NLP: обзор

Евгений Соколов

esokolov@hse.ru

НИУ ВШЭ, 2024

Задачи обработки языка (NLP)

- Классификация текстов/документов
- Распознавание именных сущностей (NER)
- Машинный перевод
- Вопрос-ответные системы (QA)
- Извлечение информации (IR)
- Суммаризация текстов/документов
- Генерация текста

и множество других

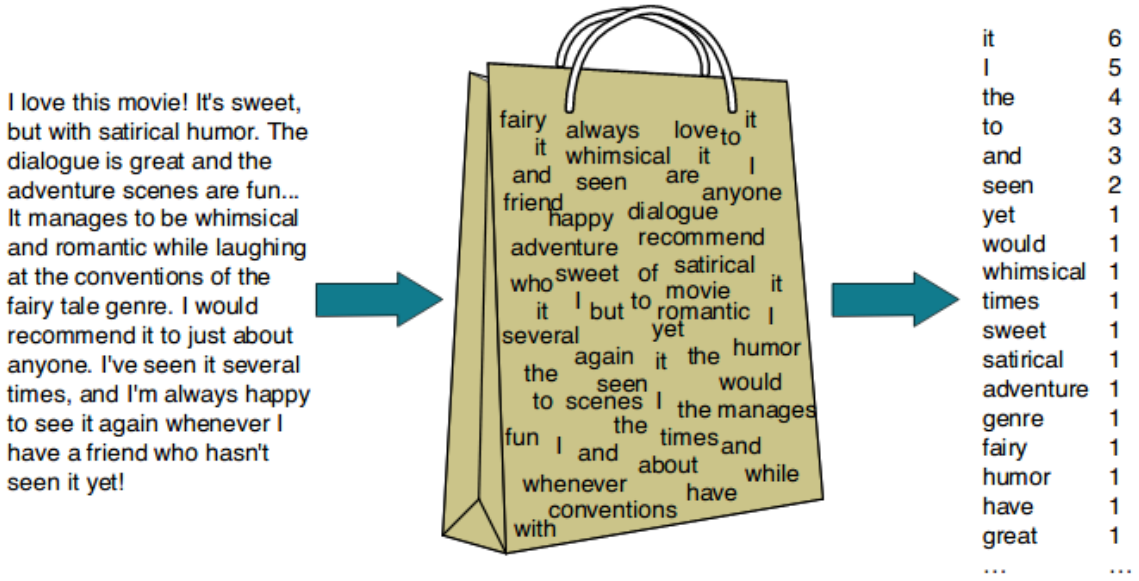
Реальные кейсы применения NLP

- Поисковые системы (Google, Yandex, etc.)
- Машинный перевод (Google Translate, Abbyy Lingvo, Linguee, etc.)
- Виртуальные ассистенты, голосовые помощники etc.
- Фильтр спама (e-mail / телефонный / etc.)
- Дополнение текста, автокоррекция
- Авторазметка отзывов пользователей
- Чат-боты
- Автосуммаризация текста

Подходы к решению NLP-задач

Классическое машинное обучение:

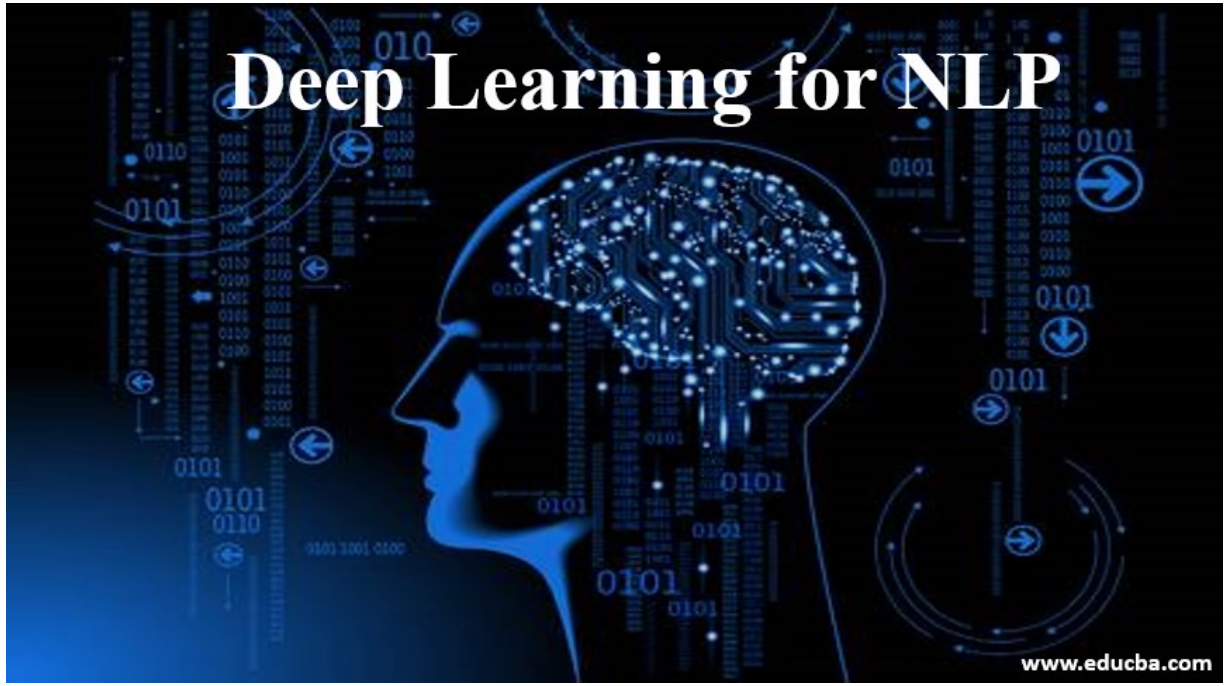
- Извлекаем признаки из текстов (bag of words, tf-idf)
- На этих признаках обучаем ML-модель



Подходы к решению NLP-задач

Глубинное обучение:

- Нейронные сети самостоятельно извлекают необходимую информацию из текстов



Рекуррентные модели

Марковские модели

- Предположение: наличие конкретного слова в тексте объясняется только k словами перед ним
- $p(w_1, \dots, w_n) = p(w_1)p(w_2|w_1) \dots p(w_n|w_{n-1}, \dots, w_{n-k})$
- $p(w_n|w_{n-1}, \dots, w_{n-k})$ — можно оценить
- Как часто встречается слово w_n после последовательности из слов $w_{n-1}, \dots, w_{n-k}$?
- Обычно делают со сглаживанием

Марковские модели

```
'I am a master armorer , lords of Westeros , sawing out each bay and peninsula
'Jon Snow is with the Night's Watch . I did not survive a broken hip , a leath
'Jon Snow is with the Hound in the woods . He won't do it . " Please don't'
'Where are the chains , and the Knight of Flowers to treat with you , Imp . "'
'Those were the same . Arianne demurred . " So the fishwives say , " It was Ty
'He thought that would be good or bad for their escape . If they can truly giv
'I thought that she was like to remember a young crow he'd met briefly years b
```

https://www.kdnuggets.com/2019/11/markov-chains-train-text-generation.html

Идея

- Мы читаем текст последовательно
- И постепенно всё лучше понимаем, о чём он

Рекуррентные сети (RNN)

- Последовательность: $x_1, x_2, \ldots, x_n, \ldots$
- Читаем слева направо
- h_t — накопленная информация после чтения t элементов (вектор)

Рекуррентные сети (RNN)

- Последовательность: $x_1, x_2, \ldots, x_n, \ldots$
- x_i — либо one-hot вектор, либо векторное представление (word2vec, fasttext, …)

Рекуррентные сети (RNN)

- Последовательность: $x_1, x_2, \ldots, x_n, \ldots$
- Читаем слева направо
- h_t — накопленная информация после чтения t элементов (вектор)
- $h_t = f(W_{xh}x_t + W_{hh}h_{t-1})$
- Если хотим что-то выдавать на каждом шаге: $o_t = f_o(W_{ho}h_t)$

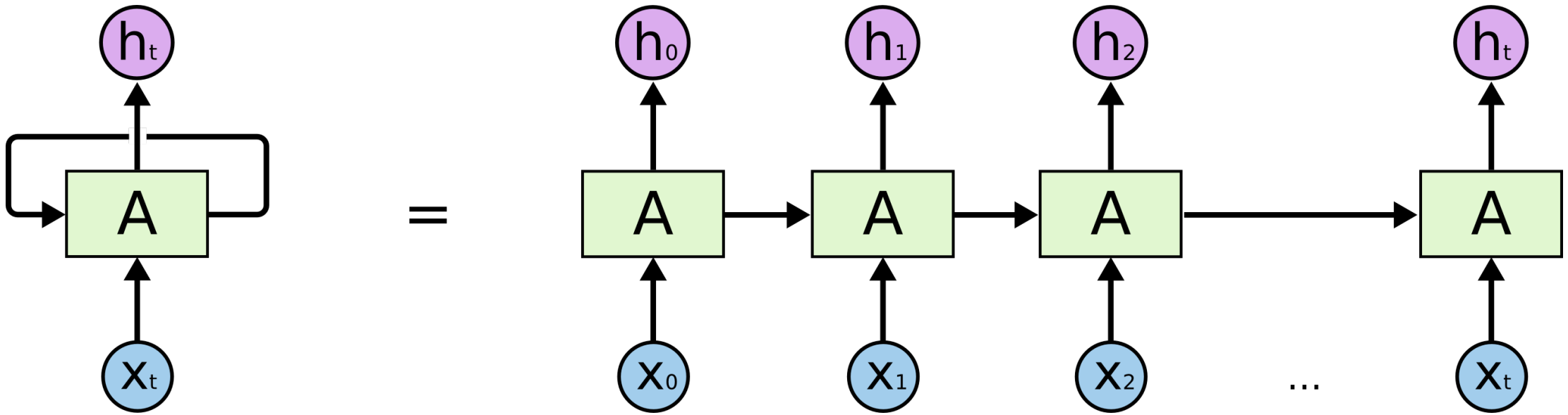
Рекуррентные сети (RNN)

- Типичный случай: $o_t \in \mathbb{R}^N$
- N — размер словаря
- То есть предсказываем вероятность того, что здесь стоит конкретное слово
- Предсказываем следующее слово

Рекуррентные сети (RNN)

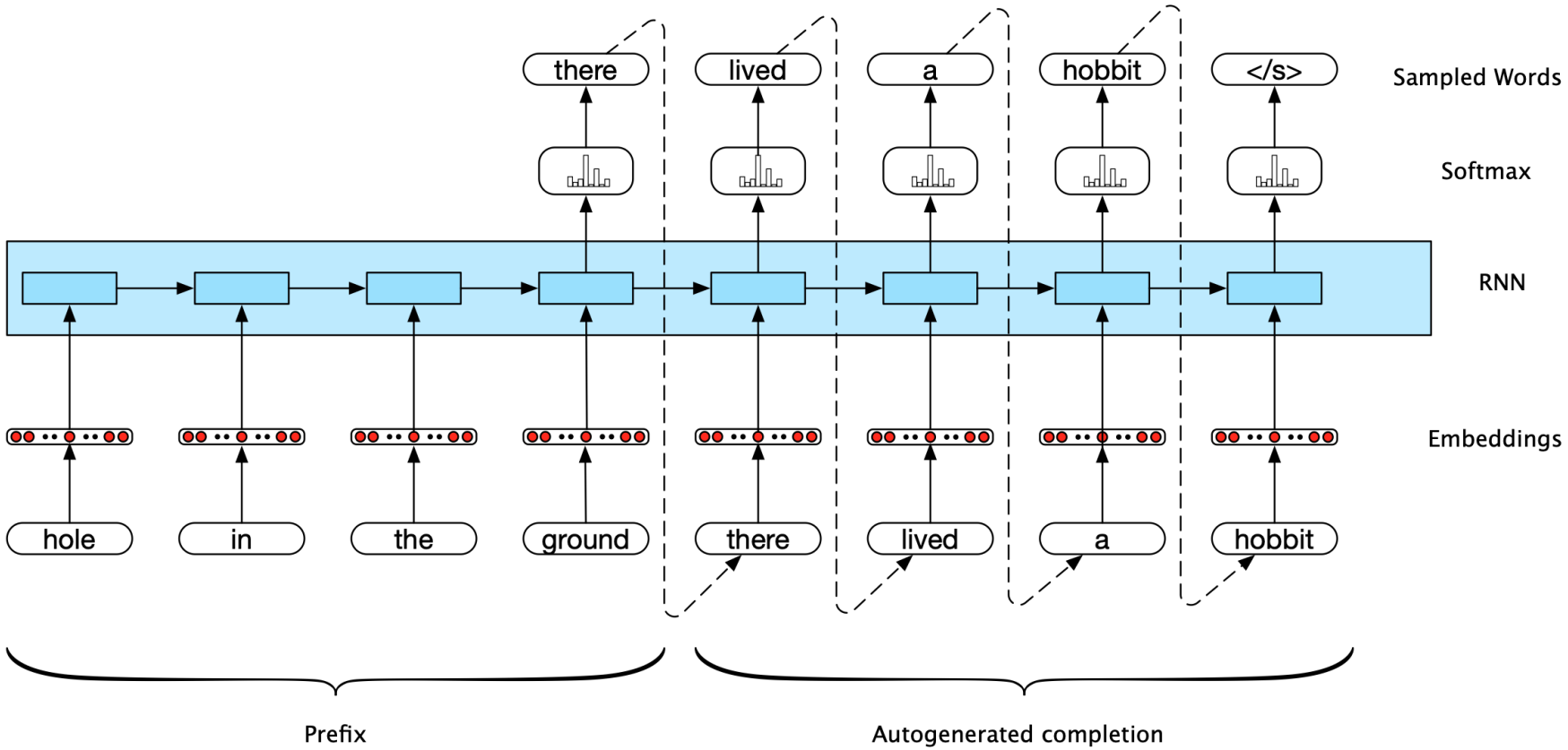
- Типичный случай: $o_t \in \mathbb{R}^N$
- N — количество частей речи
- POS tagging

Рекуррентные сети (RNN)



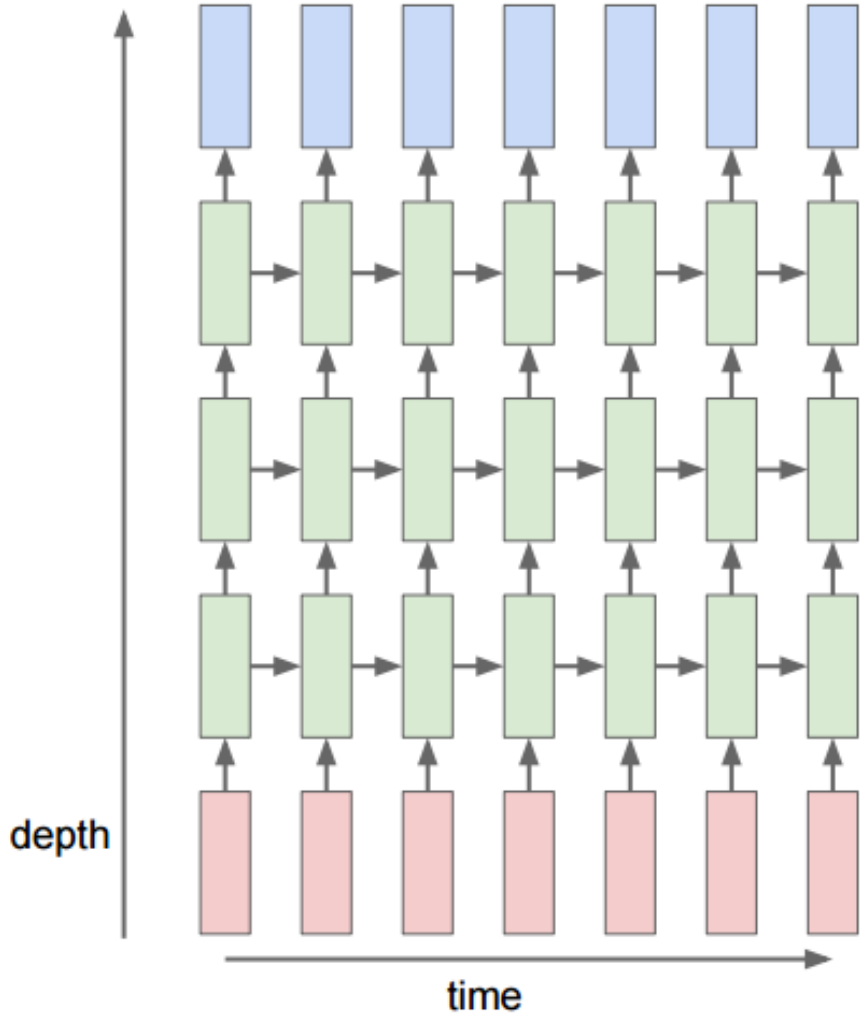
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Рекуррентные сети (RNN)



https://web.stanford.edu/~jurafsky/slp3/10.pdf

Можно делать многослойные RNN



https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/recurrent-neural-network/recurrent_neural_networks

Примеры

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Примеры

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariccoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ??, and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M^\bullet} = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ??. It may replace S by $X_{spaces,étale}$ which gives an open subspace of X and T equal to S_{Zar}, see Descent, Lemma ??. Namely, by Lemma ?? we see that R is geometrically regular over S.

