

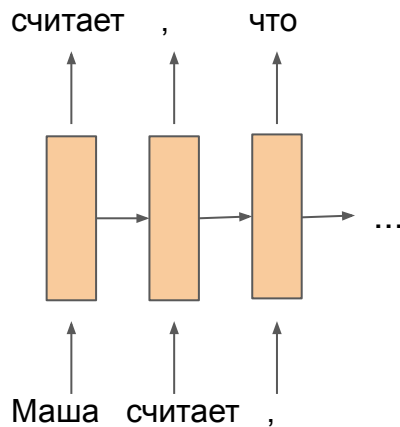
Seq2Seq, Attention

Seq2seq

Seq2seq — архитектура нейронных сетей для задач, в которых на входе и на выходе — последовательность.

Примеры:

- языковые модели
- машинный перевод
- денойзинг аудио



Задача машинного перевода

Задача машинного перевода

$source = (x_1, x_2, \dots, x_n)$

The cat sits on the floor

$target = (y_1, y_2, \dots, y_m)$

Кошка сидит на полу

Задача машинного перевода: найти наиболее вероятную последовательность на target языке (перевод) при условии входящей последовательности на source языке

$$\widehat{target} = \underset{target}{argmax} P(target | source, \theta)$$


Задача машинного перевода

$source = (x_1, x_2, \dots, x_n)$

The cat sits on the floor

$target = (y_1, y_2, \dots, y_m)$

Кошка сидит на полу

$$\widehat{target} = \underset{target}{\operatorname{argmax}} \quad P(target \mid source, \theta)$$


$$\begin{aligned} P(target \mid source) &= P(y_1 \ y_2 \ \dots \ y_m \mid source) = \\ &= P(y_1 \mid source) \cdot P(y_2 \mid y_1, source) \dots P(y_m \mid y_1, \dots, y_{m-1}, source) \end{aligned}$$

Задача машинного перевода

$source = (x_1, x_2, \dots, x_n)$

The cat sits on the floor

$target = (y_1, y_2, \dots, y_m)$

Кошка сидит на полу

$$\begin{aligned} P(target \mid source) &= P(y_1 \ y_2 \ \dots \ y_m \mid source) = \\ &= P(y_1 \mid source) \cdot P(y_2 \mid y_1, source) \dots P(y_m \mid y_1, \dots, y_{m-1}, source) \end{aligned}$$

Conditional language model (условная языковая модель). Обусловлена source (предложением на исходном языке)

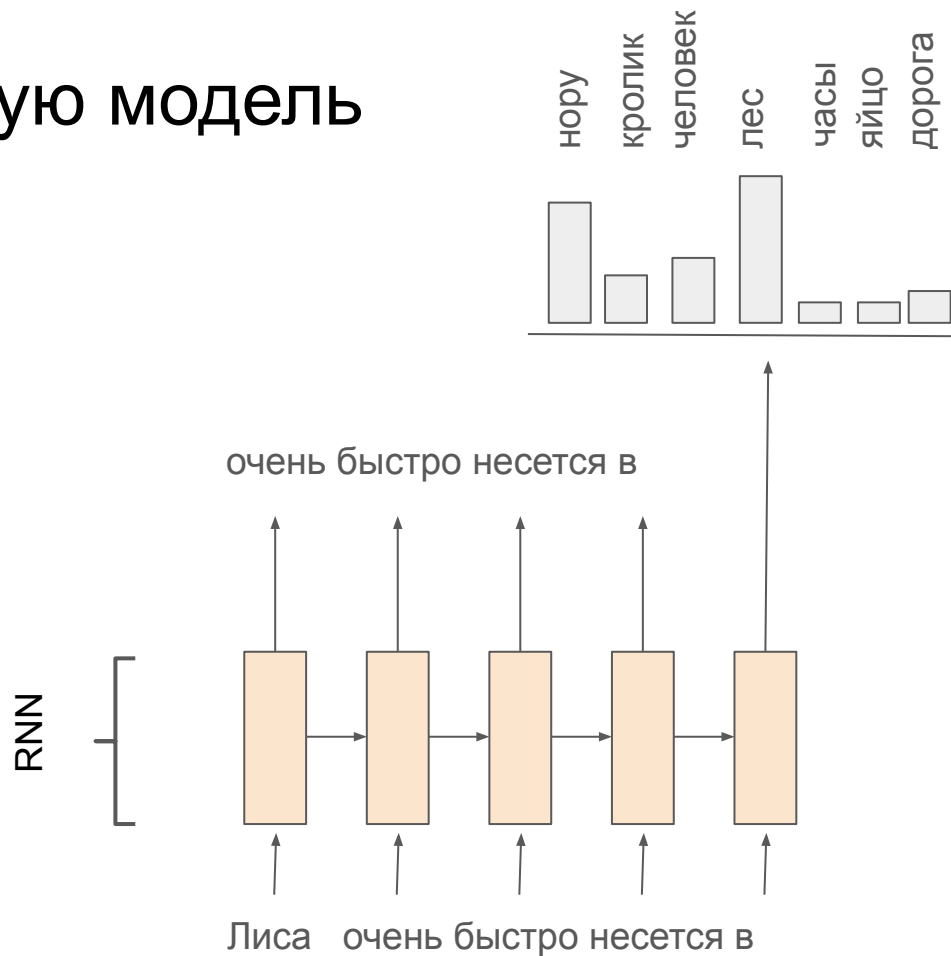
Решение задачи машинного перевода

- правилые системы (rule-based) (1950-е годы)
- статистические модели (statistics-based) (1960-2010)
- нейронные модели (neural-based) (2010-е годы)

Seq2Seq

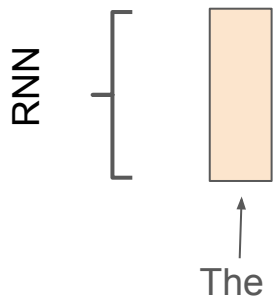
или как решать задачу машинного перевода с помощью нейронных сетей

Вспомним языковую модель

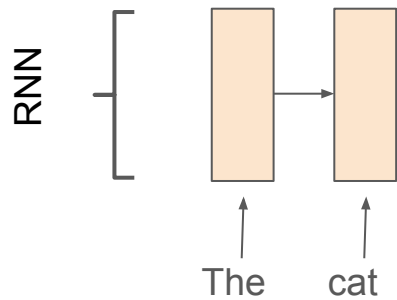


Encoder-Decoder

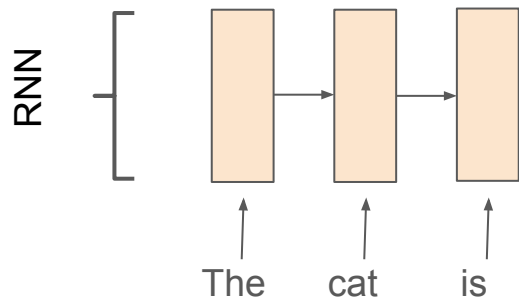
Задача: перевести предложение “The cat is pretty” на русский язык



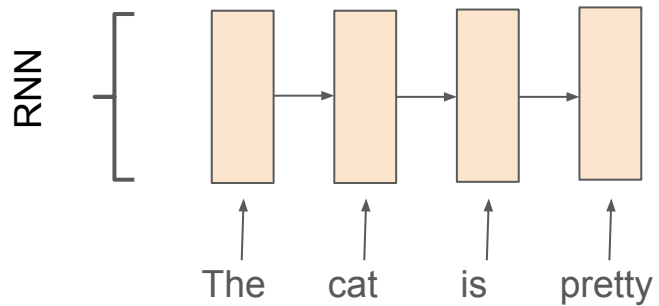
Encoder-Decoder



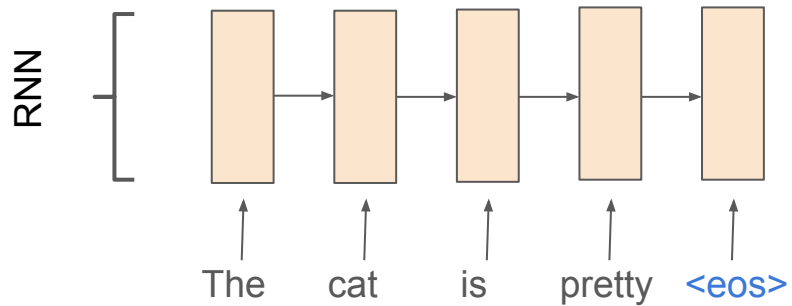
Encoder-Decoder



Encoder-Decoder

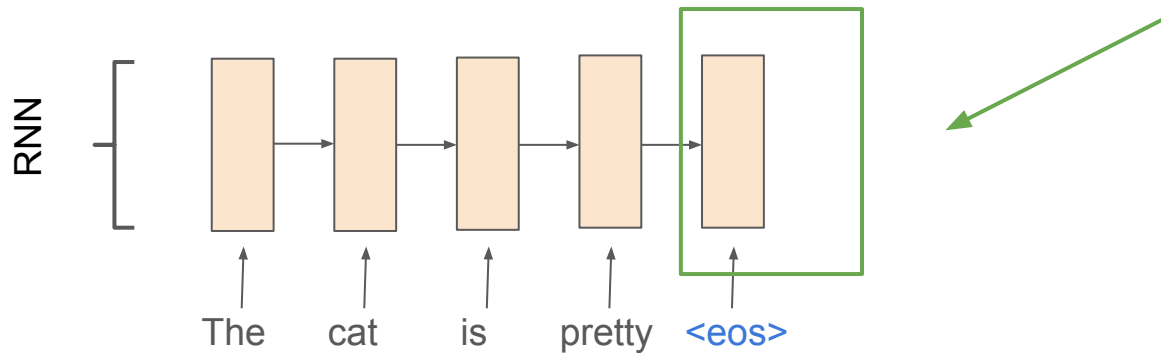


Encoder-Decoder

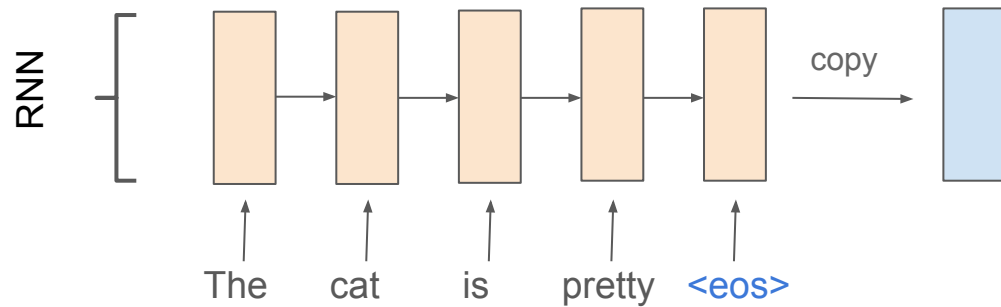


Encoder-Decoder

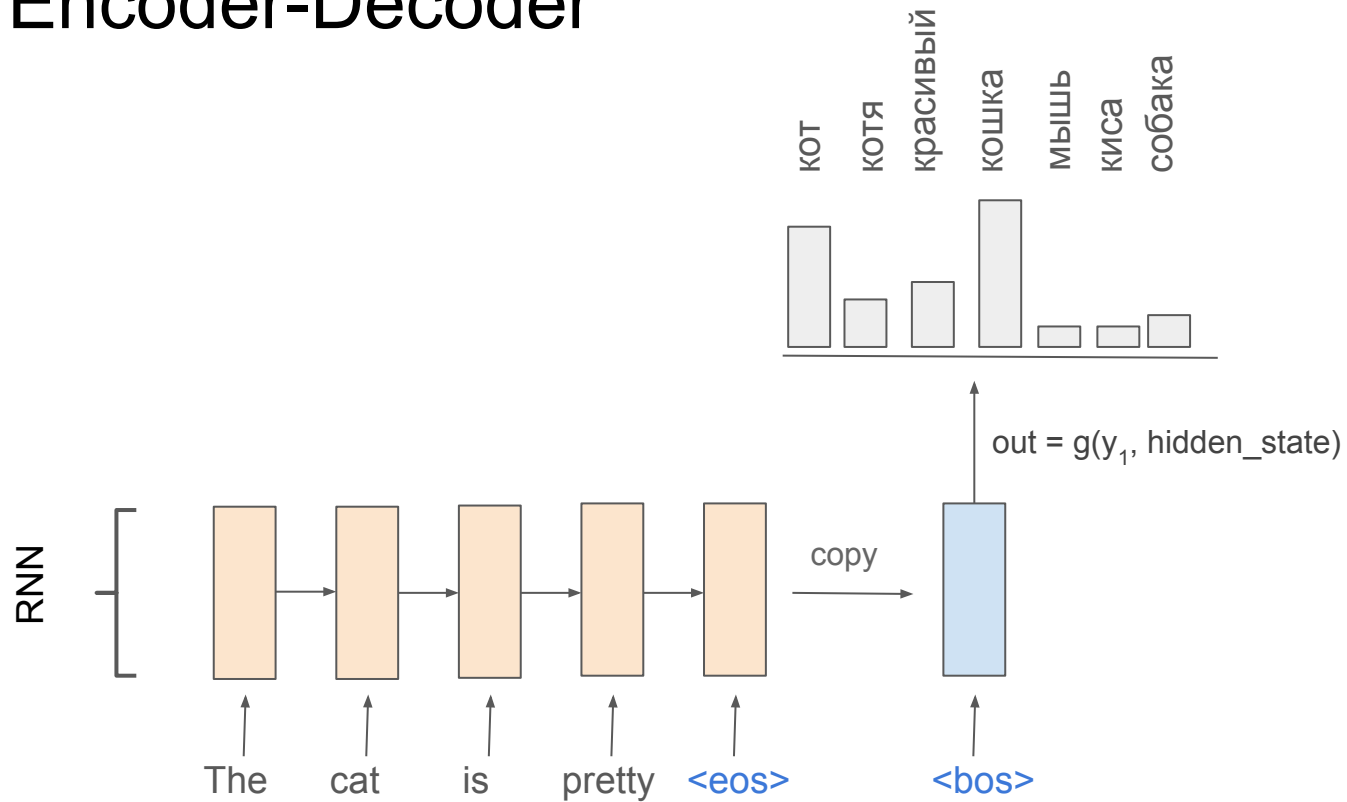
Скрытое состояние нейронной сети в последний момент времени содержит информацию обо всем входном предложении



