average) negative log likelihood:

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \neq 0}} \log P(w_{t+j} \mid w_t; \theta)$$

Word2Vec Objective

We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\leq j\leq m \\ j\neq 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$

Question: How to calculate $P\left(w_{t+j} \mid w_t; \theta\right)$?

Answer: We will *use two* vectors per word *w*:

- v_w when *w* is a center word
- u_w when *w* is a context word

Exponentiation makes anything positive

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

Dot product compares similarity of *o* and *c*.

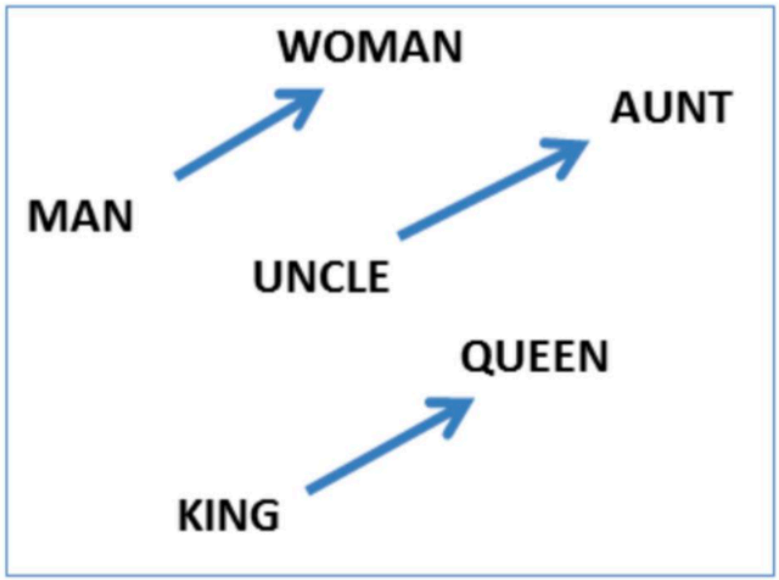
$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$

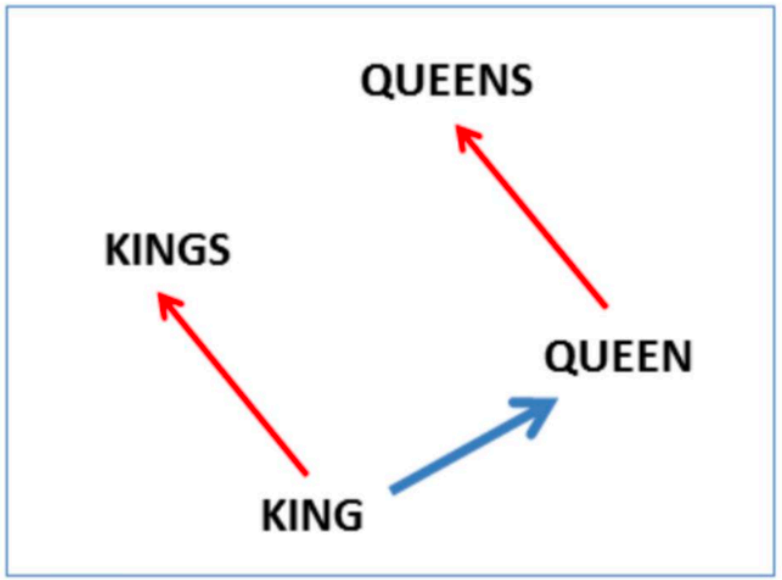
Larger dot product = larger probability

Normalize over entire vocabulary to give probability distribution

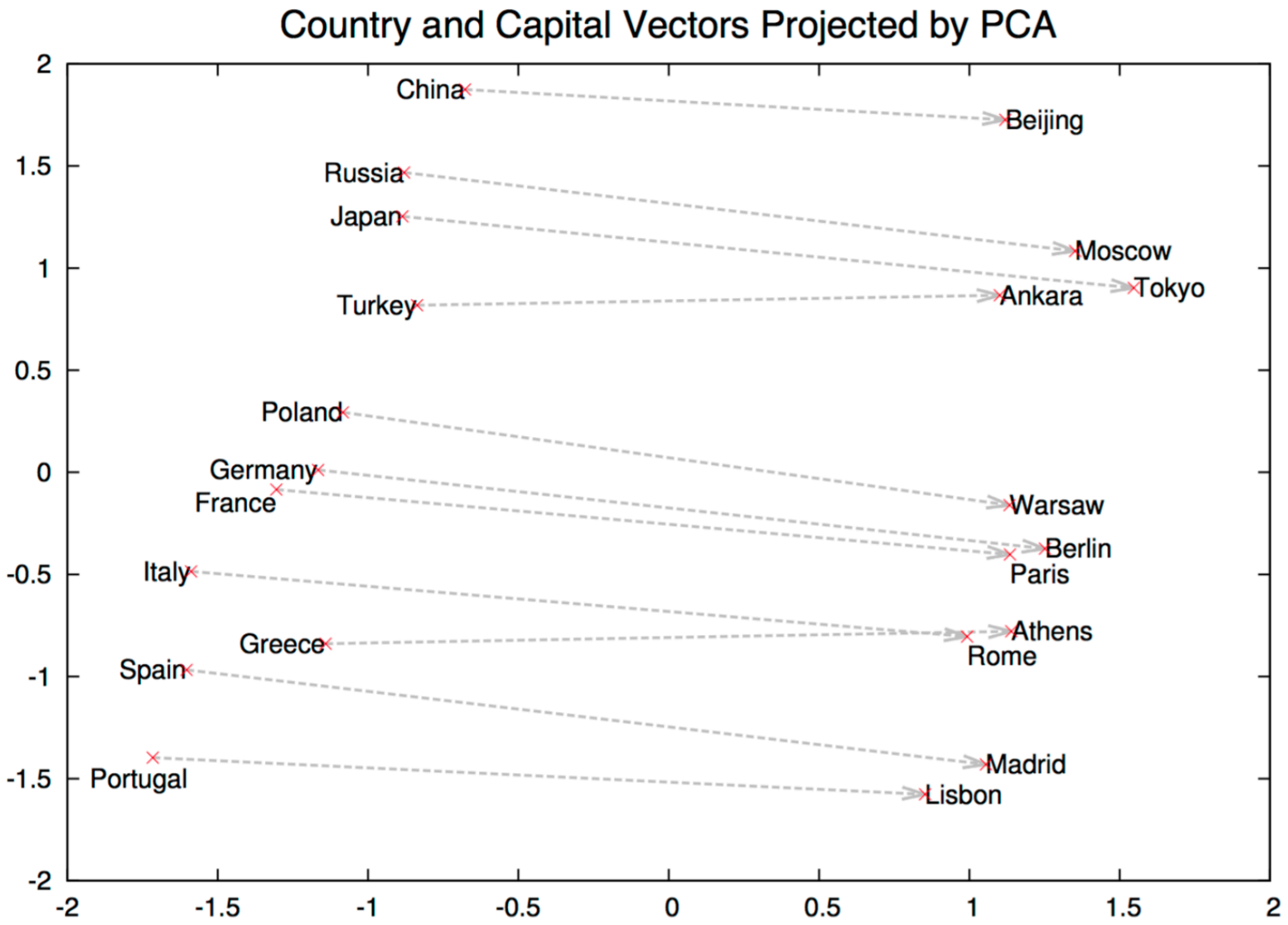
Word2Vec: Relations between vectors

$v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$

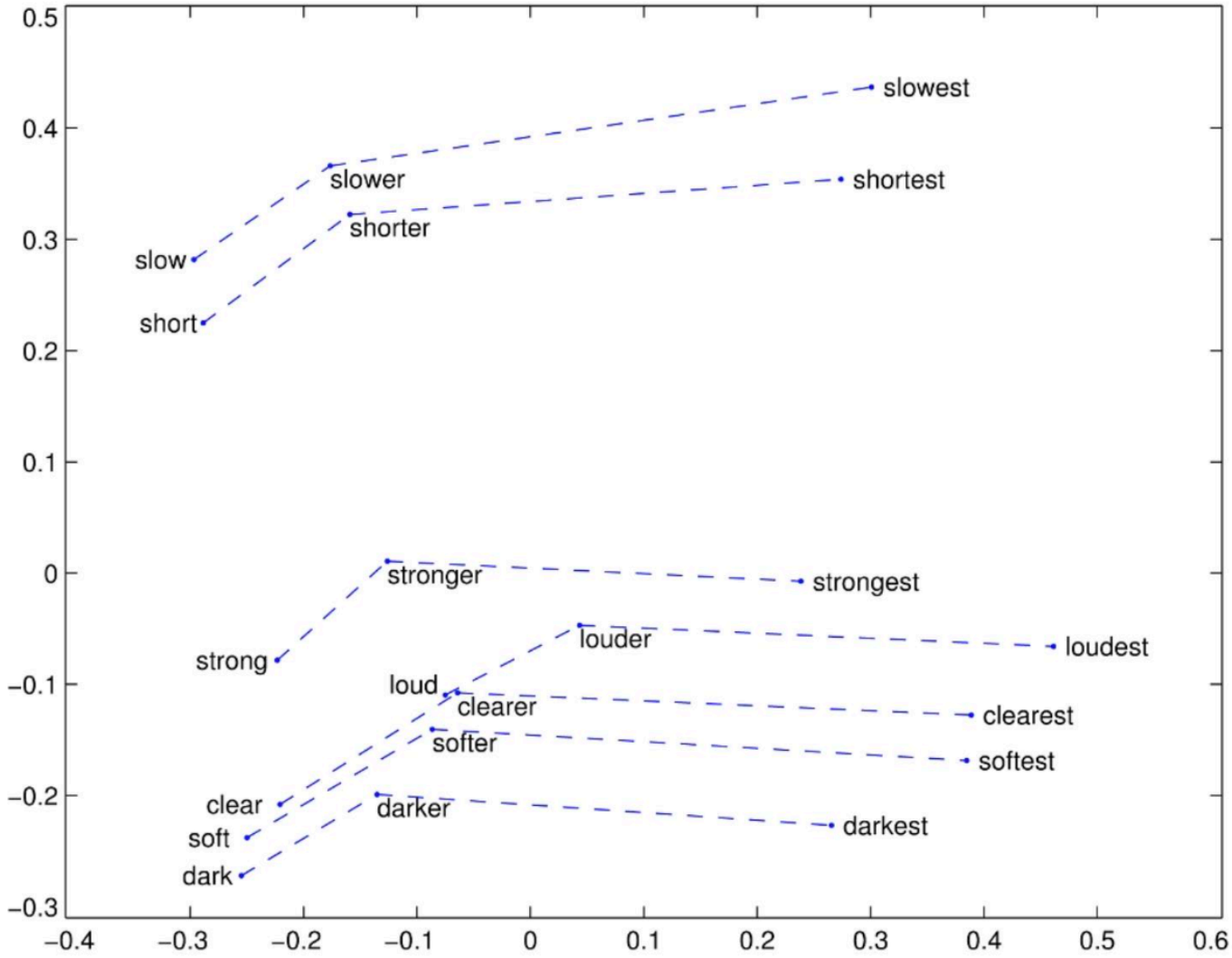




Word2Vec: Relations between vectors



Word2Vec: Relations between vectors



(http://nlp.stanford.edu/projects/glove/)

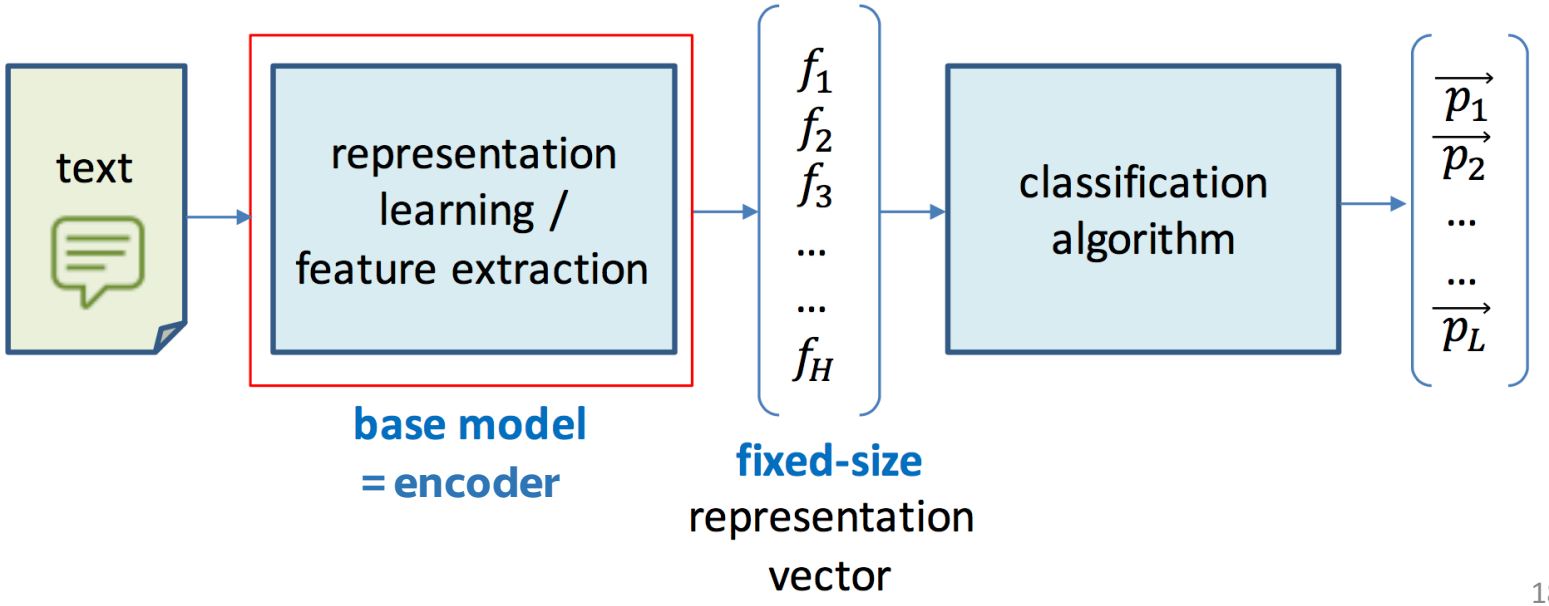
Text Encoders in NLP

Common case: initializing the embedding layer with pre-trained embeddings

- Vanilla RNN
- Fancy LSTM, BiLSTM, GRU
- Transformer-based encoders: ELMo, BERT, RoBERTa, XLNet, T5, ...

Text classification problem

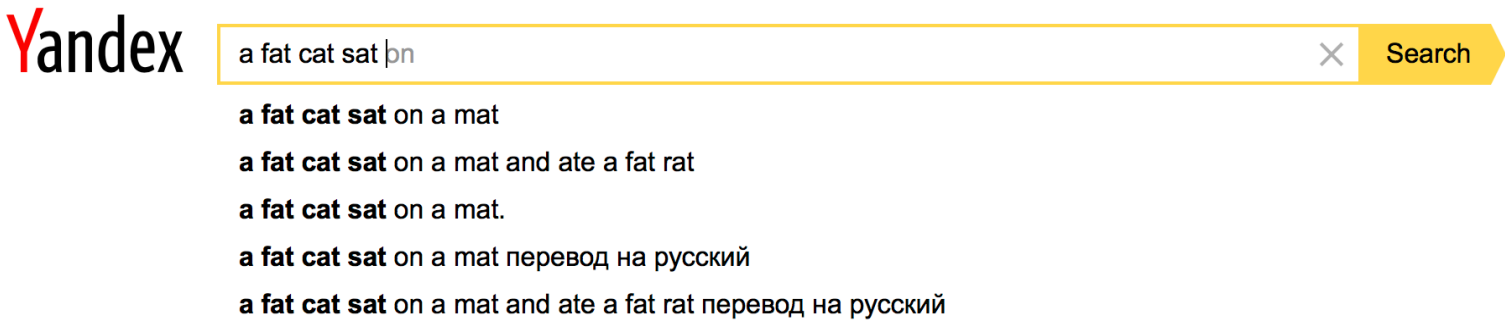
- Dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ with $x \in X, y \in Y$
- Label set $\mathcal{L} = \{1, 2, \ldots, L\}$
- The pairs (x_n, y_n) are i.i.d. following $P(\mathbf{X}, \mathbf{Y})$ over a sample space $X \times Y$
- Multi-class classification: $Y = \mathcal{L}$
- Multi-label classification: $Y = 2^{\mathcal{L}}$



Language models

- Language model: estimates the “probability” of a text

$$P(\boldsymbol{y}) = P(y_1)P(y_2|y_1)P(y_3|y_1, y_2) \dots P(y_T|y_1, y_2, \dots, y_{T-1})$$



Vanilla RNN language model

Output distribution

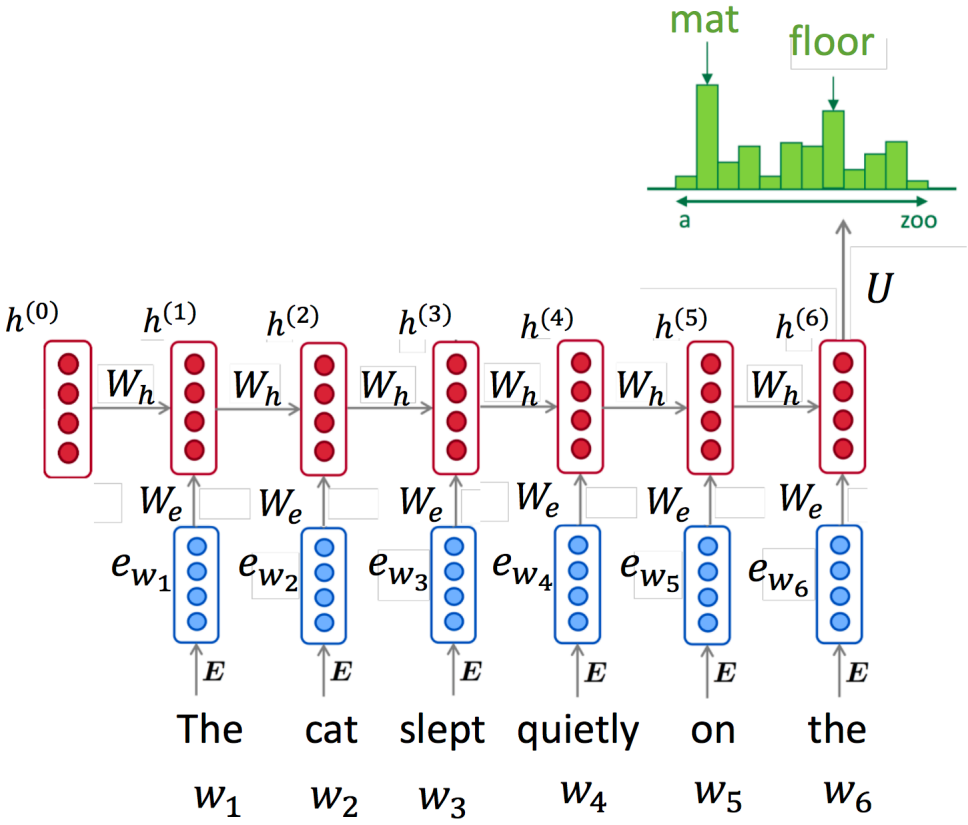
$$\hat{y}_t = \text{softmax}\left(Uh^{(t)} + b_2\right) \in \mathbb{R}^{|V|}$$

Hidden states

$$h^{(t)} = \sigma\left(W_h h^{(t-1)} + W_e e_{w_{t-1}} + b_1\right)$$

Word embeddings

Words/tokens

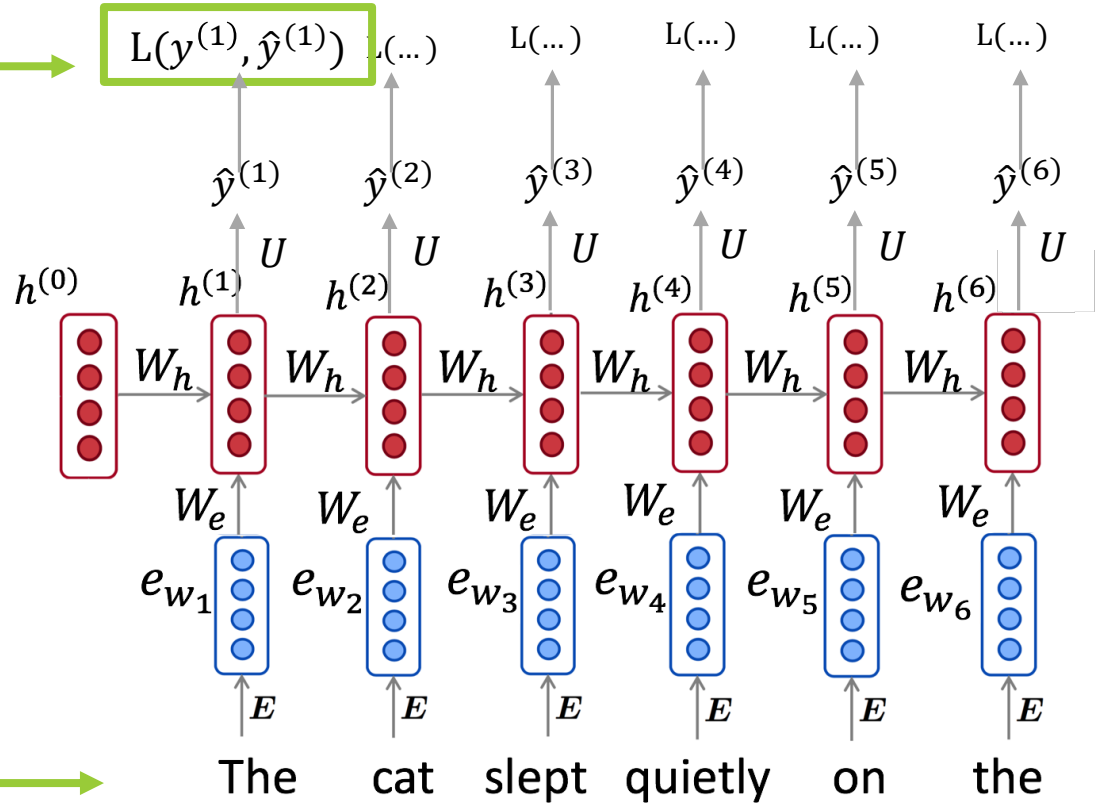


Training RNN-LM

negative log prob of "cat"

Loss

$$\mathrm{L}\left(y^{(t)}, \hat{y}^{(t)}\right) = \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$

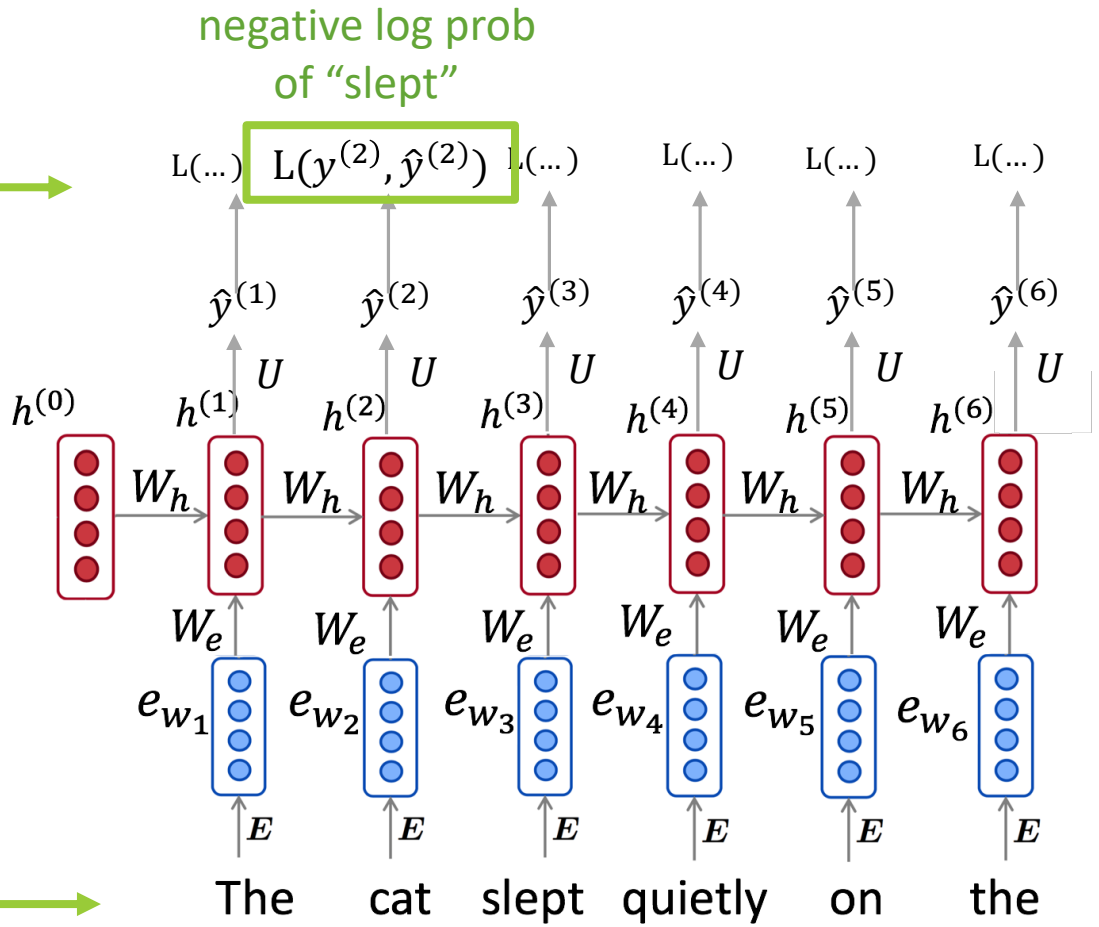


corpus

Training RNN-LM

Loss

$$\mathrm{L}\left(y^{(t)}, \hat{y}^{(t)}\right) = \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$