Lemma 0.1. *Assume (3) and (3) by the construction in the description.*

Suppose $X = \lim |X|$ *(by the formal open covering* X *and a single map* $\underline{Proj}_X(\mathcal{A}) = \mathrm{Spec}(B)$ *over* U *compatible with the complex*

$$Set(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{Z/X}$ *is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If* T *is surjective we may assume that* T *is connected with residue fields of* S*. Moreover there exists a closed subspace* $Z \subset X$ *of* X *where* U *in* X' *is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem*

(1) f *is locally of finite type. Since* $S = \mathrm{Spec}(R)$ *and* $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\ldots,n} U_i$ be the scheme X over S at the schemes $X_i \to X$ and $U = \lim_i X_i$. □

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\ldots,0}$.

Lemma 0.2. *Let* X *be a locally Noetherian scheme over* S, $E = \mathcal{F}_{X/S}$. *Set* $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. *Since* $\mathcal{I}^n \subset \mathcal{I}^n$ *are nonzero over* $i_0 \leq \mathfrak{p}$ *is a subset of* $\mathcal{J}_{n,0} \circ \overline{A}_2$ *works.*

Lemma 0.3. *In Situation ??. Hence we may assume* $\mathfrak{q}' = 0$.

Proof. We will use the property we see that $\mathfrak{p}$ is the mext functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F-algebra where δ_{n+1} is a scheme over S. □

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Примеры

```
/*
 * 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;
}
```

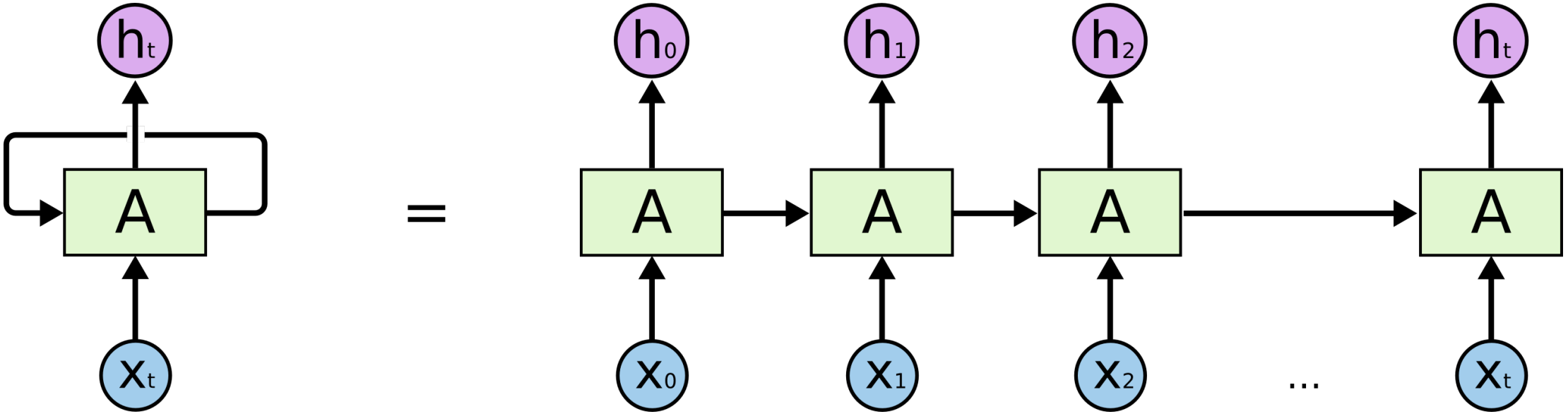
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Seq2seq

Sequence to sequence

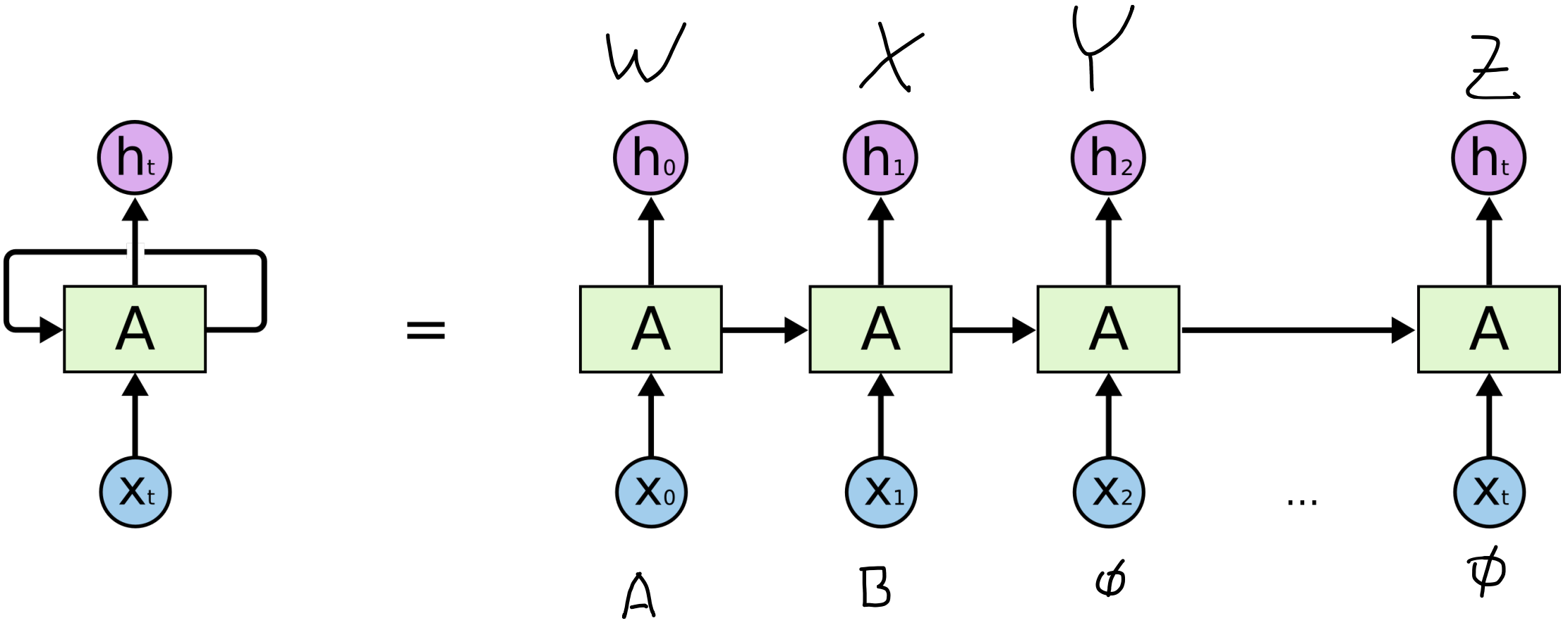
- Машинный перевод
- Суммаризация текста
- Генерация комментариев к коду
- Математические преобразования
- Смена стиля текста

Seq2seq Machine Translation

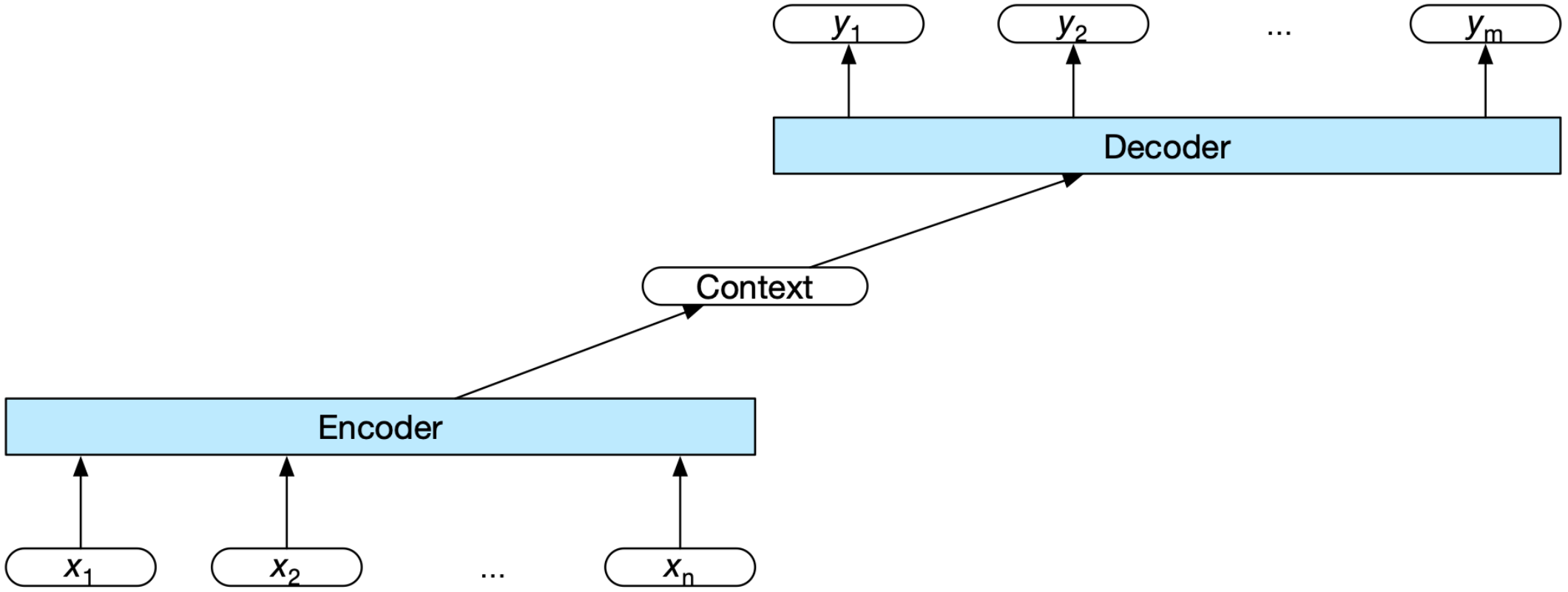


Что делать, если длины входного и выходного текстов разные?