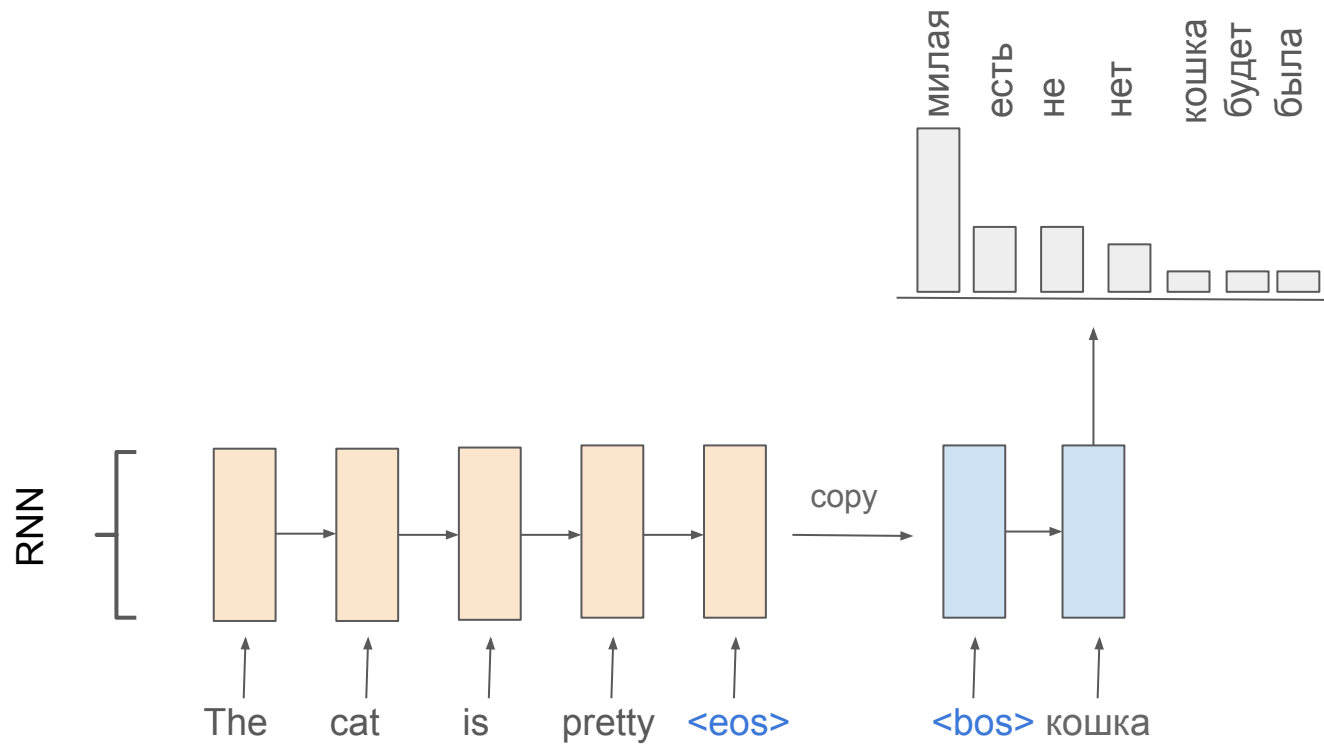
Encoder-Decoder



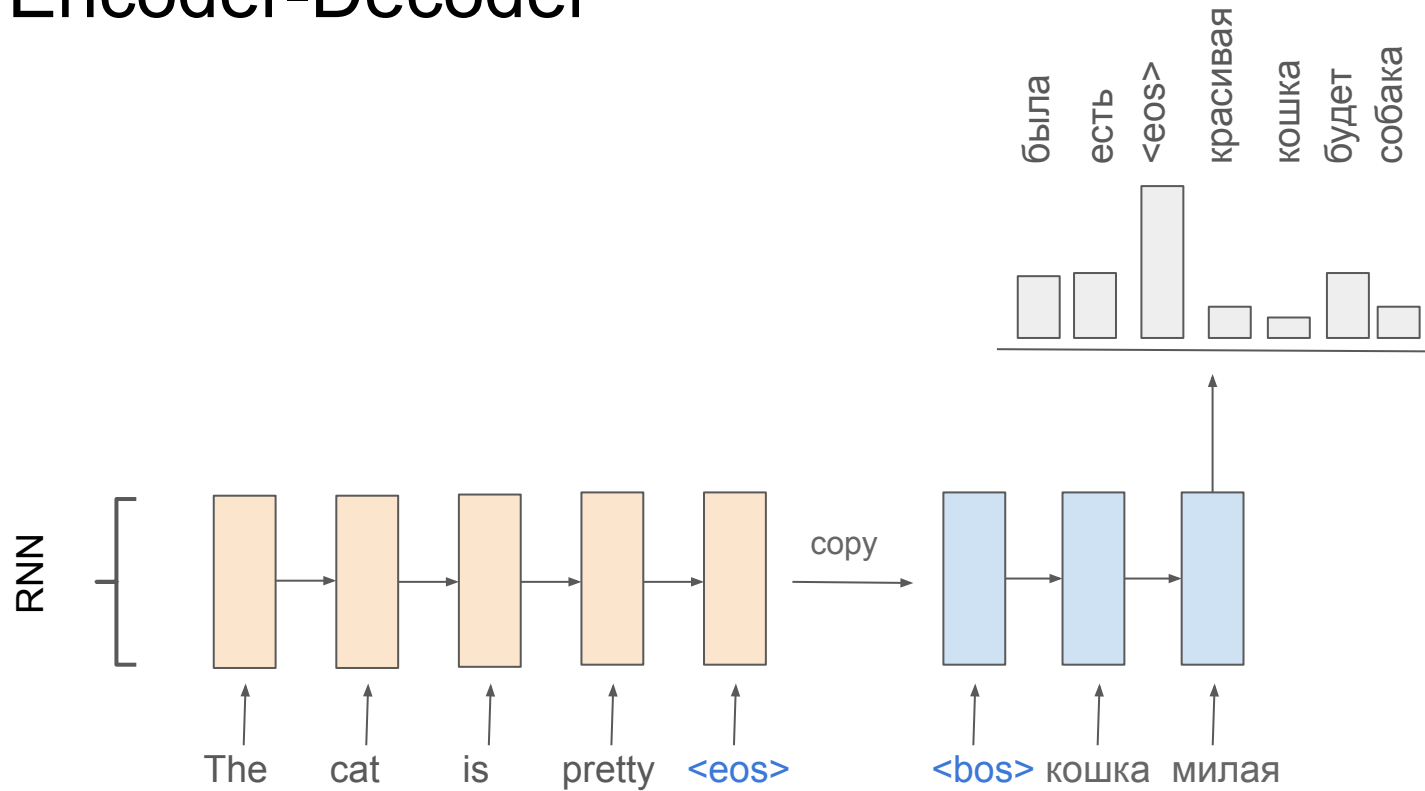
Encoder-Decoder



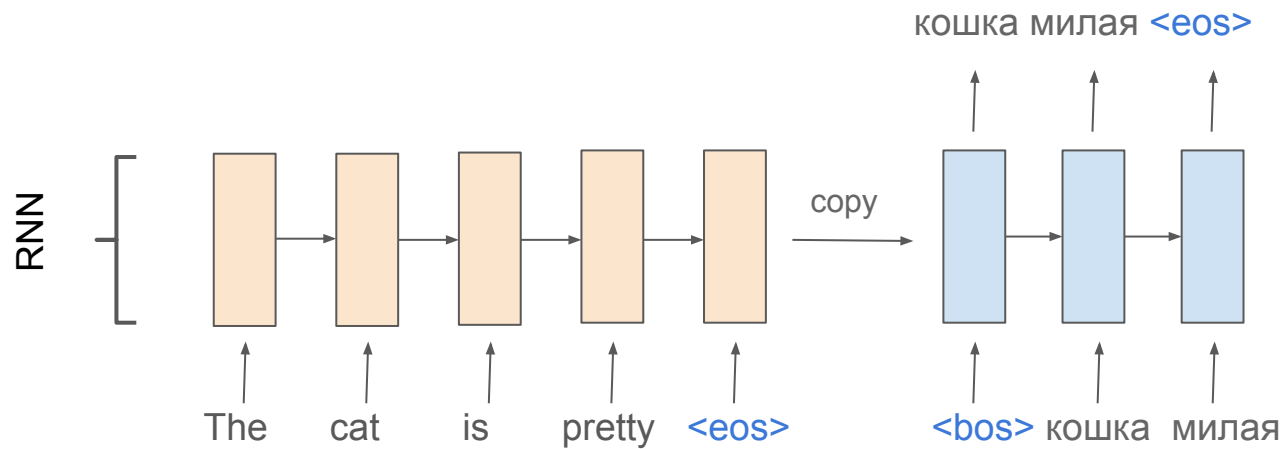
Encoder-Decoder



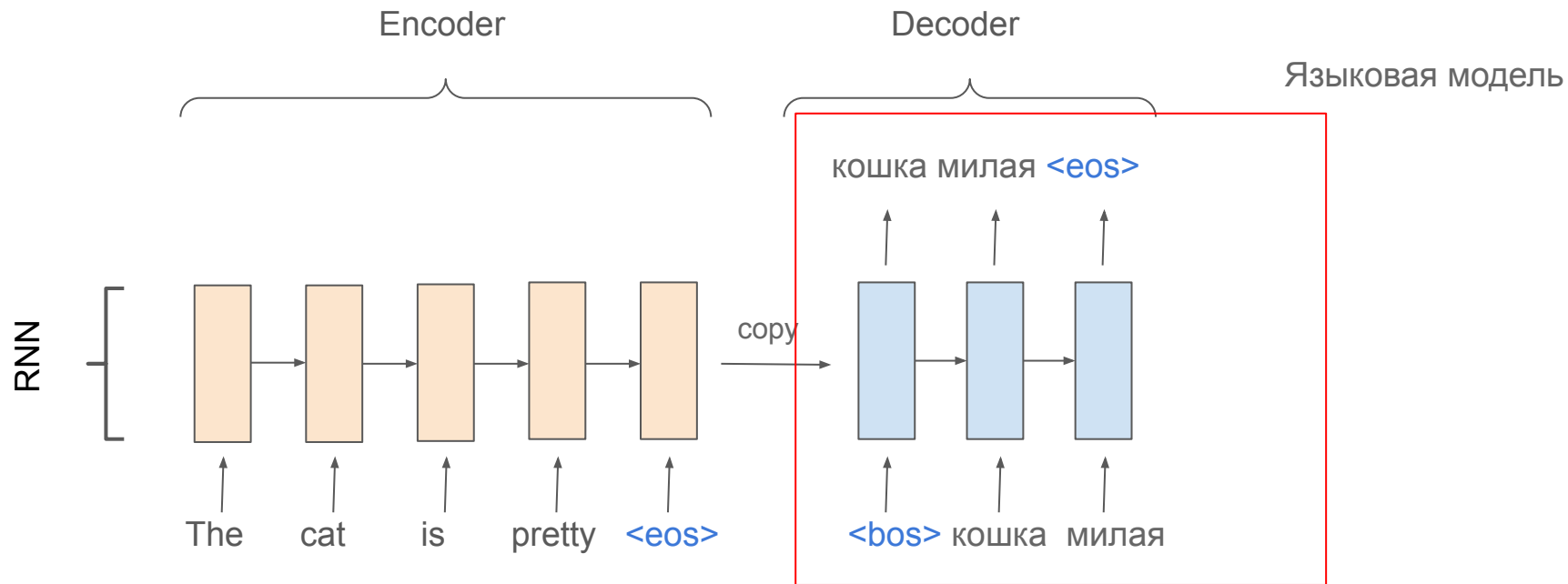
Encoder-Decoder



Encoder-Decoder

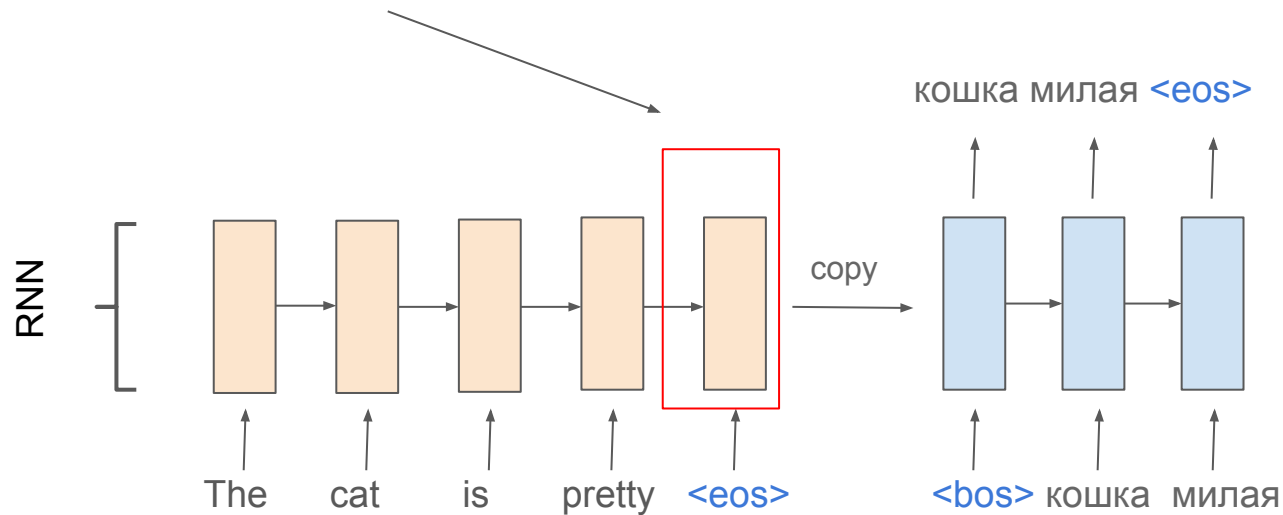


Encoder-Decoder

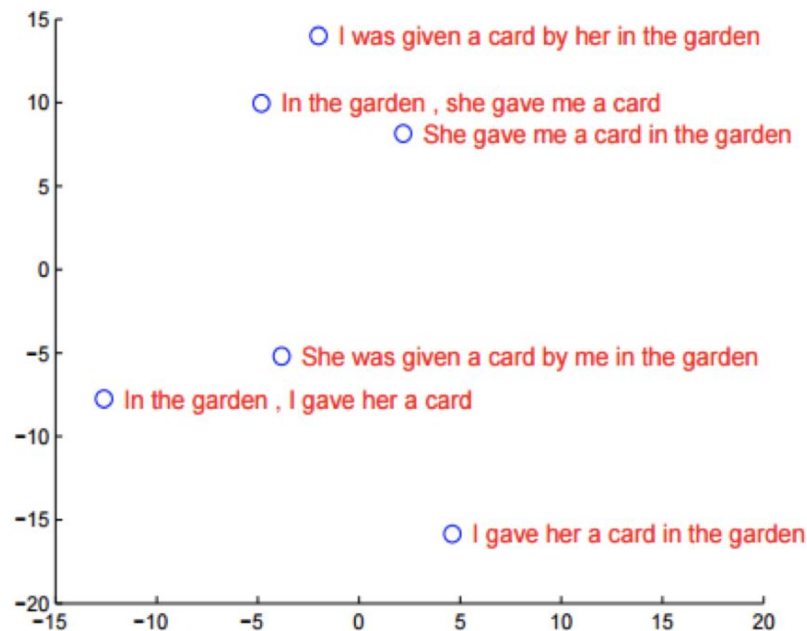
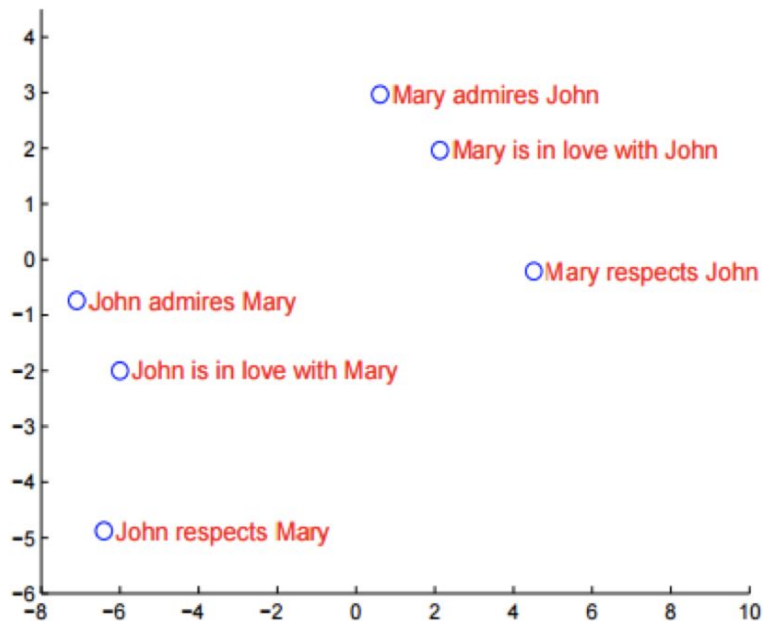


Encoder-Decoder

Эмбеддинг предложения



Sentence embedding



(Sutskever et al., 2014)

Encoder-Decoder

- состоит из двух частей: энкодера и декодера
- энкодер собирает информацию о предложении на source языке. Может быть и не RNN-сетью, а сетью любой архитектуры.
- Декодер использует информацию, собранную энкодером, чтобы сгенерировать предложение-перевод на target языке.