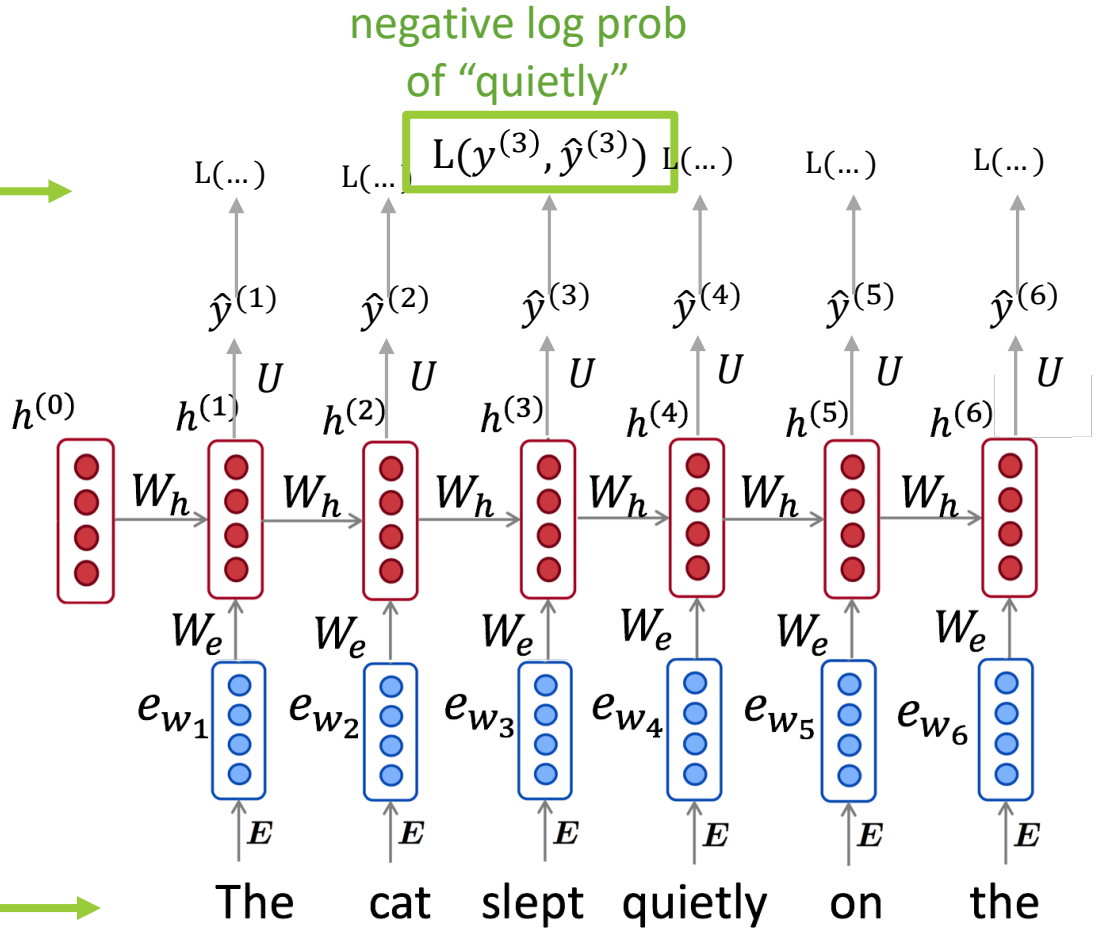


corpus

Training RNN-LM

Loss

$$\mathrm{L}\left(y^{(t)}, \hat{y}^{(t)}\right) = \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$

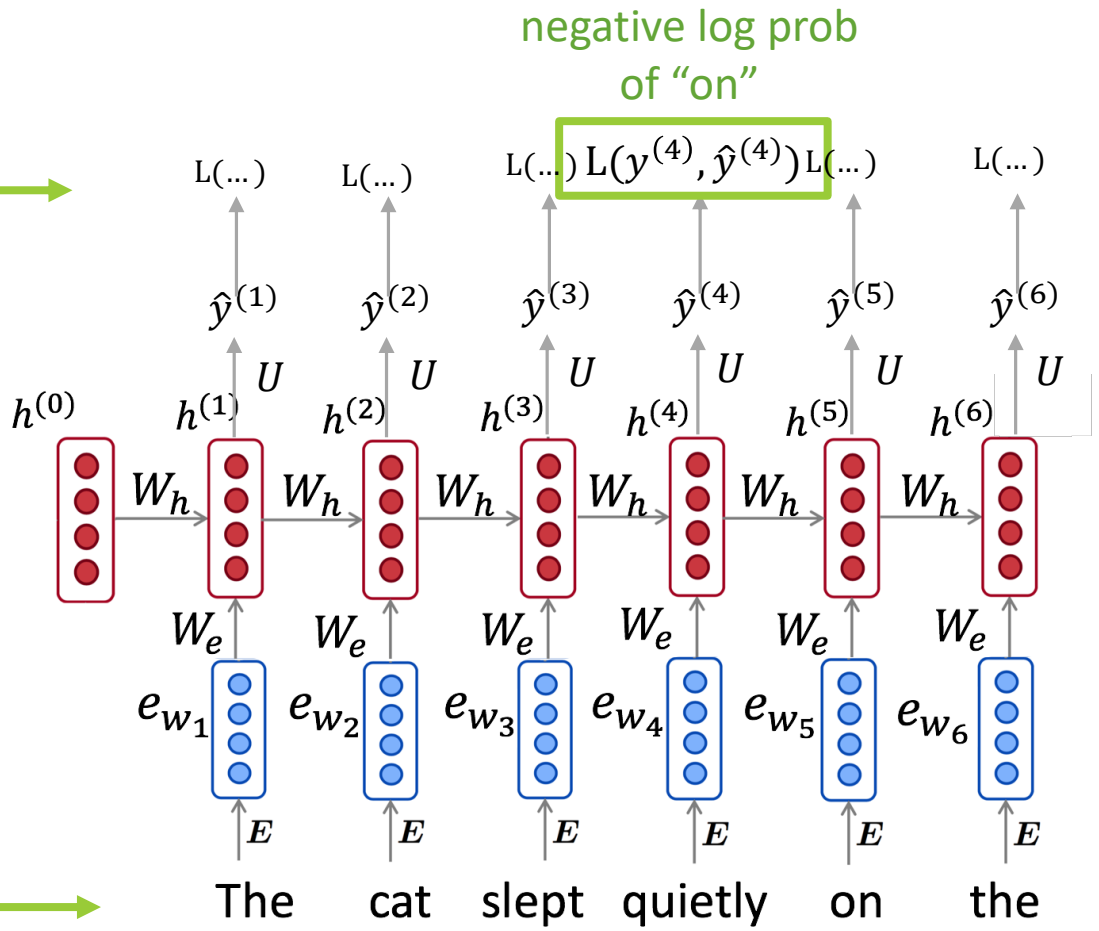


corpus

Training RNN-LM

Loss

$$\mathrm{L}\left(y^{(t)}, \hat{y}^{(t)}\right) = \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$

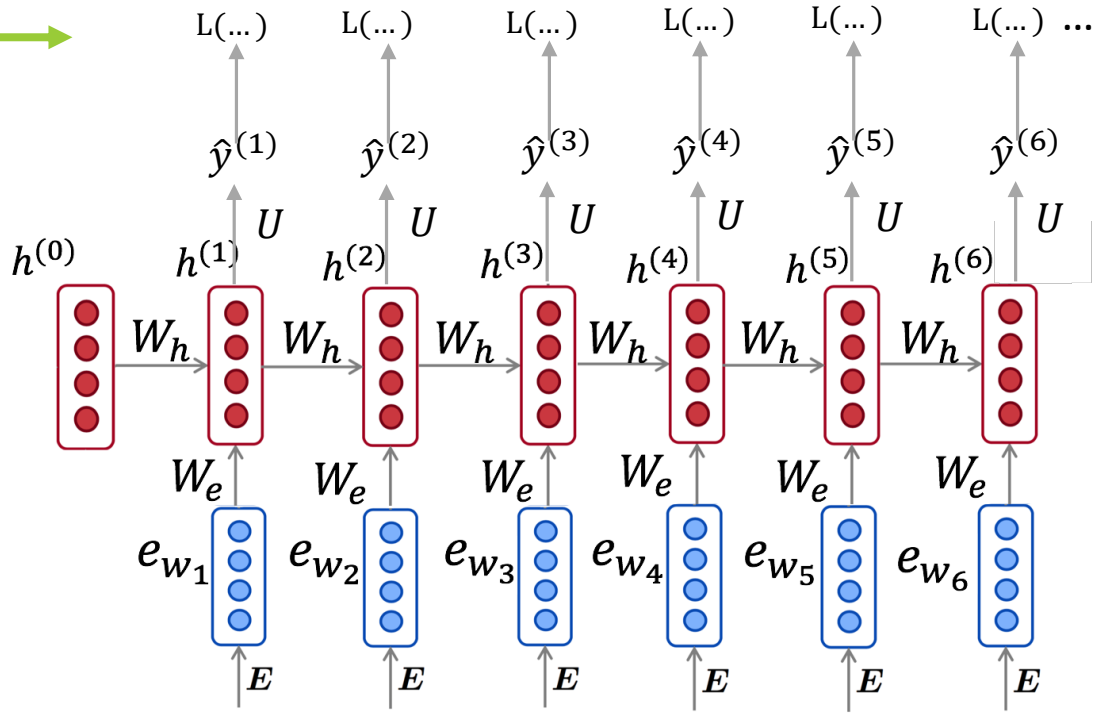


corpus

Training RNN-LM

Loss

$$\mathrm{L}\left(y^{(t)}, \hat{y}^{(t)}\right) = \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$



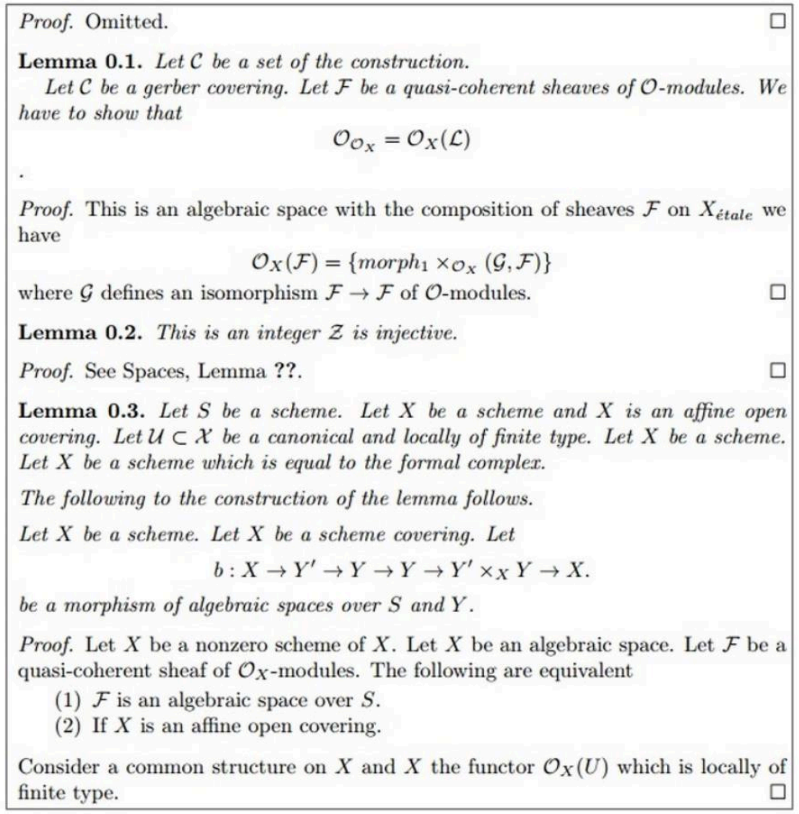
sum

$$\sum_{t=1}^{T} L(y^{(t)}, \hat{y}^{(t)})$$

corpus

Visualizing and Understanding Recurrent Networks

Char-based LM trained on Latex of book on algebraic geometry



Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

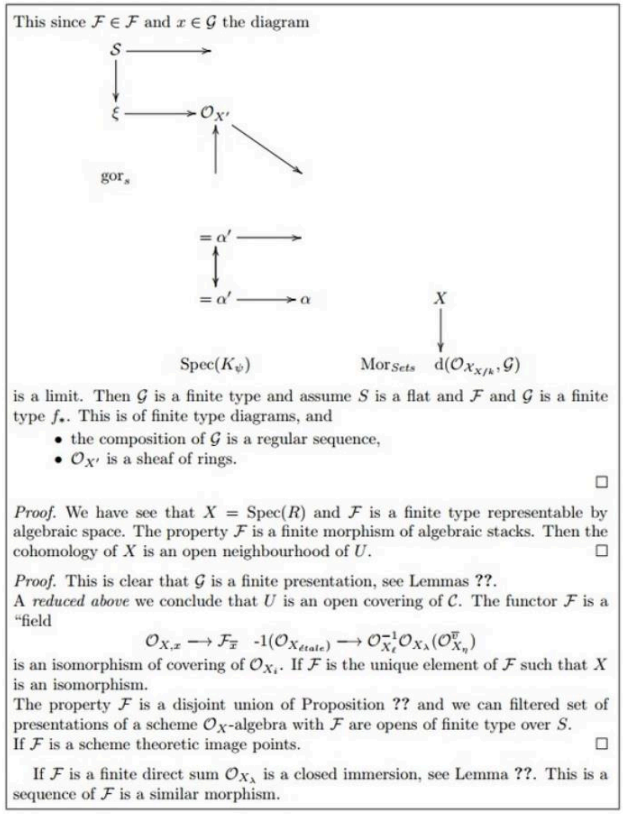
$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □



This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram

$$\begin{array}{ccc} \mathcal{S} & & \\ \downarrow & & \\ \xi & \to & \mathcal{O}_{X'} \\ & & \uparrow \\ \text{gor}_s & & \\ & & \\ = \alpha' & \to & \\ \uparrow & & \\ = \alpha' & \to & \alpha \end{array} \qquad \begin{array}{c} X \\ \downarrow \\ \text{d}(\mathcal{O}_{\mathcal{X}_{X/k}}, \mathcal{G}) \end{array}$$

$$\text{Spec}(K_\psi) \qquad \text{Mor}_{Sets}$$

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \text{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.

A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_i}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{x}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.

The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.

If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

More hallucinated algebraic geometry. Nice try on the diagram (right).

Karpathy et al, ICLR workshop, 2016 https://arxiv.org/pdf/1506.02078.pdf

Visualizing and Understanding Recurrent Networks

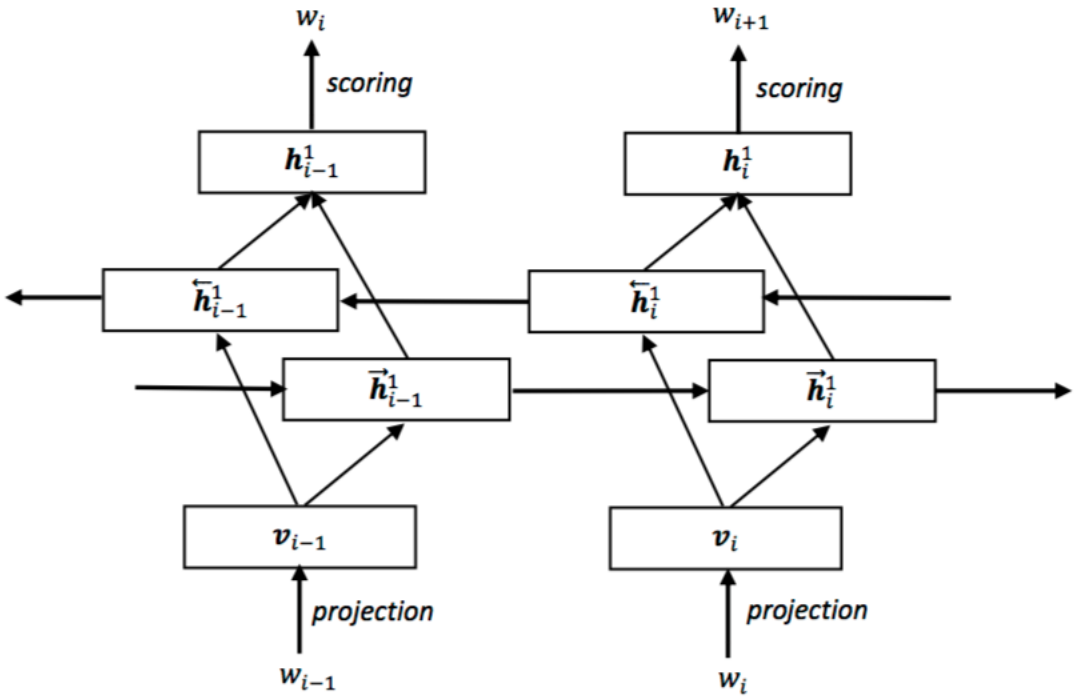
Char-based LM trained on Linux Source Code

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

Karpathy et al, ICLR workshop, 2016 https://arxiv.org/pdf/1506.02078.pdf

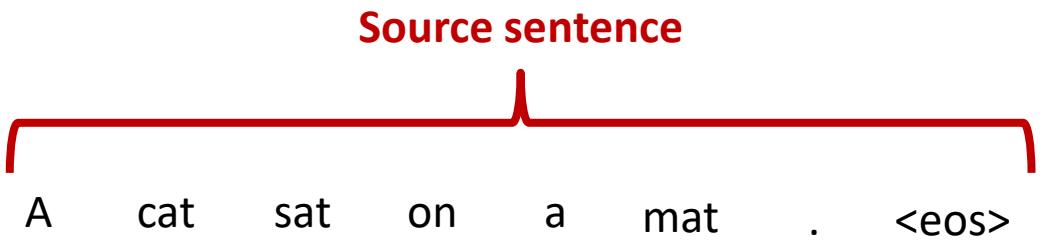
Bi-RNN Language Models

- Can take into account both left and right contexts
- Can not be used for generation
- Then, where can it be used?

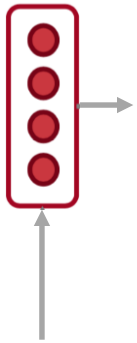


https://arxiv.org/pdf/1602.06064.pdf

RNN Encoder-Decoder (Vanilla)

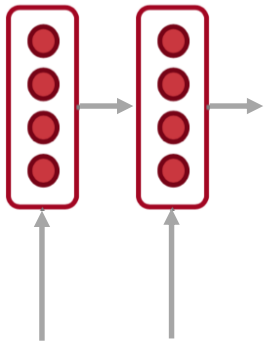


RNN Encoder-Decoder (Vanilla)



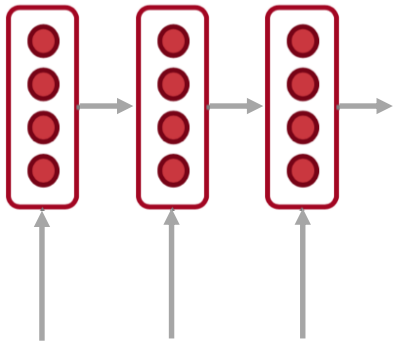
A cat sat on a mat . <eos>

RNN Encoder-Decoder (Vanilla)



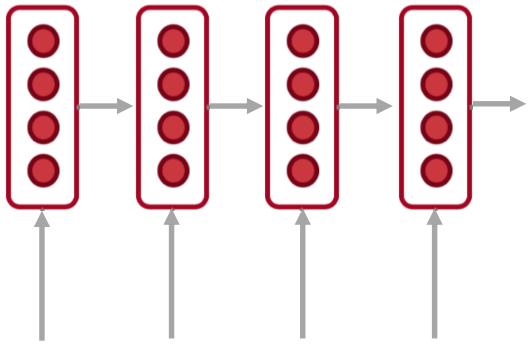
A cat sat on a mat . <eos>

RNN Encoder-Decoder (Vanilla)



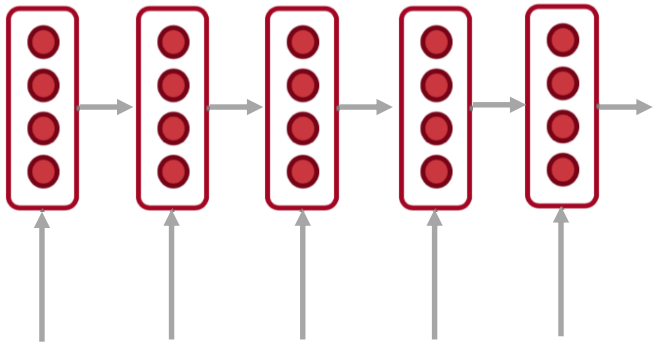
A cat sat on a mat . <eos>

RNN Encoder-Decoder (Vanilla)

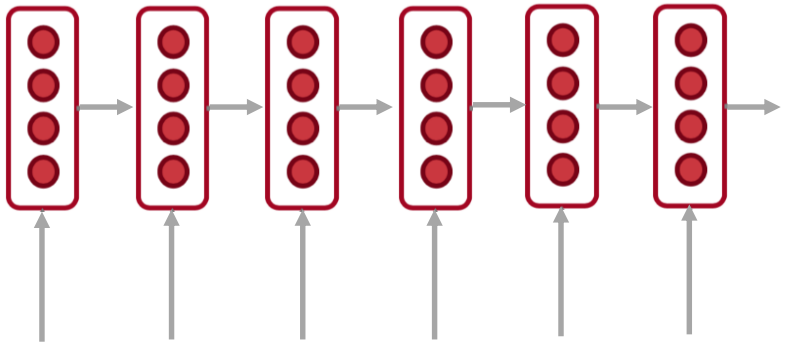


A cat sat on a mat . <eos>

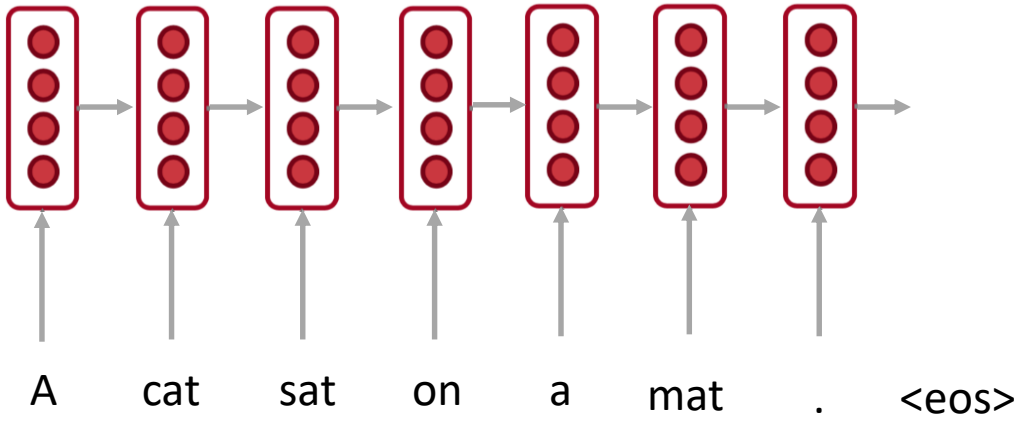
RNN Encoder-Decoder (Vanilla)



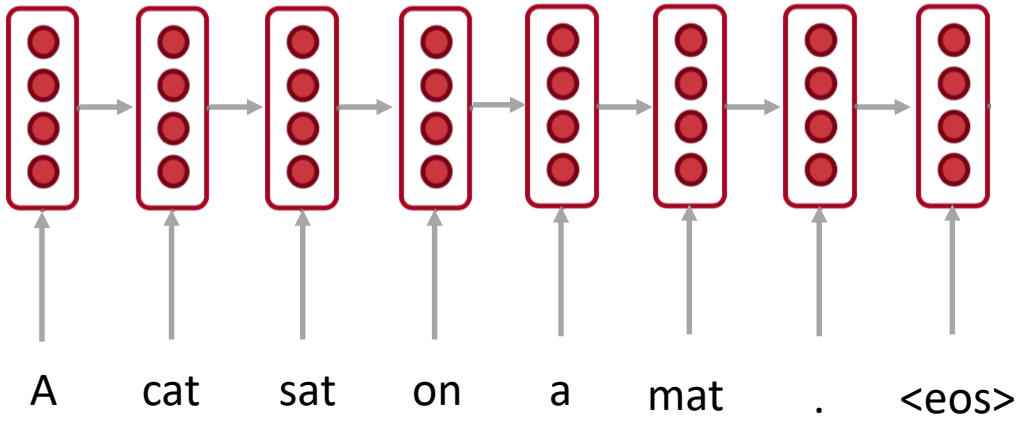
RNN Encoder-Decoder (Vanilla)