Seq2seq Machine Translation

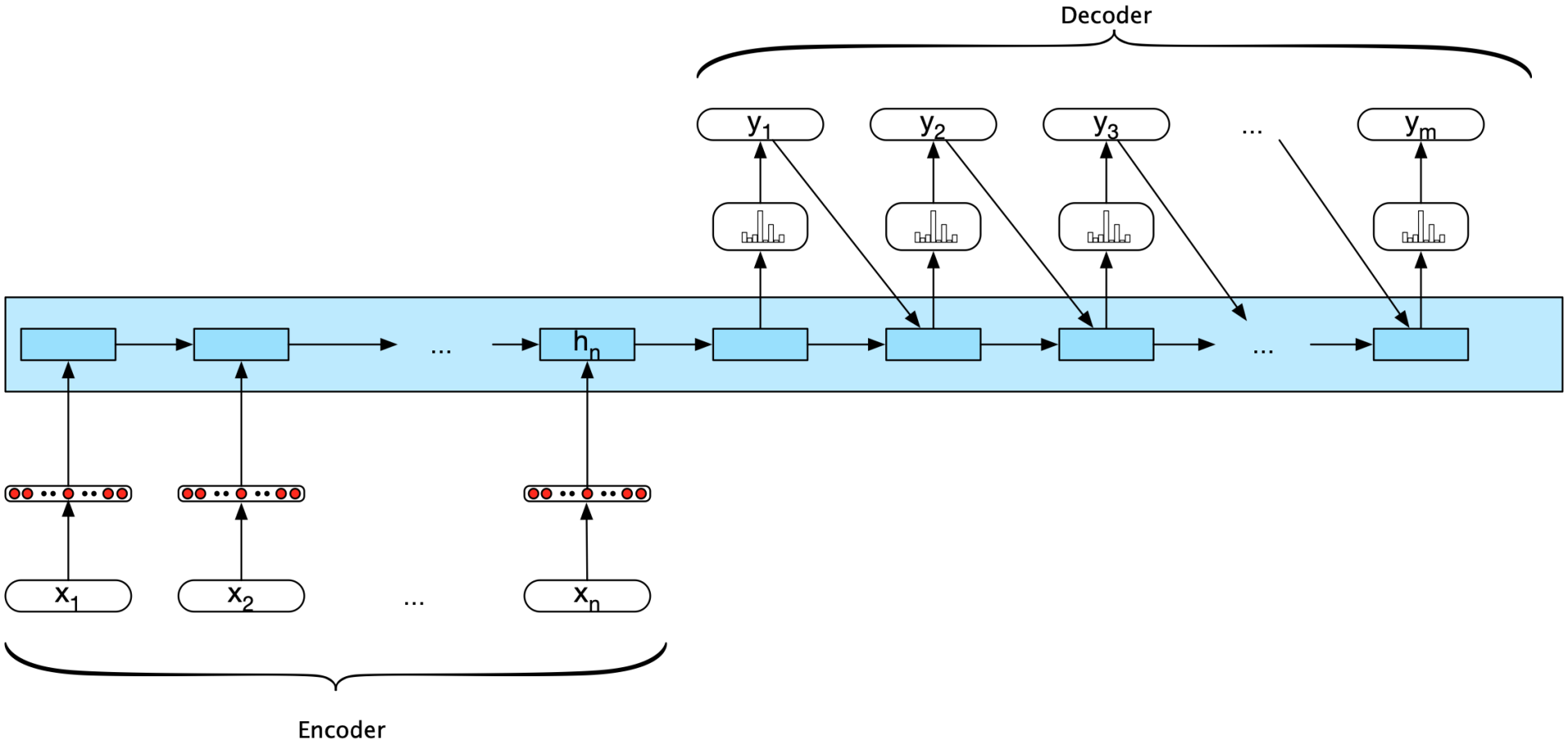


Seq2seq Machine Translation



https://web.stanford.edu/~jurafsky/slp3/10.pdf

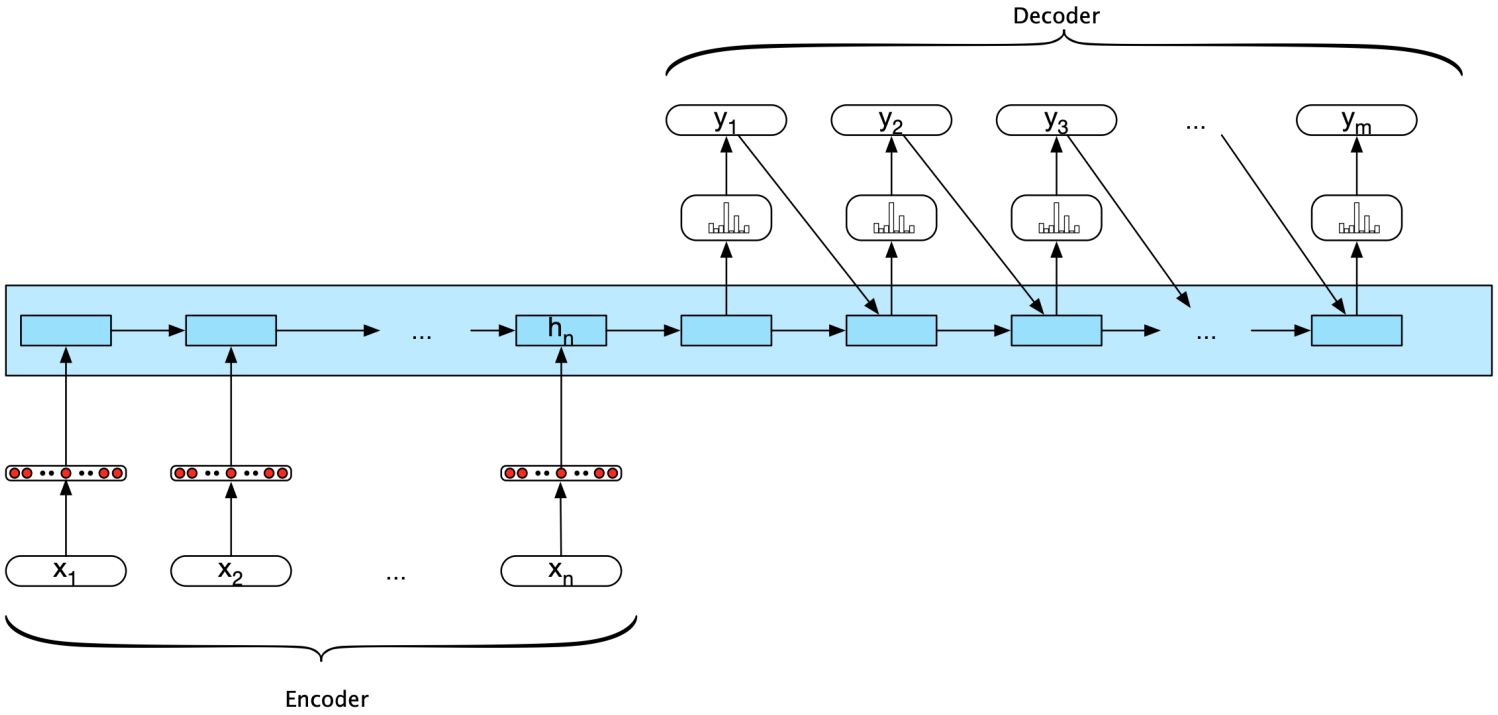
Seq2seq Machine Translation



https://web.stanford.edu/~jurafsky/slp3/10.pdf

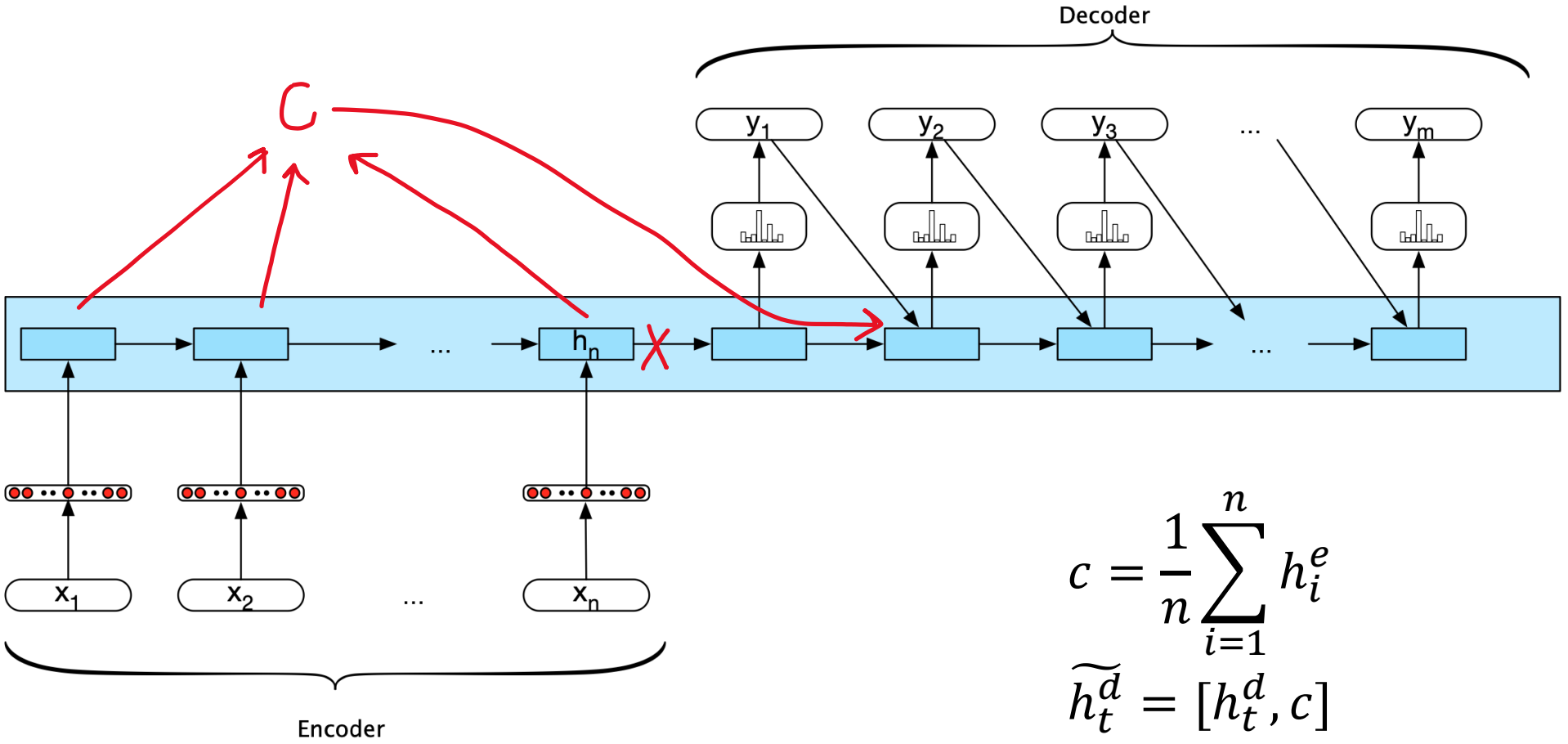
Механизм внимания

Seq2seq



- Читаем текст слева направо, собираем информацию о словах внутри скрытого вектора h_t
- Вряд ли можно уместить полный смысл текста в одном векторе
- Скрытый вектор всего входного текста — «бутылочное горлышко» в такой архитектуре

Механизм внимания: версия 1



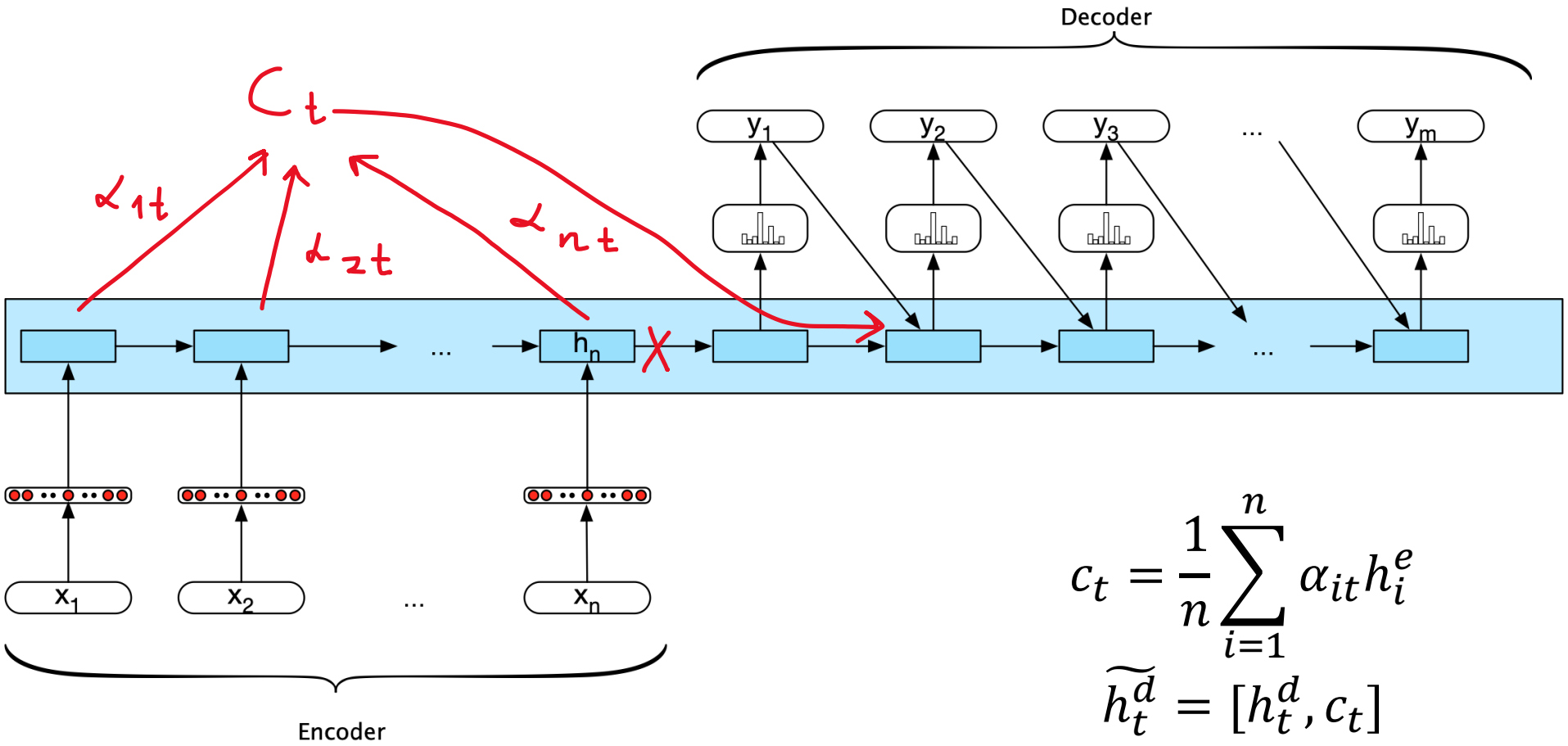
$$c = \frac{1}{n}\sum_{i=1}^{n} h_i^e$$

$$\widetilde{h_t^d} = [h_t^d, c]$$

Механизм внимания: версия 1

- При генерации всех выходов мы используем одну и ту же информацию о входном тексте
- Это мало чем отличается от seq2seq-архитектуры
- Хочется, чтобы каждое выходное слово могло «обращать внимание» на «свои» входные слова

Механизм внимания: версия 2



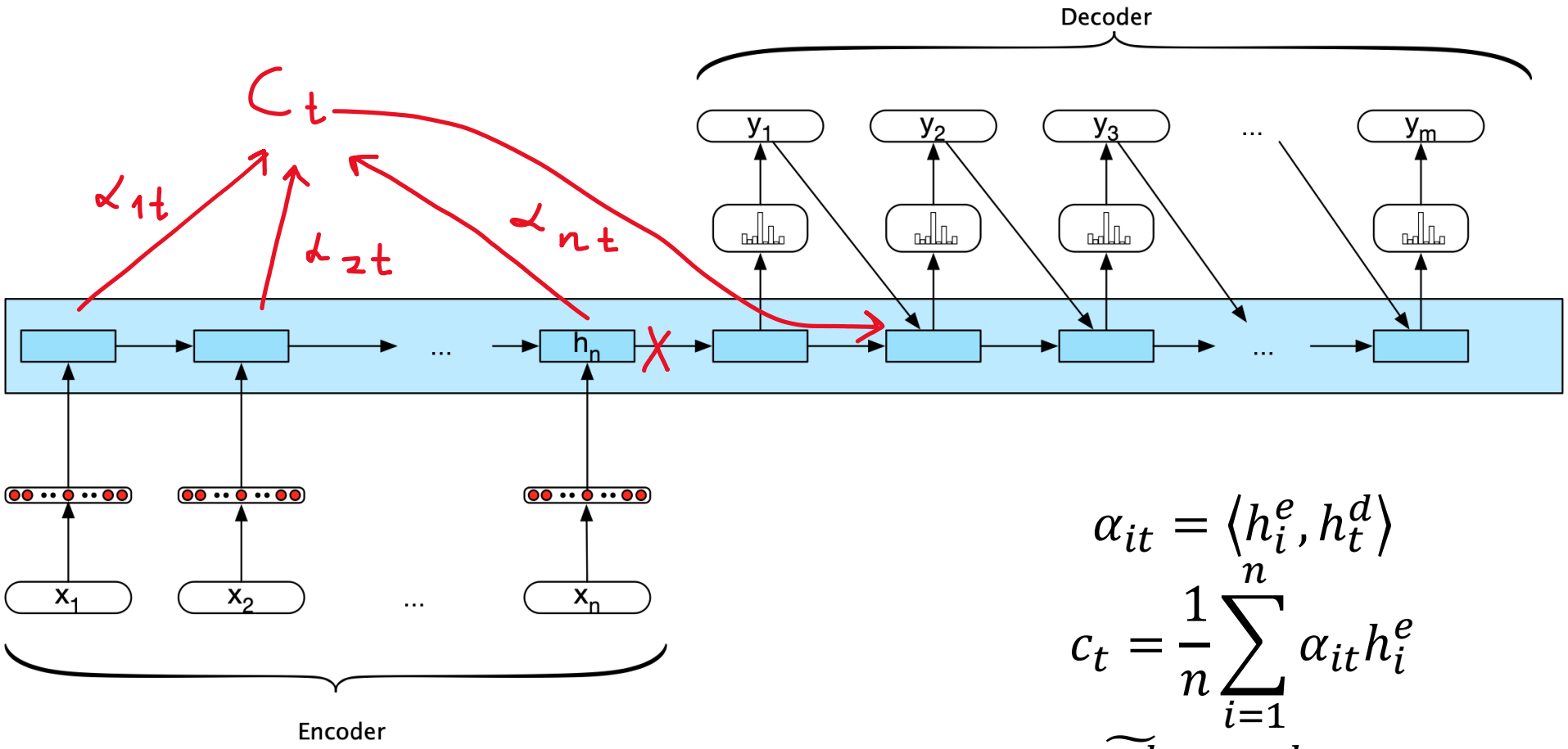
$$c_t = \frac{1}{n}\sum_{i=1}^{n} \alpha_{it} h_i^e$$

$$\widetilde{h_t^d} = [h_t^d, c_t]$$

Механизм внимания: версия 2

- Надо зафиксировать раз и навсегда, как i-е входное слово влияет на t-е выходное слово
- Эти влияния очень зависят от конкретных текстов
- Надо вычислять α_{it} в зависимости от данных