Обучение Encoder-Decoder

Для обучения машинного перевода нужен датасет с параллельными предложениями.

Параллельные предложения — пара

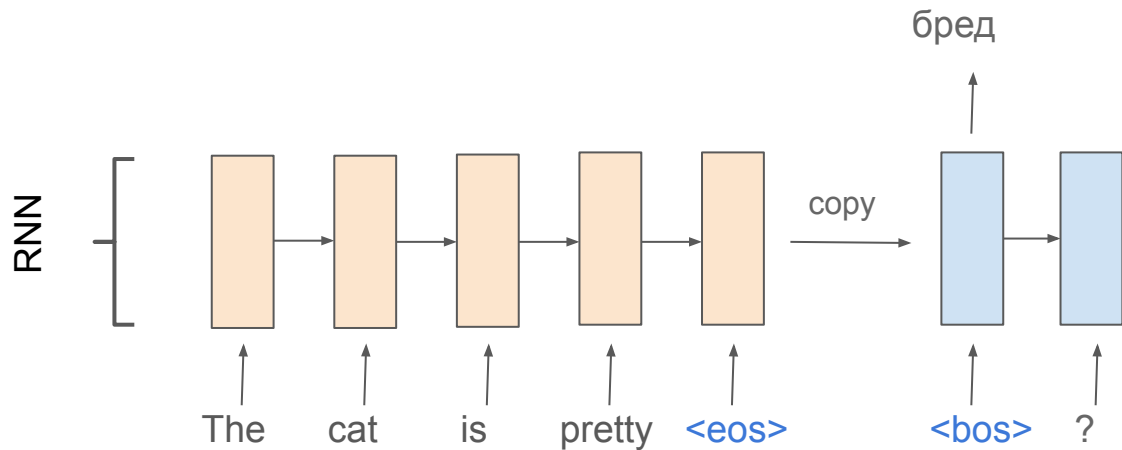
(“предложение на source языке”, “предложение на target языке”)

(“The cat is pretty”, “Кошка милая”)

Такие датасеты чаще всего собираются скроллингом интернета: Википедии, новостных сайтов, etc.

Обучение Encoder-Decoder

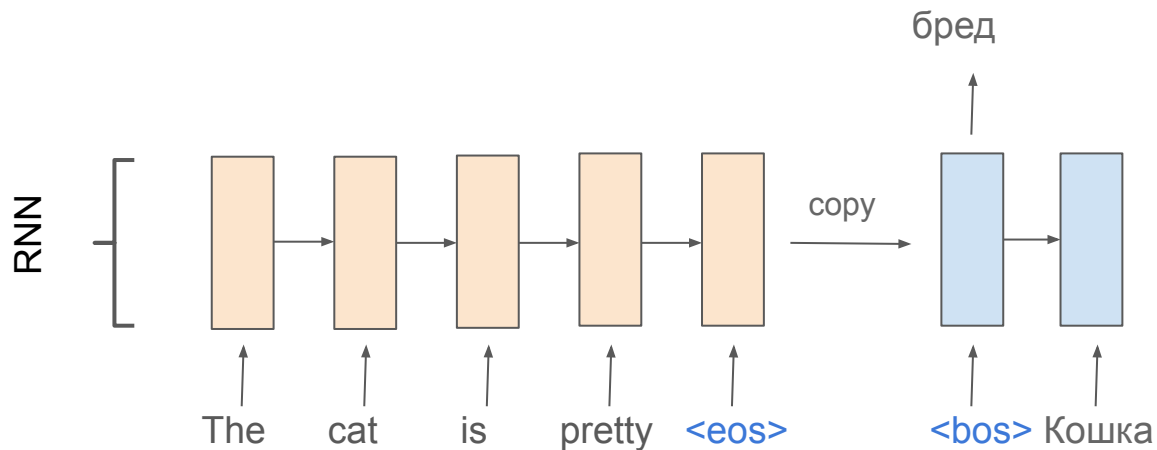
Пусть у нас в тренин датасете есть пара предложений:
“The cat is pretty” и “Кошка милая”



Обучение Encoder-Decoder

Пусть у нас в трейн датасете есть пара предложений:
“The cat is pretty” и “Кошка милая”

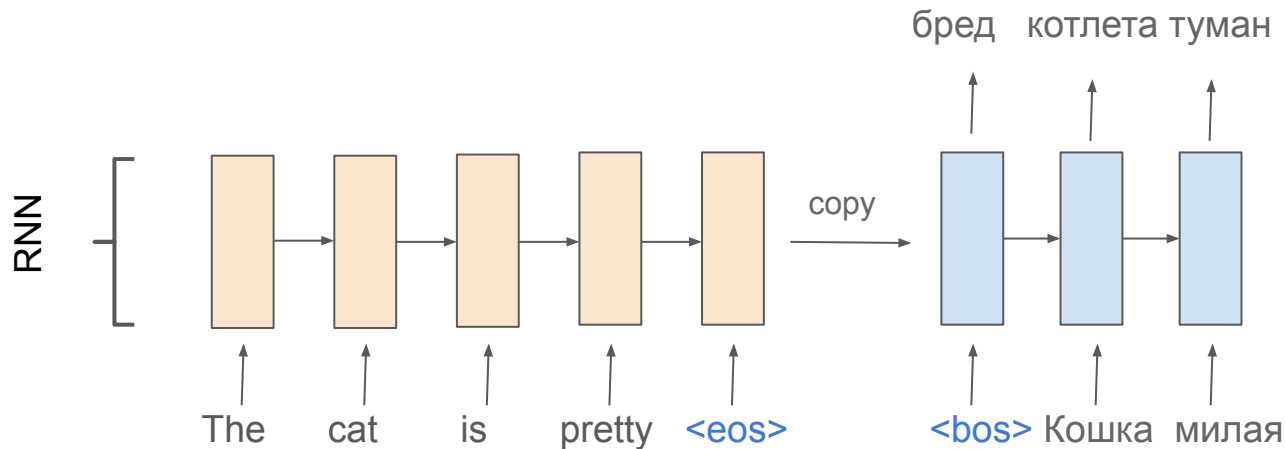
Используем **teacher-forcing**



Обучение Encoder-Decoder

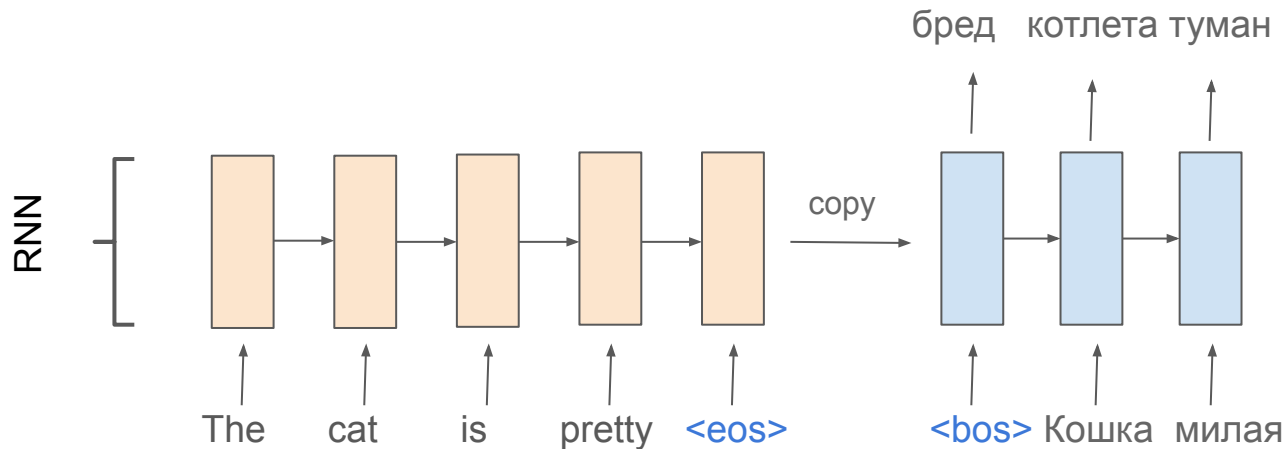
Пусть у нас в тренин датасете есть пара предложений:
“The cat is pretty” и “Кошка милая”

Используем **teacher-forcing**



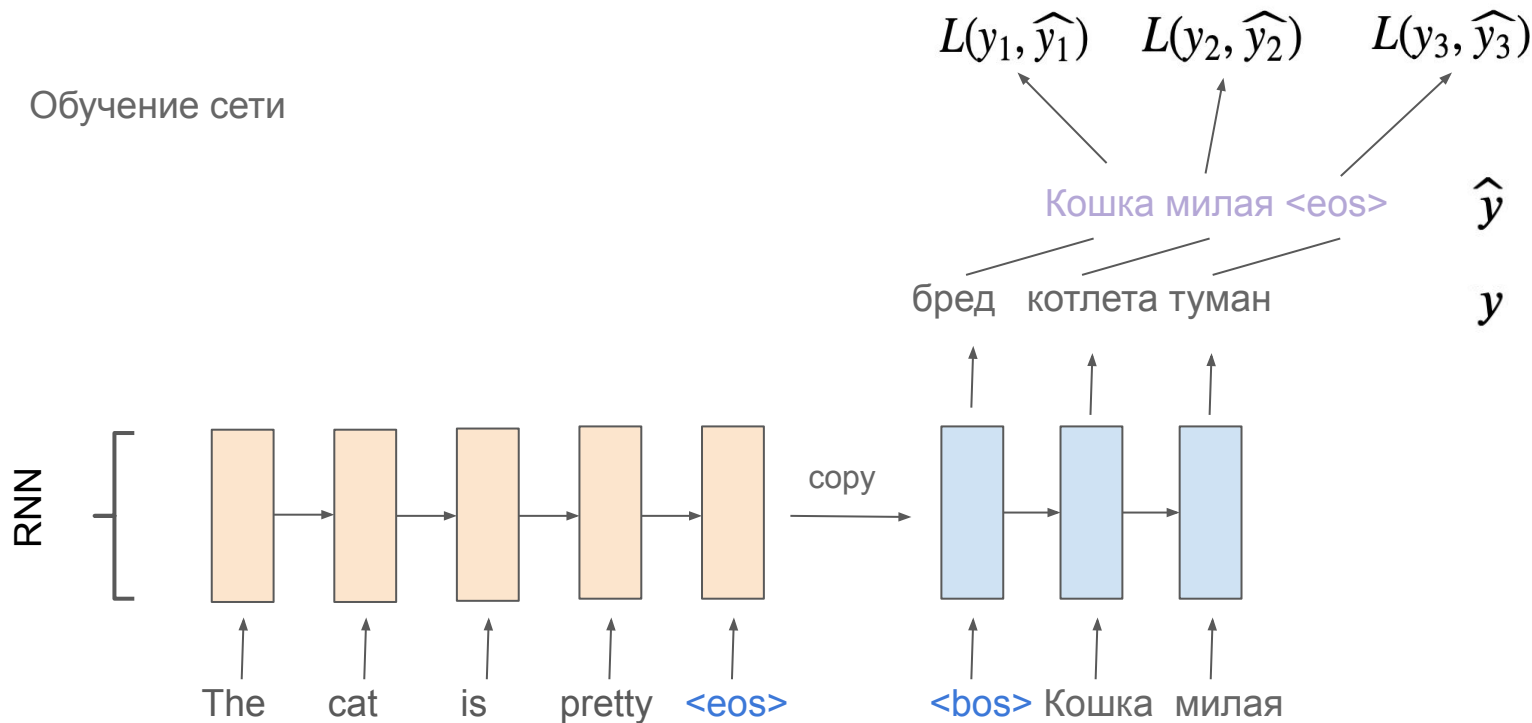
Обучение Encoder-Decoder

По мере обучения сети можно ослабевать teacher-forcing и с некоторой вероятностью подавать на вход сети не правильное слово, а то, которое сеть сгенерировала сама на предыдущем шаге.



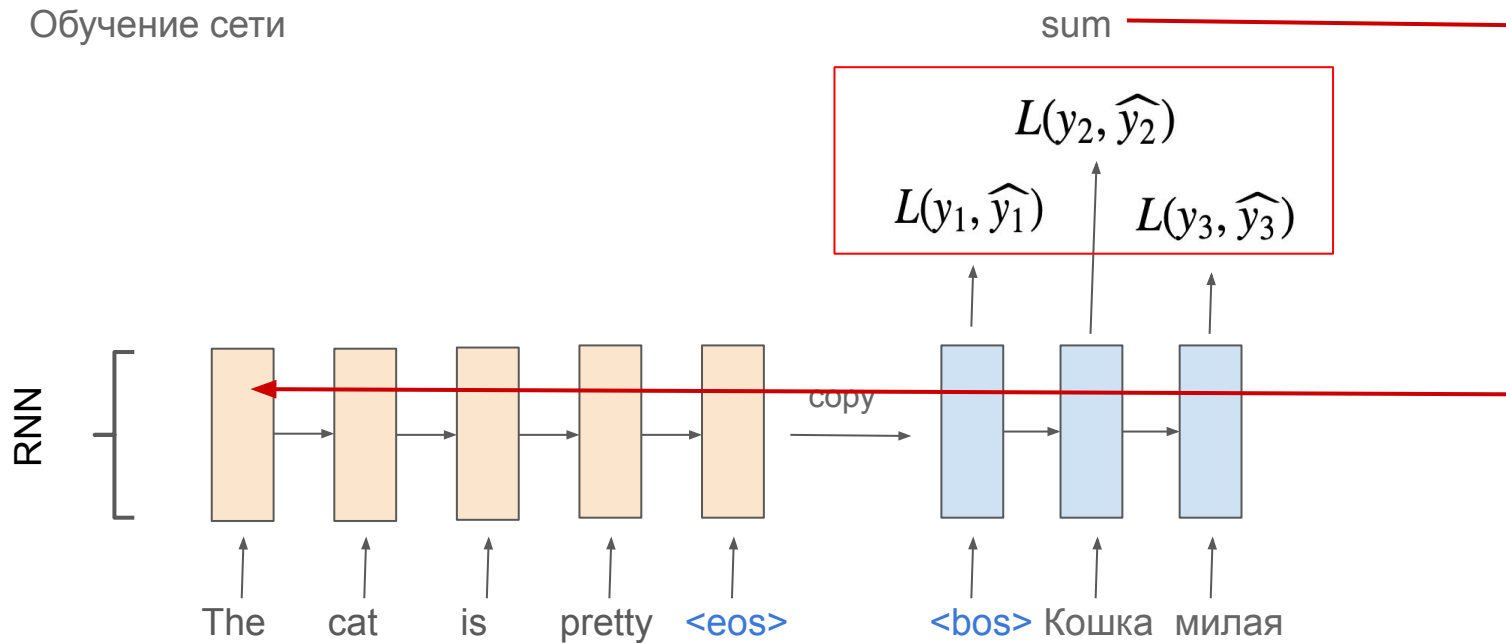
Обучение Encoder-Decoder

Обучение сети



Обучение Encoder-Decoder

Обучение сети



Метрика качества машинного перевода

Метрику качества для задачи машинного перевода придумать сложно. Сложно потому, что для одного предложения на английском языке есть несколько хороших переводов на русский язык.