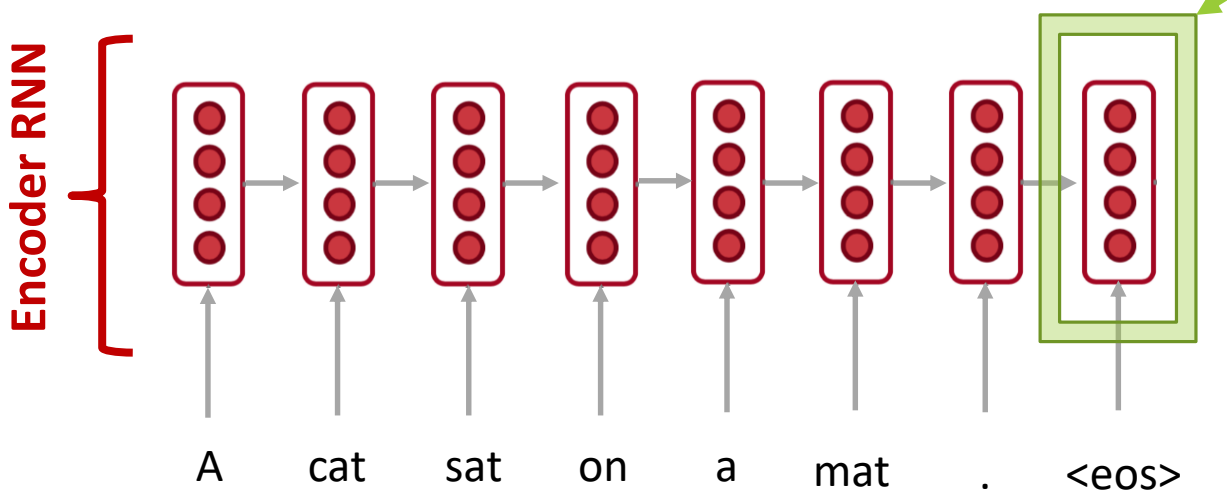
RNN Encoder-Decoder (Vanilla)



RNN Encoder-Decoder (Vanilla)

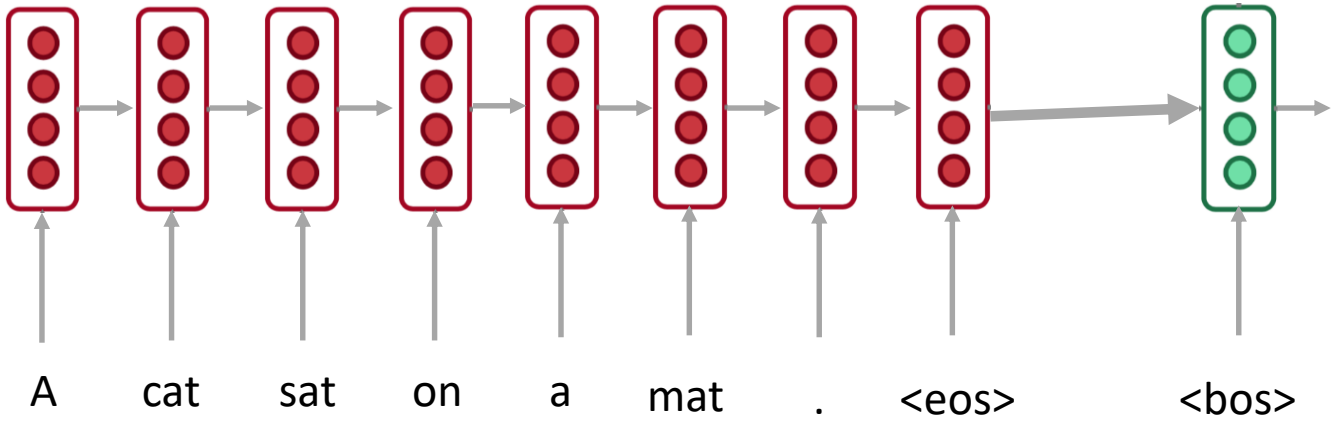


RNN Encoder-Decoder (Vanilla)



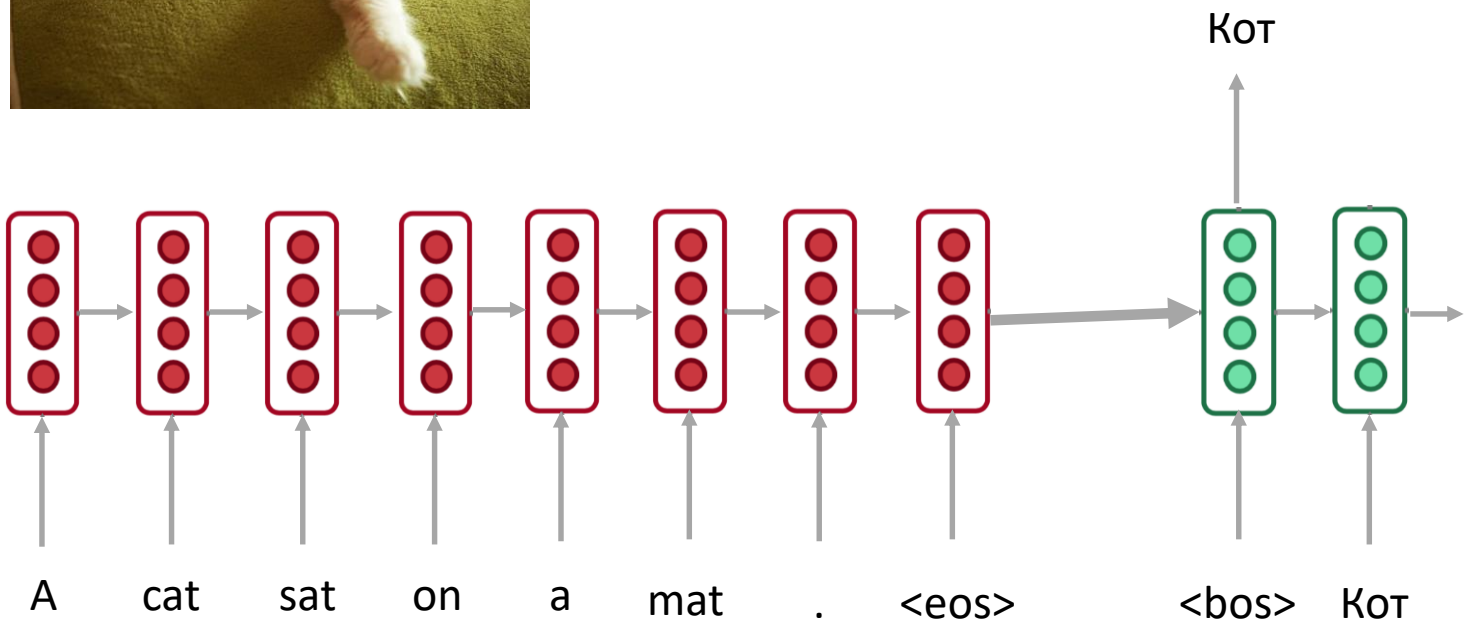
Encoding of a source sentence.
Provides an initial state for decoder RNN

RNN Encoder-Decoder (Vanilla)

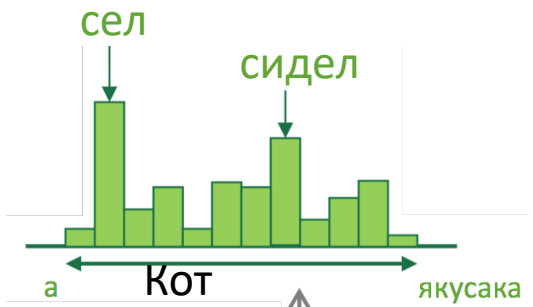


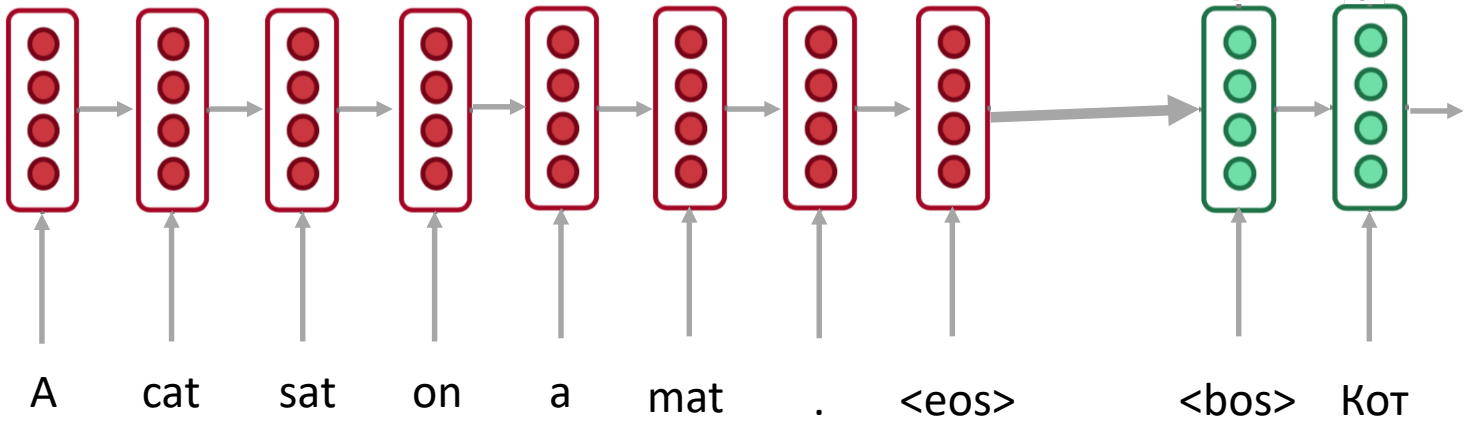
RNN Encoder-Decoder (Vanilla)

RNN Encoder-Decoder (Vanilla)

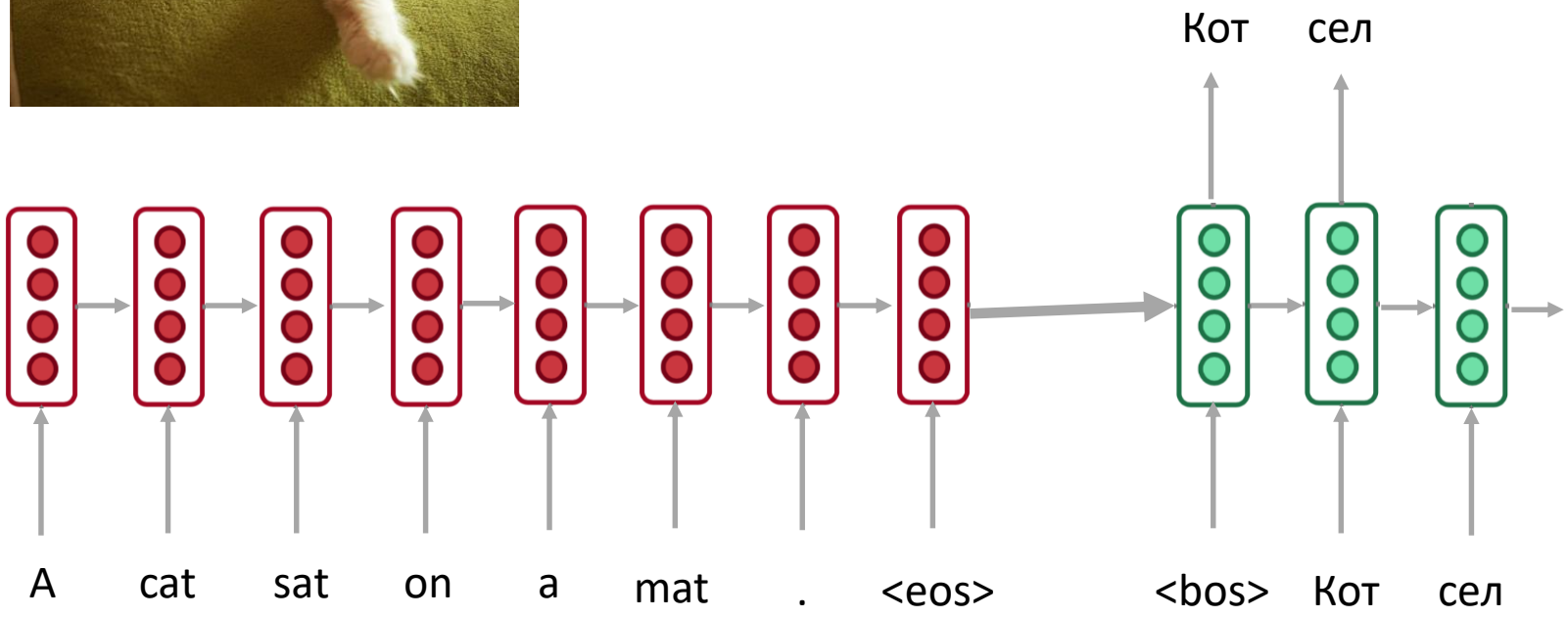


RNN Encoder-Decoder (Vanilla)



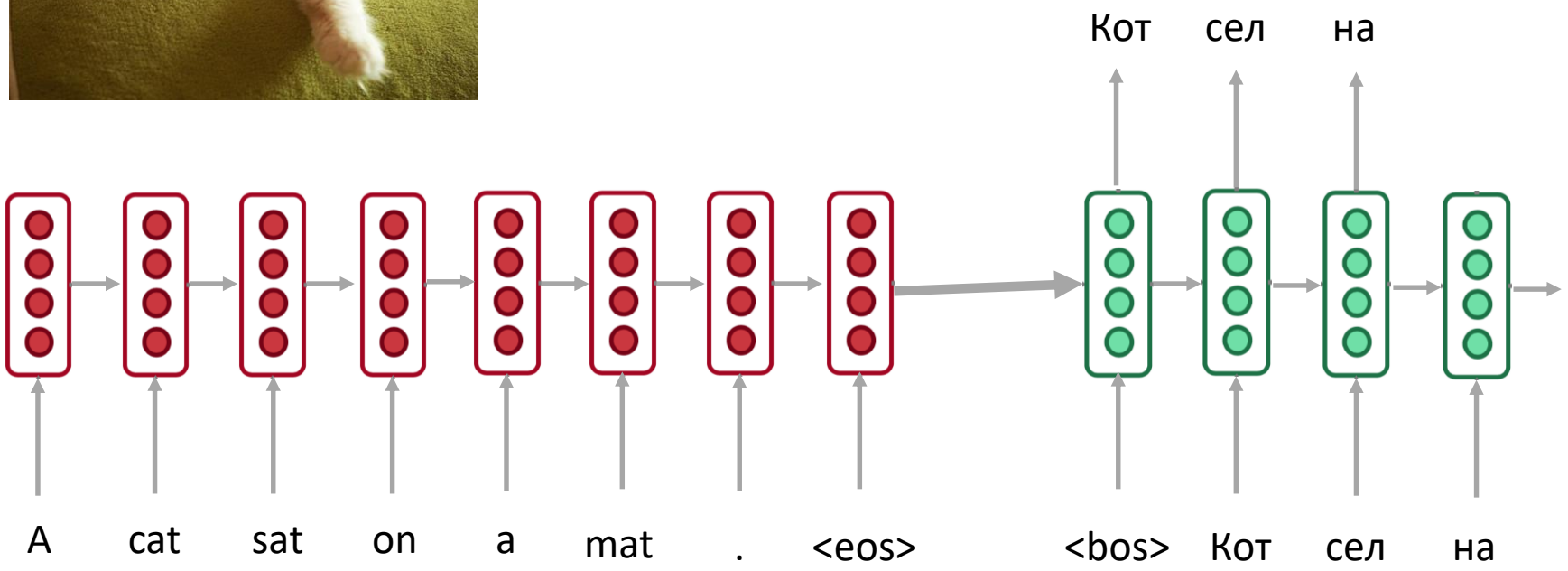


RNN Encoder-Decoder (Vanilla)



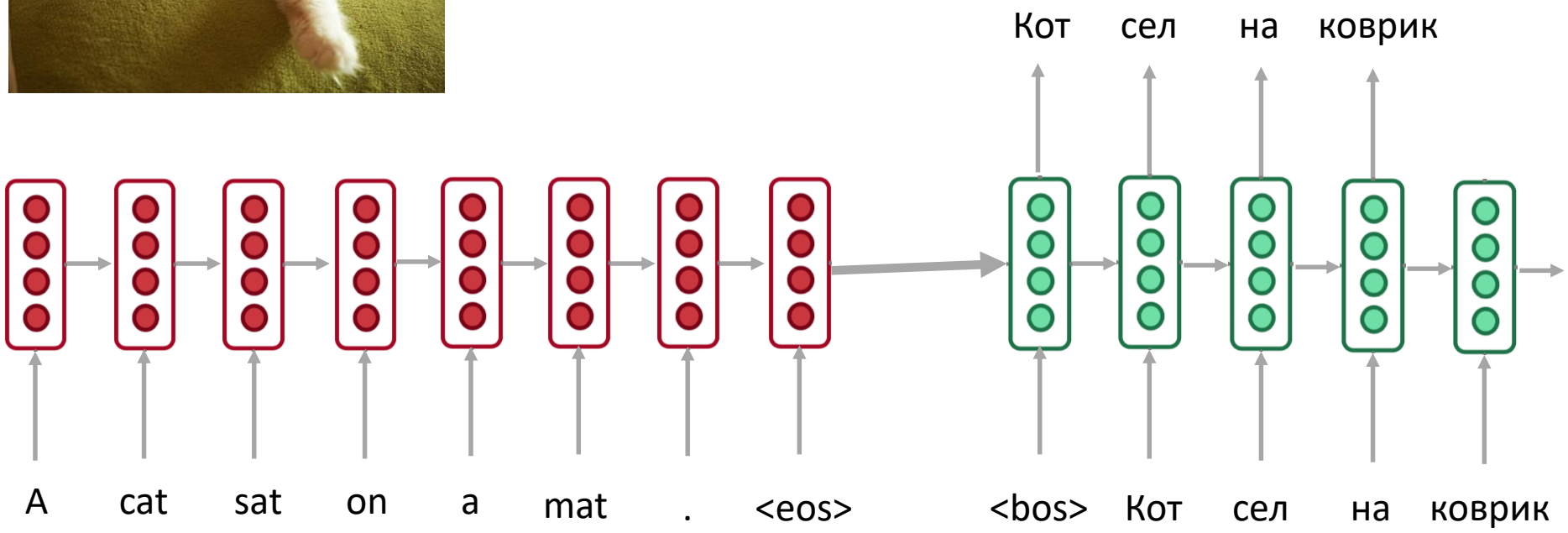
RNN Encoder-Decoder (Vanilla)

RNN Encoder-Decoder (Vanilla)



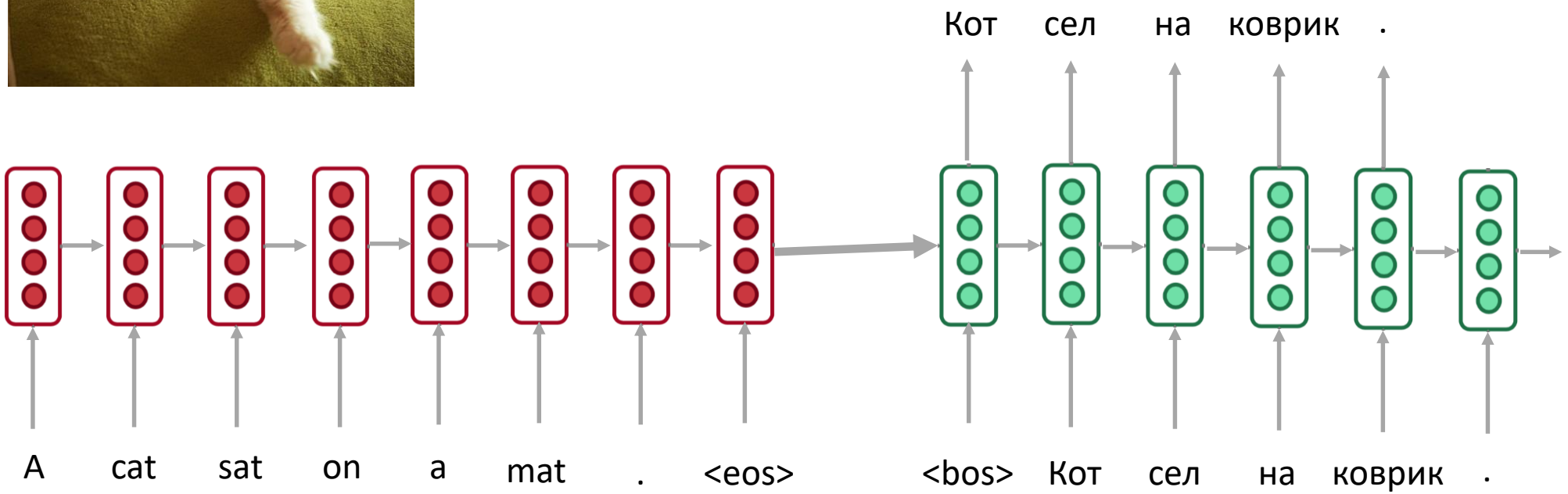
RNN Encoder-Decoder (Vanilla)

RNN Encoder-Decoder (Vanilla)



RNN Encoder-Decoder (Vanilla)

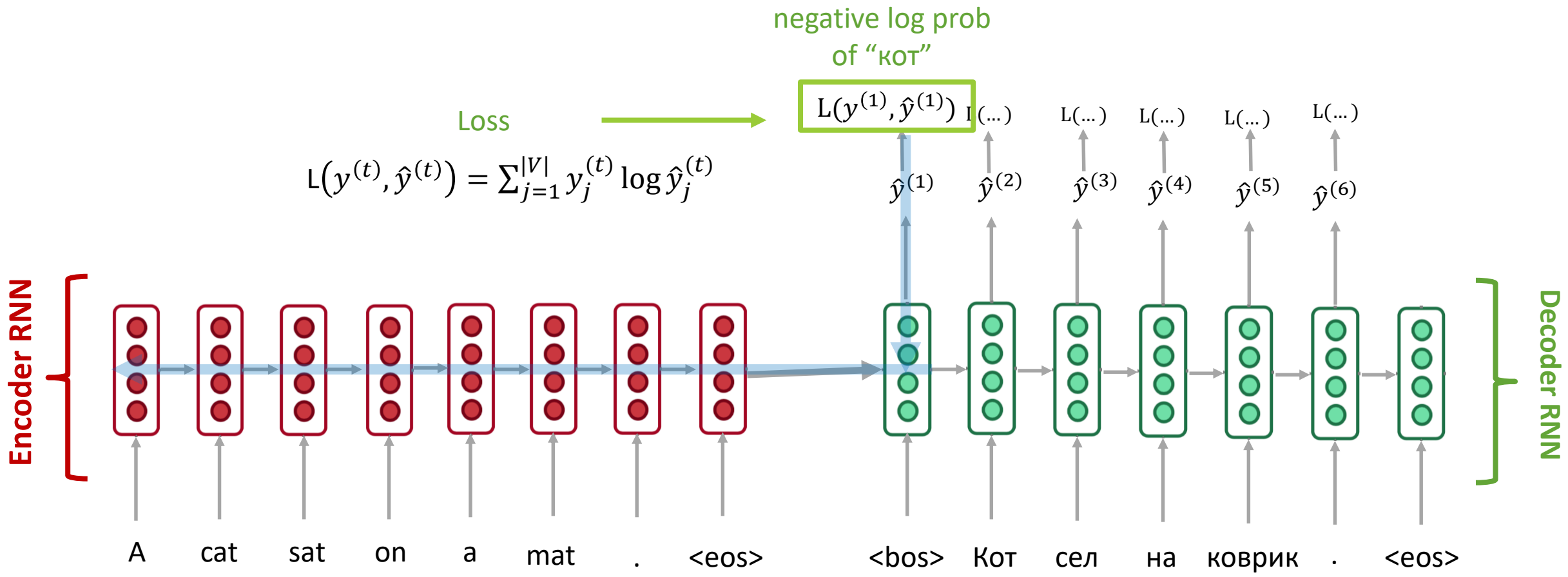
RNN Encoder-Decoder (Vanilla)