Механизм внимания: версия 3



$$\alpha_{it} = \langle h_i^e, h_t^d \rangle$$

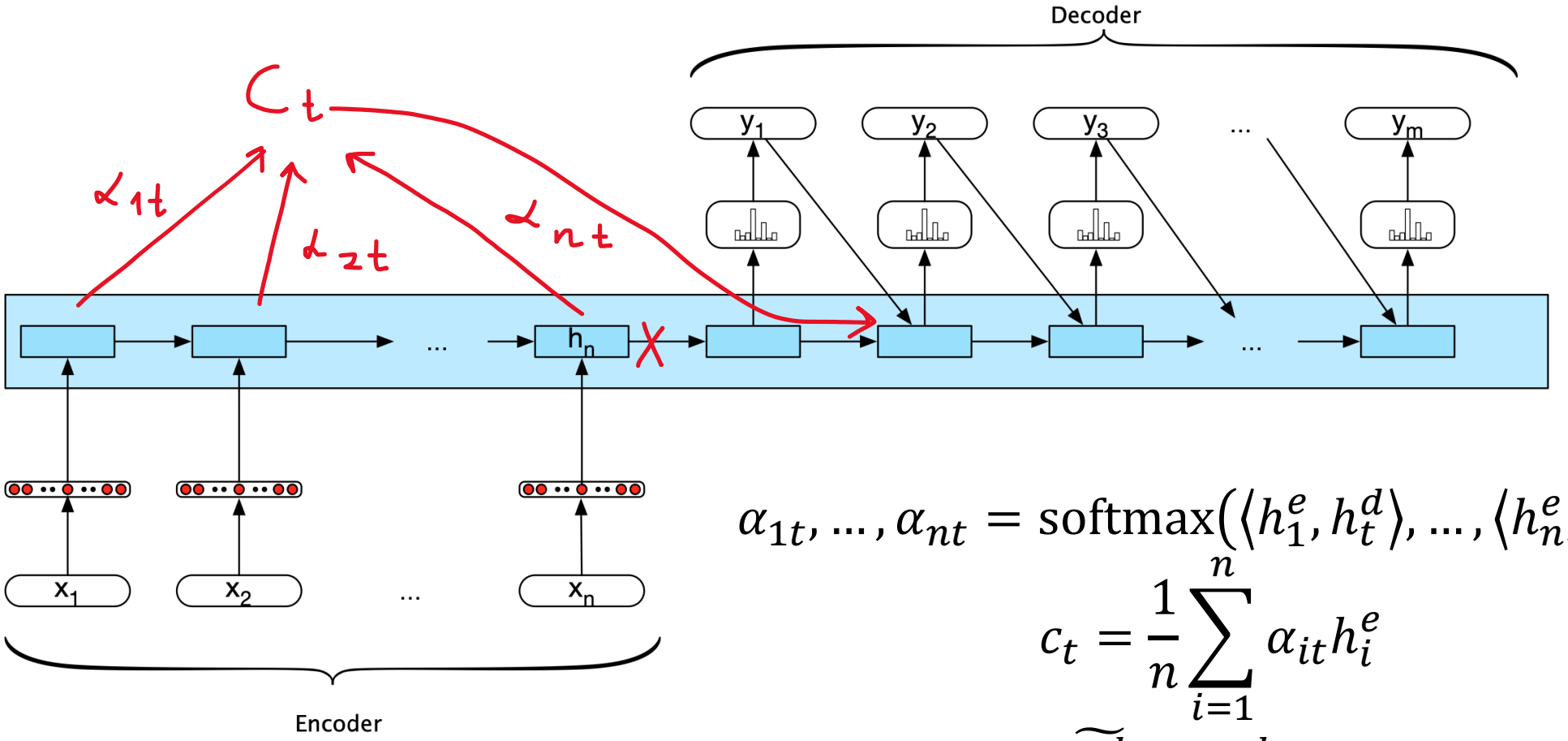
$$c_t = \frac{1}{n} \sum_{i=1}^{n} \alpha_{it} h_i^e$$

$$\widetilde{h_t^d} = [h_t^d, c_t]$$

Механизм внимания: версия 3

- Есть проблема с тем, что веса α_{it} могут принимать совершенно любые значения
- Почему это проблема?
  - Масштаб c_t может зависеть от длины входной последовательности
  - Скорее всего, все веса будут сильно ненулевыми, то есть все входные слова будут влиять на все выходные слова

Механизм внимания: версия 4



$$\alpha_{1t}, \dots, \alpha_{nt} = \text{softmax}\left(\left\langle h_1^e, h_t^d \right\rangle, \dots, \left\langle h_n^e, h_t^d \right\rangle\right)$$

$$c_t = \frac{1}{n}\sum_{i=1}^{n} \alpha_{it} h_i^e$$

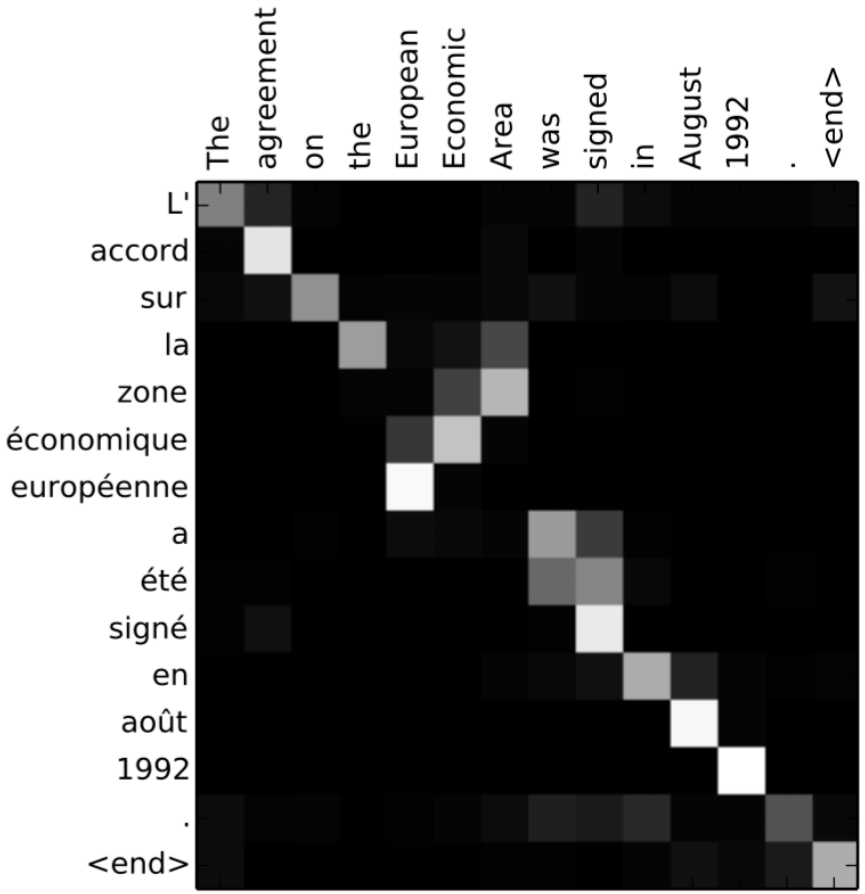
$$\widetilde{h_t^d} = [h_t^d, c_t]$$

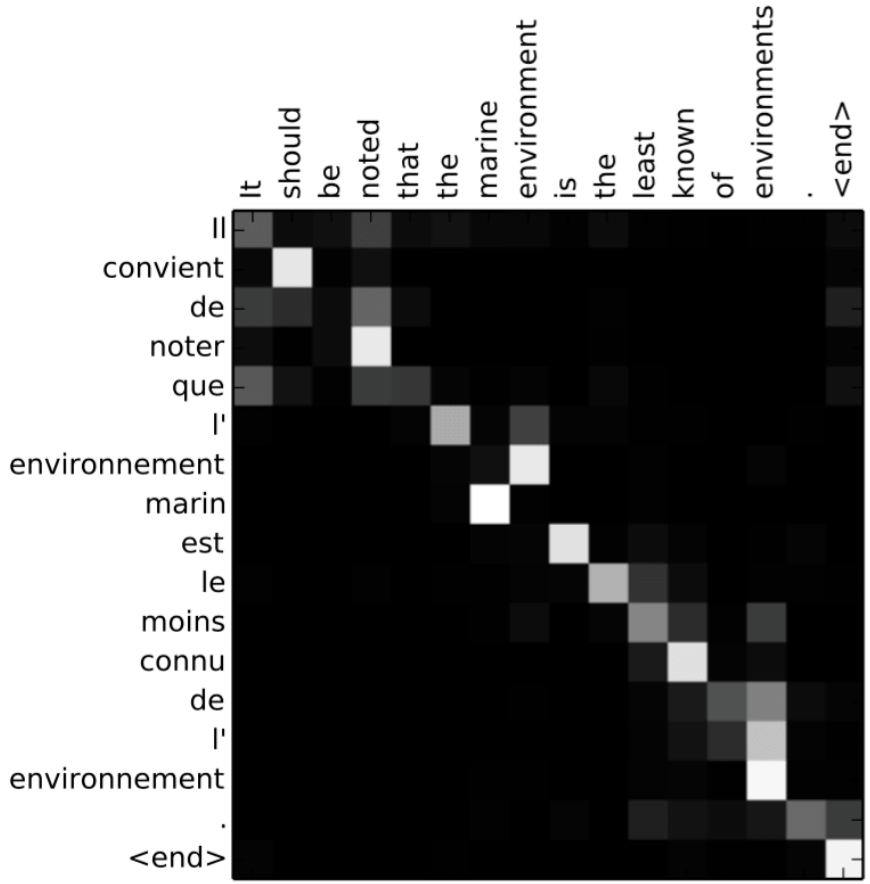
softmax

$$(a_1, \dots, a_n) \to \left(\frac{\exp(a_1)}{\sum_{i=1}^{n} \exp(a_i)}, \dots, \frac{\exp(a_n)}{\sum_{i=1}^{n} \exp(a_i)} \right)$$

- [-5, 1, 10] -> [0, 0, 1]
- [1, 1, 1] -> [0.33, 0.33, 0.33]
- [1, 2, 0] -> [0.24, 0.67, 0.09]

Механизм внимания

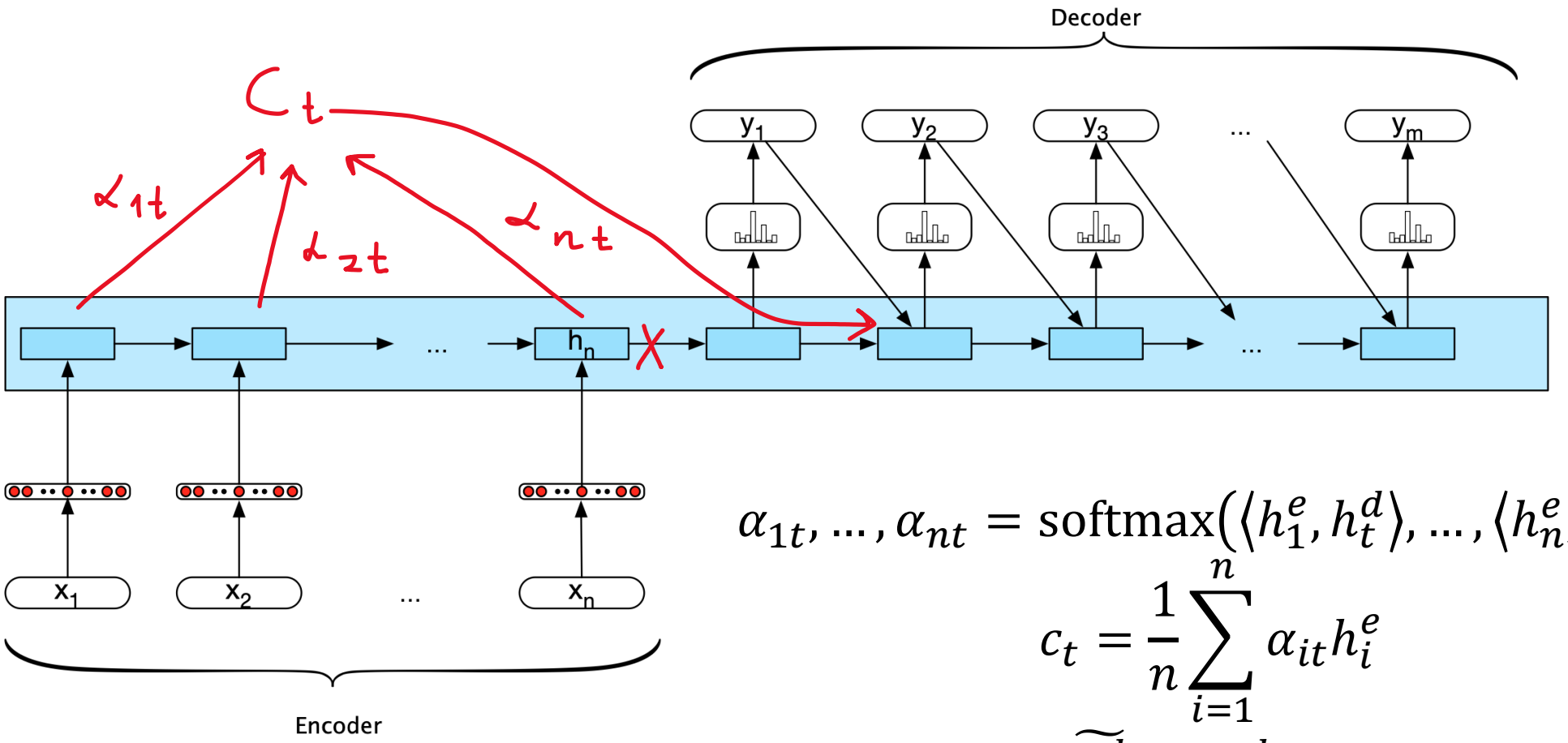




https://arxiv.org/pdf/1409.0473.pdf

Трансформер: основы self-attention

Механизм внимания



$$\alpha_{1t}, \dots, \alpha_{nt} = \text{softmax}\left(\langle h_1^e, h_t^d \rangle, \dots, \langle h_n^e, h_t^d \rangle\right)$$

$$c_t = \frac{1}{n} \sum_{i=1}^{n} \alpha_{it} h_i^e$$

$$\widetilde{h_t^d} = [h_t^d, c_t]$$

Как усилить архитектуру?

- Мы почему-то читаем входной текст слева направа
- Есть варианты с двунаправленными кодировщиками, но всё ещё мы пытаемся имитировать поведение людей
- Долой эти аналогии!