Source: It seemed like there was going to be a hurricane

Target #1: Похоже, надвигался ураган

Target #2: Мне показалось, что скоро будет ураган

Target #3: Казалось, что нас вот-вот настигнет ураган.

Метрика качества машинного перевода

Самая распространенная метрика качества для машинного перевода — **BLEU** (bilingual evaluation understudy)

BLEU считает точность по униграммам, биграммам и т.д.

BLEU

Source: Кошка сидит

Target: the cat is sitting.

Generated: the the cat

точность n-грамм: сколько из сгенерированных моделью n-грамм действительно встречается в target предложении



	set of grams	precision score
unigram	“the”, ‘the”, “cat”	$(1+1+1)/3 = 1$
bigram	“the the”, “the cat”	$(0 + 1)/2 = 1/2$
trigram	“the the cat”	$0/1 = 0$

BLEU

Source: Кошка сидит

Итоговый скор BLEU: $1 + \frac{1}{2} + 0 = 1.5$

Target: the cat is sitting.

Generated: the the cat



	set of grams	precision score
unigram	“the”, ‘the”, “cat”	$(1+1+1)/3 = 1$
bigram	“the the”, “the cat”	$(0 + 1)/2 = 1/2$
trigram	“the the cat”	$0/1 = 0$

BLEU

BLEU — совершенно неидеальная метрика качества для машинного перевода. Улучшение метрики BLEU не всегда показывает улучшение качества перевода моделью.

Table 9: Human side-by-side evaluation scores of WMT En→Fr models.

Model	BLEU	Side-by-side averaged score
PBMT [15]	37.0	3.87
NMT before RL	40.35	4.46
NMT after RL	41.16	4.44
Human		4.82

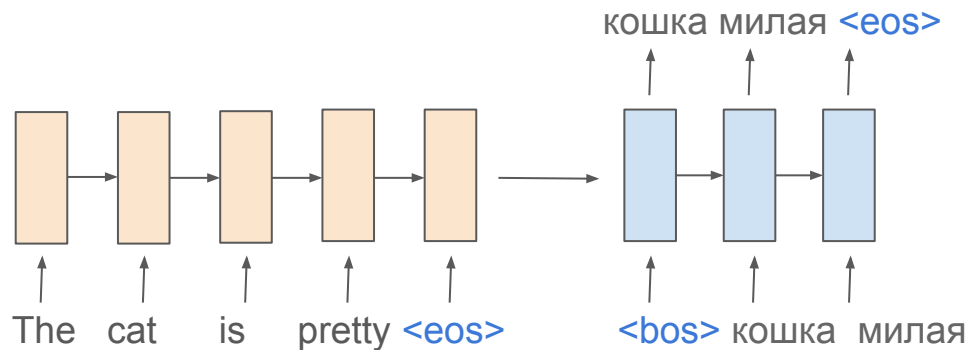
Encoder-Decoder

- состоит из двух частей: энкодера и декодера
- энкодер собирает информацию о предложении на source языке. Может быть и не RNN-сетью, а сетью любой архитектуры.
- Декодер использует информацию, собранную энкодером, чтобы сгенерировать предложение-перевод на target языке.

Какие проблемы есть у описанного подхода encoder-decoder?

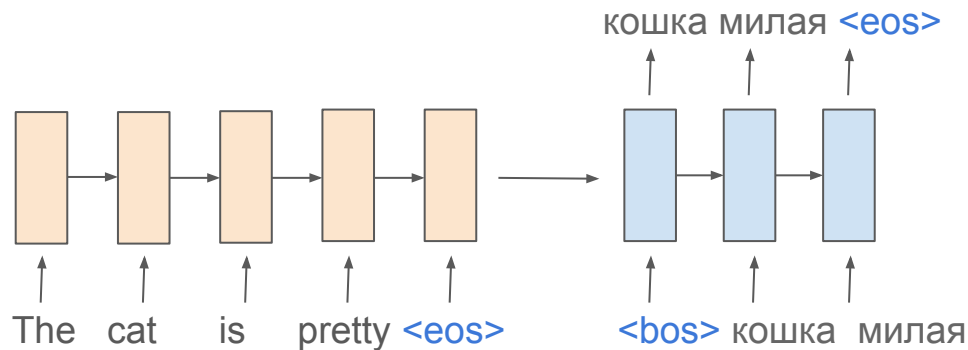
Encoder-Decoder: проблемы

- “жадное” декодирование
- “забывчивость” RNN: энкодер может забывать начало длинных предложений
- вся информация о длинном предложении записывается в один вектор!

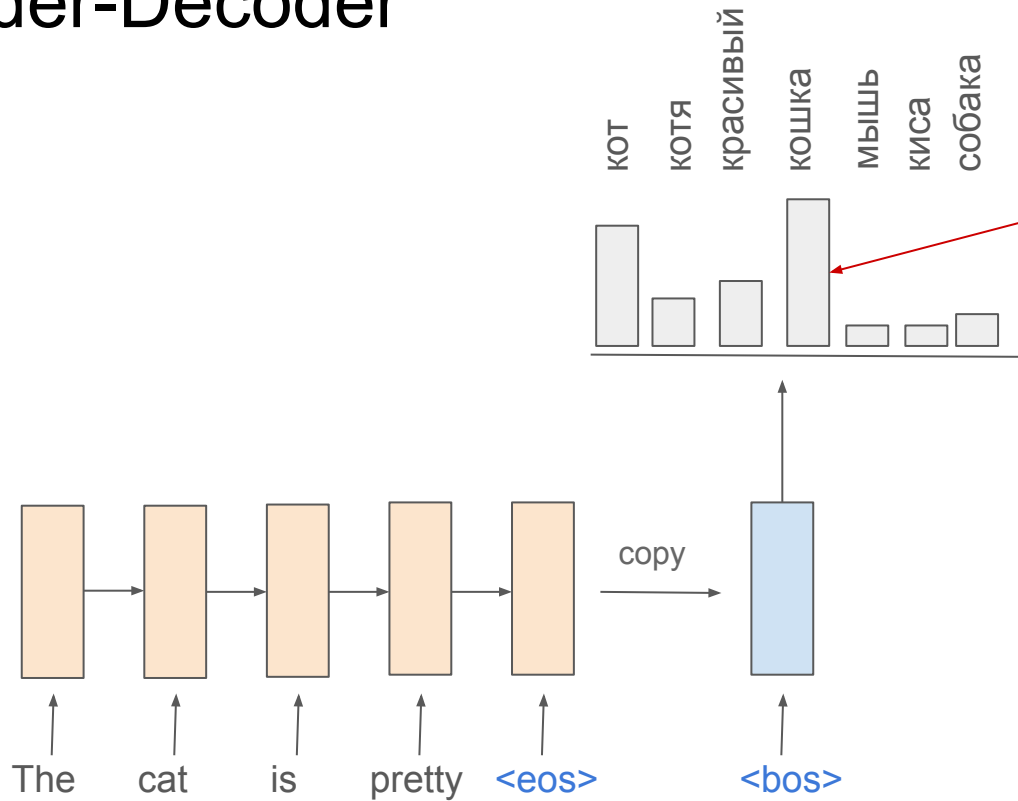


Encoder-Decoder: проблемы

- “жадное” декодирование
- “забывчивость” RNN: энкодер может забывать начало длинных предложений
- вся информация о длинном предложении записывается в один вектор!



Encoder-Decoder



Каждый раз выбираем argmax .
Хорошая ли это идея?

Greedy decoding

Пример, когда жадное декодирование может сработать плохо:

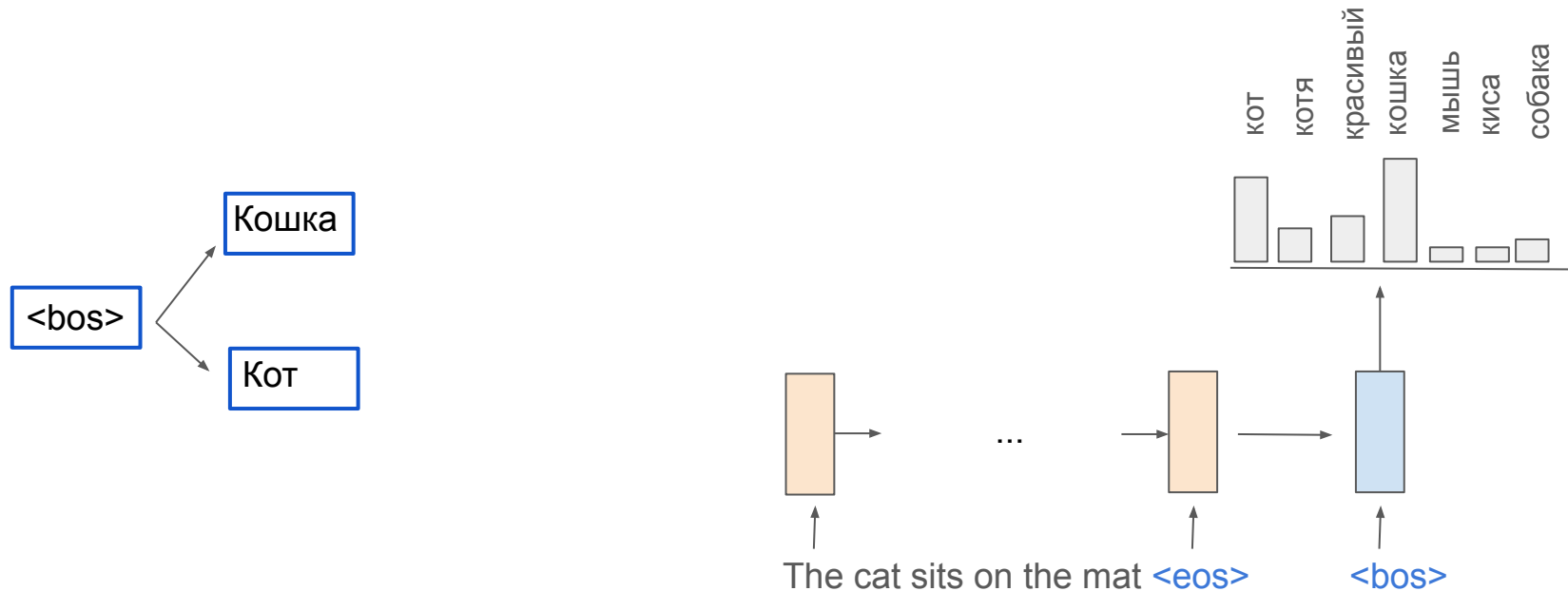
Had there been a vaccine, would he be alive.

Была тут вакцина, был бы он жив

Если бы там была вакцина, то он бы выжил

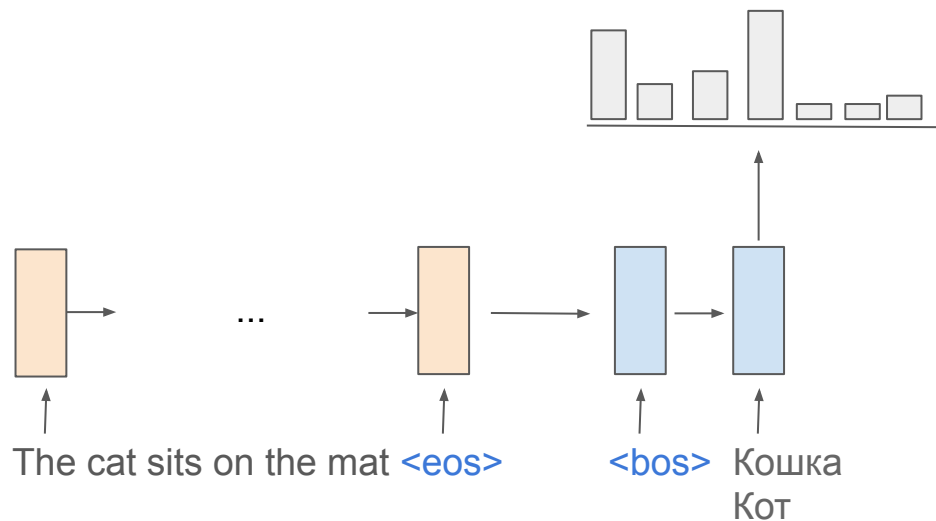
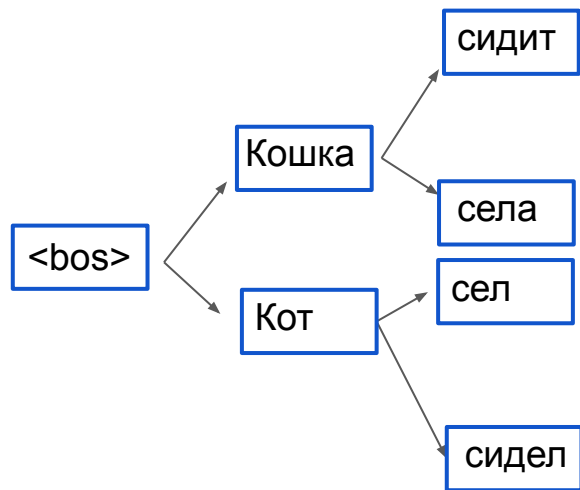
Beam search