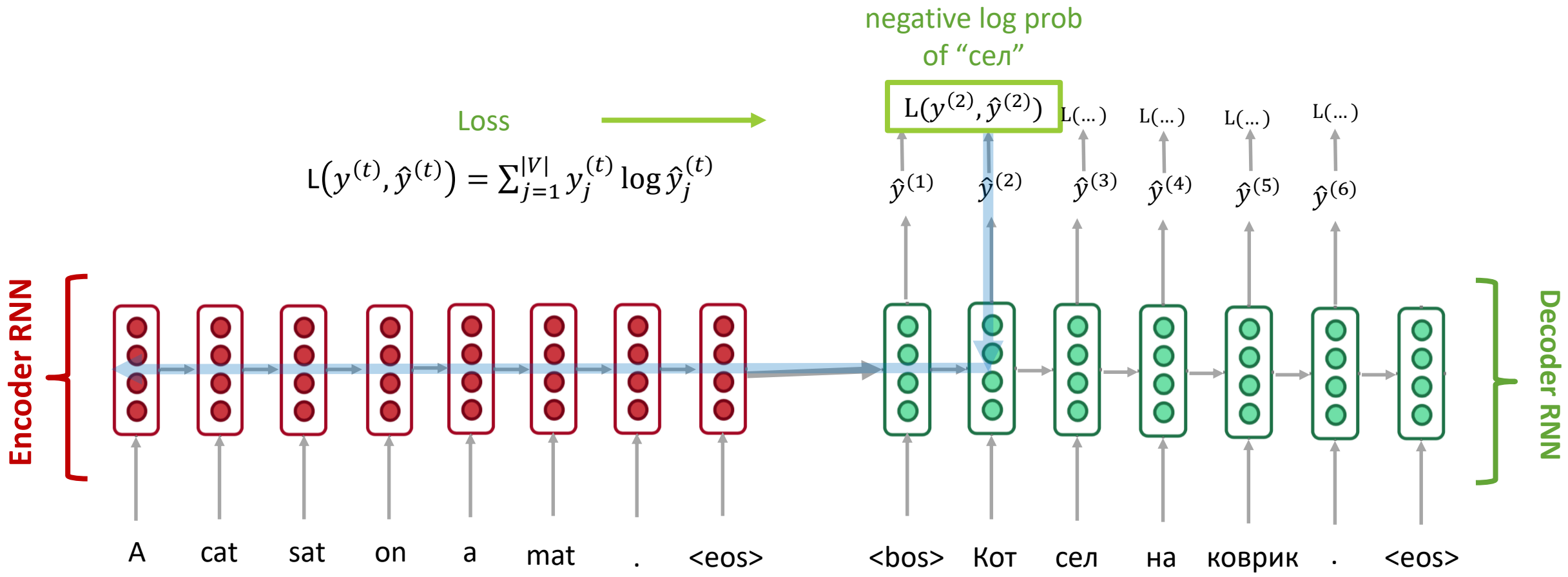
RNN Encoder-Decoder (Vanilla)

RNN Encoder-Decoder (Vanilla)

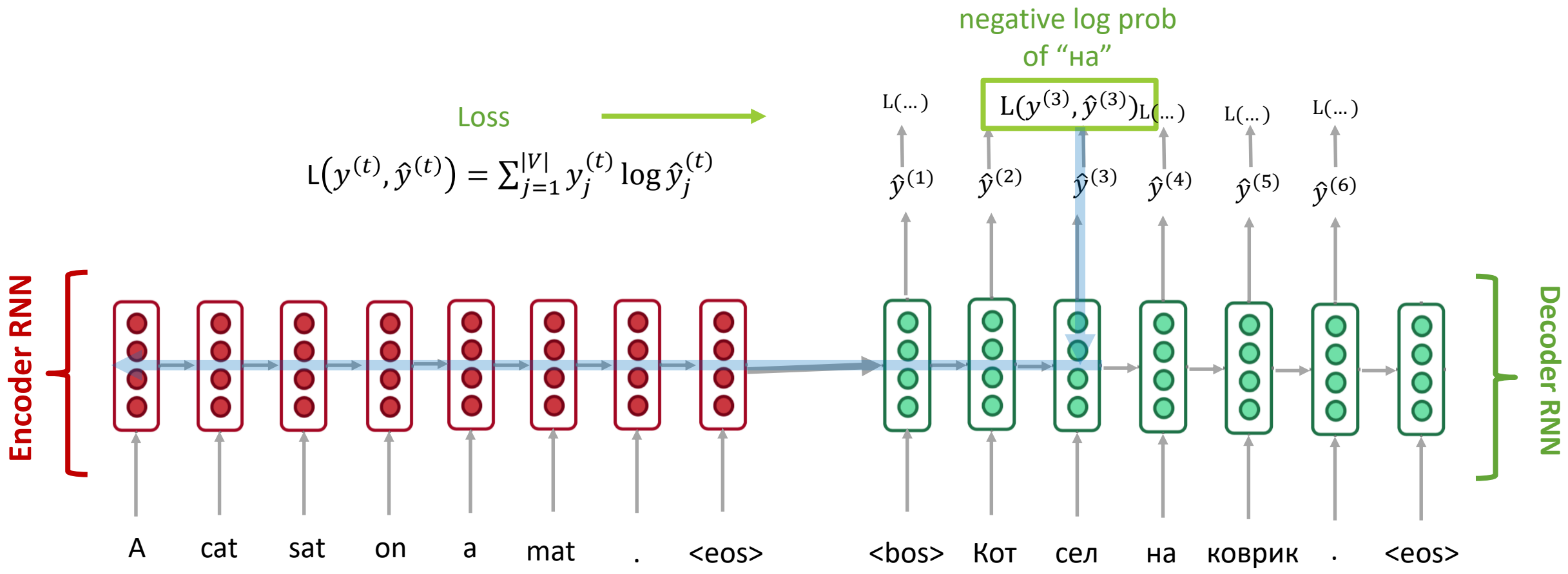
RNN Encoder-Decoder (Vanilla)



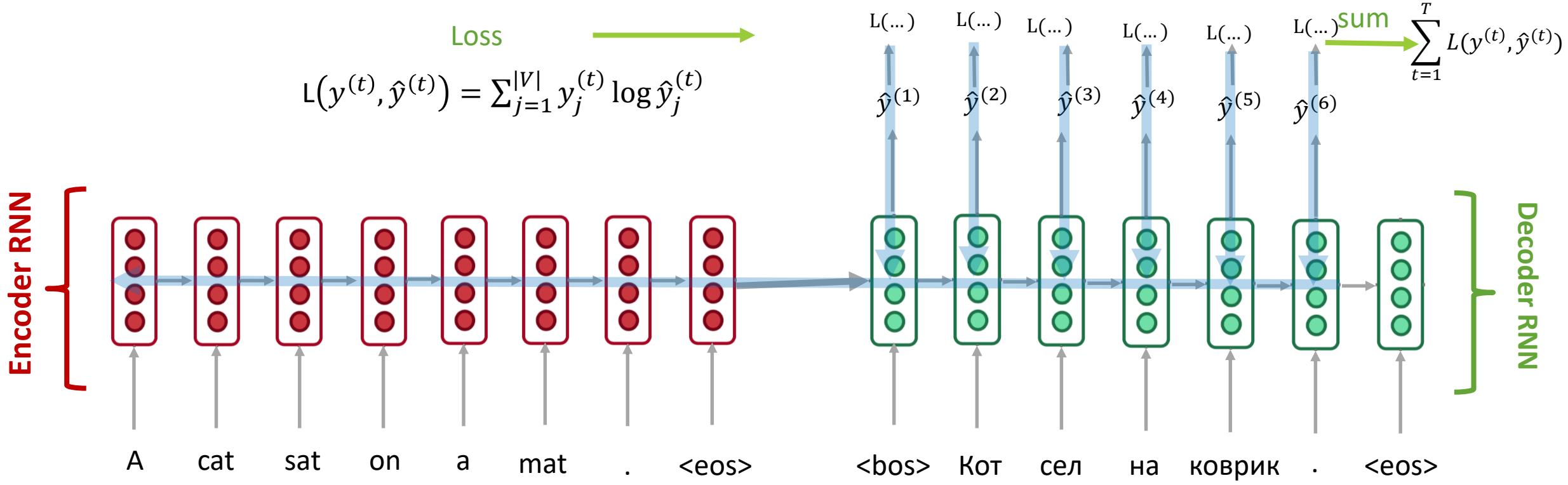
RNN Encoder-Decoder (Vanilla)



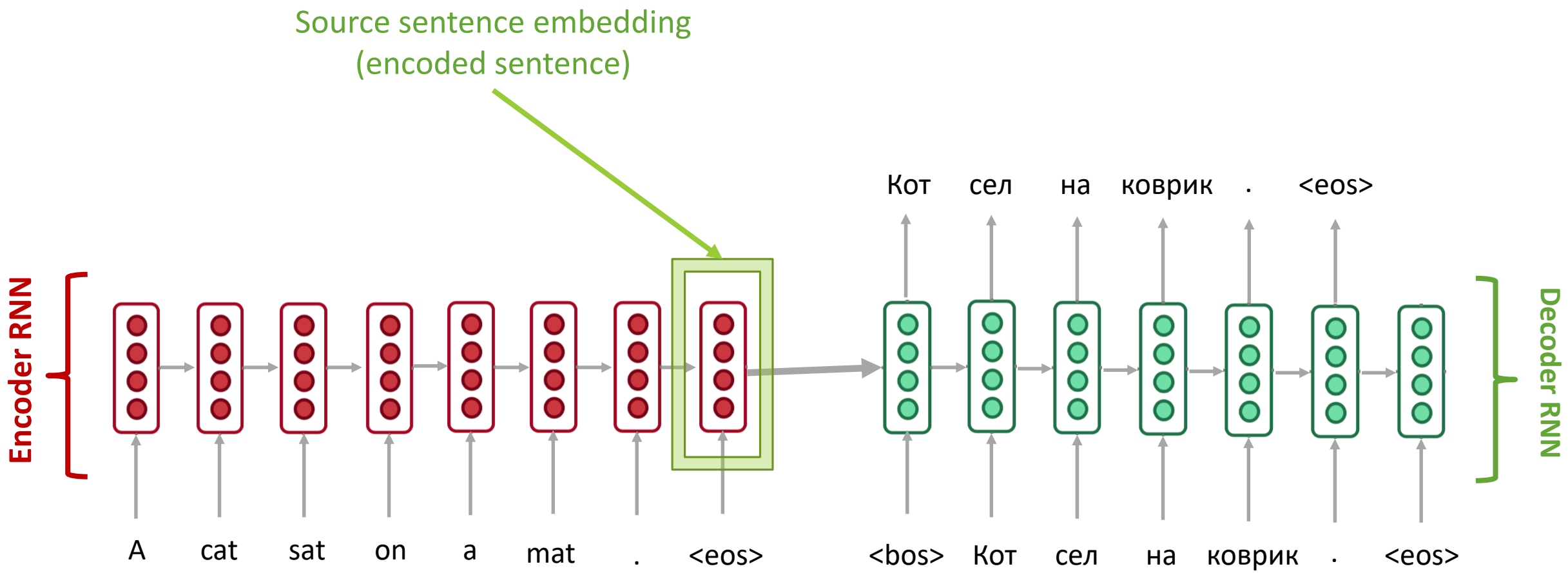
RNN Encoder-Decoder (Vanilla)



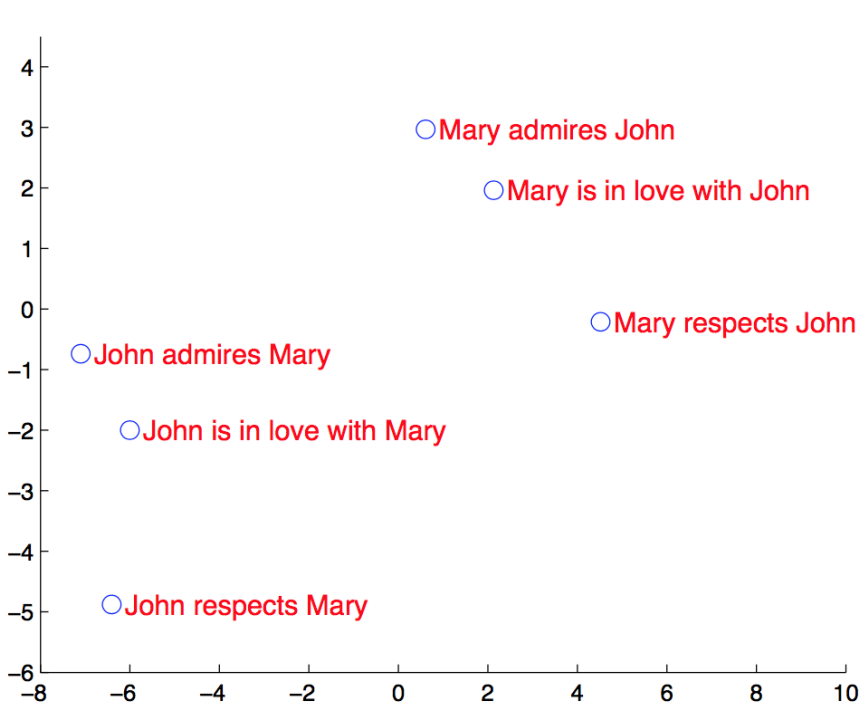
RNN Encoder-Decoder (Vanilla)

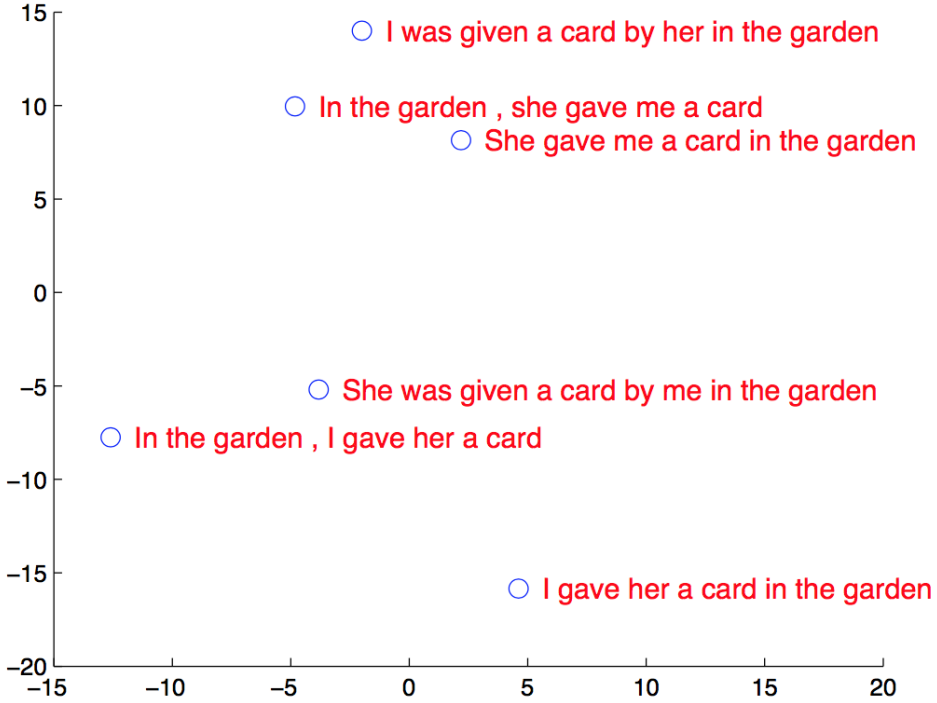


Let’s look at sentence embedding



Let's look at sentence embedding

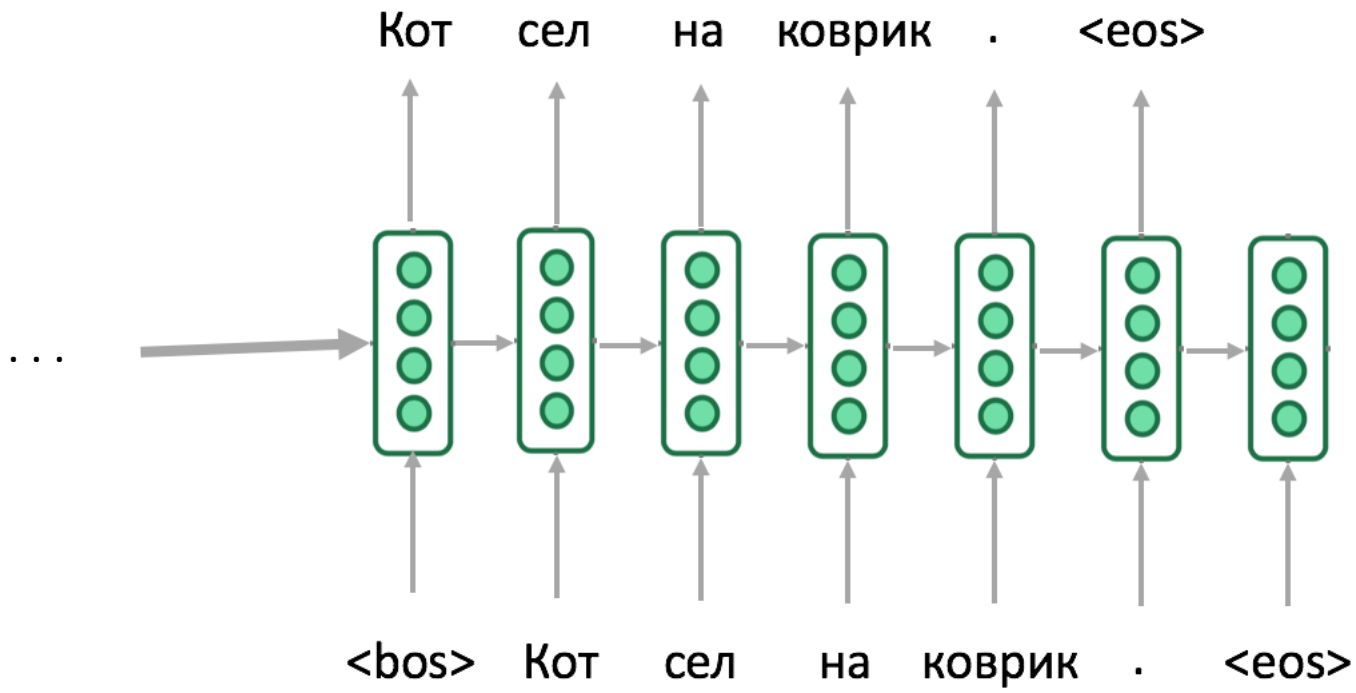




Sutskever et al, NIPS 2014, https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

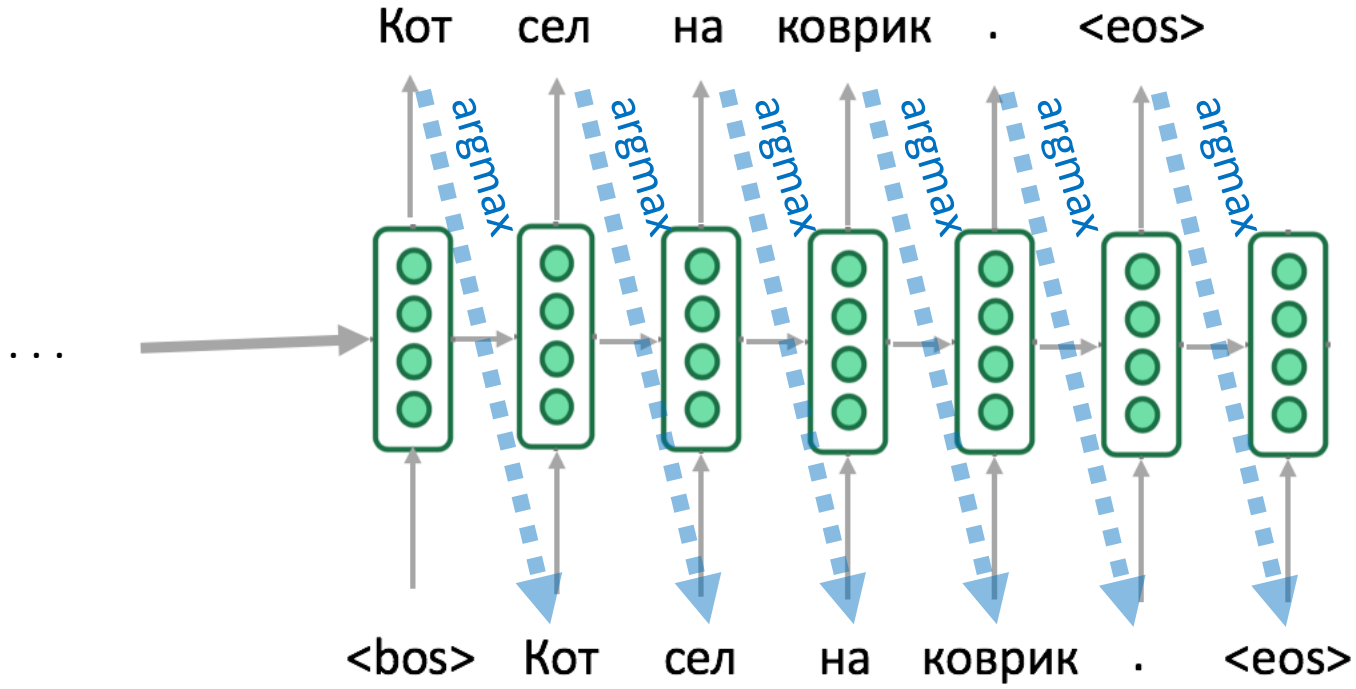
Decoding

Decoding: how to generate a sequence?



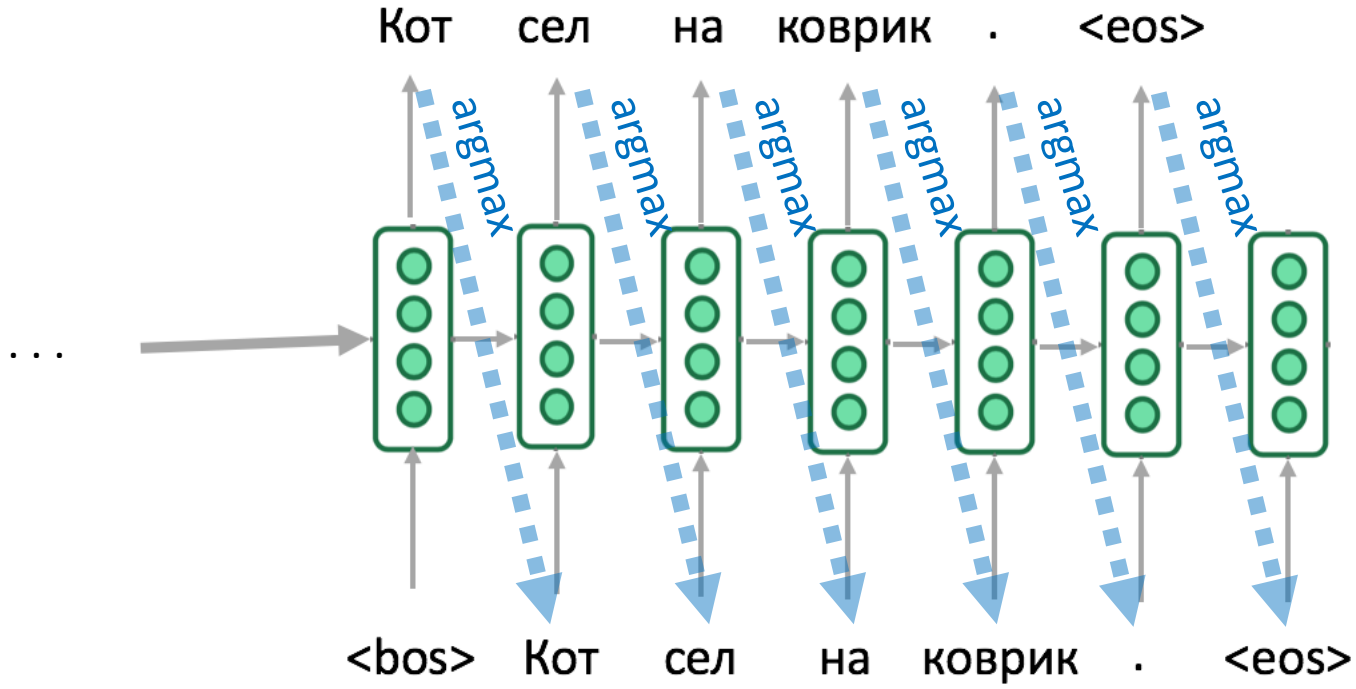
Decoding: how to generate a sequence?

1. Greedy decoding

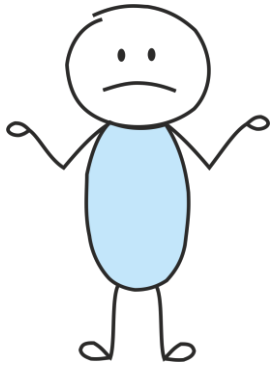


Decoding: how to generate a sequence?

1. Greedy decoding



Is it really good?



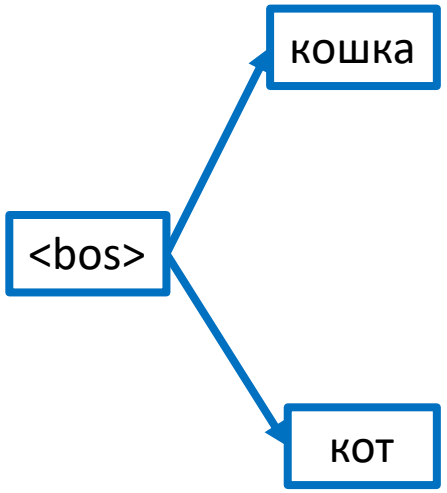
Decoding: how to generate a sequence?