Кодировщик

- Начнём с качественного прочтения входного текста
- Попробуем обогатить каждое входное слово информацией обо всём тексте
- Назовём это «вниманием на себя» (self-attention)

Кодировщик: версия 1

- Мы подмешиваем к слову t информацию из слова i на основе сходства этих слов
- Наверное, мы хотим смешивать информацию более хитро — например, смотреть на слова той же части речи или той же части предложения

Кодировщик: версия 2

- Будем для каждого слова x_i обучать три вектора:
  - Запрос (query) $q_i = W_Q x_i$
  - Ключ (key) $k_i = W_K x_i$
  - Значение (value) $v_i = W_V x_i$
- «Важность» слова x_i для слова x_j: $\langle q_j, k_i \rangle$

Кодировщик: версия 2

- Вклад слова x_i в новое представление слова x_j:

$$\alpha_{ij} = \frac{\exp\left(\frac{\langle q_j, k_i \rangle}{\sqrt{d}}\right)}{\sum_{p=1}^{n} \exp\left(\frac{\langle q_j, k_p \rangle}{\sqrt{d}}\right)}$$

- d — размерность векторов q_j и k_i
- n — число слов во входной последовательности

Кодировщик: версия 2

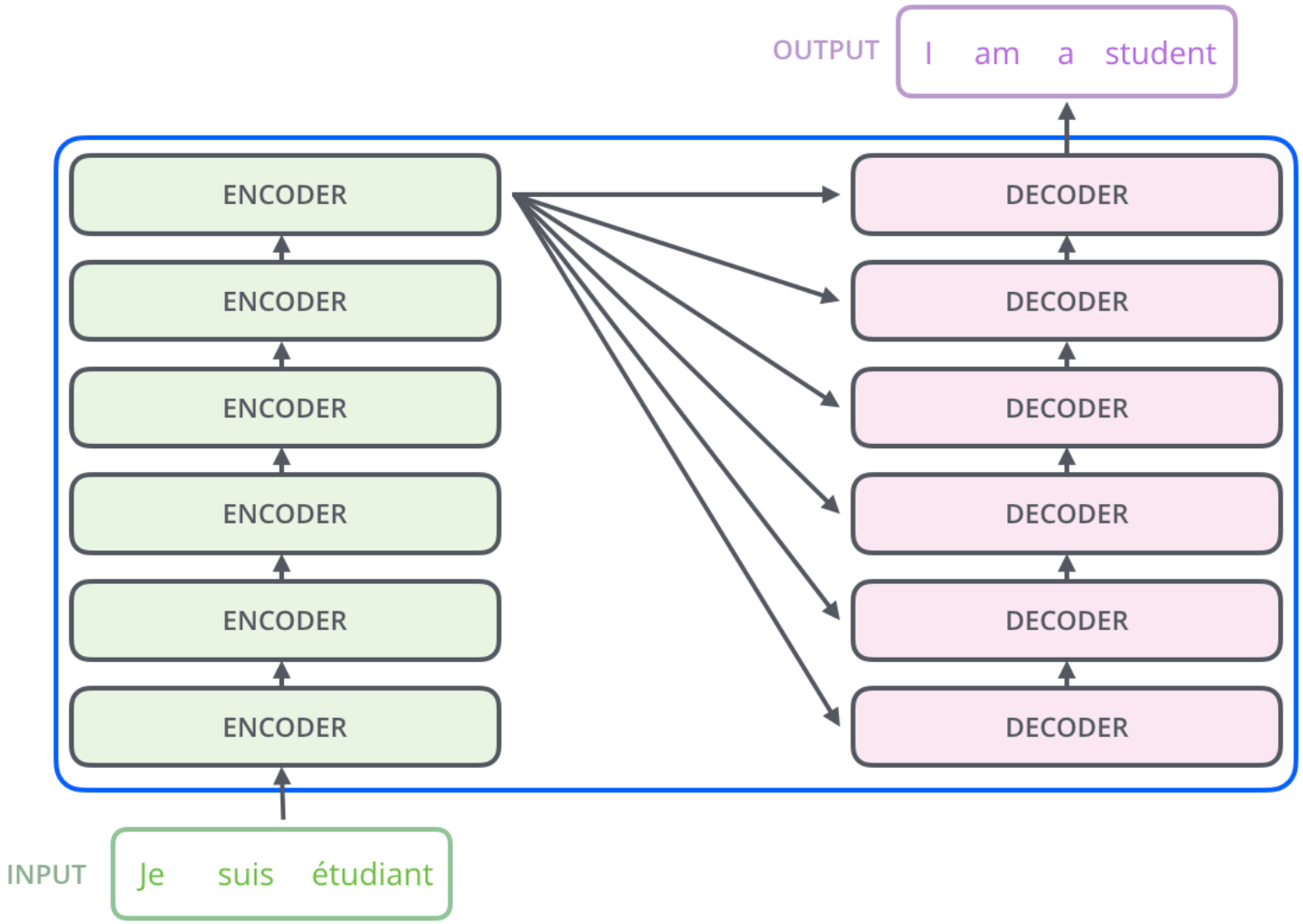
Новое представление слова x_j:

$$\widetilde{x_j} = \sum_{i=1}^{n} \alpha_{ij} v_i$$

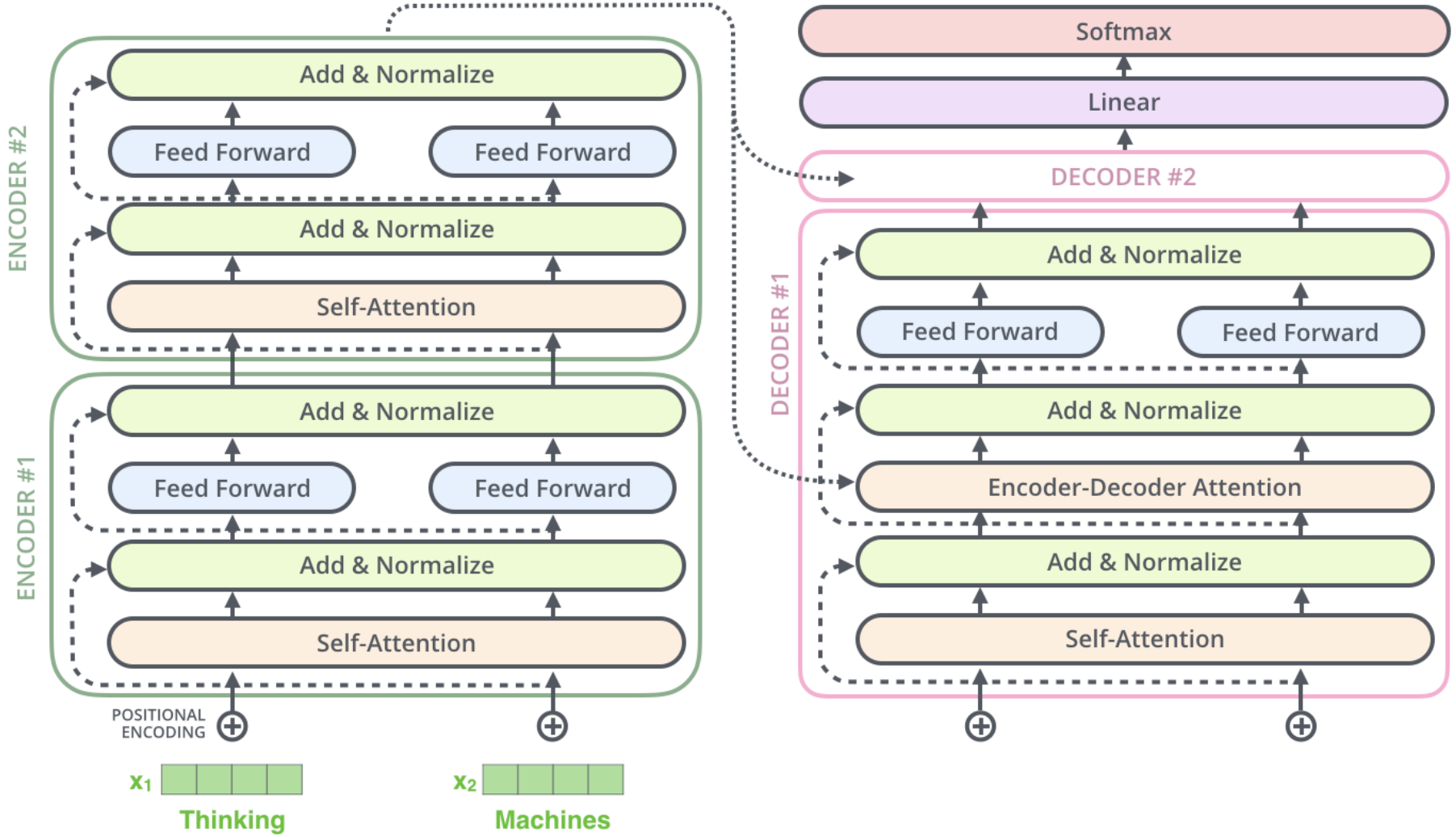
Кодировщик: версия 2

- Кодировщик задаётся тремя мини-моделями, вычисляющими по вектору слова векторы запроса, ключа и значения
- Каждая мини-модель — линейный слой