Решение: поддерживать в памяти фиксированное количество гипотез



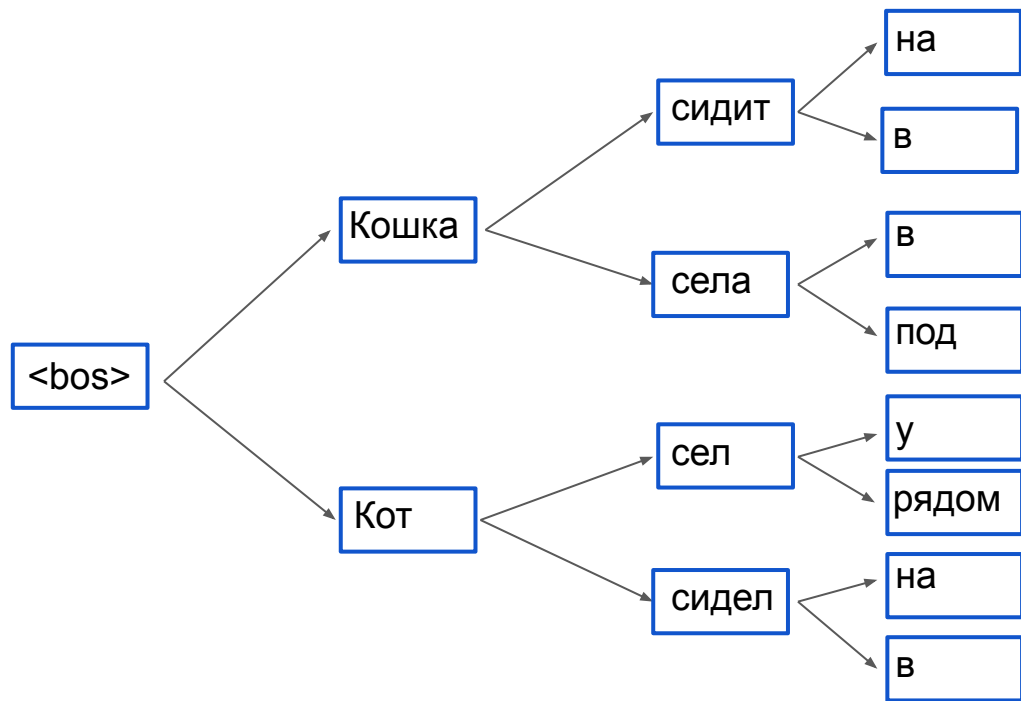
Beam search

Решение: поддерживать в памяти фиксированное количество гипотез



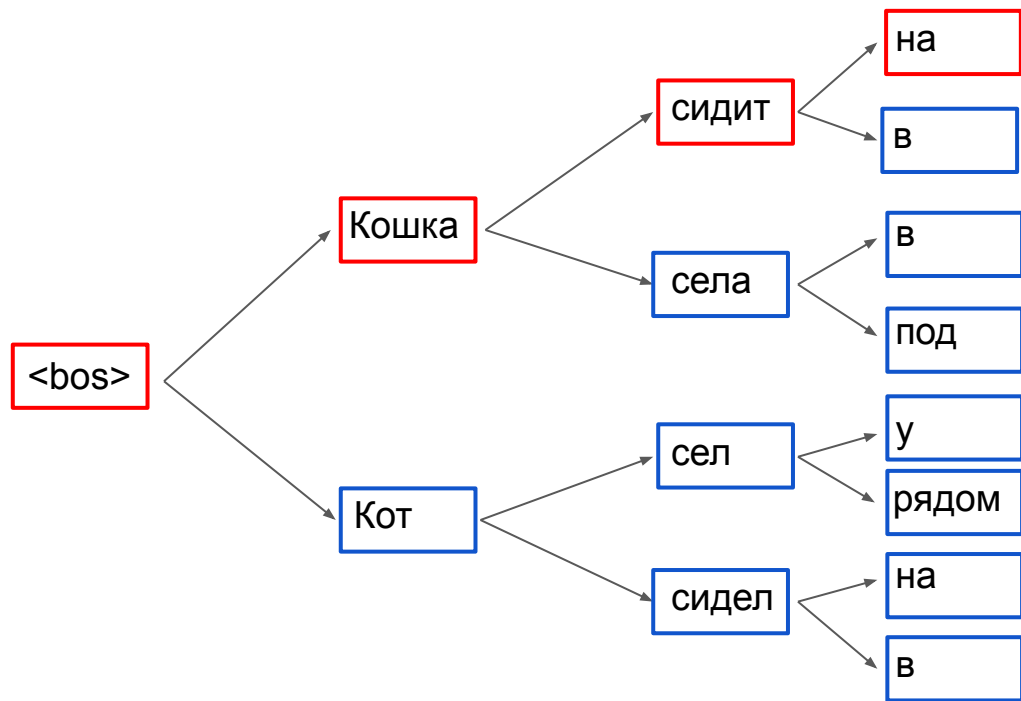
Beam search

Решение: поддерживать в памяти фиксированное количество гипотез



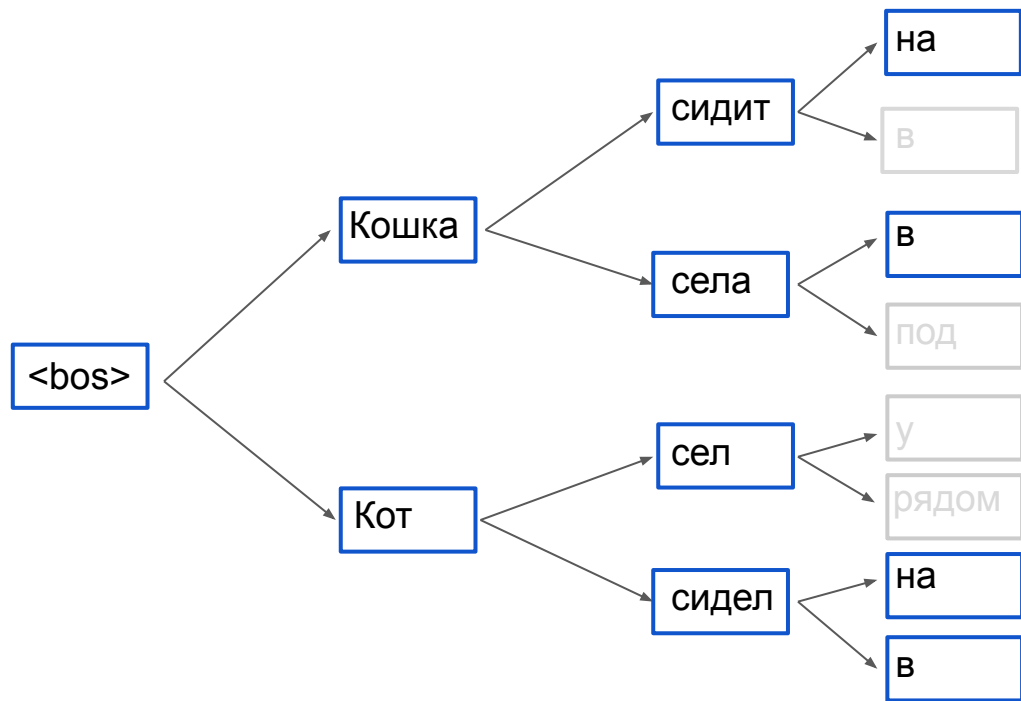
Beam search

$$P(\text{Кошка сидит на}) = P(\text{на} \mid \text{кошка сидит}) * P(\text{сидит} \mid \text{кошка}) * P(\text{кошка})$$



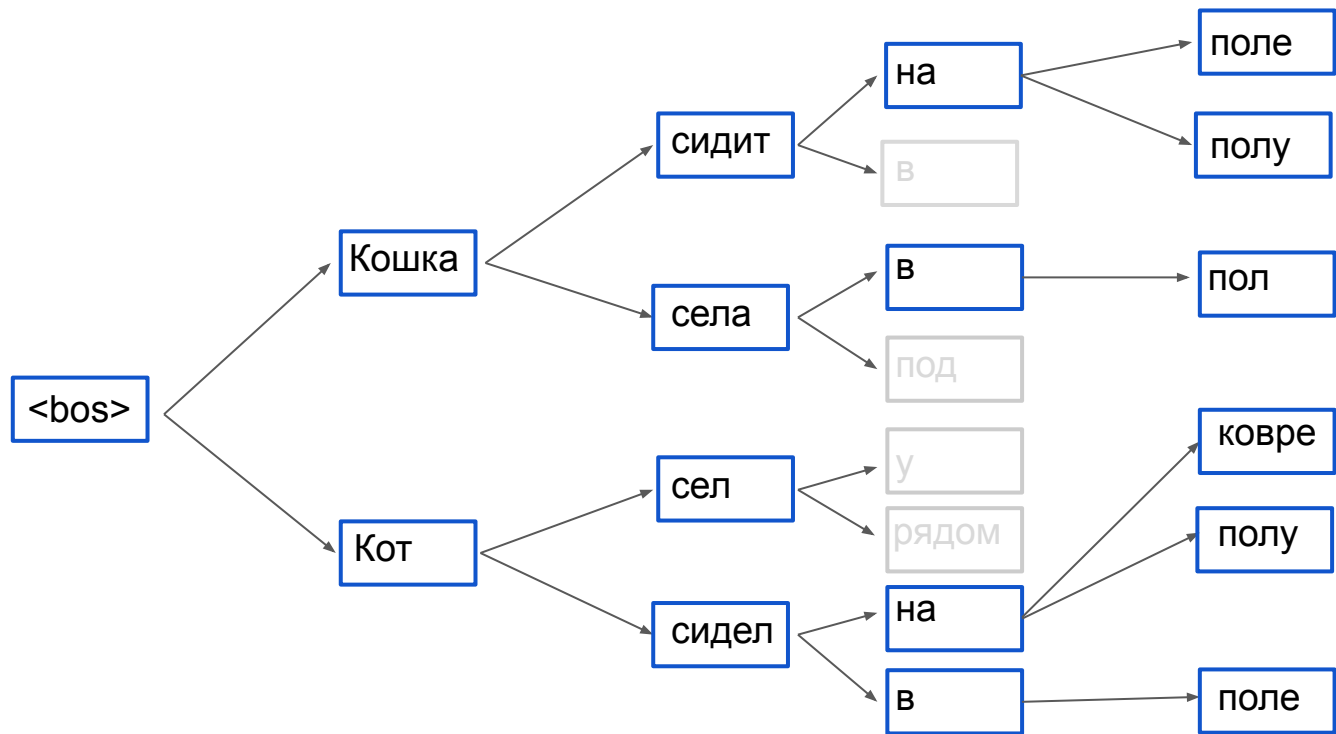
Beam search

Оставляем только фиксированное количество гипотез (4 в нашем случае)

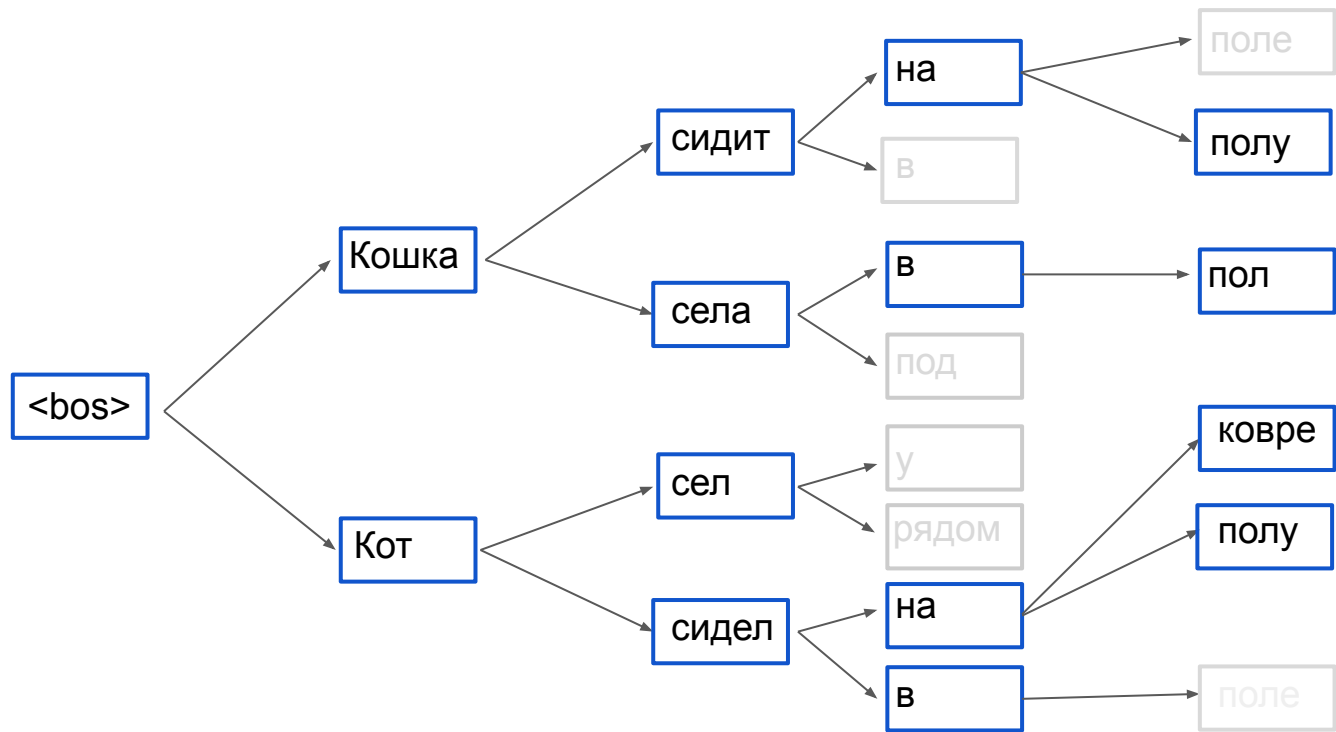


Beam search

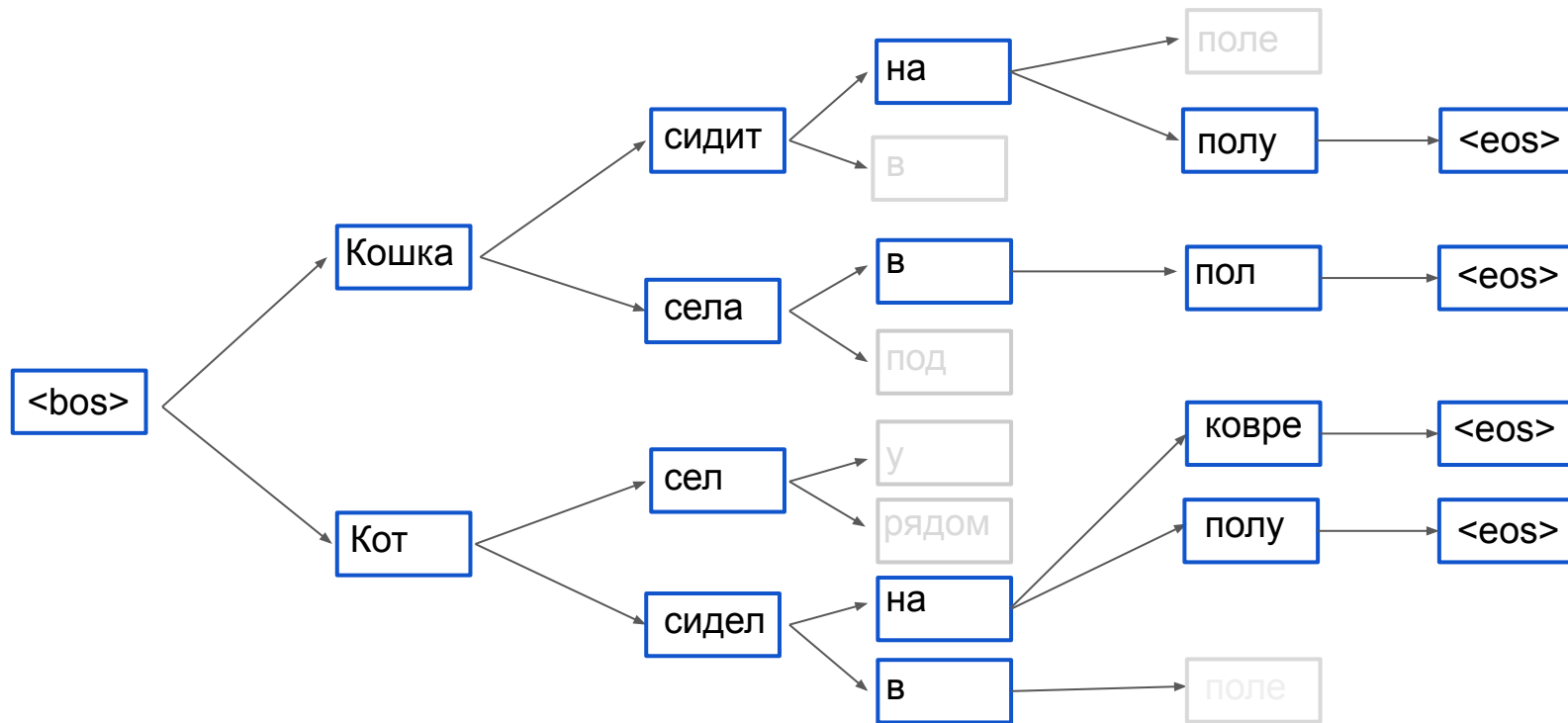
Продолжаем разворачивать лучшие гипотезы