2. Beam search

<bos>

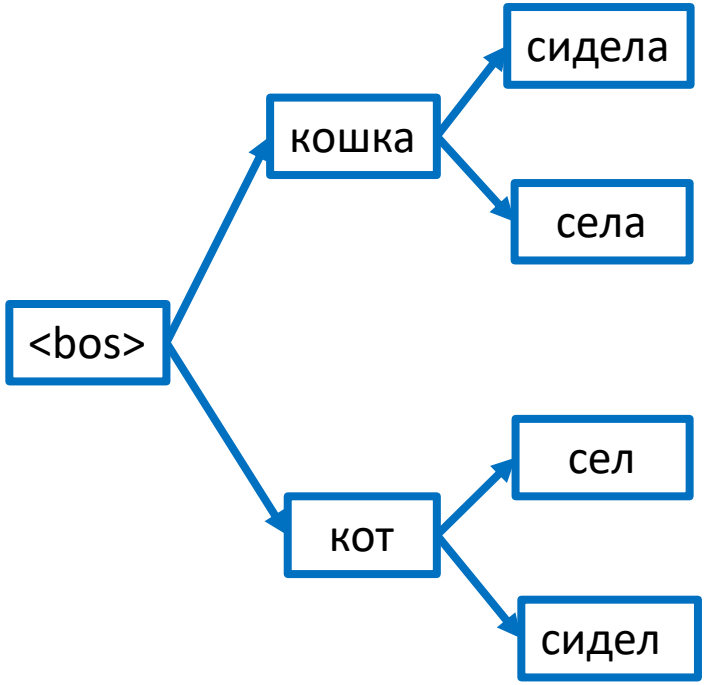
Decoding: how to generate a sequence?

2. Beam search



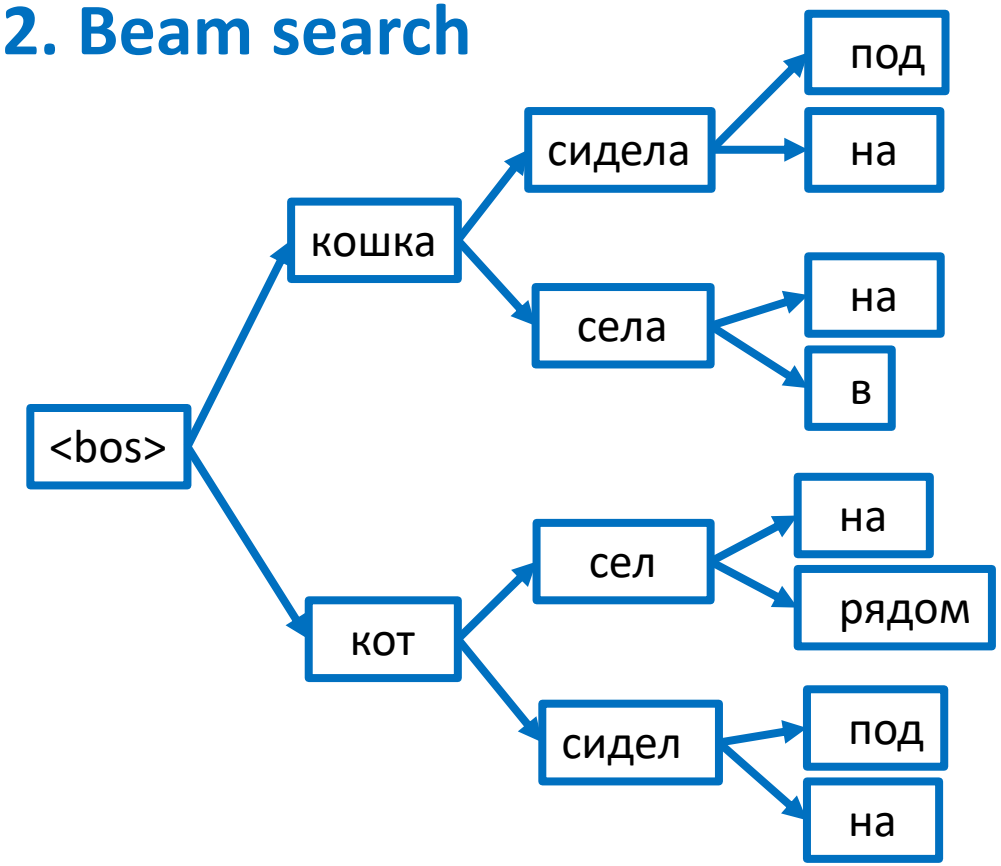
Decoding: how to generate a sequence?

2. Beam search



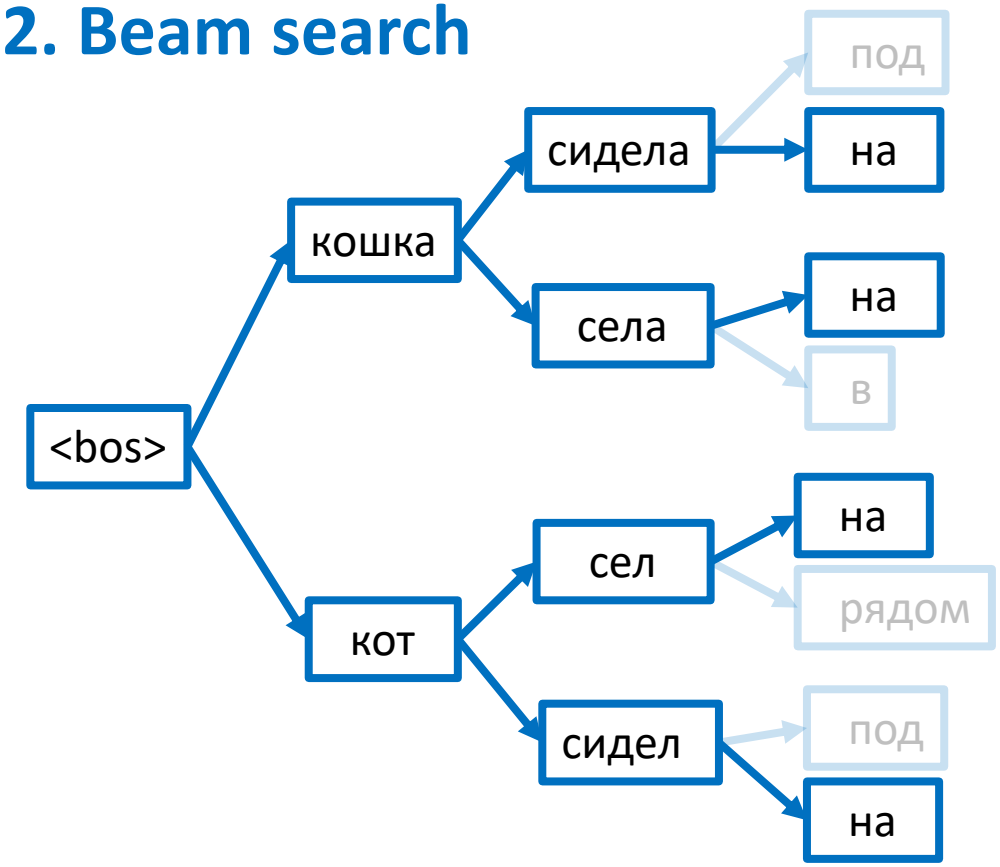
Decoding: how to generate a sequence?

2. Beam search



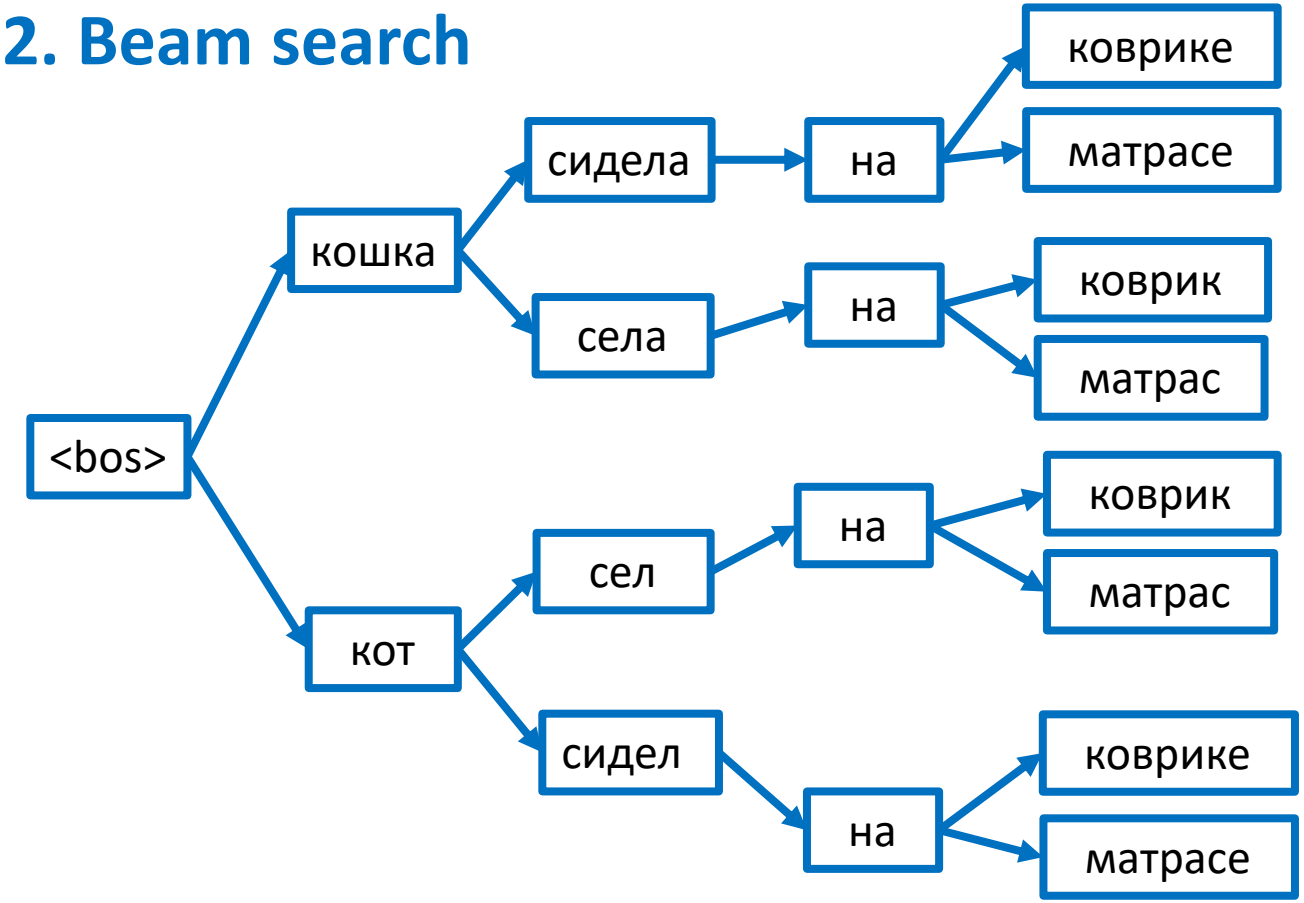
Decoding: how to generate a sequence?

2. Beam search



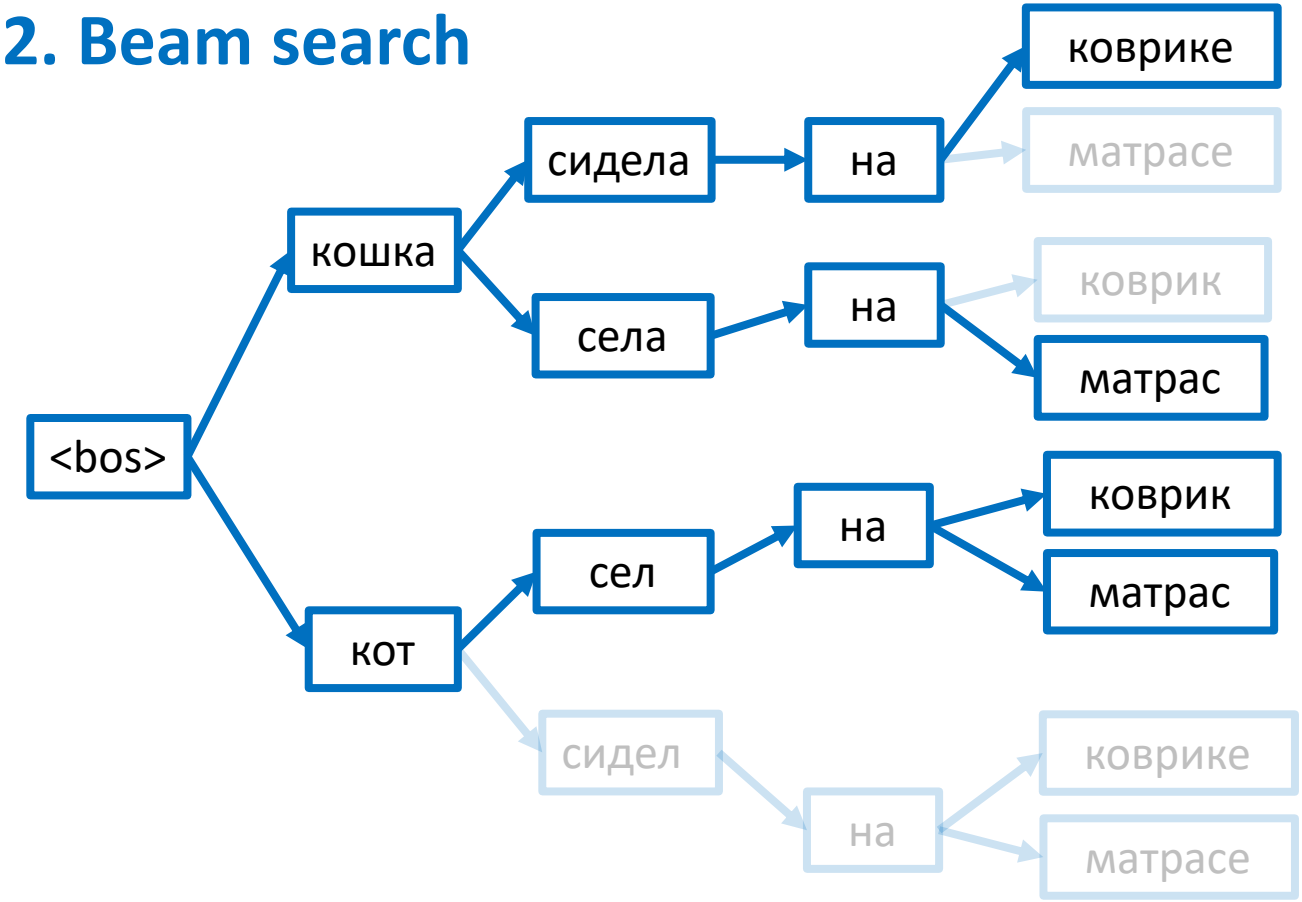
Decoding: how to generate a sequence?

2. Beam search



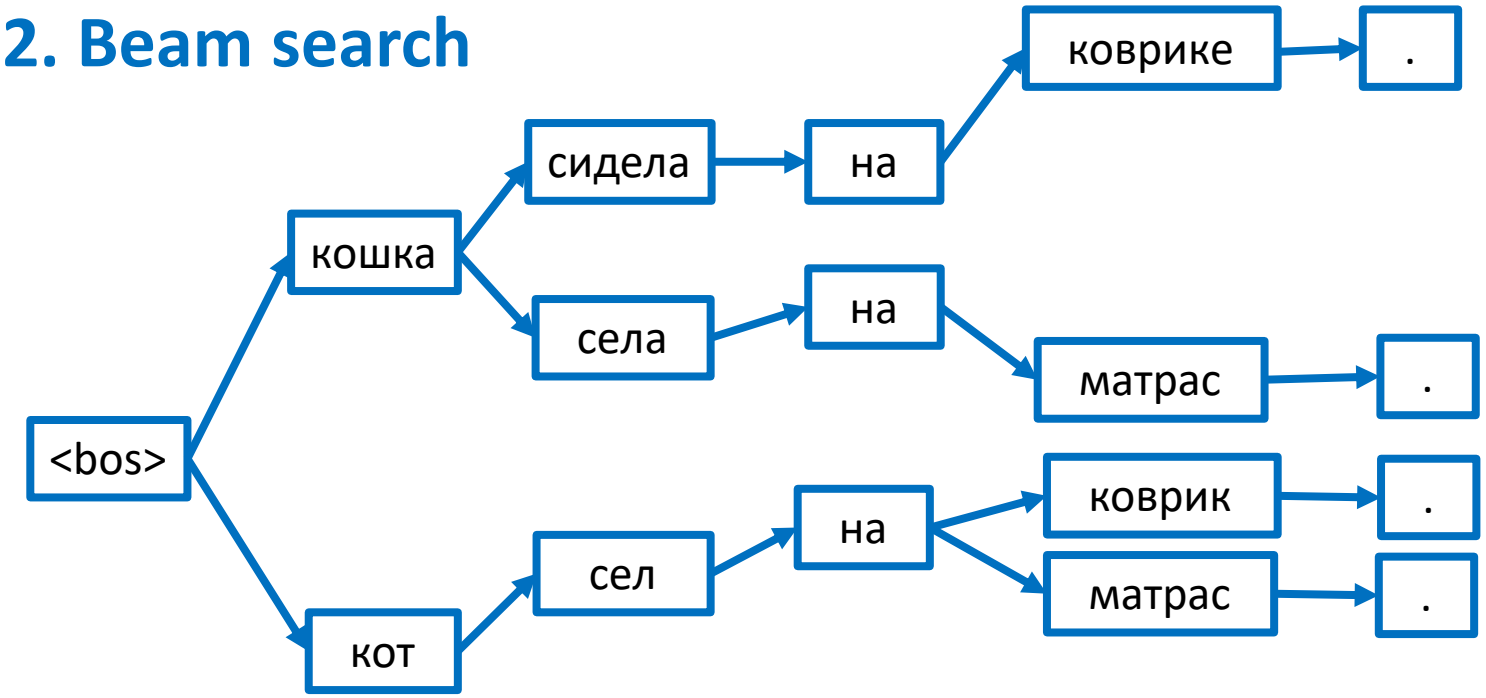
Decoding: how to generate a sequence?

2. Beam search



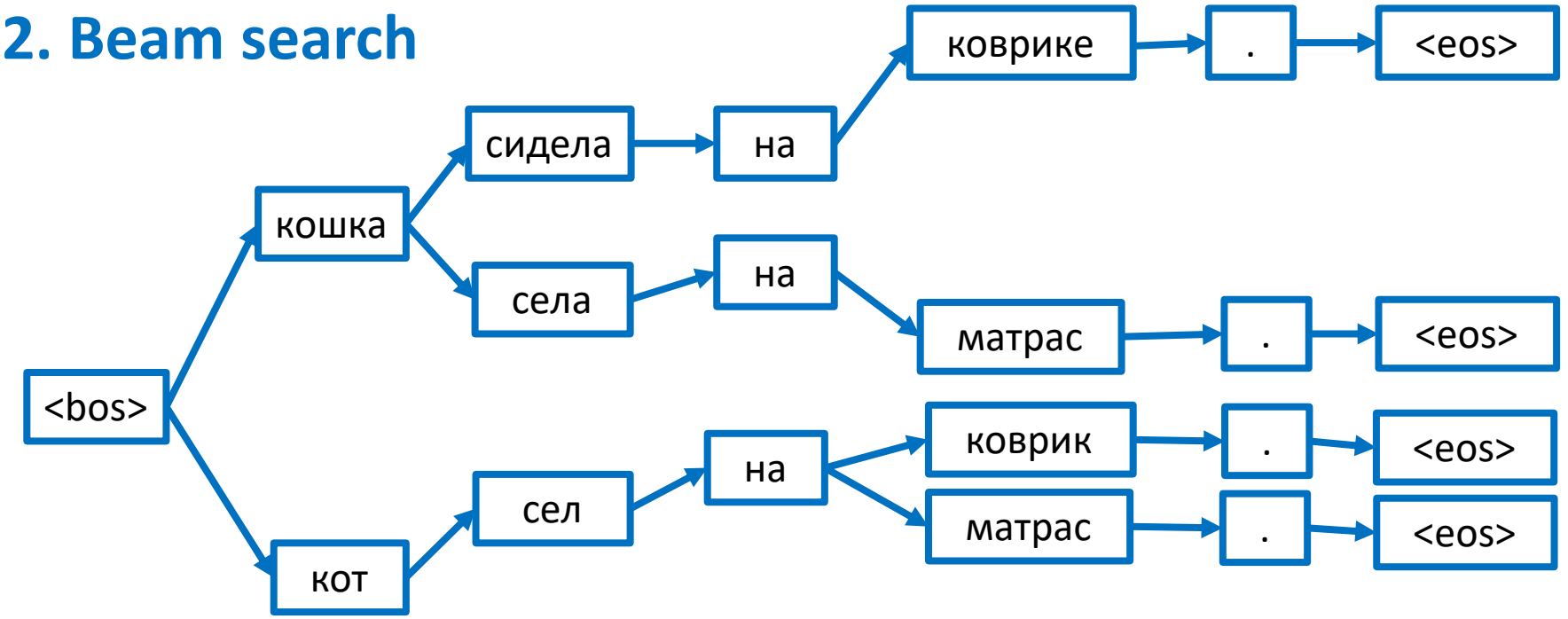
Decoding: how to generate a sequence?

2. Beam search



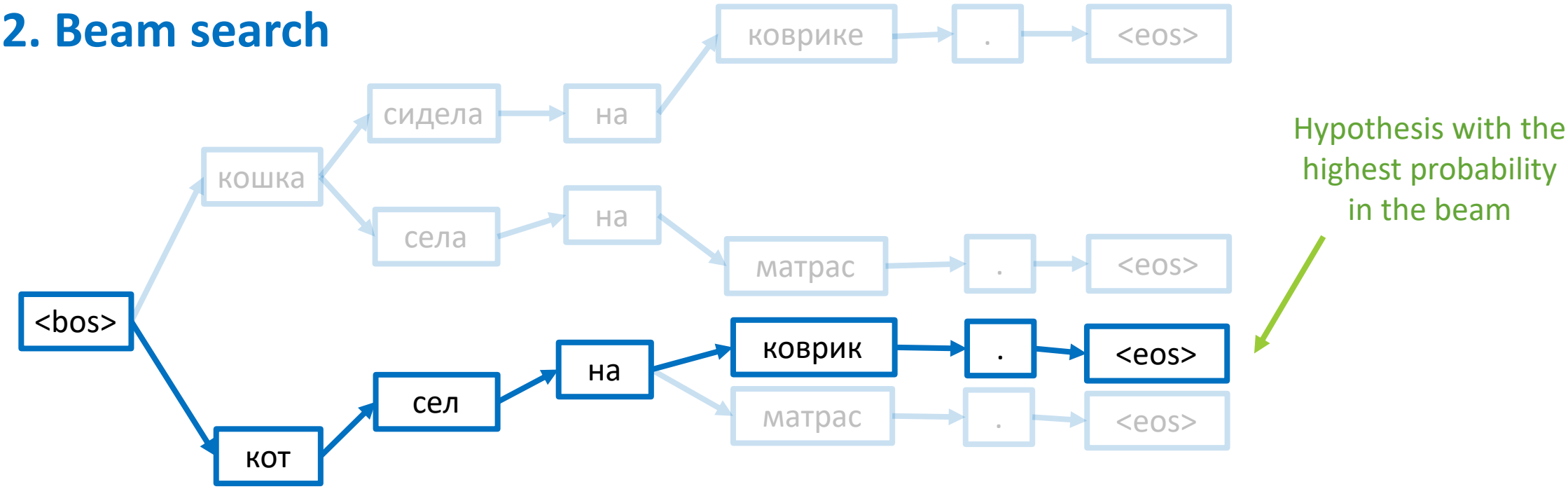
Decoding: how to generate a sequence?