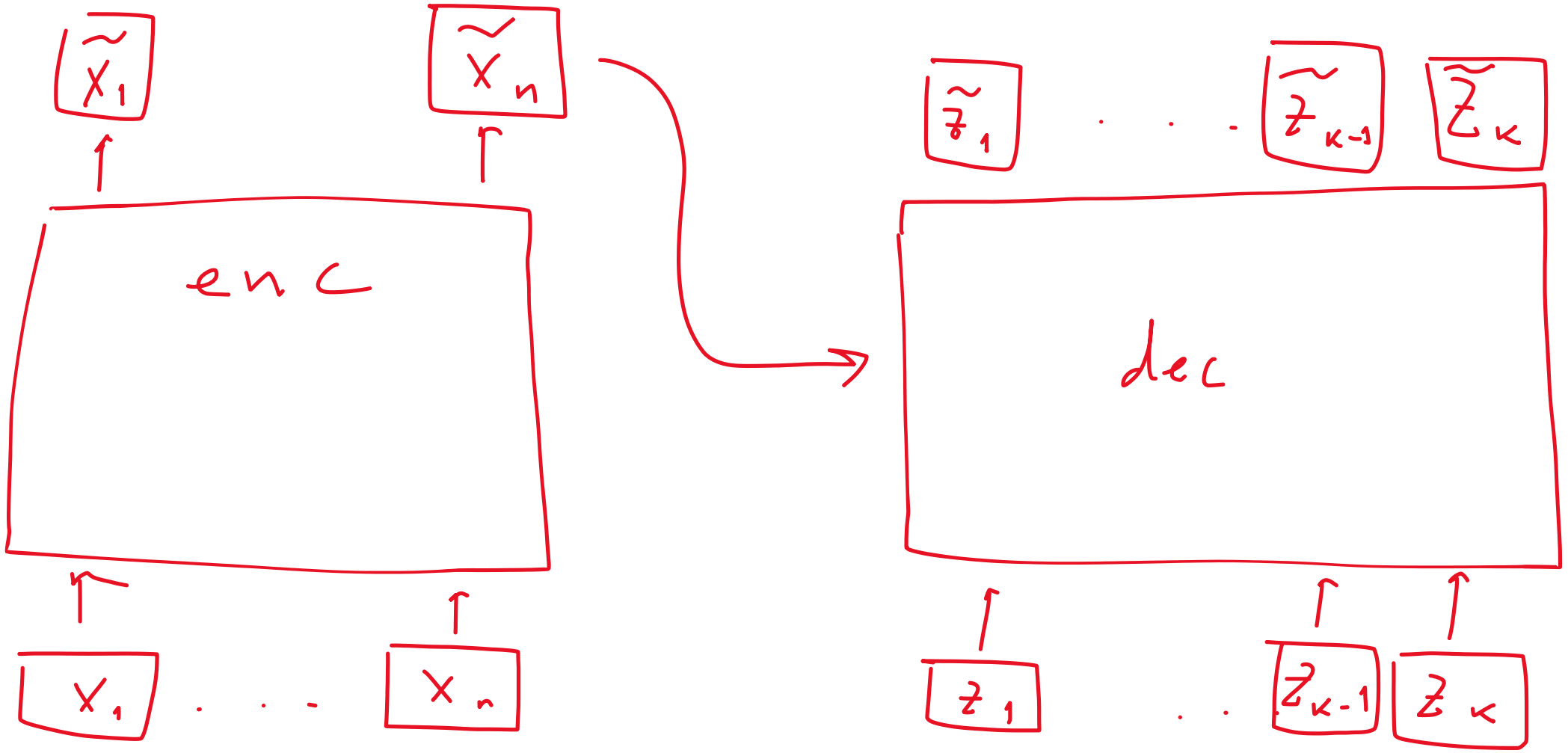
Трансформер



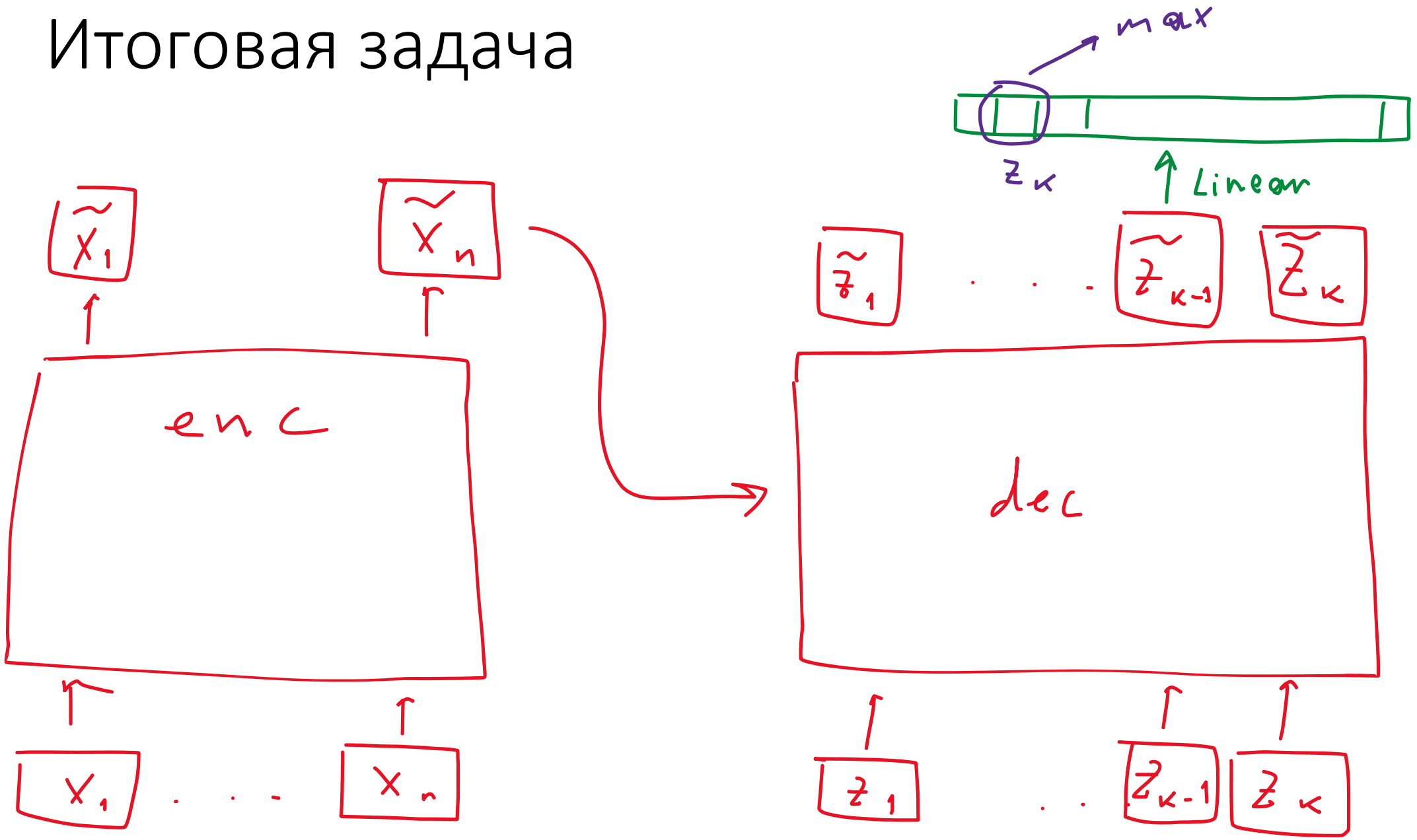
Деодировщик в трансформере



Итоговая задача

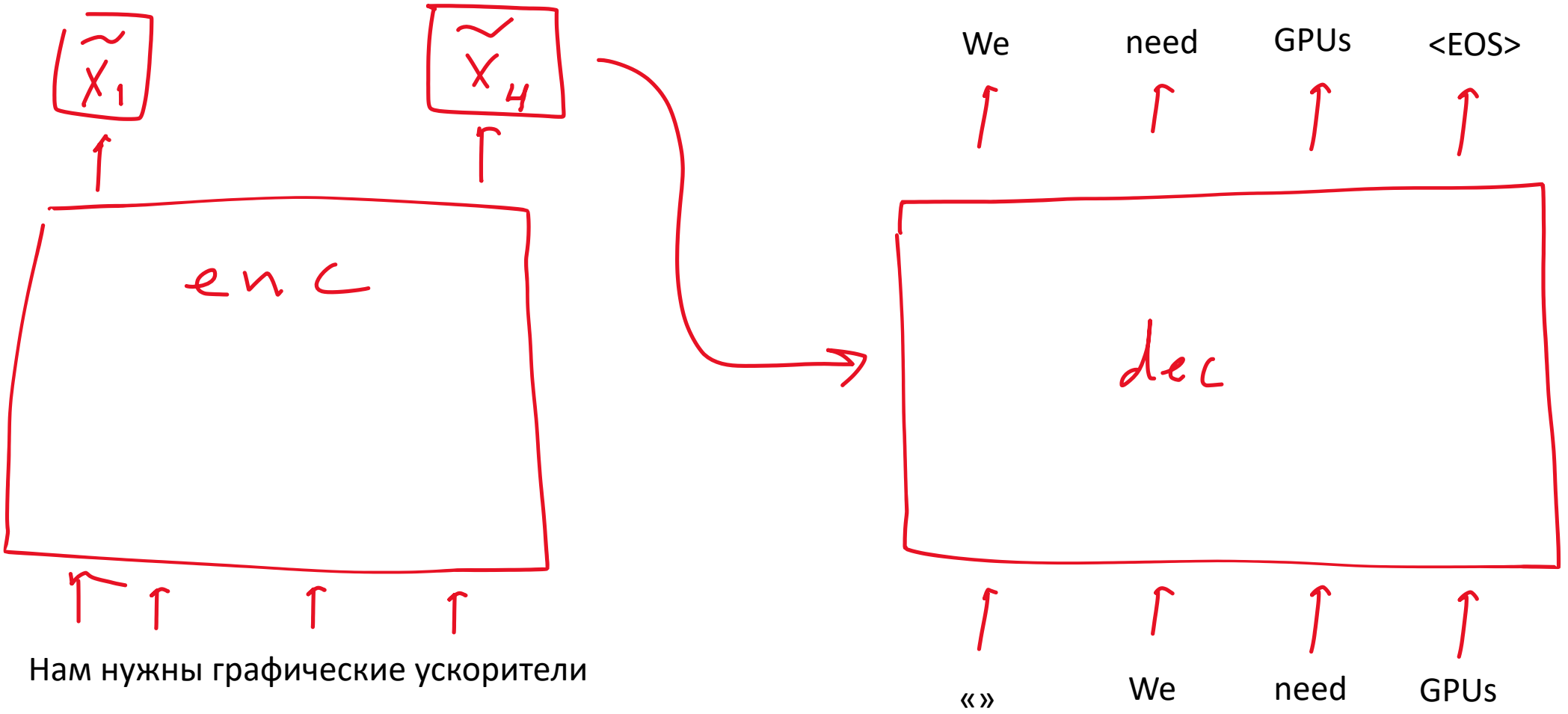


Итоговая задача



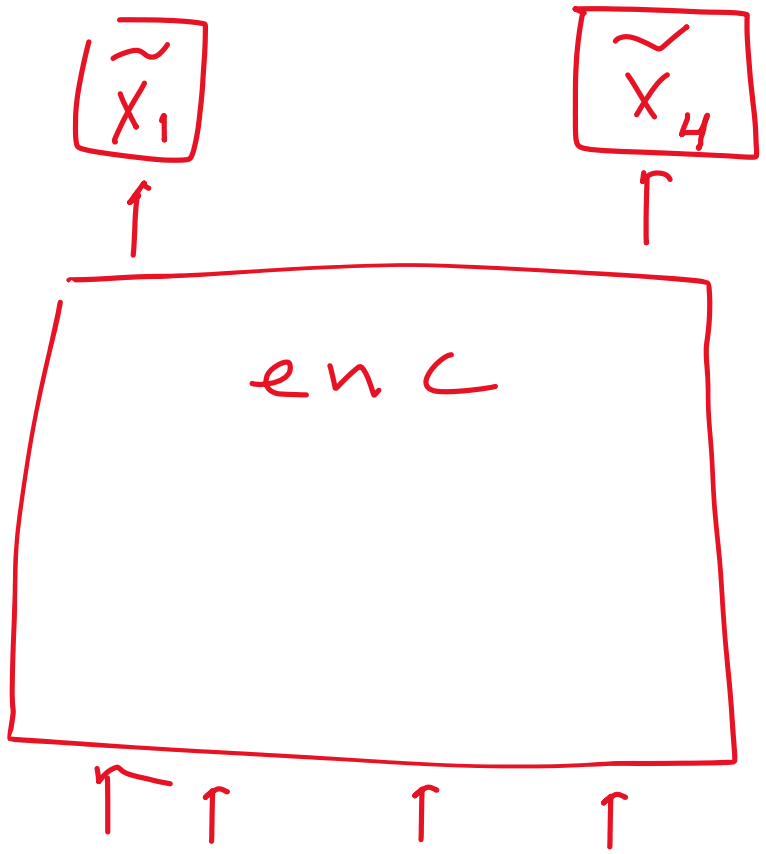
Трансформер: режимы работы

Режим 1: seq2seq



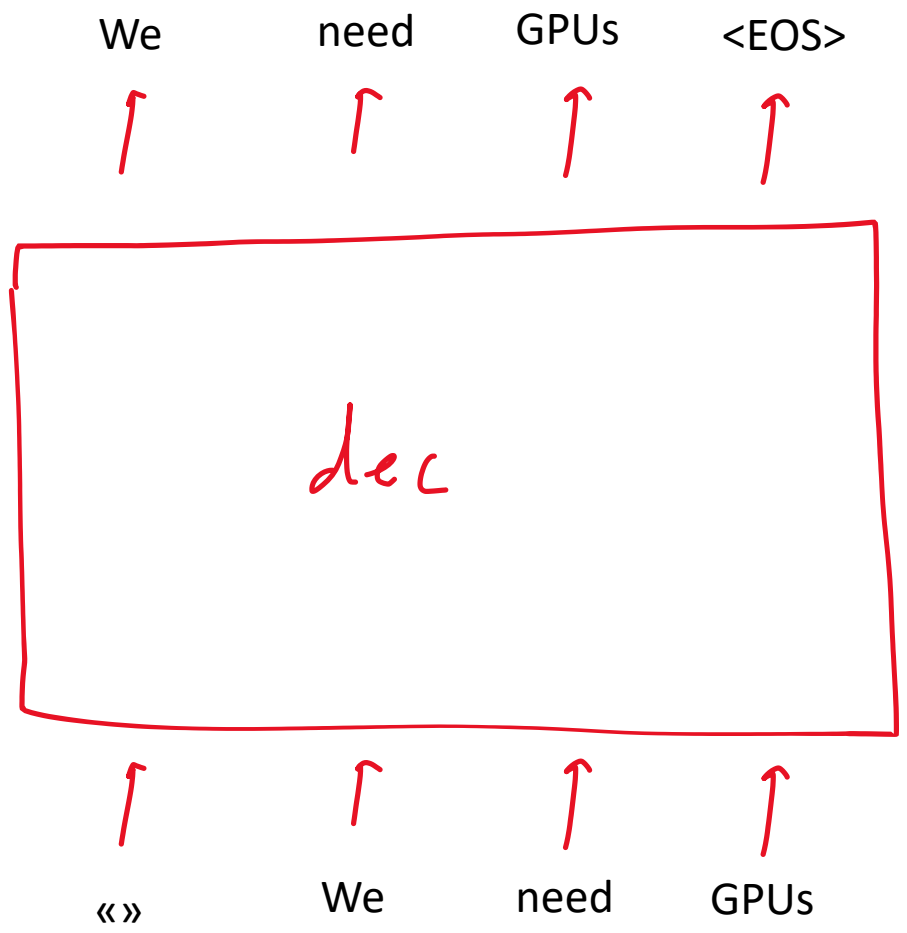
Нам нужны графические ускорители

Режим 2: только кодировщик



Нам нужны графические ускорители

Режим 3: только декодировщик



Обучение трансформеров с нуля

- Только кодировщик:
  - Классификация текстов
  - Сравнение текстов
  - Классификация слов в тексте
  - Отбор ответов на вопрос
  - …
- Только декодировщик:
  - Генерация текста
  - Генерация ответов на задания

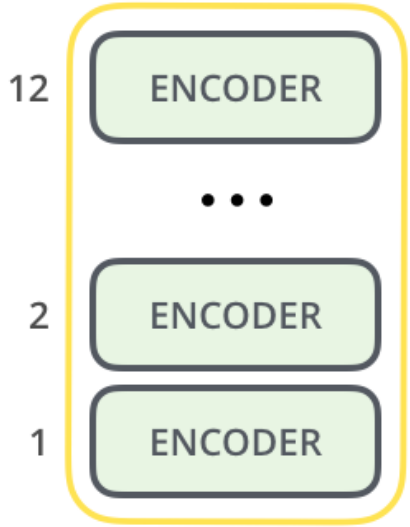
Обучение трансформеров с нуля

- Требует огромного объёма данных
- Требует колоссальных вычислительных мощностей

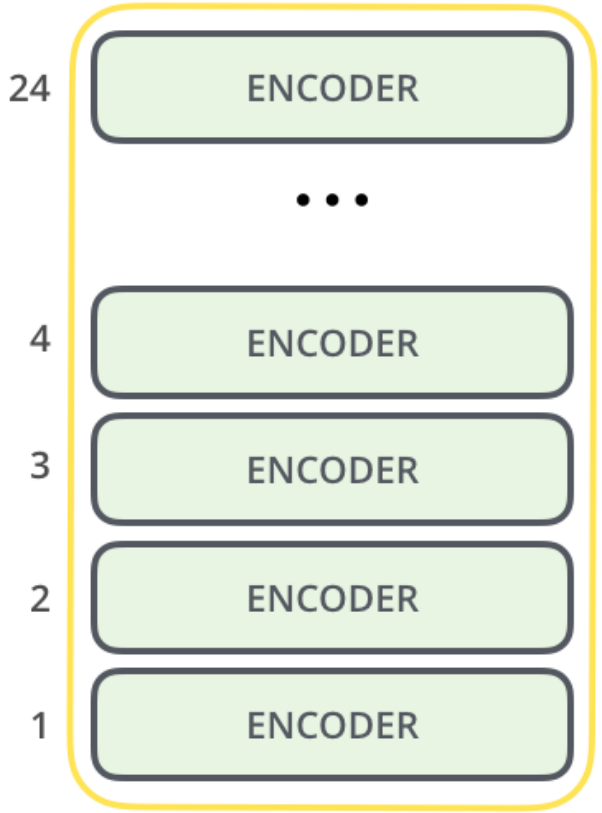
- Можно ли заранее обучить большую модель так, чтобы она легко донастраивалась на новые задачи?