Beam search

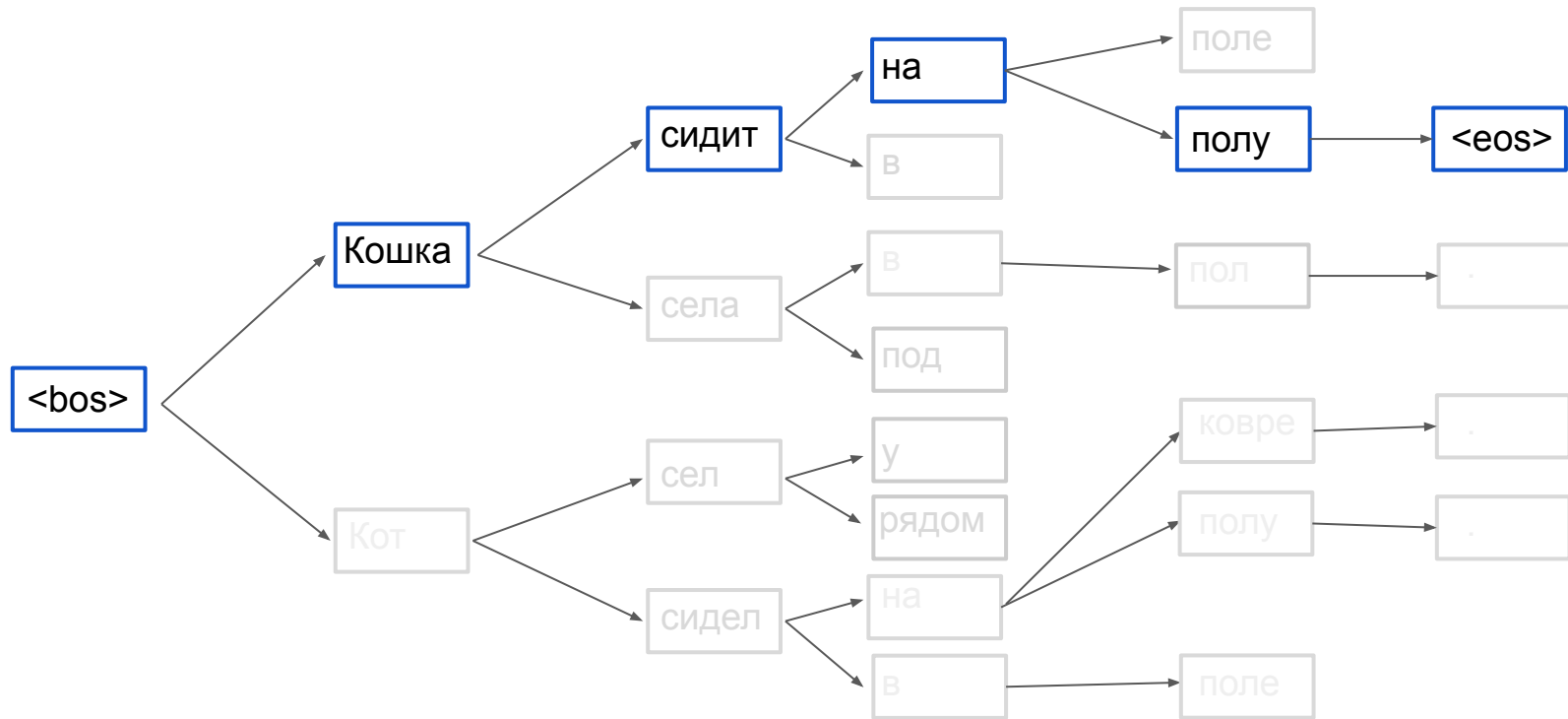


Beam search



Beam search

Оставляем только лучшую гипотезу



Beam search

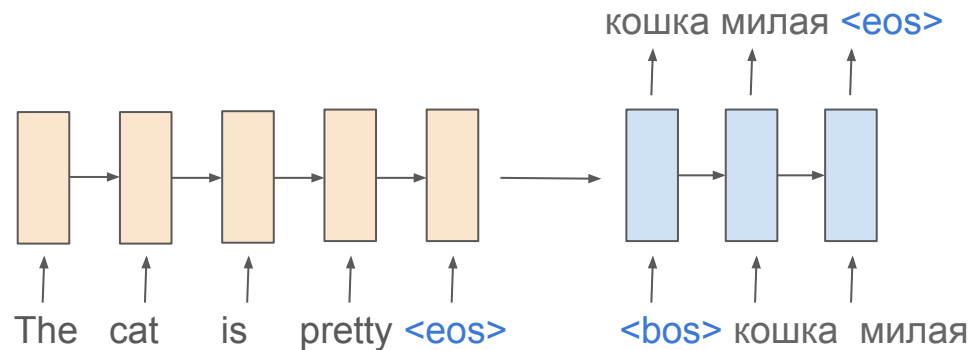
Beam search: при инференсе поддерживаем фиксированное количество гипотез.

Оптимальный размер бина — $\sim 4-8$

Если размер бина слишком большой, качество перевода падает.

Encoder-Decoder: проблемы

- “жадное” декодирование
- “забывчивость” RNN: энкодер может забывать начало длинных предложений
- вся информация о длинном предложении записывается в один вектор!



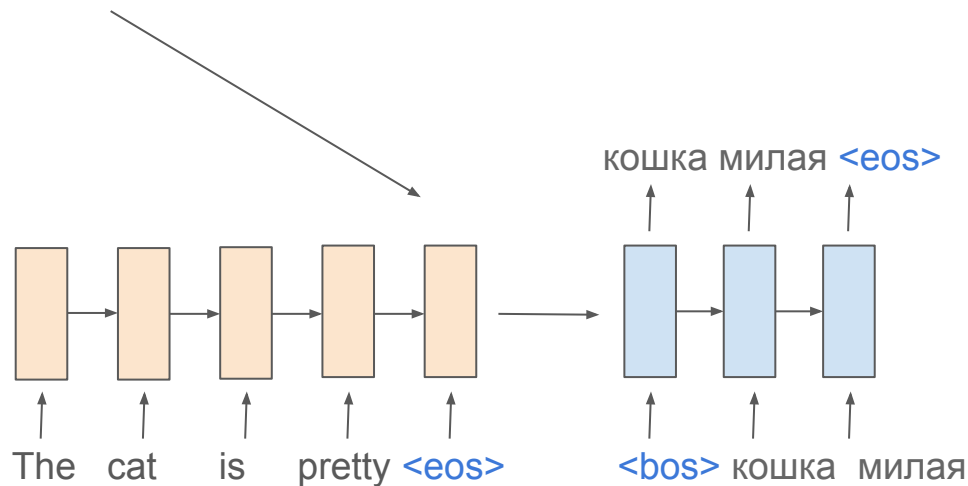
Attention

Немного истории

- **До 2015:** NMT решалось с помощью Encoder-Decoder, но показывало результаты хуже, чем статистические методы
- **2015:** году придумали Attention
- **2016:** NMT в продакшене гугла на translate.google.com
- **2017:** статья Google: “Attention is all you need”, начало эры трансформеров
- **2017-сейчас:** Attention используется везде, не только в NLP, но и в CV и других областях.

Encoder-Decoder: проблемы

- “забывчивость” RNN: энкодер может забывать начало длинных предложений
- вся информация о длинном предложении записывается в один вектор!



Attention: идея

Пусть нам нужно перевести длинное предложение:

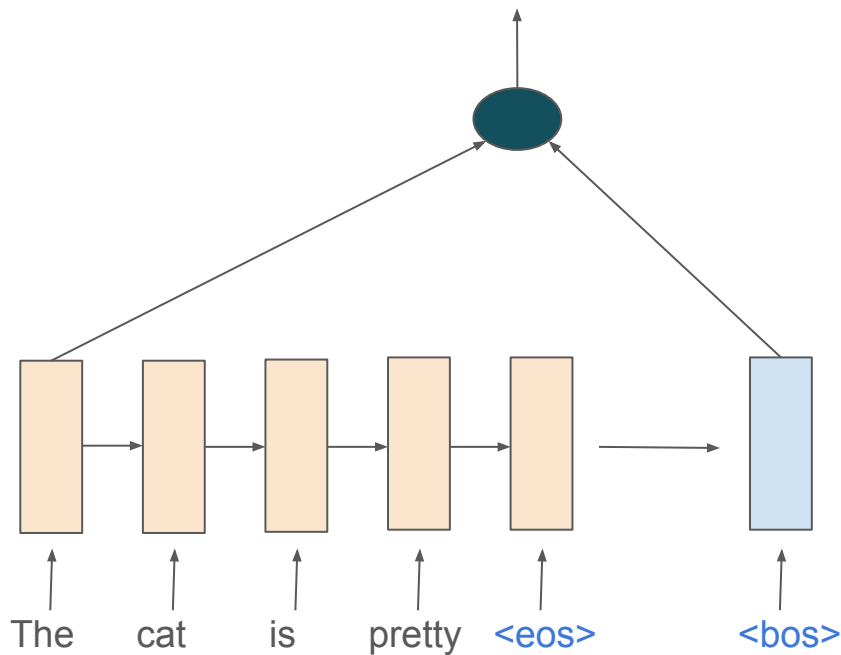
A man in a nice cowboy hat and a parrot on his shoulder has entered the bar.

Когда мы, как человек, будем переводить это предложение, в процессе перевода мы несколько раз взглянем на части изначального предложения.

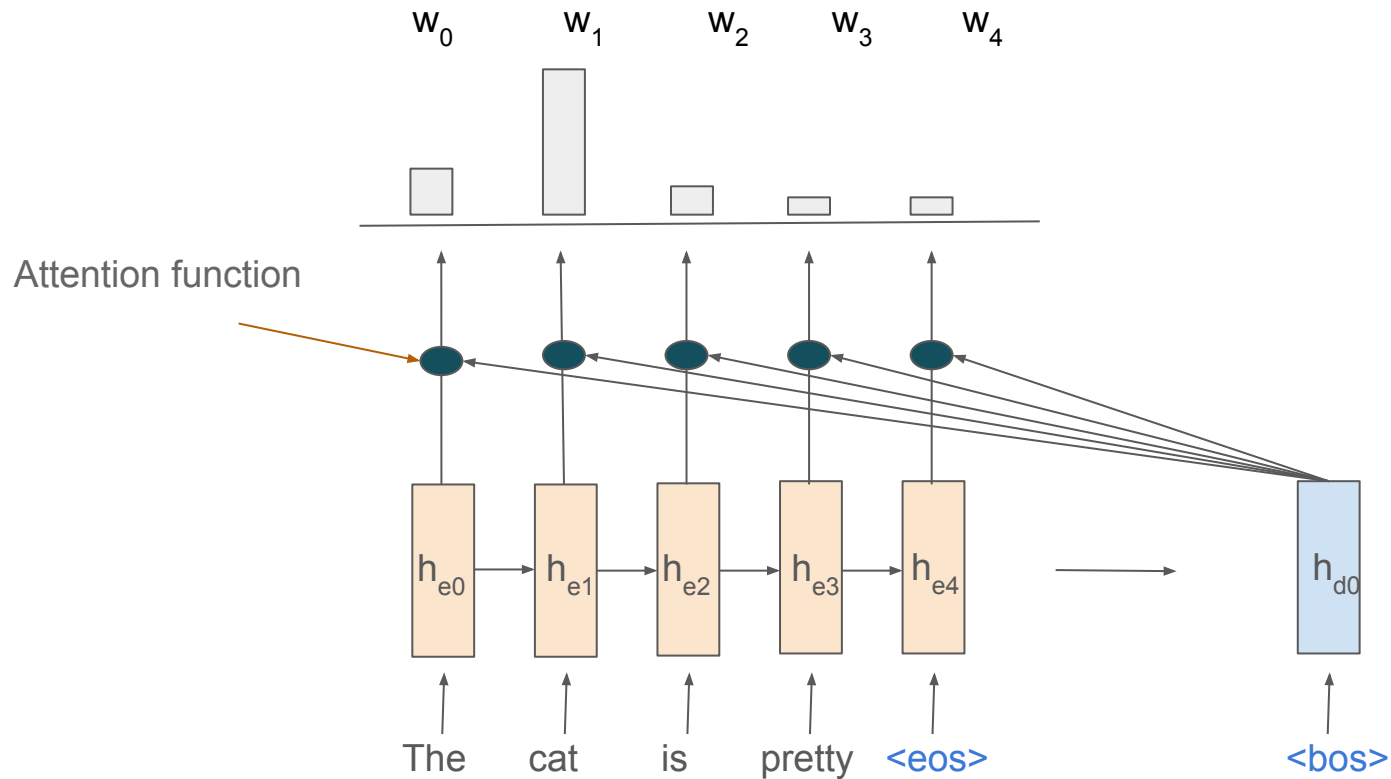
Человек в красивой ковбойской шляпе и попугаем на плече зашел в бар.

Attention

насколько декодеру сейчас важно посмотреть на слово “the”,
чтобы правильно перевести следующее слово?