2. Beam search



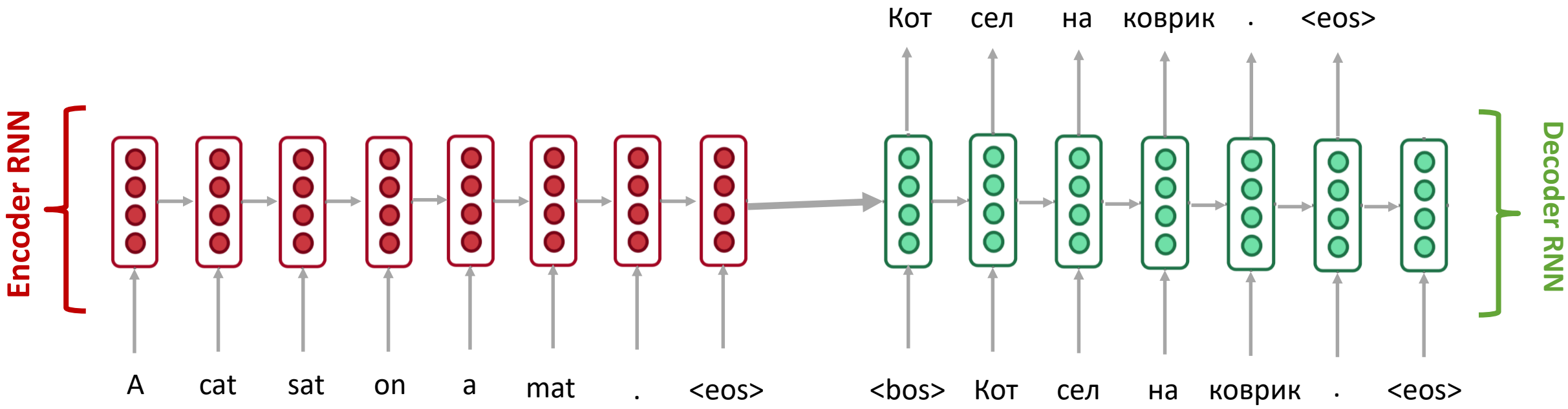
Decoding: how to generate a sequence?

2. Beam search



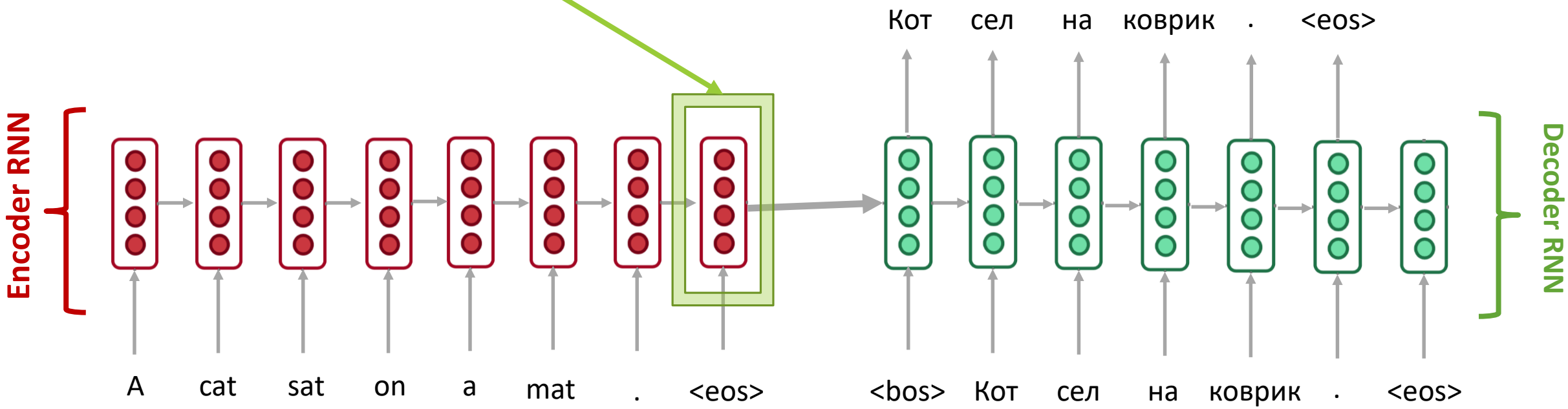
Attention-based models

What is the problem with such models?

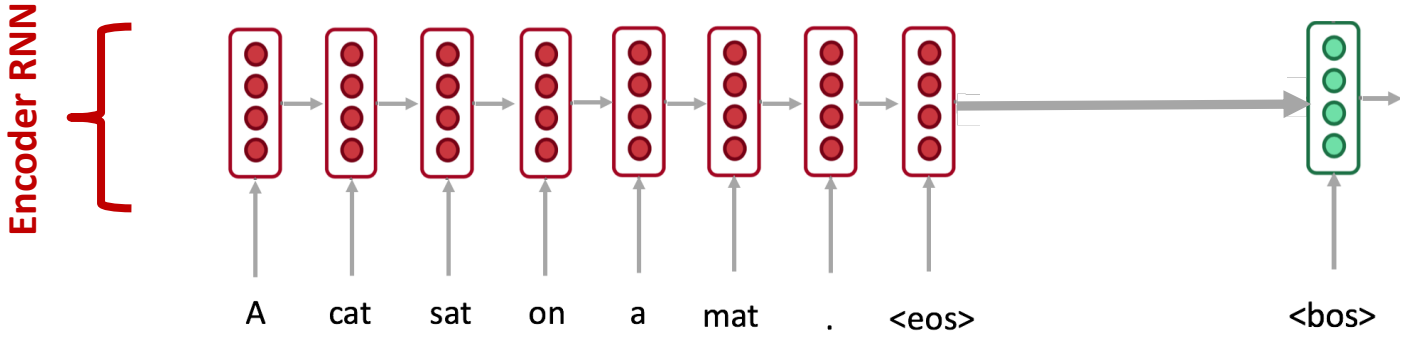


What is the problem with such models?

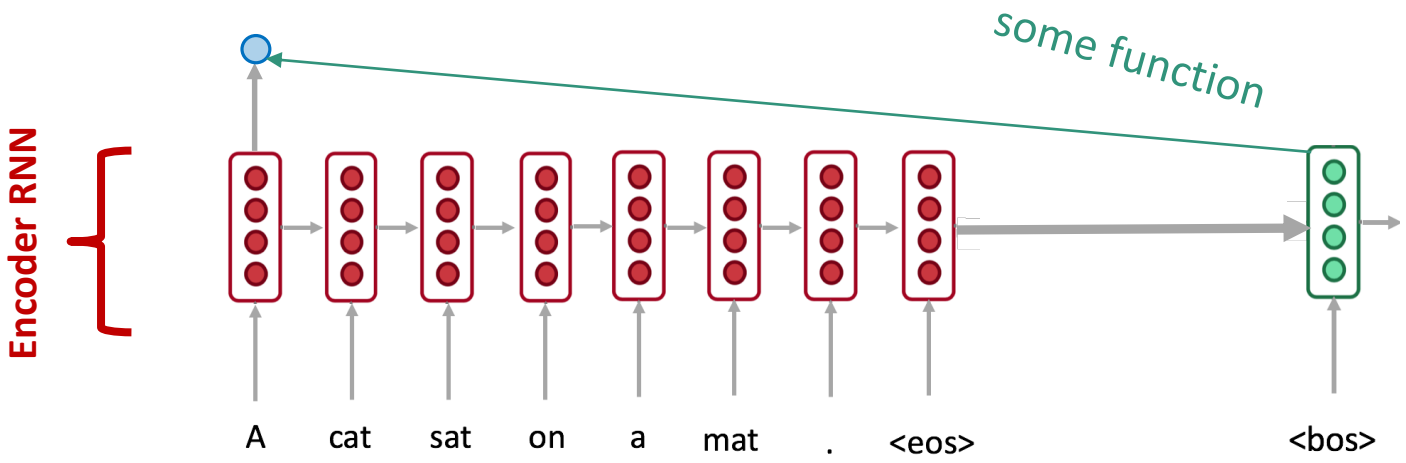
Information bottleneck:
the model has to put all information about sentence in one vector