BERT: предобучение

BERT: архитектура

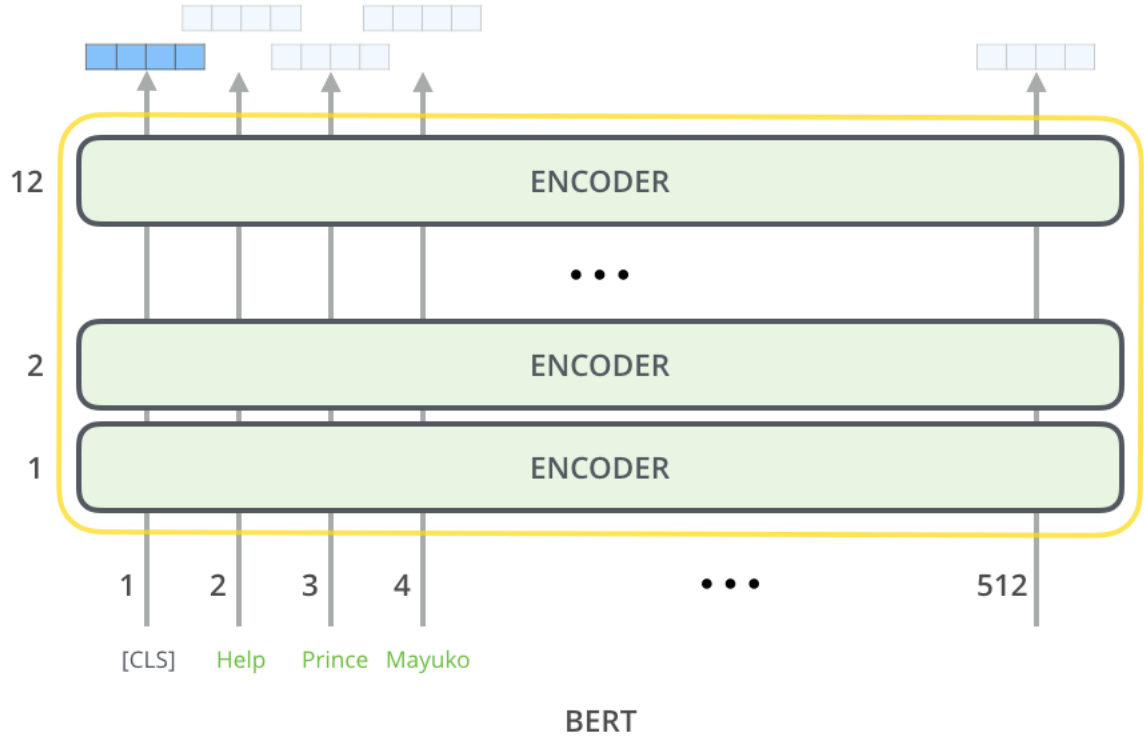


BERT$_{BASE}$

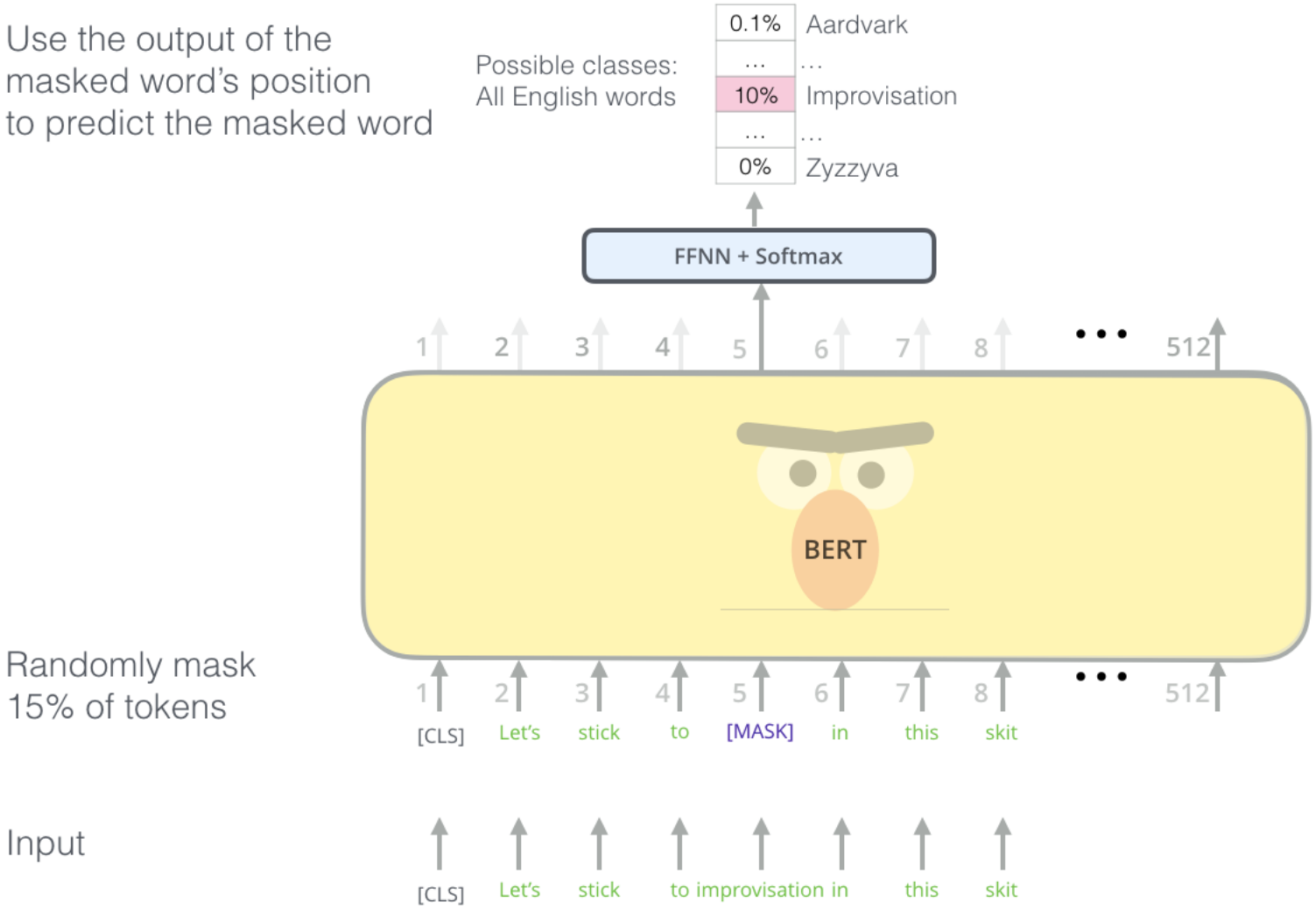


BERT$_{LARGE}$

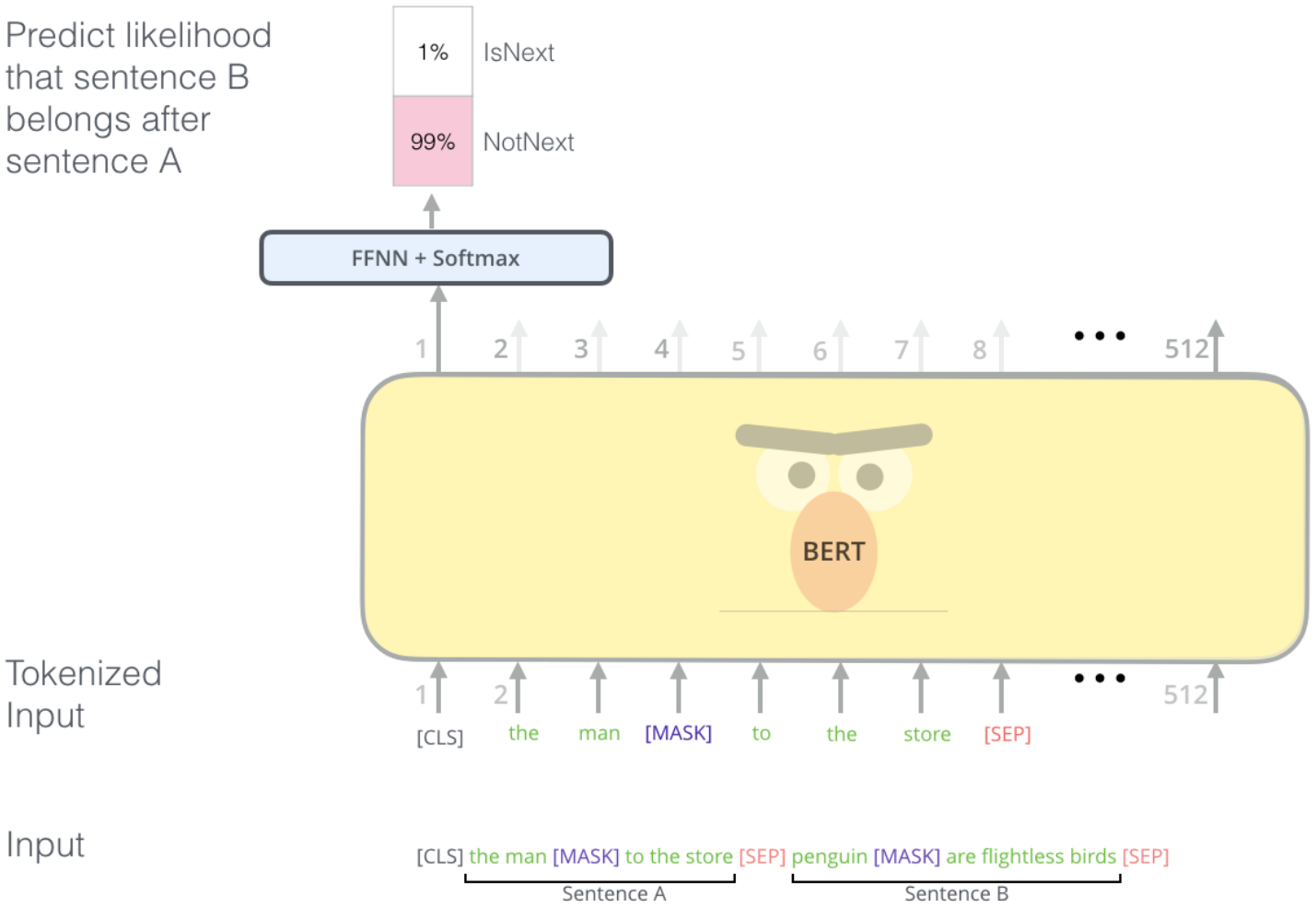
BERT: архитектура



BERT: предобучение

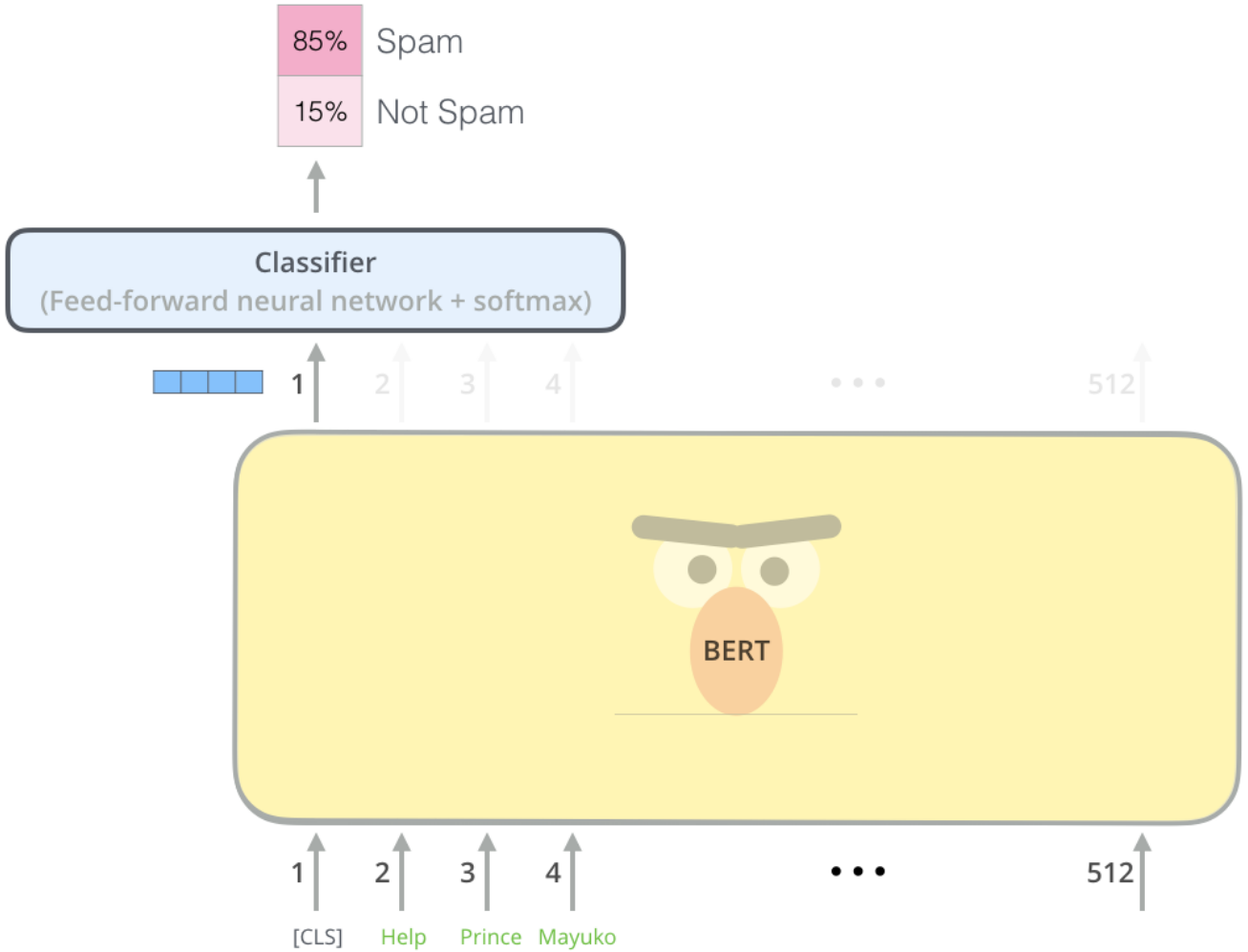


BERT: предобучение

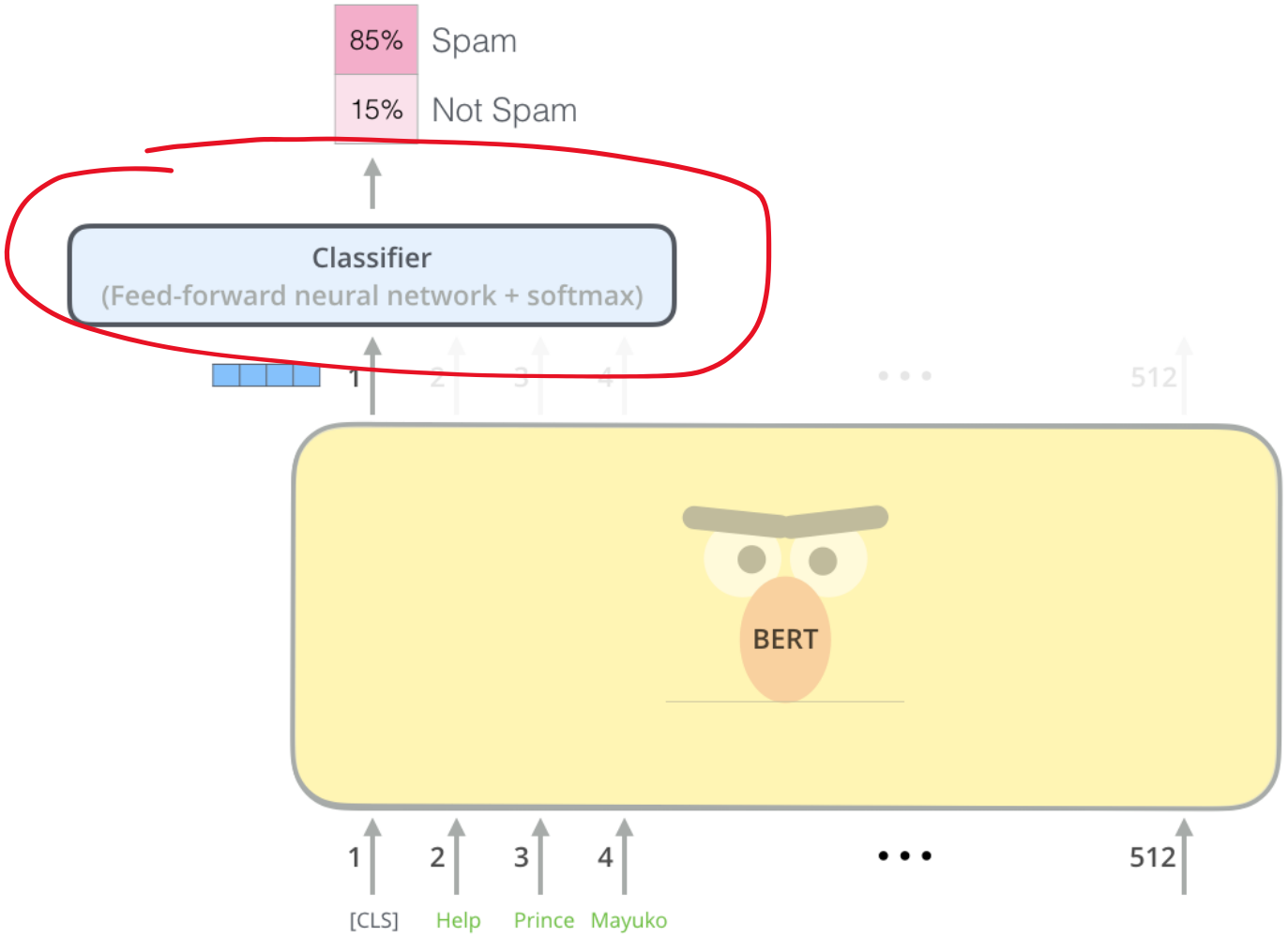


BERT: применение

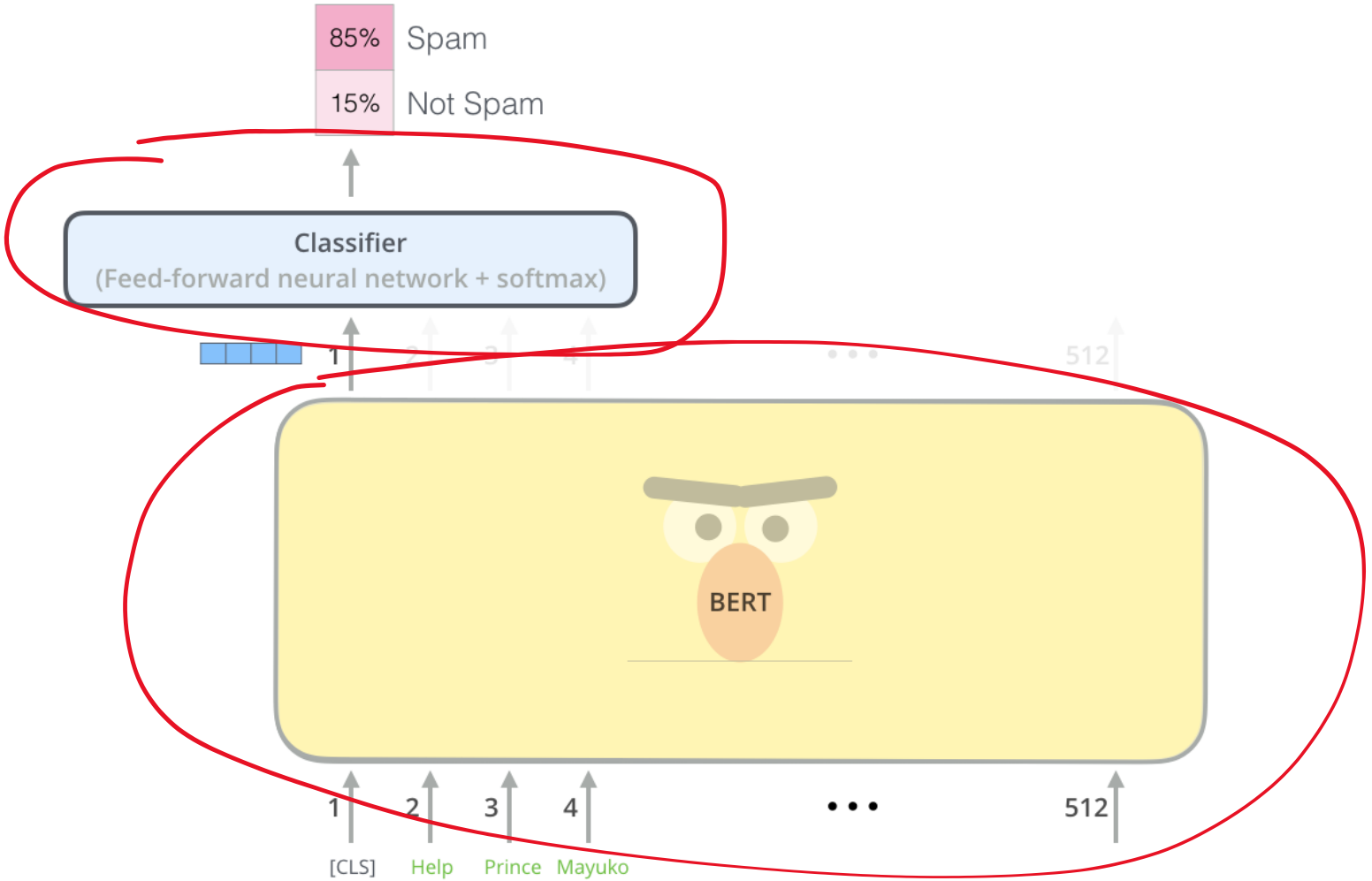
BERT: архитектура



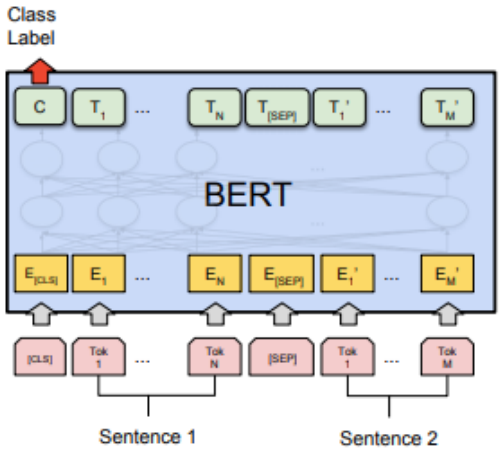
BERT: дообучение



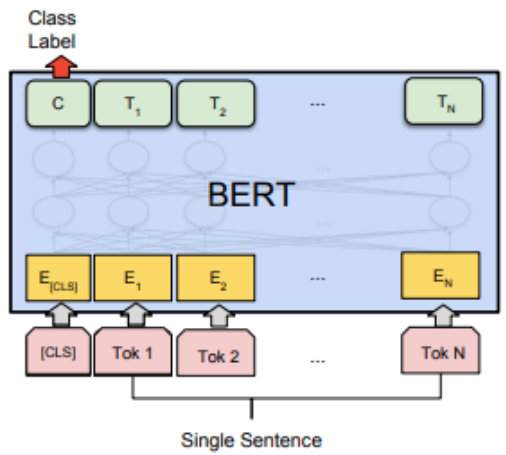
BERT: дообучение



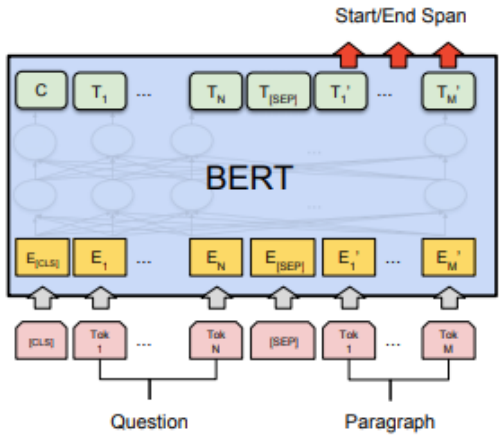
BERT: применение



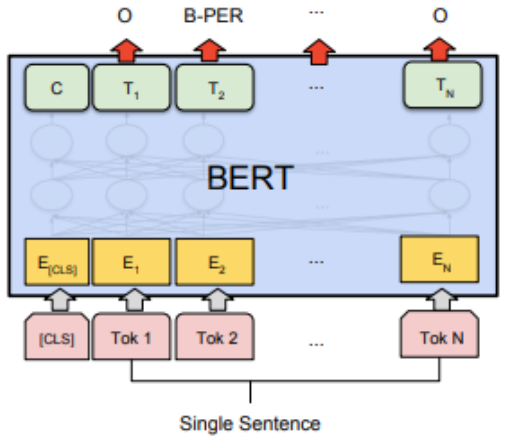
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA

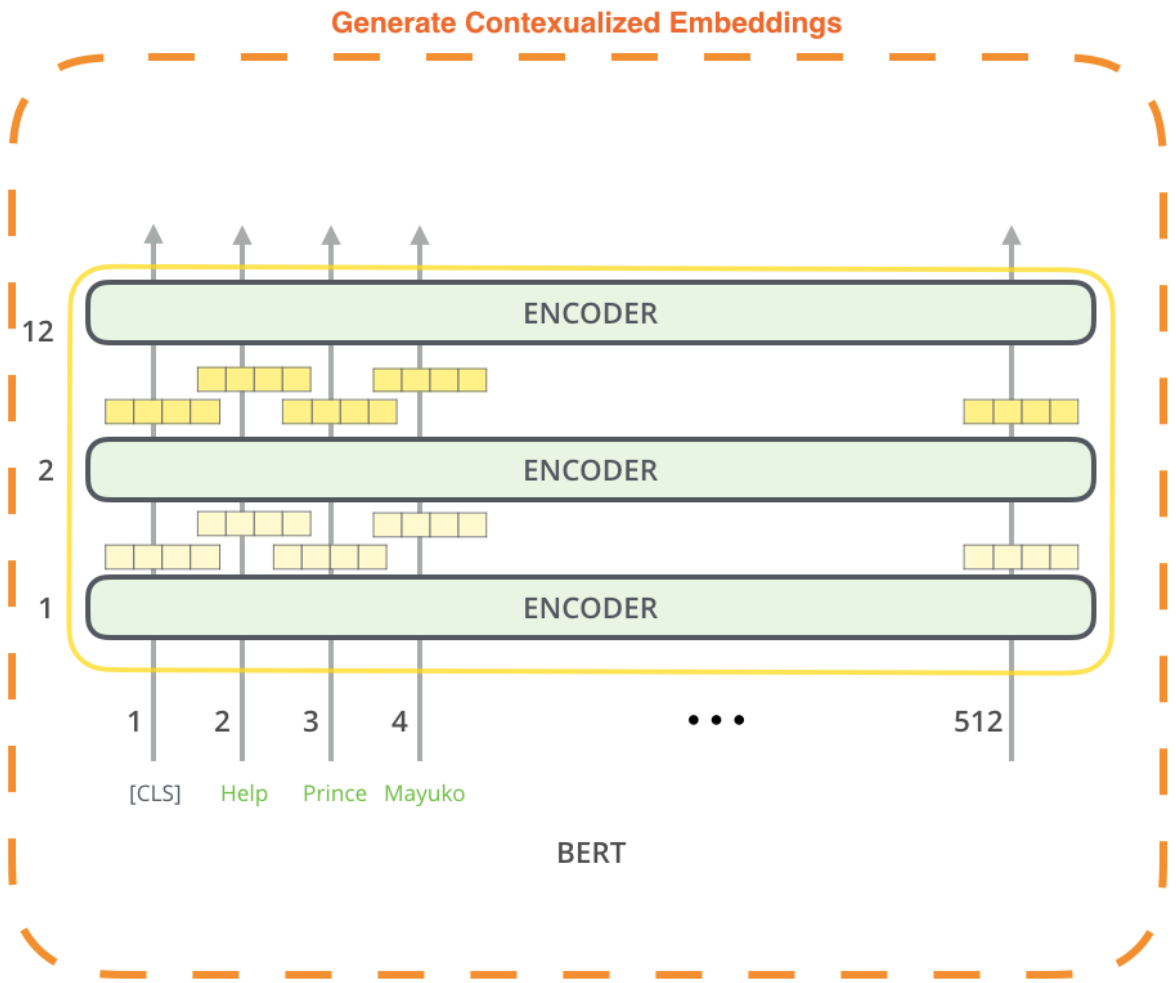


(c) Question Answering Tasks: SQuAD v1.1

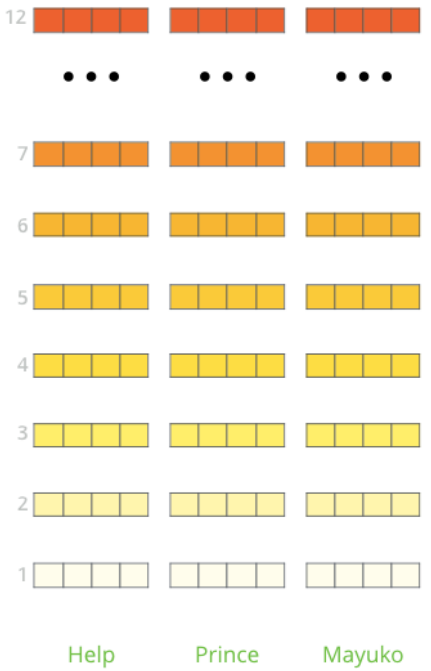


(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

BERT: применение