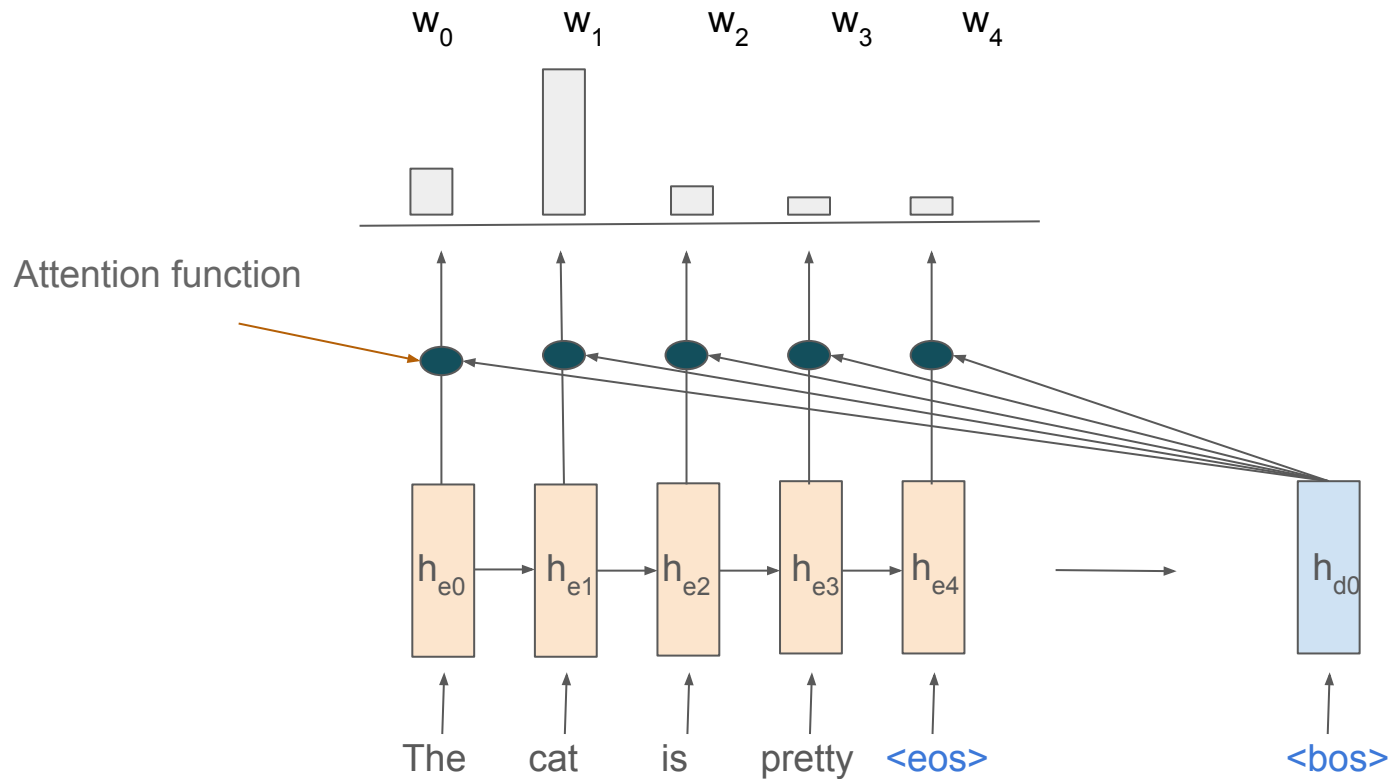
Attention



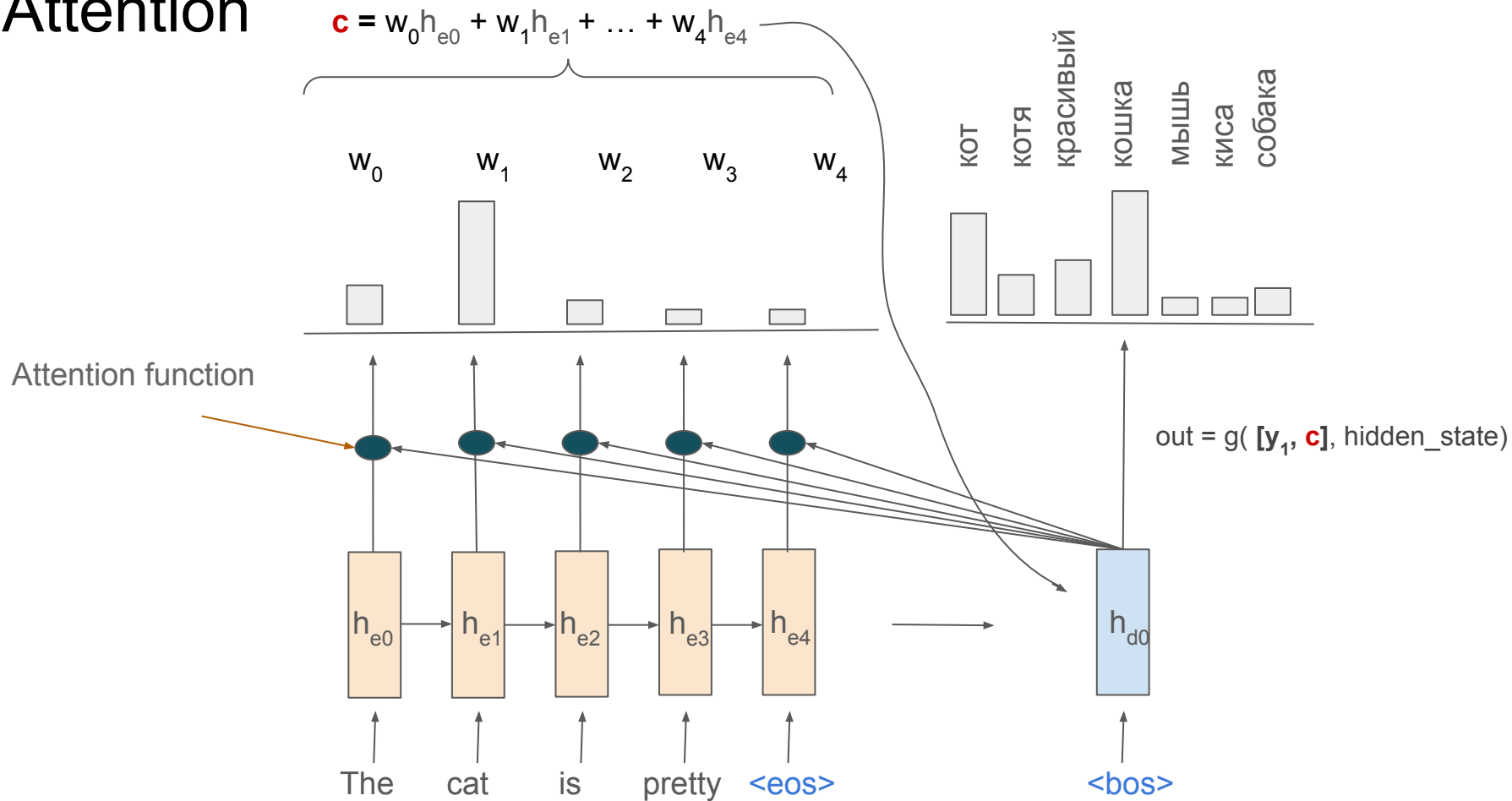
Attention

$$\mathbf{c} = w_0 h_{e0} + w_1 h_{e1} + \dots + w_4 h_{e4}$$

вектор контекста



Attention



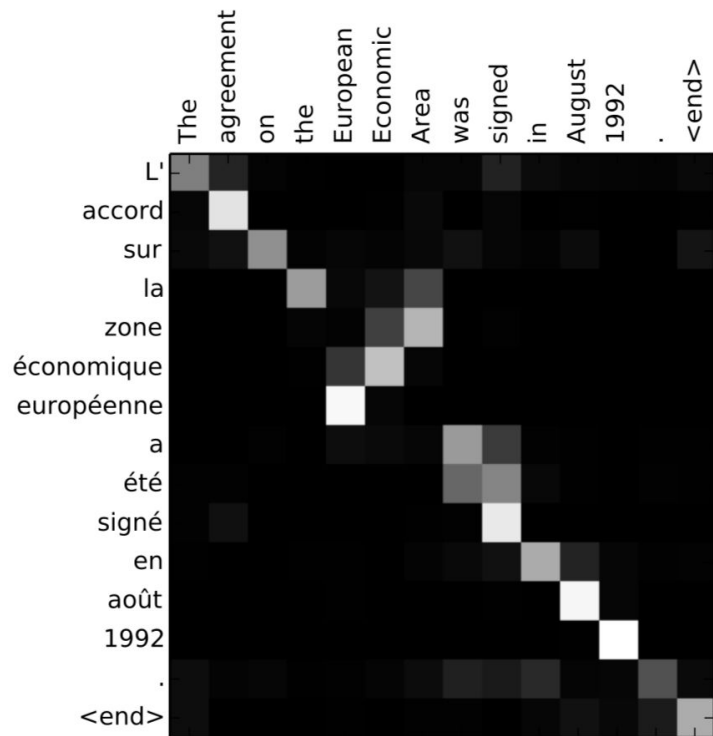
Attention

Attention на каждом шаге декодера выделяет из энкодера ту информацию, которая понадобится декодеру именно в этот момент для генерации следующего слова перевода.

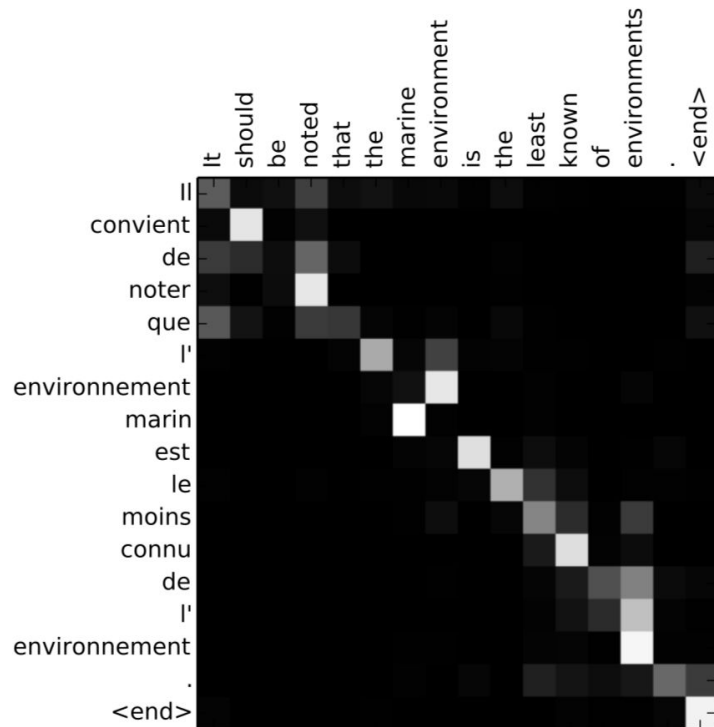
Attention может быть разной функцией. Примеры:

- dot product: $\text{Attn}(h_{e0}, h_{d0}) = \langle h_{e0}, h_{d0} \rangle$
- one-layer nn: $\text{Attn}(h_{e0}, h_{d0}) = \text{nn.Linear}(\text{torch.cat}([h_{e0}, h_{d0}]))$
- whatever

Attention



(a)



(b)

Attention

Attention действительно помогает сети переводить длинные предложения

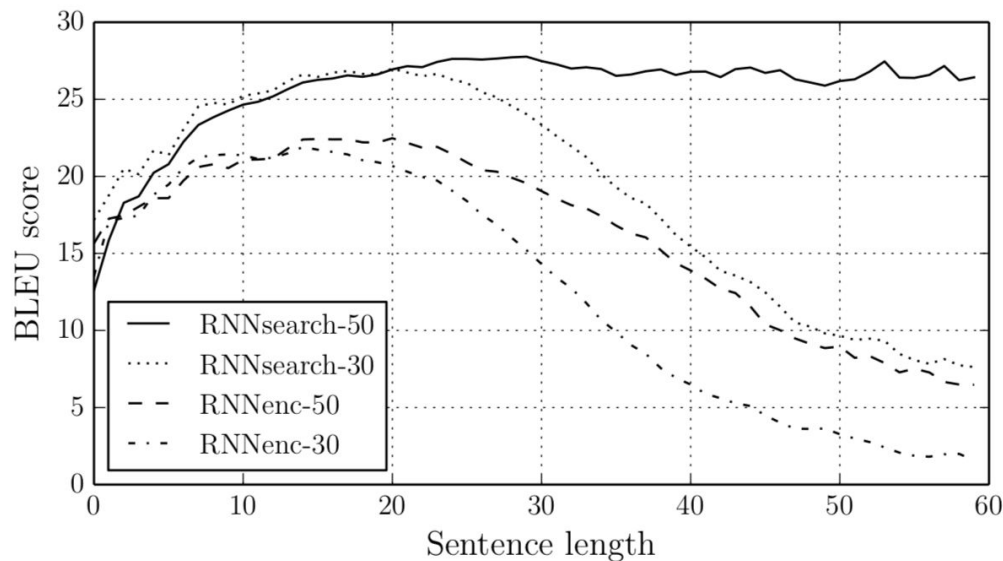


Figure 2: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. The results are on the full test set which includes sentences having unknown words to the models.

Self-Attention: идея

Пусть нам нужно перевести длинное предложение:

A man in a nice cowboy hat and a parrot on his shoulder **has entered** the bar.

Когда мы, как человек, будем переводить это предложение, в процессе перевода мы несколько раз взглянем на части изначального предложения.

Человек в красивой ковбойской шляпе и попугаем на плече ... [зашел? зашла? зашло? залетело?]

Self-Attention: идея

Пусть нам нужно перевести длинное предложение:

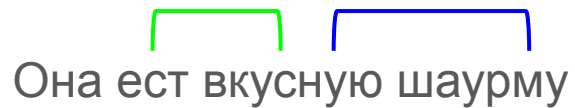
A man in a nice cowboy hat and a parrot on his shoulder **has entered** the bar.

Когда мы, как человек, будем переводить это предложение, в процессе перевода мы несколько раз взглянем на части изначального предложения.

Человек в красивой ковбойской шляпе и попугаем на плече **зашел** в бар

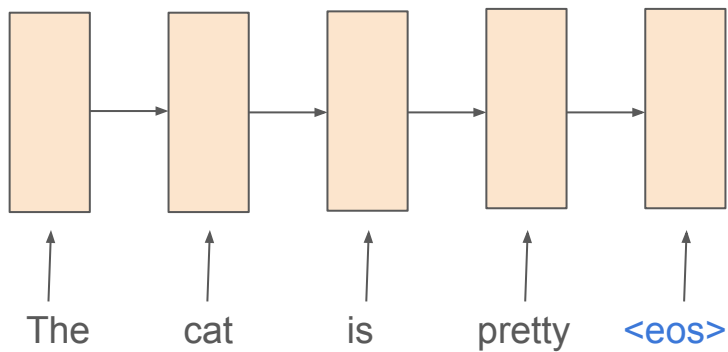
Self-attention

Она ест вкусную шаурму

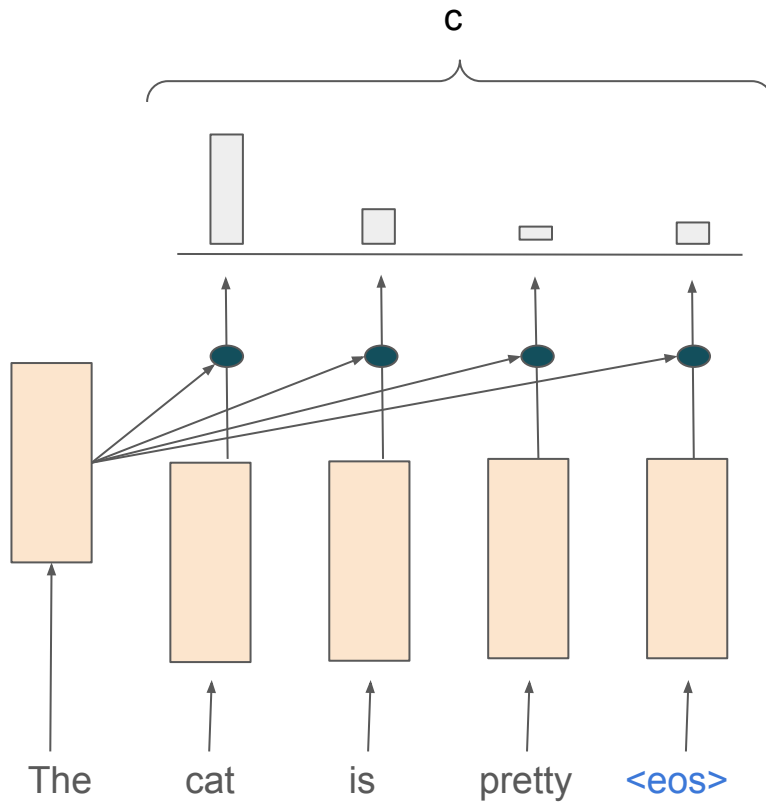


- согласование по падежу (case agreement)
- согласование по роду (gender agreement)