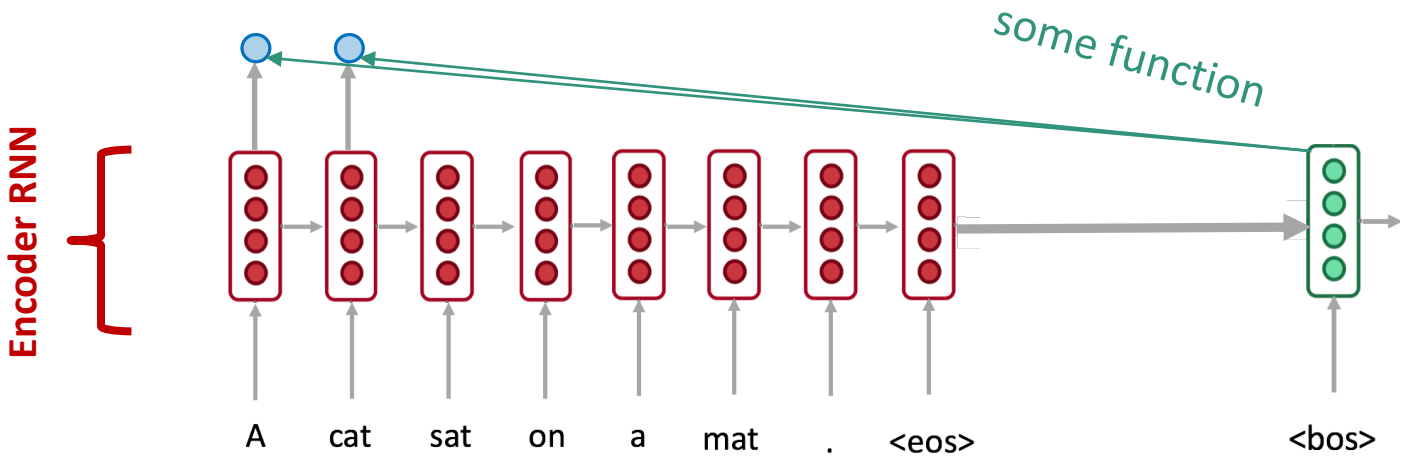
Attention



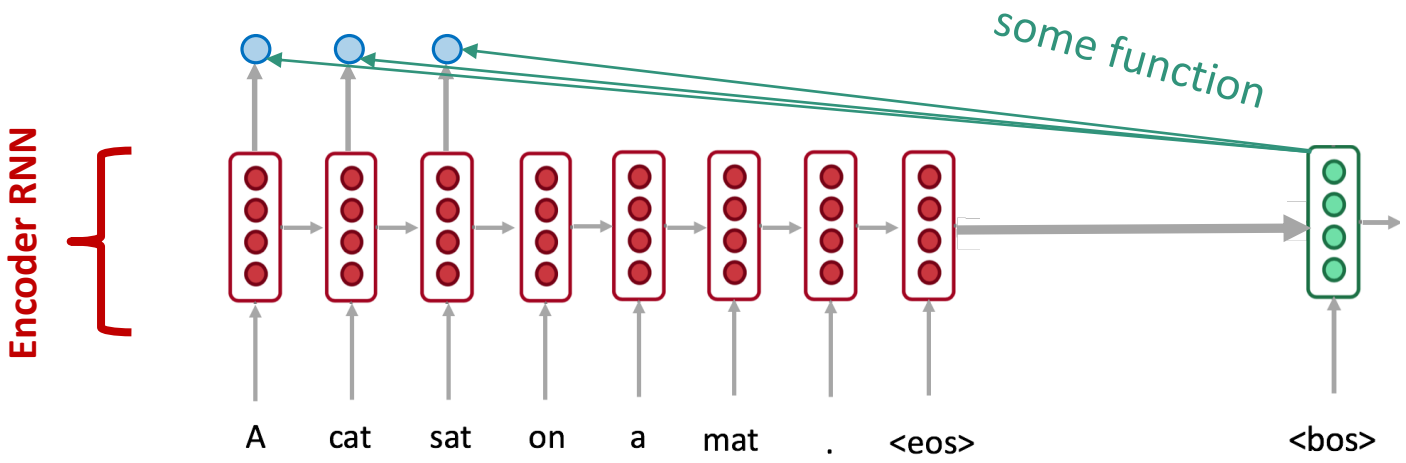
Attention



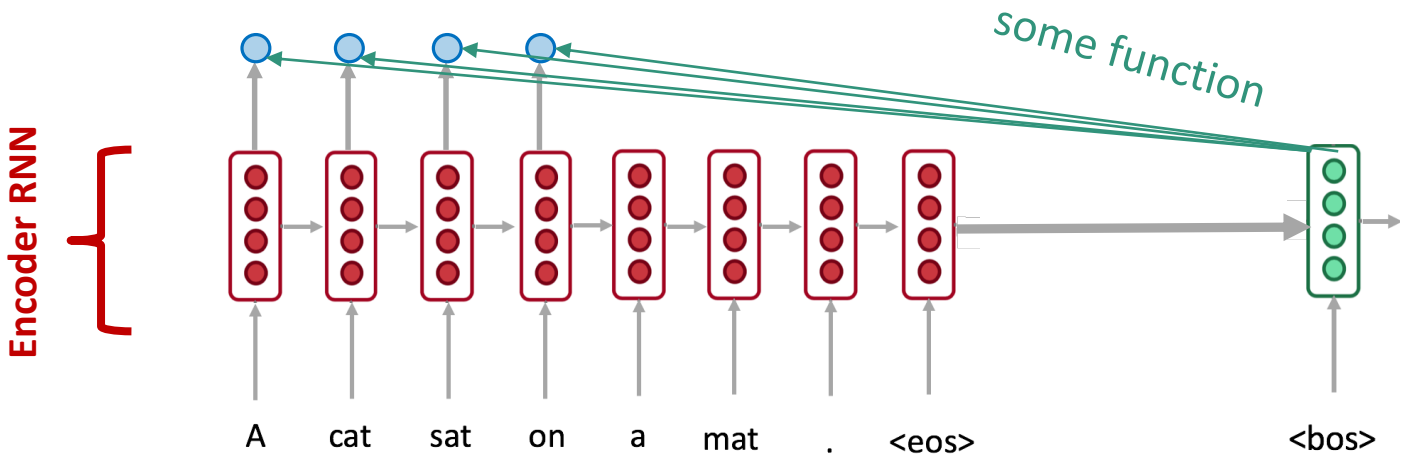
Attention



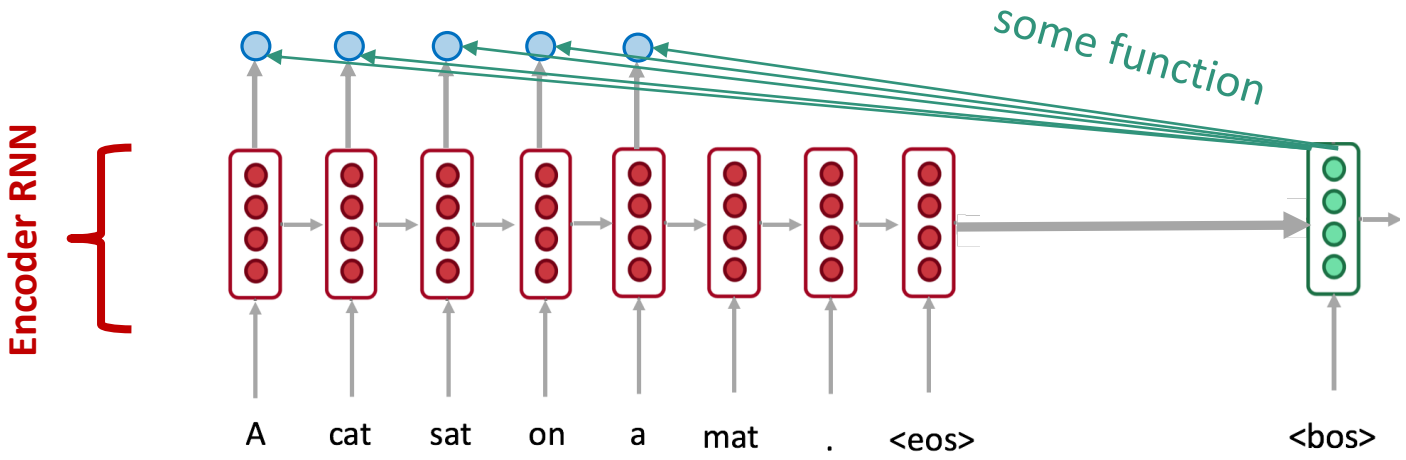
Attention



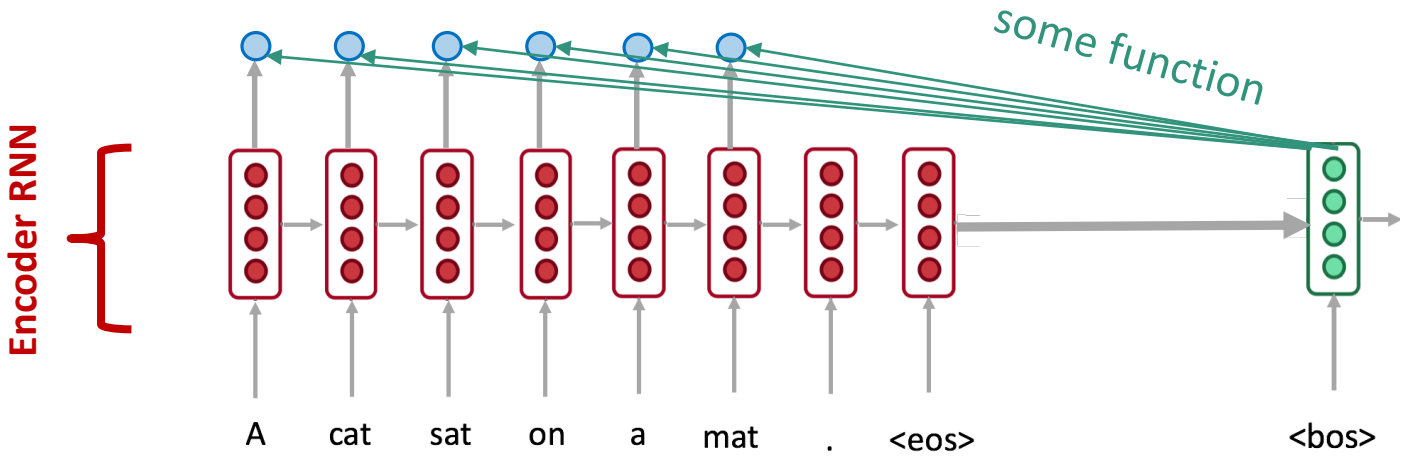
Attention



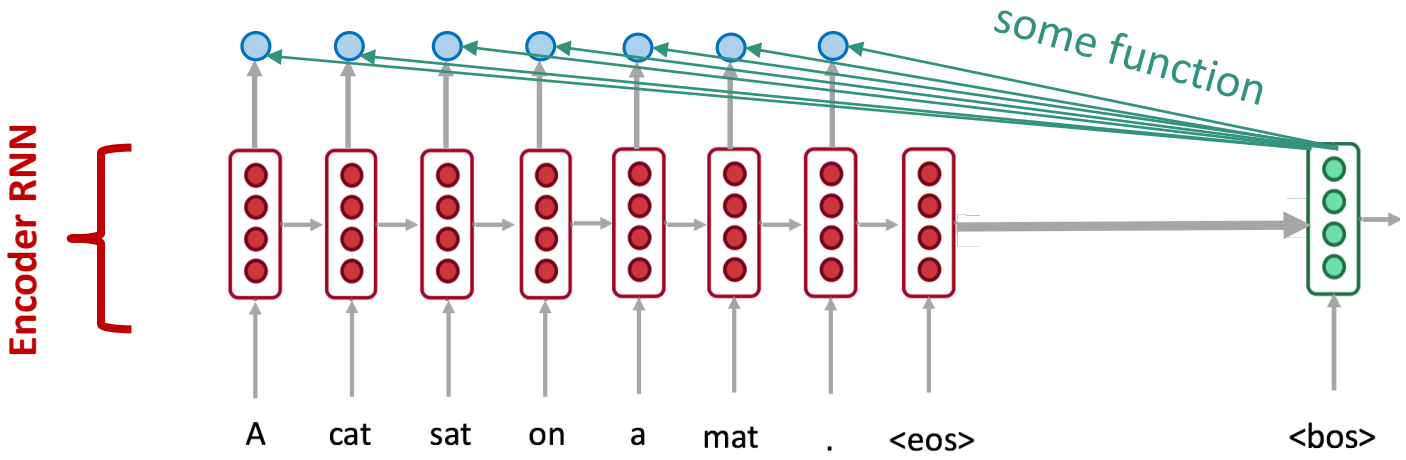
Attention



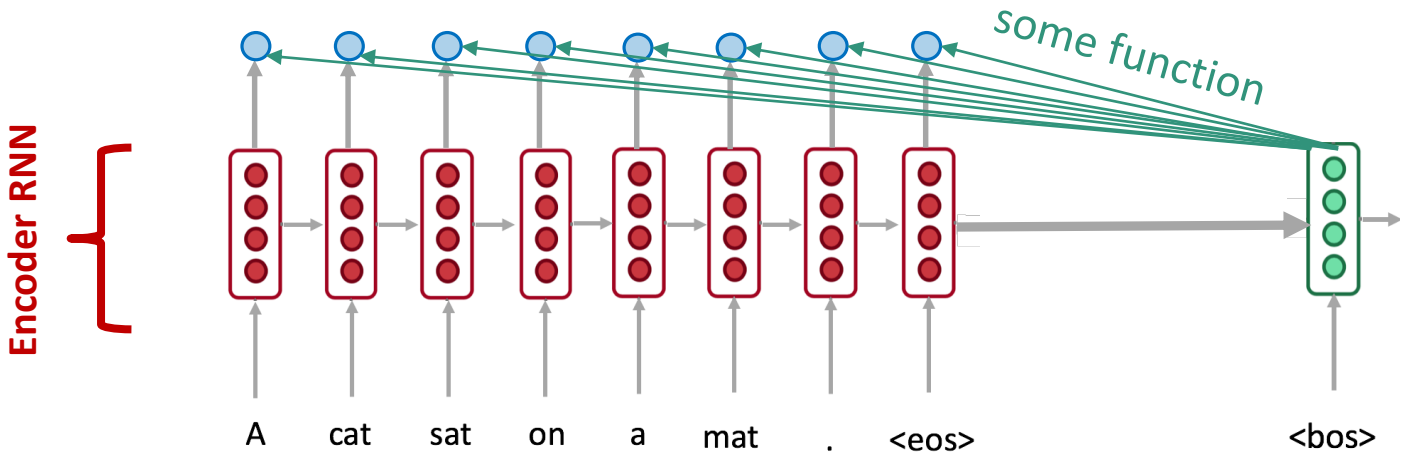
Attention



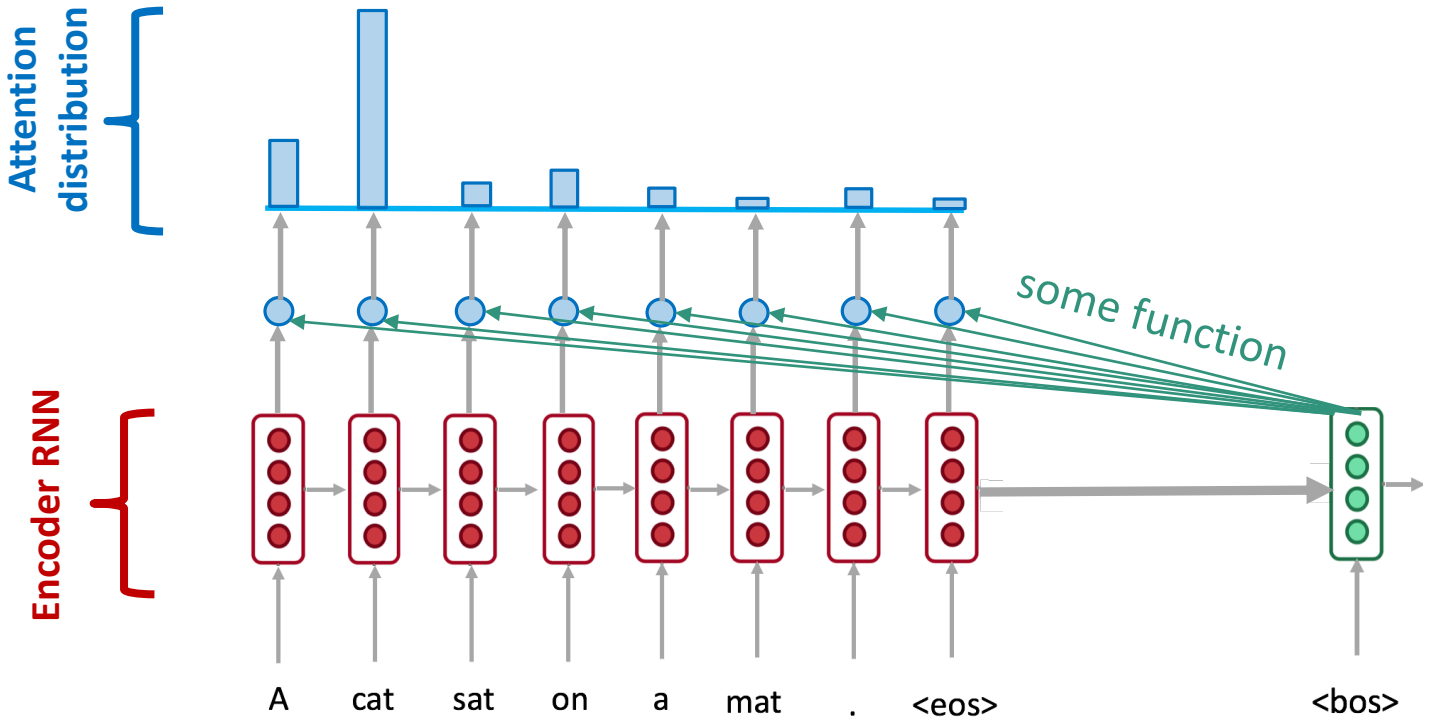
Attention



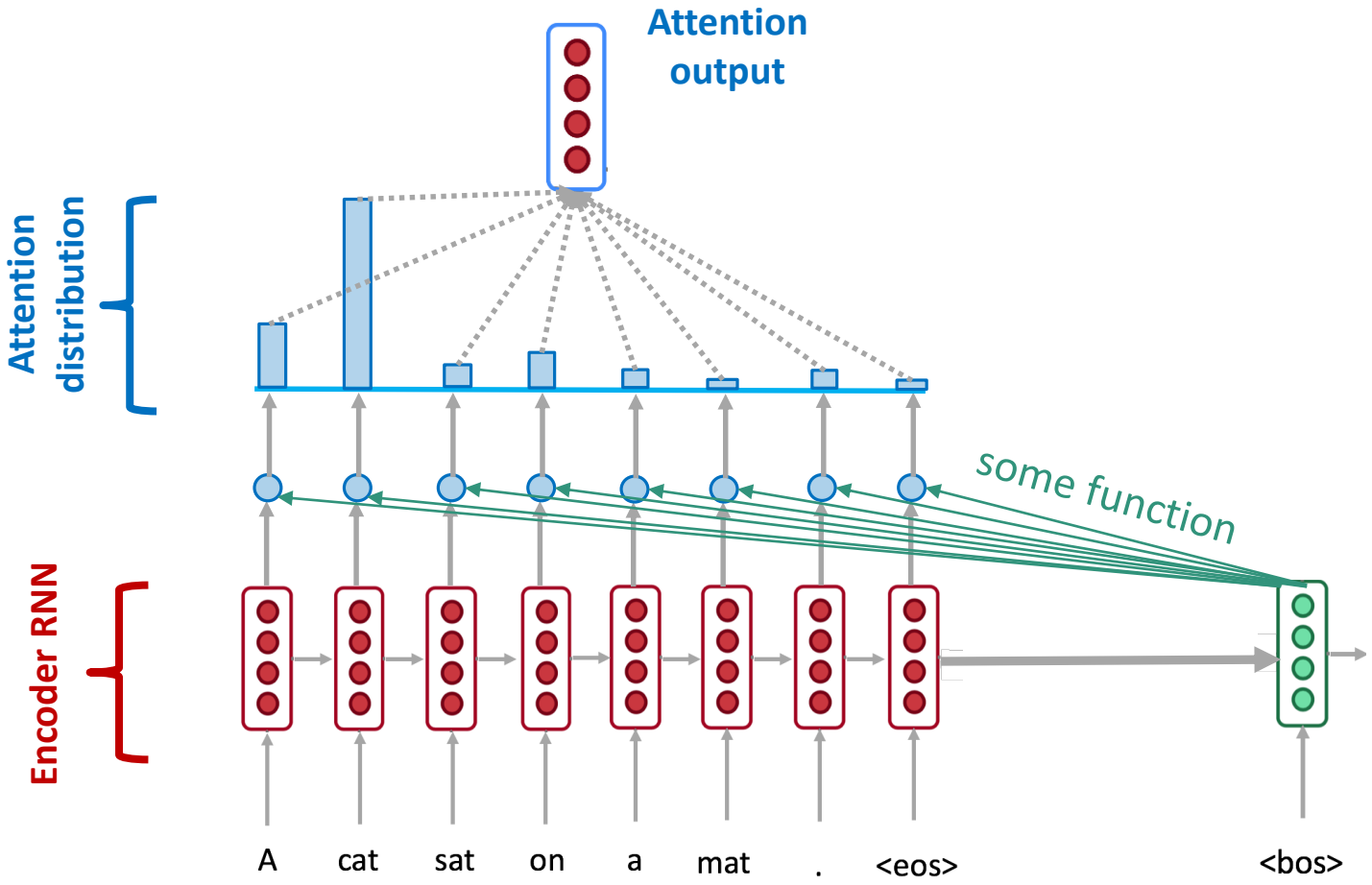
Attention



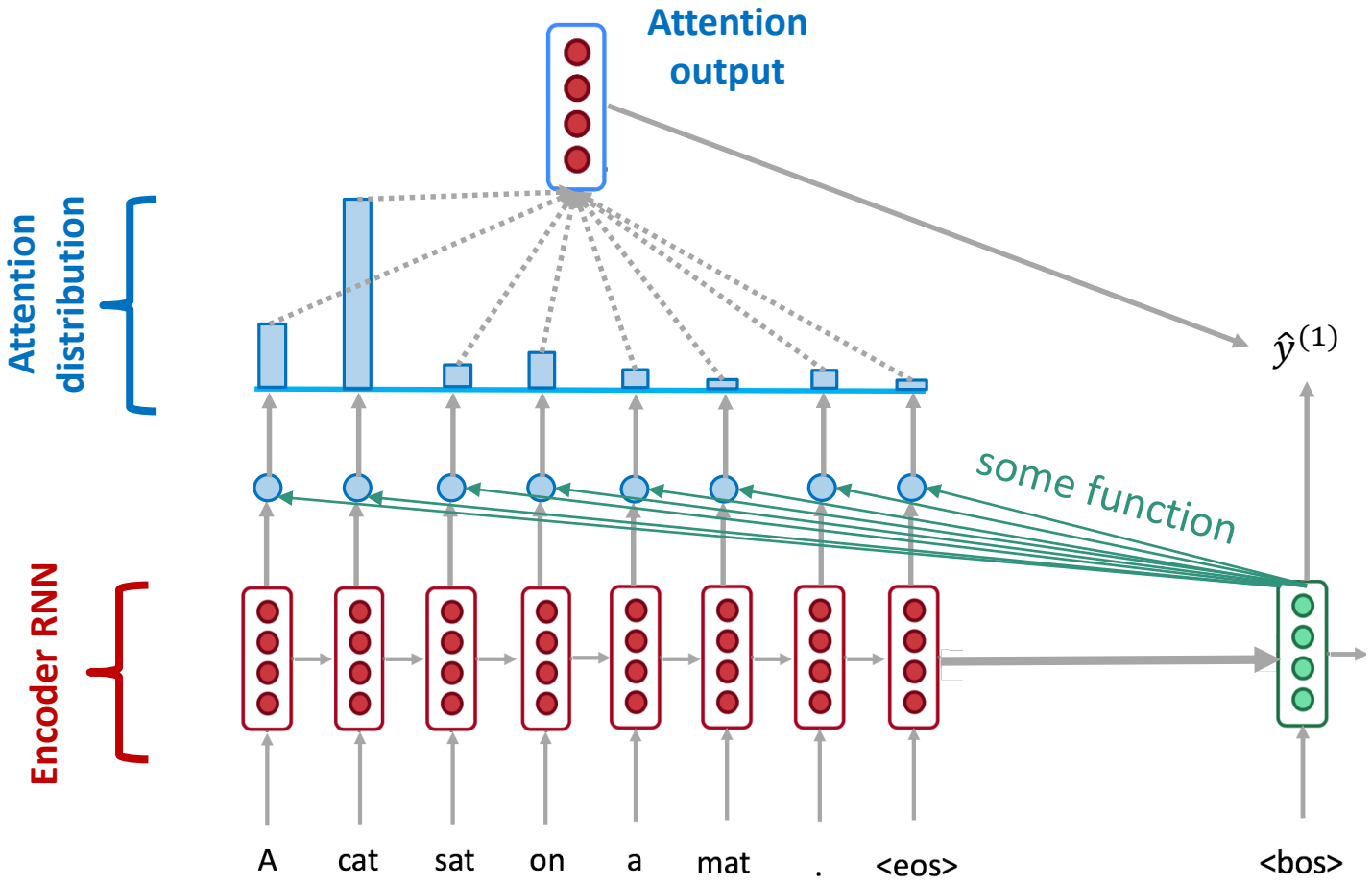
Attention


Attention distribution
Encoder RNN
some function
A
cat
sat
on
a
mat
.
<eos>
<bos>

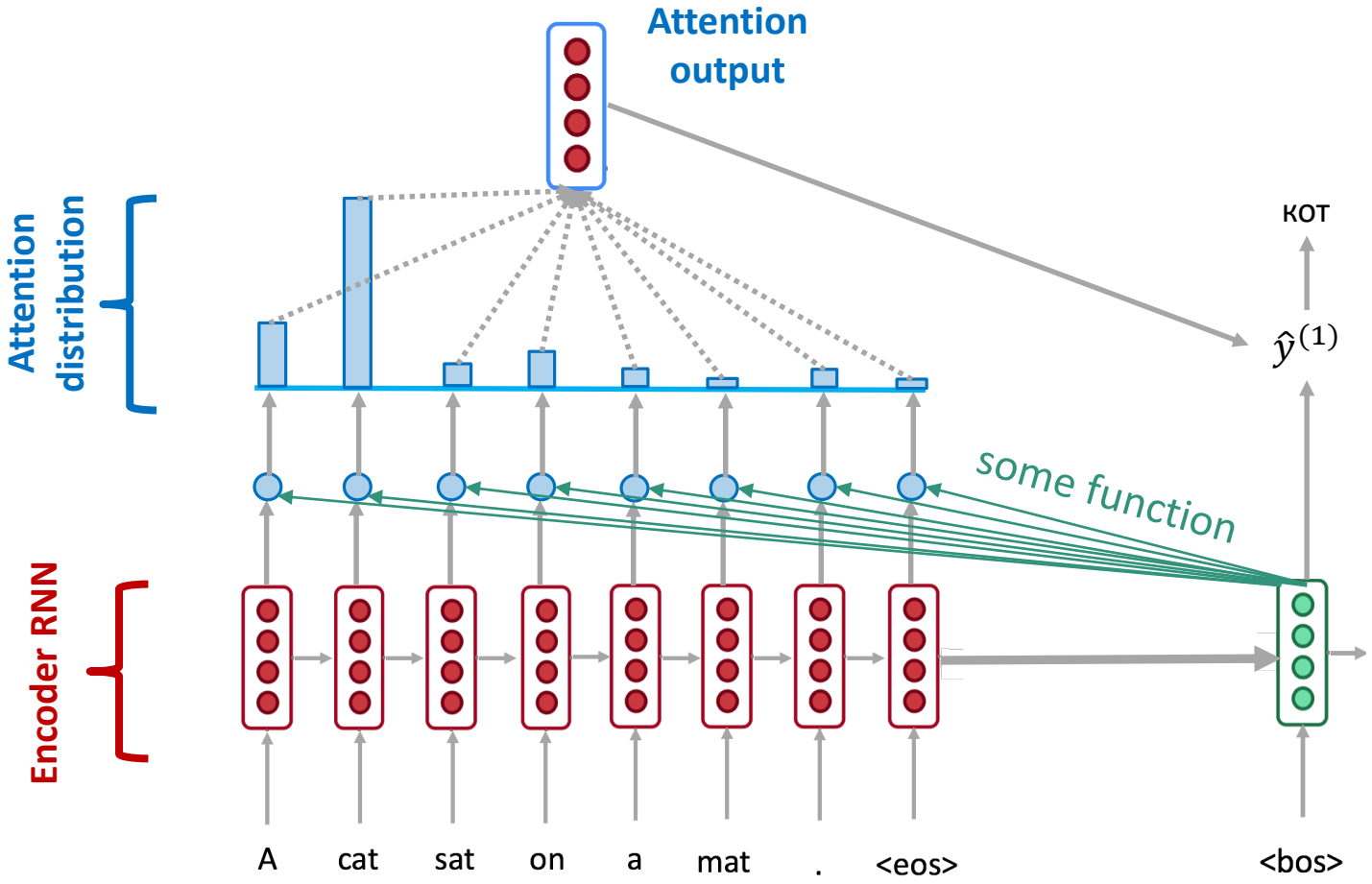
Attention


Attention output
Attention distribution
some function
Encoder RNN
A
cat
sat
on
a
mat
.
<eos>
<bos>

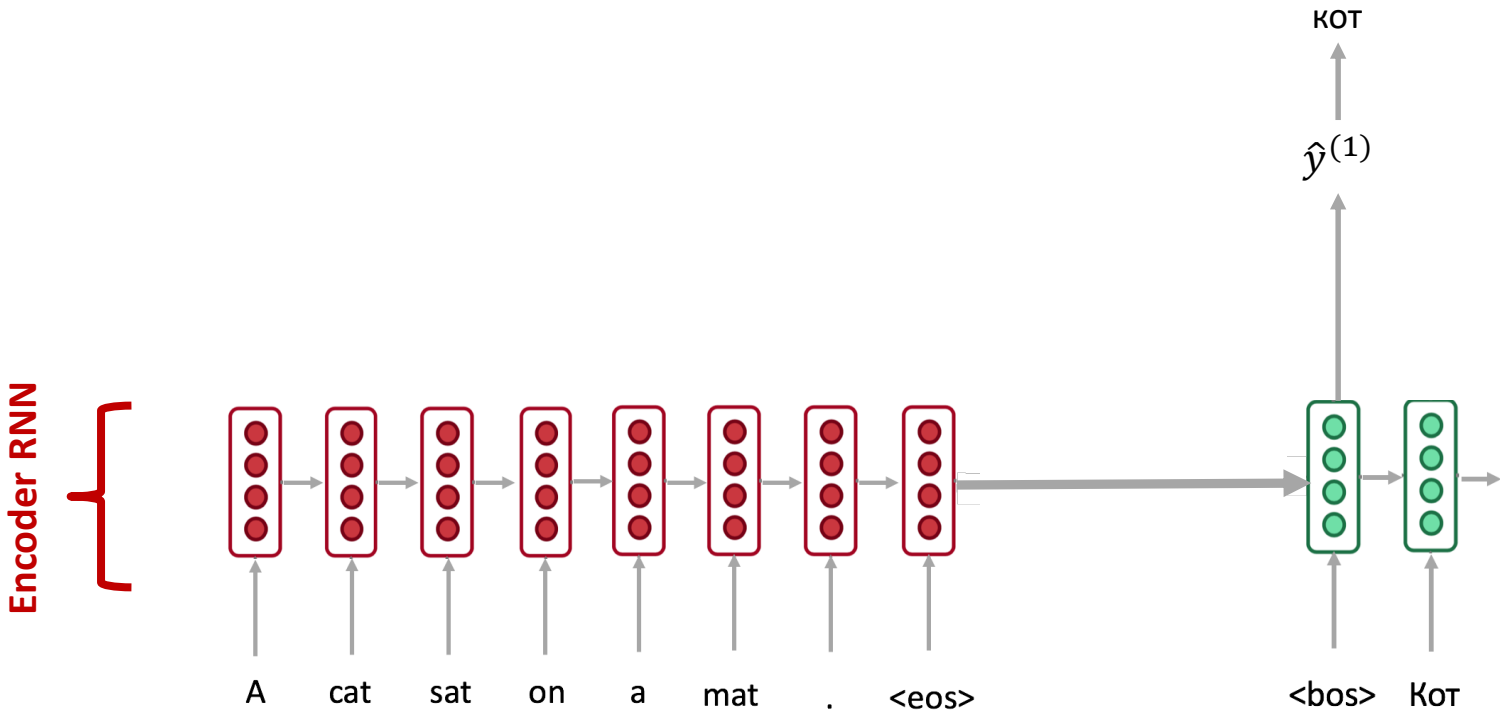
Attention


Attention output
Attention distribution
$\hat{y}^{(1)}$
some function
Encoder RNN
A
cat
sat
on
a
mat
.
<eos>
<bos>

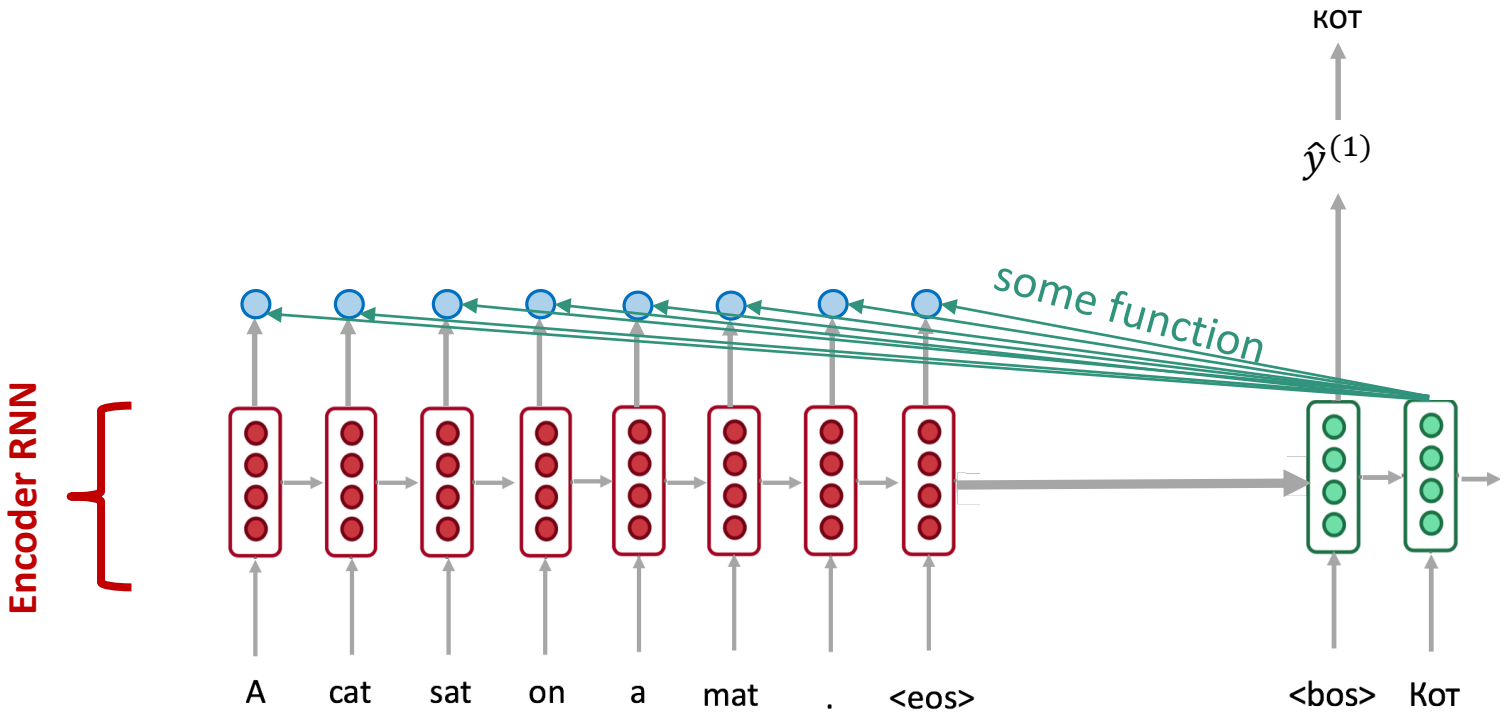
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
$\hat{y}^{(1)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>

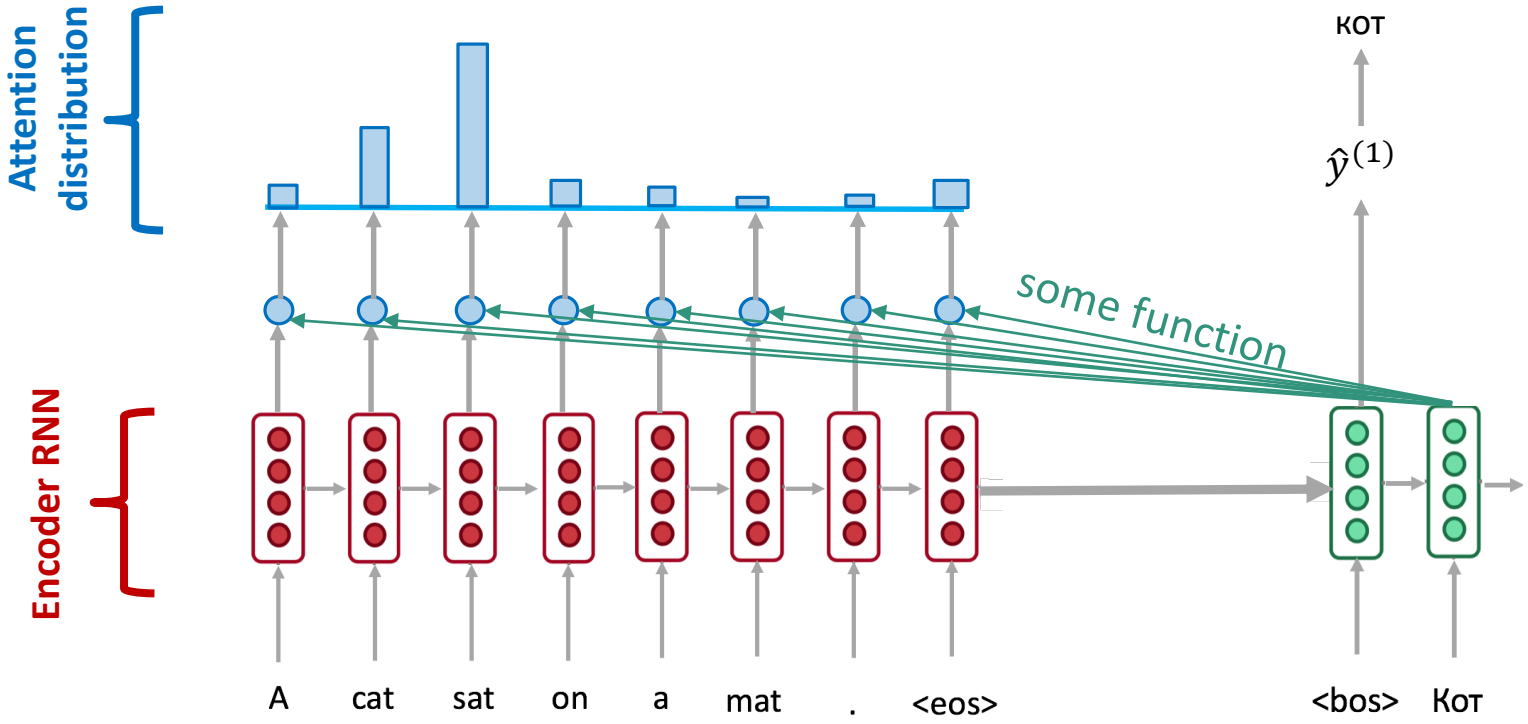
Attention



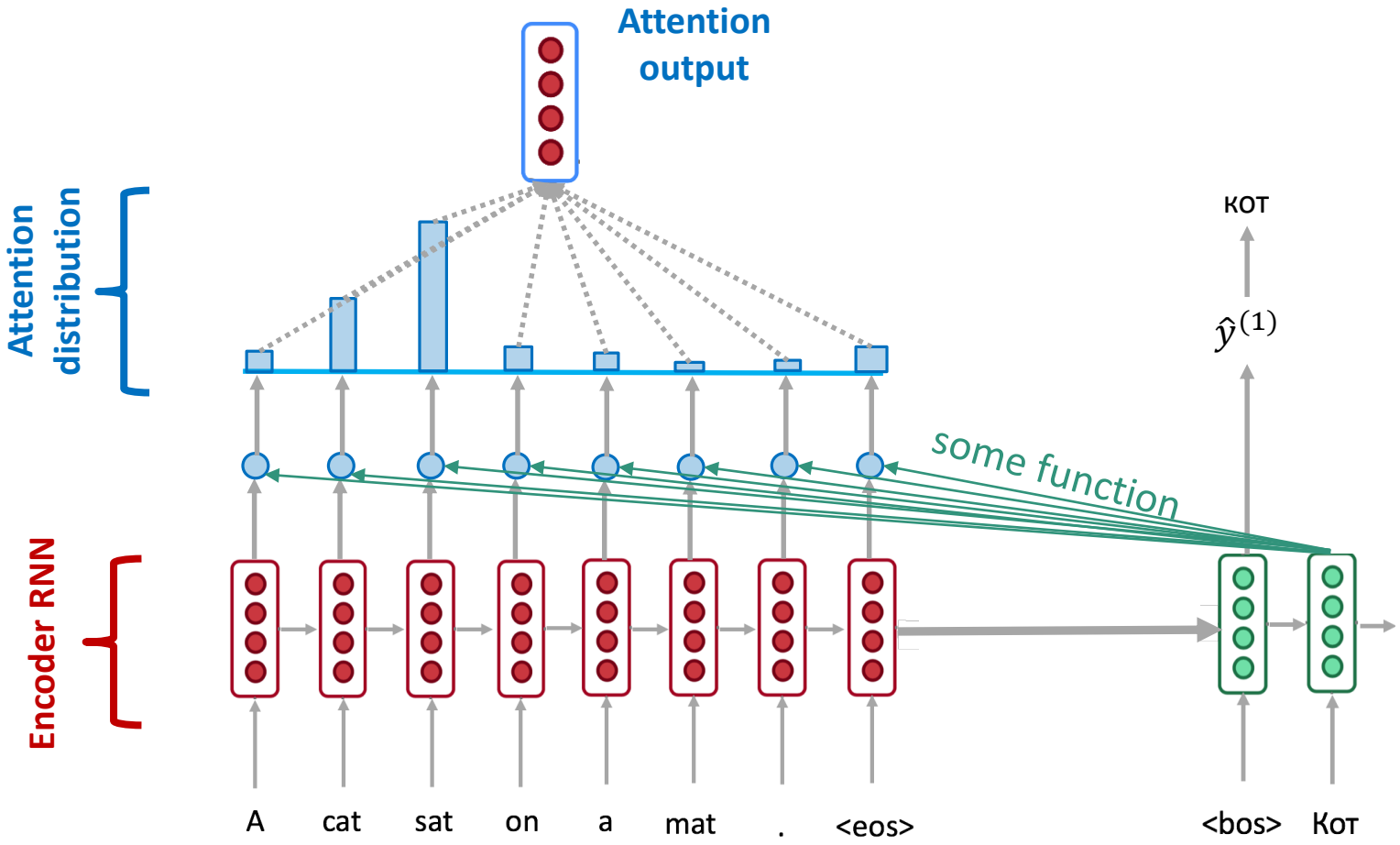
Attention


кот
$\hat{y}^{(1)}$
some function
Encoder RNN
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот