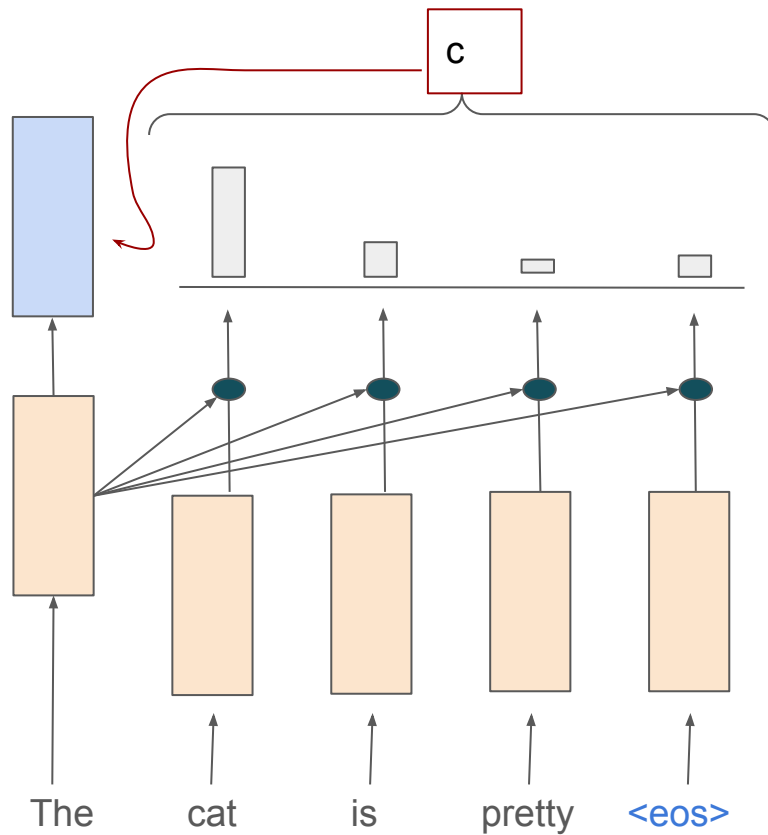
Self-attention: Encoder



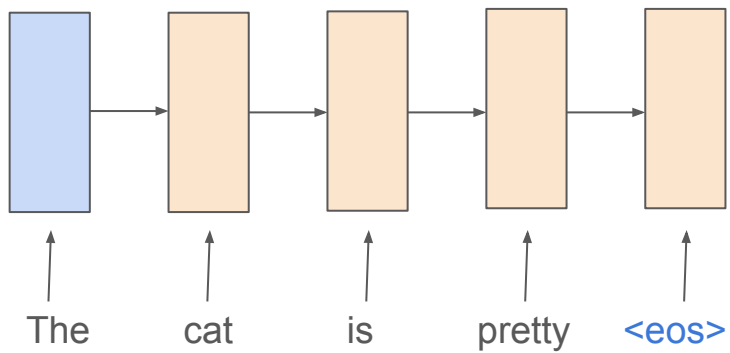
Self-attention



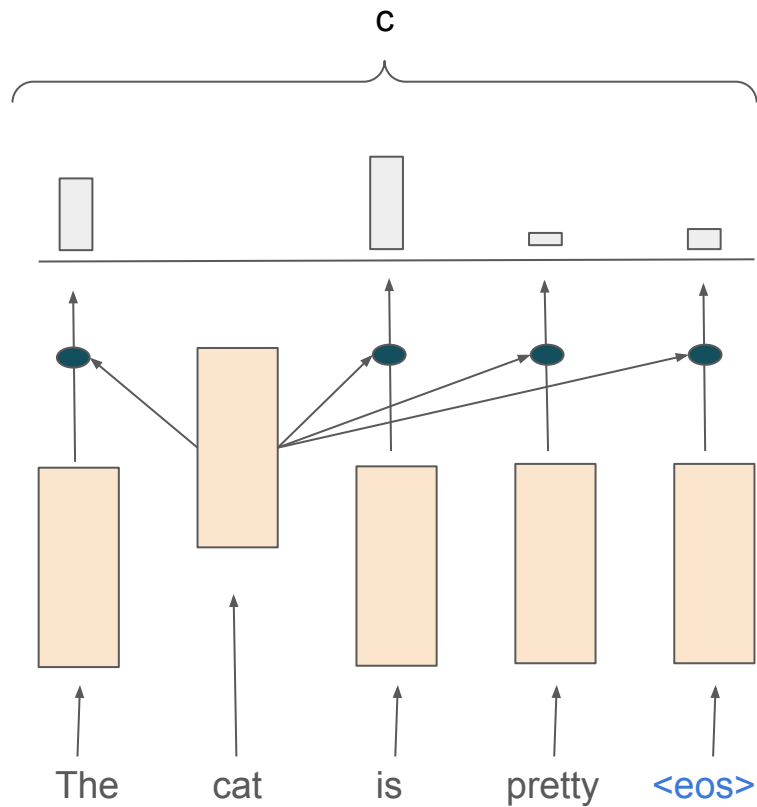
Self-attention



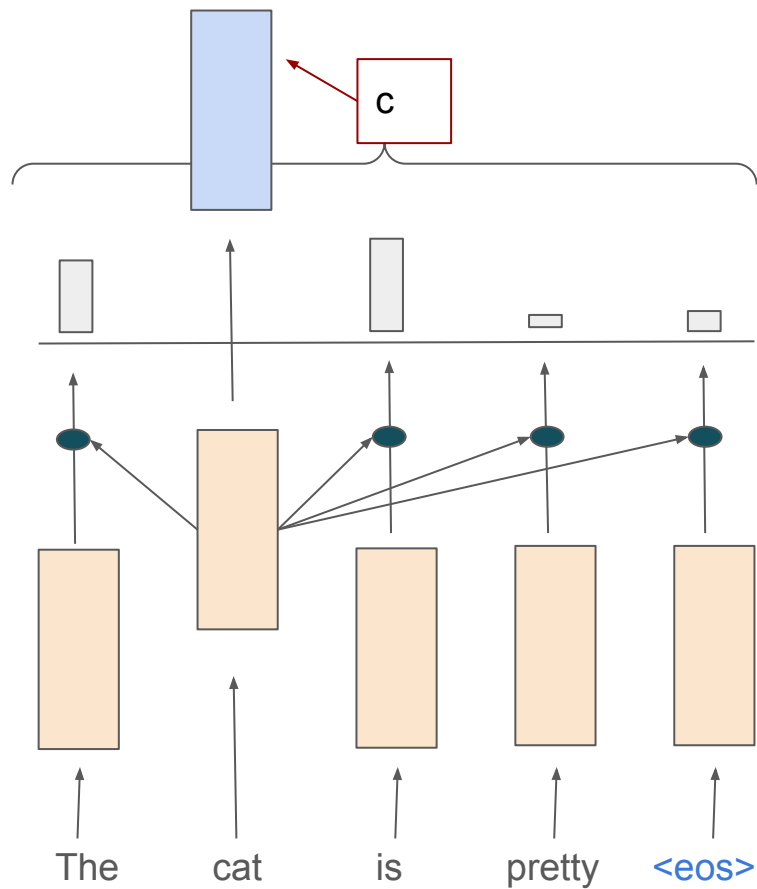
Self-attention



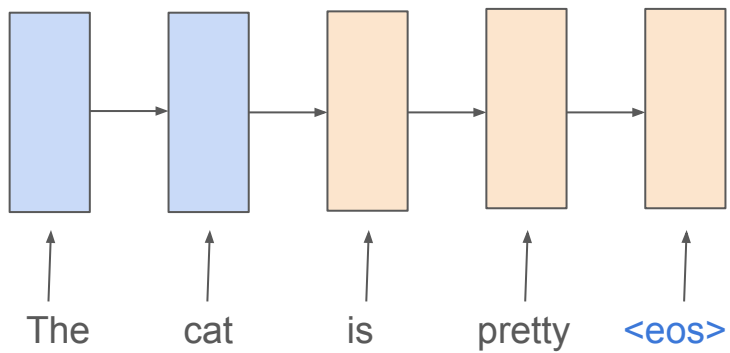
Self-attention



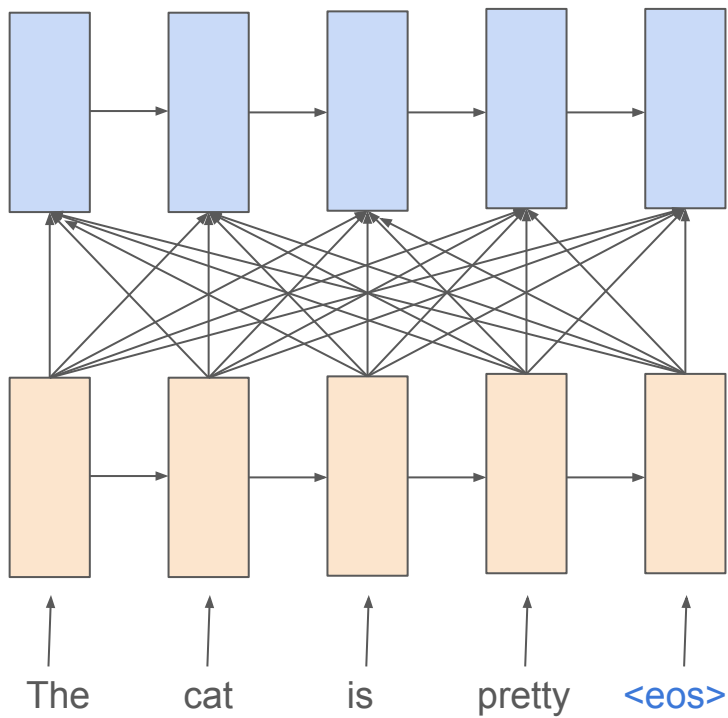
Self-attention



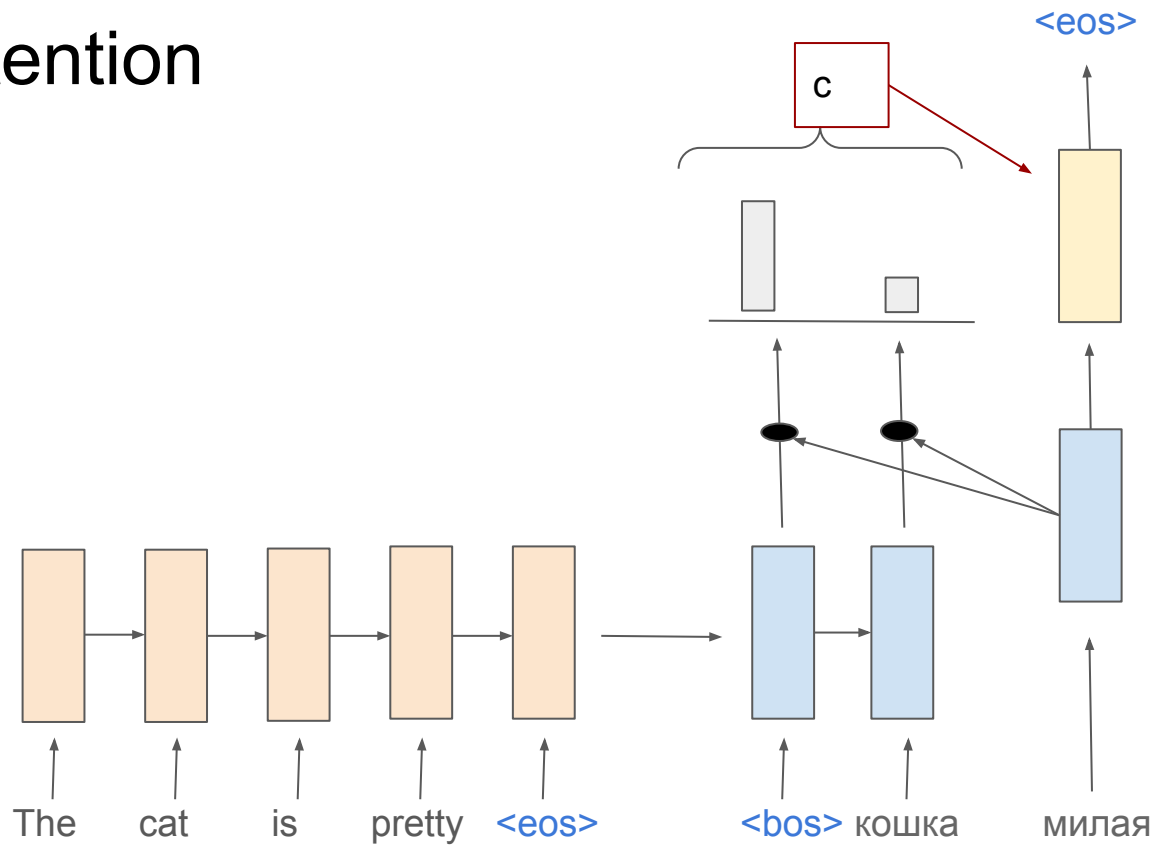
Self-attention



Self-attention




Self-Attention



Multi-head attention

Она ест вкусную шаурму



- согласование по падежу (case agreement)
- согласование по роду (gender agreement)

В модели может быть несколько Attention'ов и Self-Attention'ов. Каждый из них будет выделять определенную информацию: например, согласование по падежу/лицу/роду и т.д.

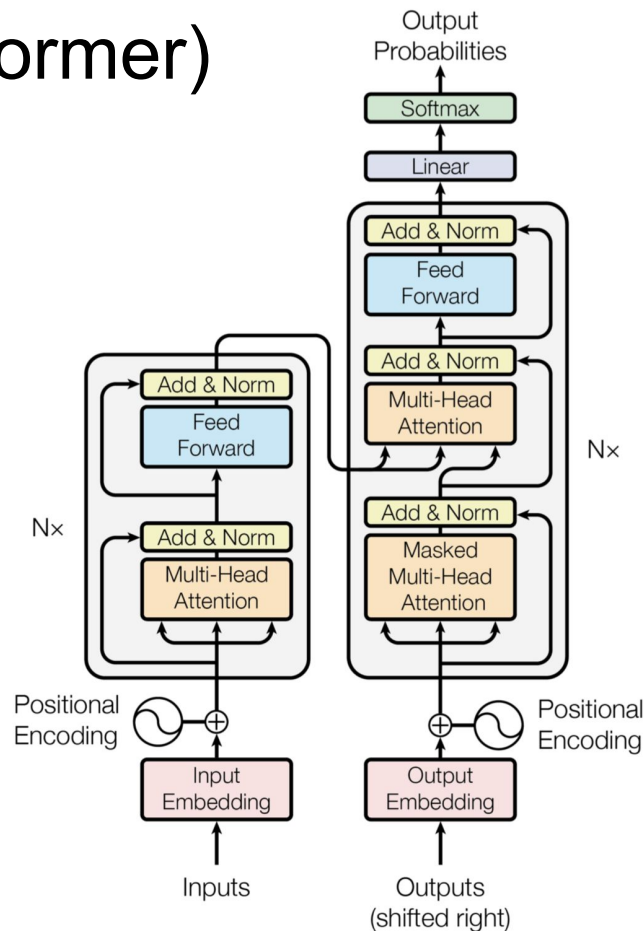
$$Multi - head\ attn = Concat(attn_1, attn_2, \dots, attn_8)W$$

Attention

- Attention позволяет модели не помнить слишком много информации. Благодаря Attention модель может “заглянуть” в части предложения и вытащить нужную информацию
- Attention’ов может быть много, и тогда каждый из них отвечает за определенную связь между словами.
- Attention — общая идея “внимания” и используется не только в NLP

Attention is all you need (~ Transformer)

- Не рекуррентный энкодер -> параллельное кодирование предложения энкодером -> модель работает быстрее
- Много attention'ов -> модели не нужно многое помнить
- Multi-head attention -> модель может обращать внимание на разные связи между словами в предложении



Transformer

Transformer

The animal didn't cross the street because it was too tired.
L'animal n'a pas traversé la rue parce qu'il était trop fatigué.

The animal didn't cross the street because it was too wide.
L'animal n'a pas traversé la rue parce qu'elle était trop large.

Transformer