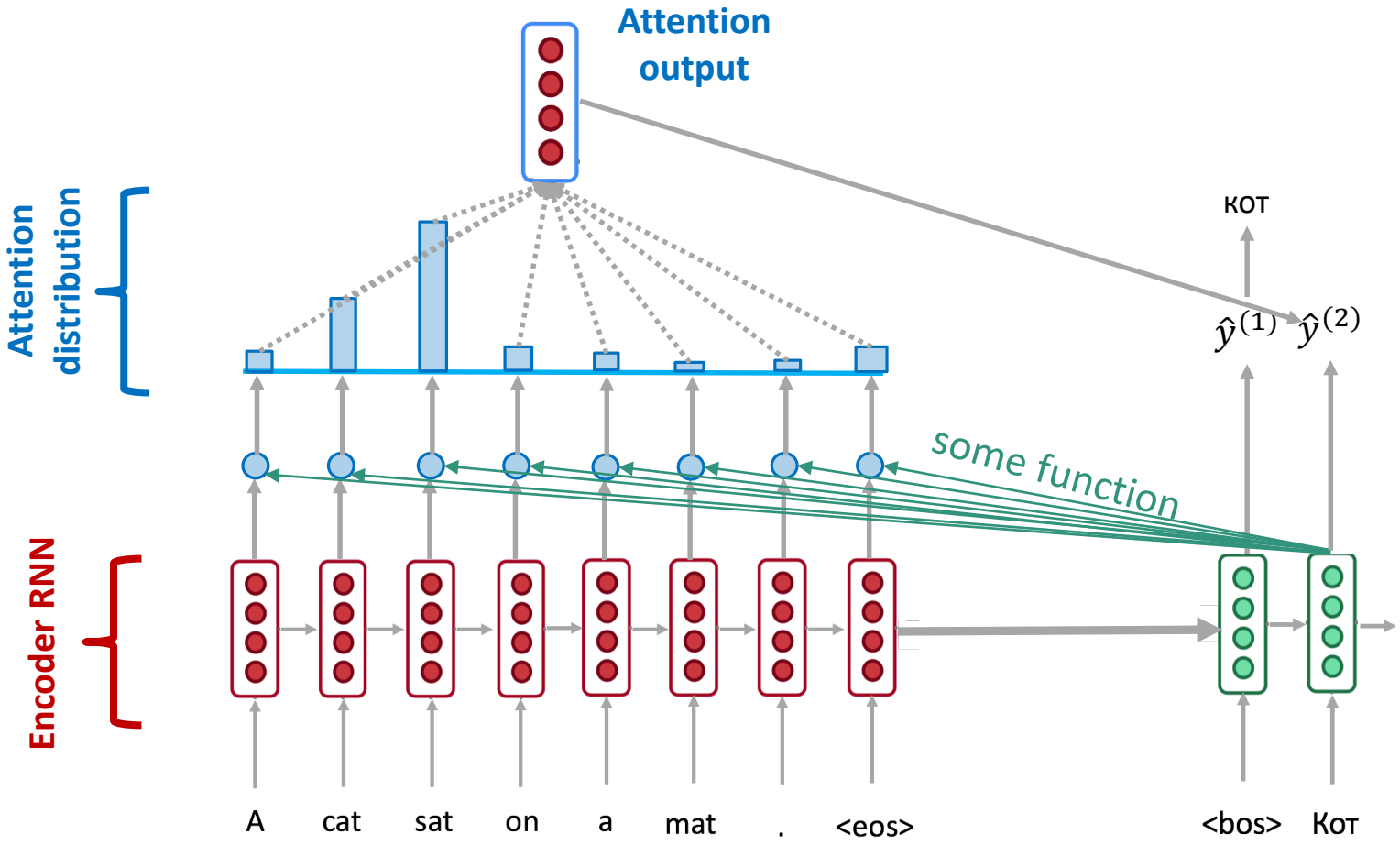
Attention


Attention distribution
Encoder RNN
some function
кот
$\hat{y}^{(1)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот

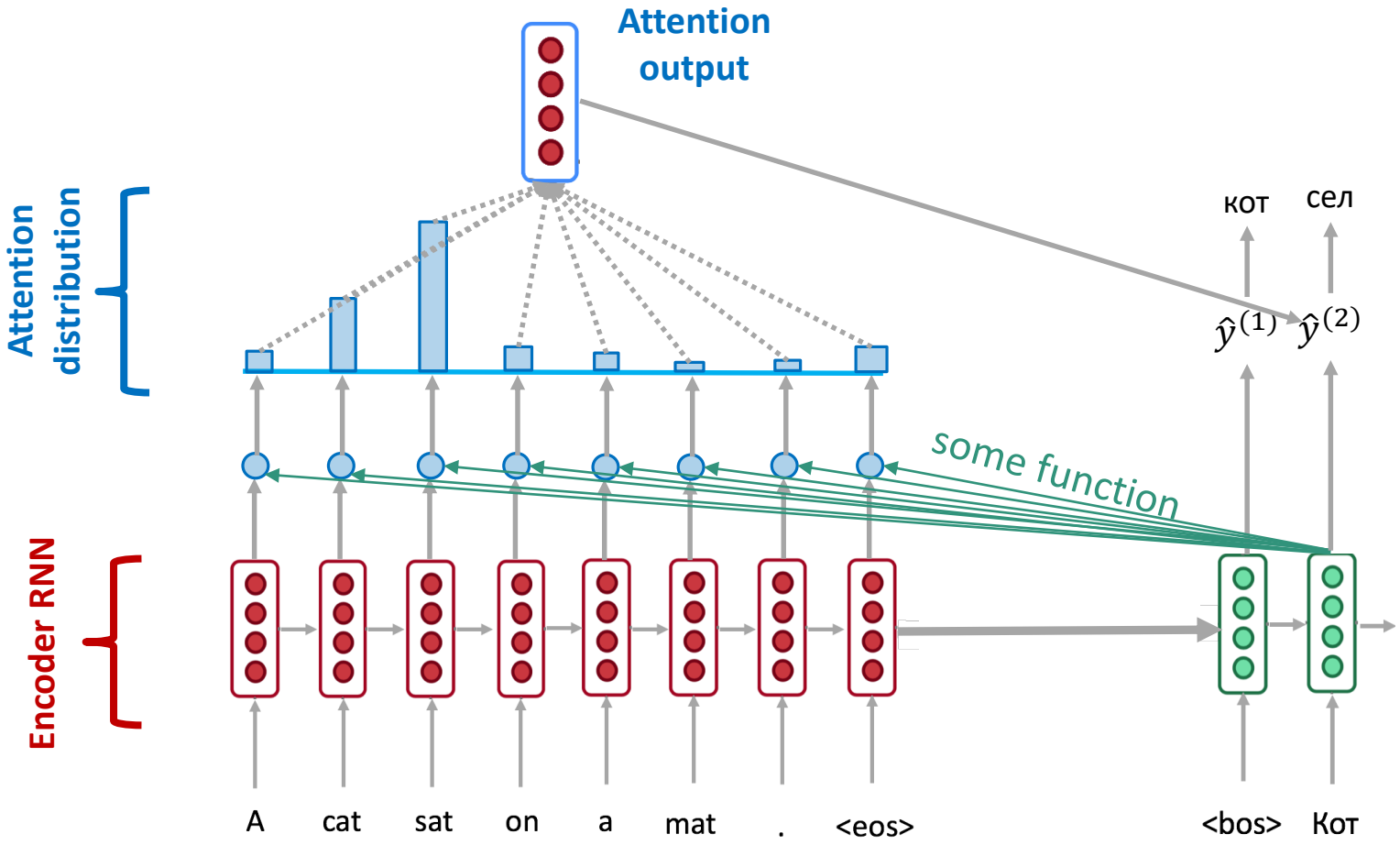
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
$\hat{y}^{(1)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот

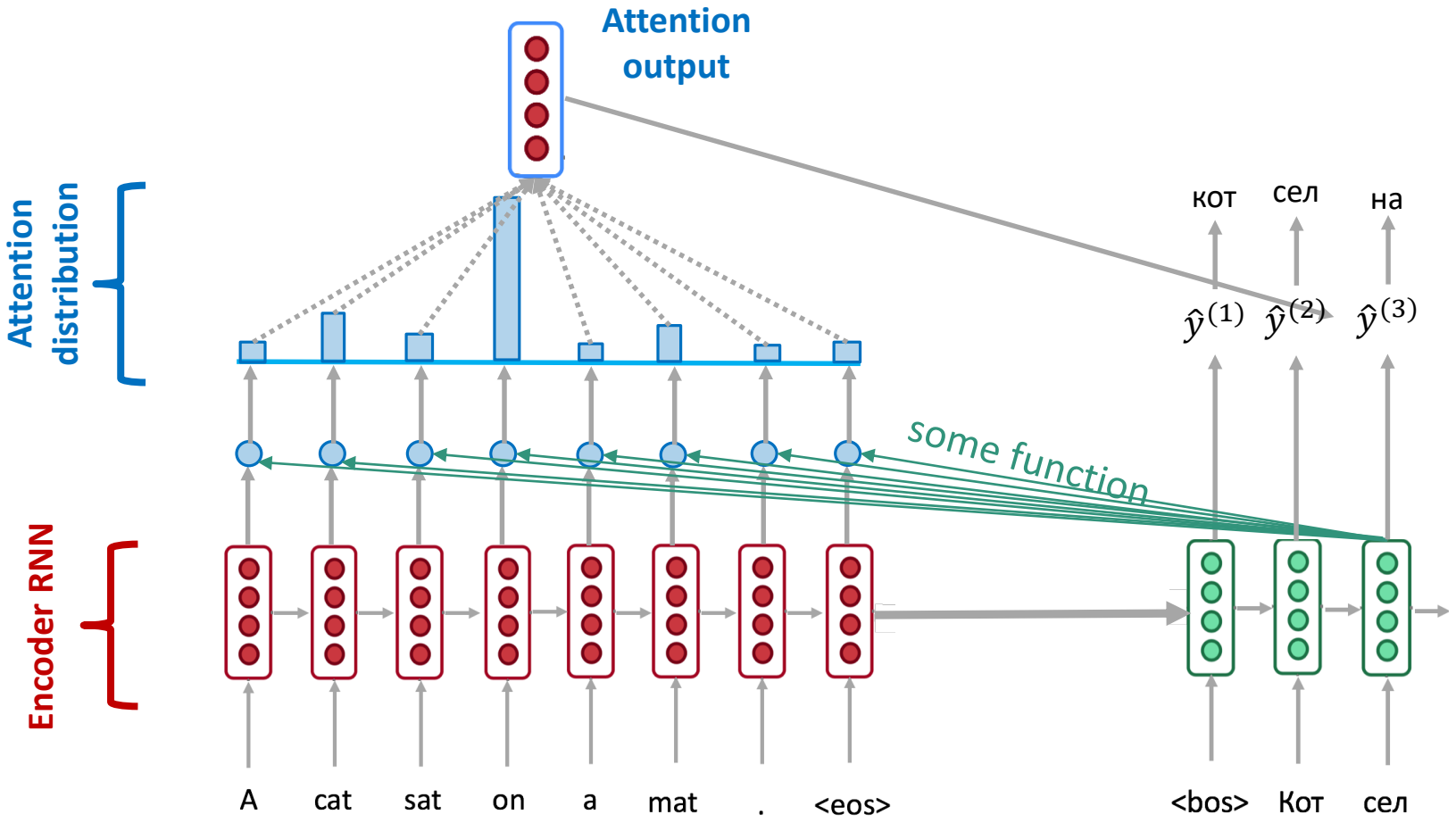
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот

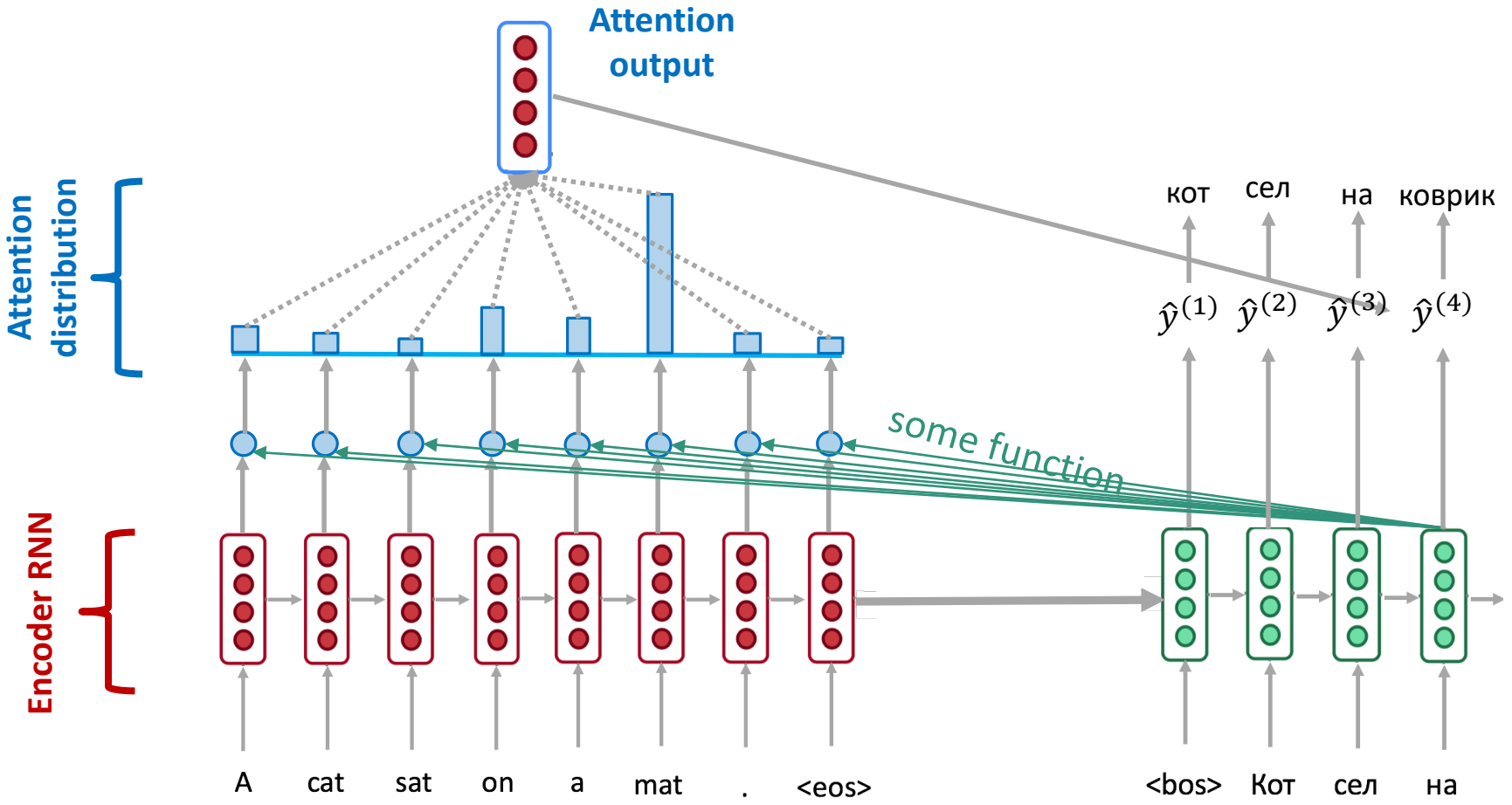
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
сел
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот

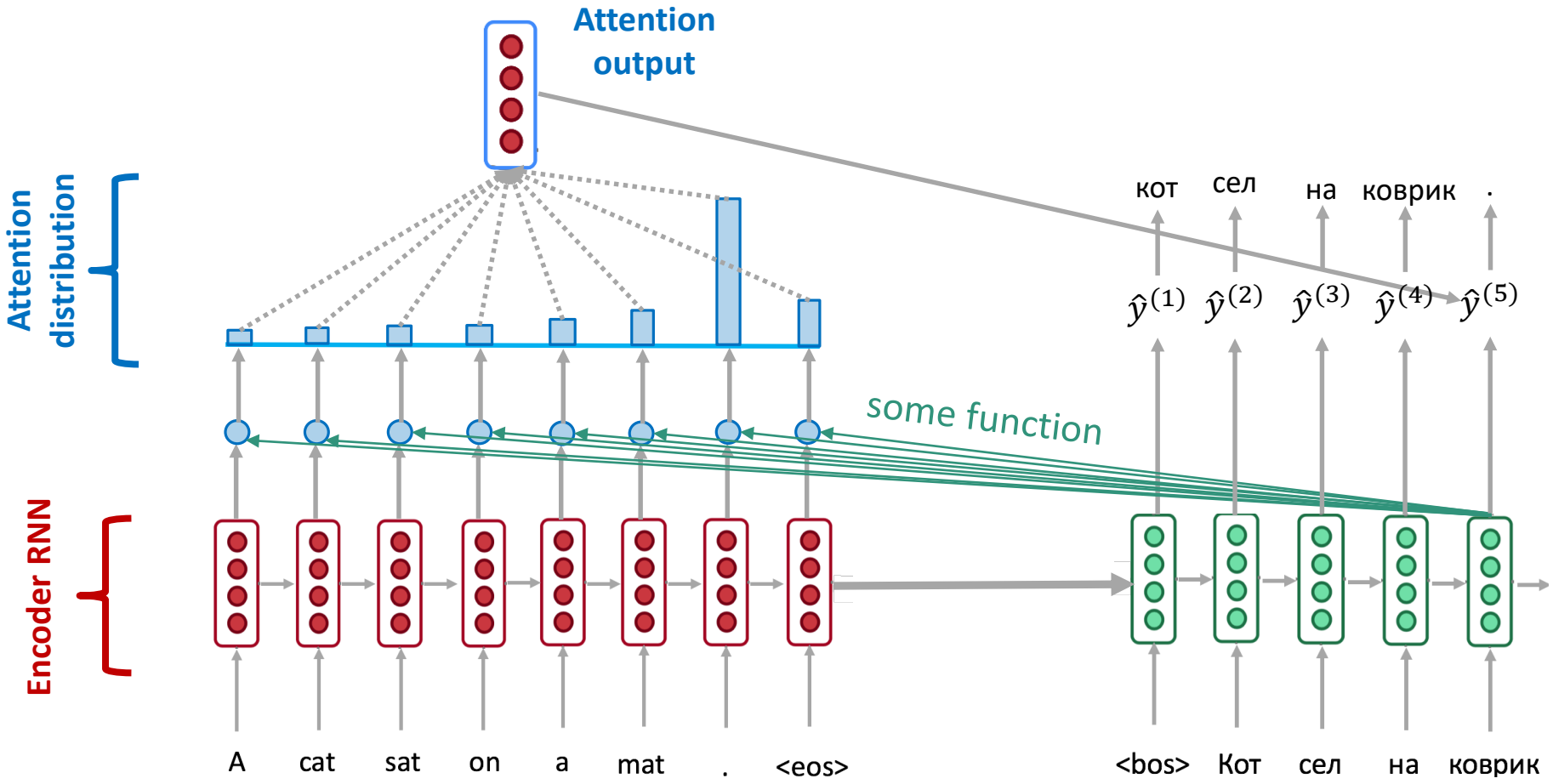
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
сел
на
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот
сел

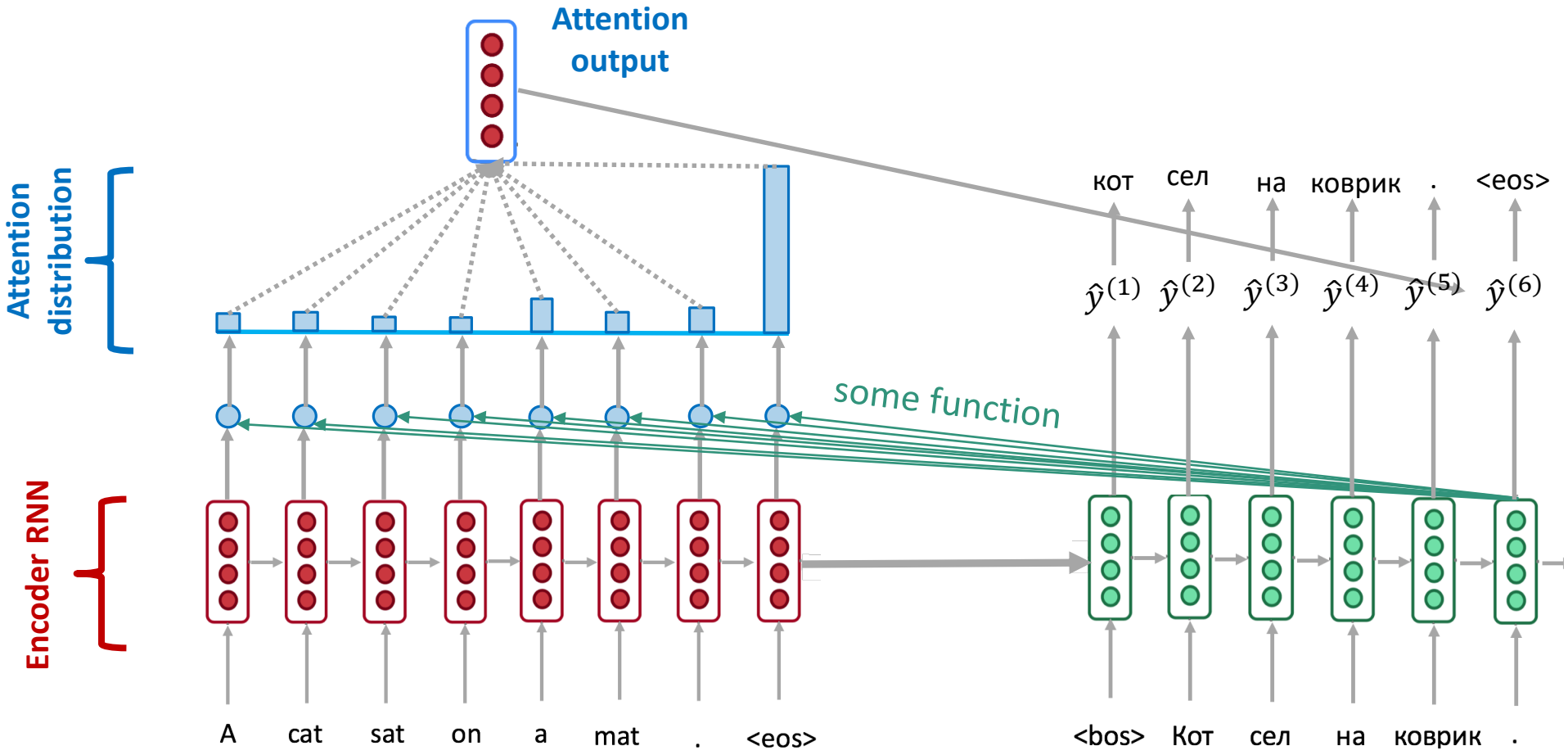
Attention


Attention output
Attention distribution
Encoder RNN
some function
кот
сел
на
коврик
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
$\hat{y}^{(4)}$
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот
сел
на

Attention


Attention output
Attention distribution
some function
Encoder RNN
A
cat
sat
on
a
mat
.
<eos>
<bos>
Кот
сел
на
коврик
кот
сел
на
коврик
.
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
$\hat{y}^{(4)}$
$\hat{y}^{(5)}$

Attention


Attention output
Attention distribution
some function
Encoder RNN
A cat sat on a mat . <eos>
<bos> Кот сел на коврик .
кот сел на коврик . <eos>
$\hat{y}^{(1)}$ $\hat{y}^{(2)}$ $\hat{y}^{(3)}$ $\hat{y}^{(4)}$ $\hat{y}^{(5)}$ $\hat{y}^{(6)}$

Attention: in equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$