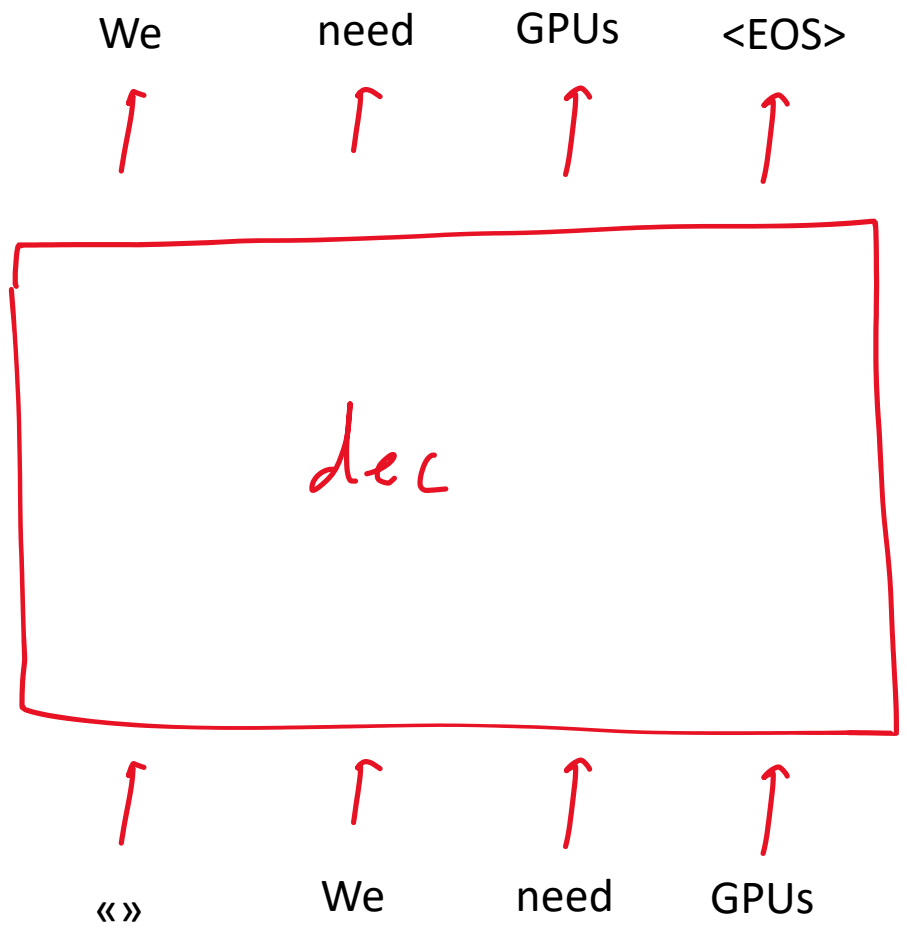
The output of each encoder layer along each token's path can be used as a feature representing that token.



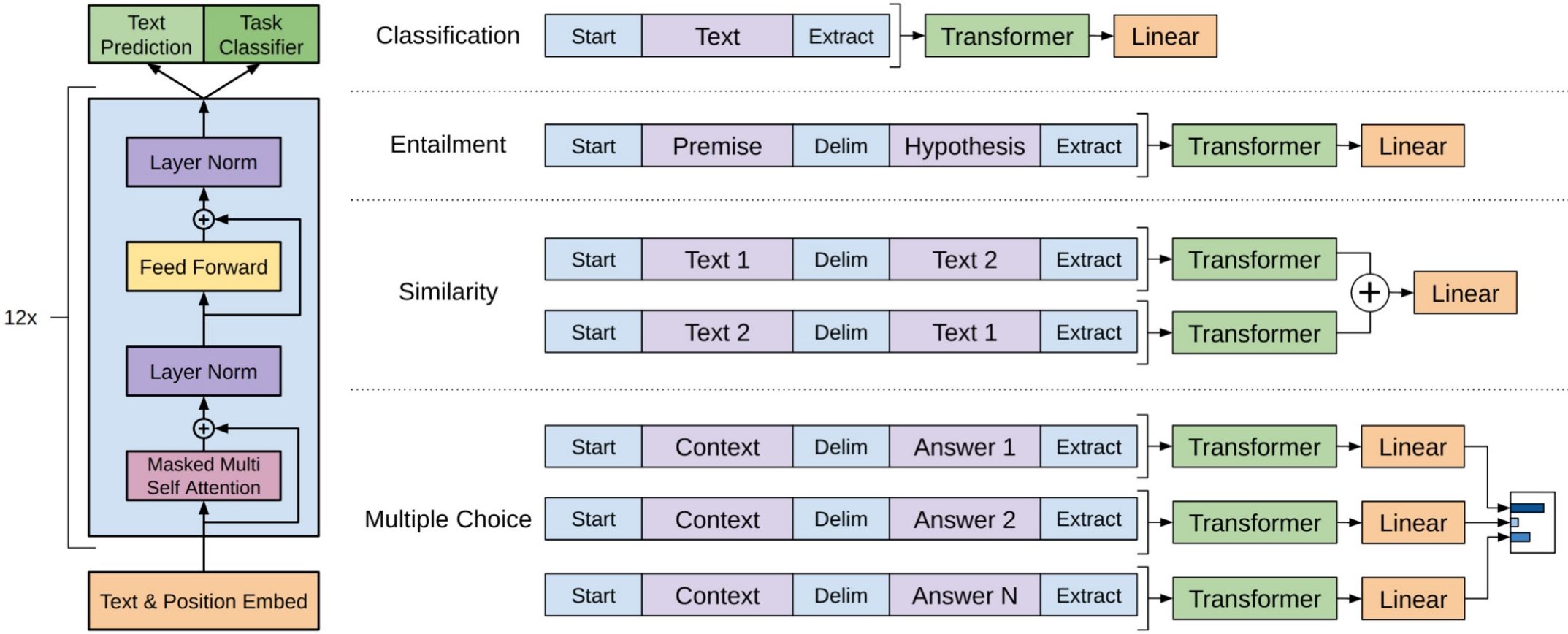
But which one should we use?

GPT: обзор

Декодировщик



GPT



117 миллионов параметров

GPT-2

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

- Больше параметров
- Модель хорошо генерирует тексты
- Модель неплохо решает ряд задач без дообучения

GPT-3

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

- Качество растёт
- Модель ещё лучше решает задачи в режиме zero-shot
- Модель показывает хорошее качество в ряде задач в few-shot режиме

GPT: обучение

Предобучение

Задача: предсказать следующее слово

Модель: трансформер

Данные: **все тексты, которые можем найти**

Supervised finetuning

Задача: предсказать следующее слово

Модель: предобученный трансформер

Данные: качественные пары «запрос-ответ», созданные ассесорами

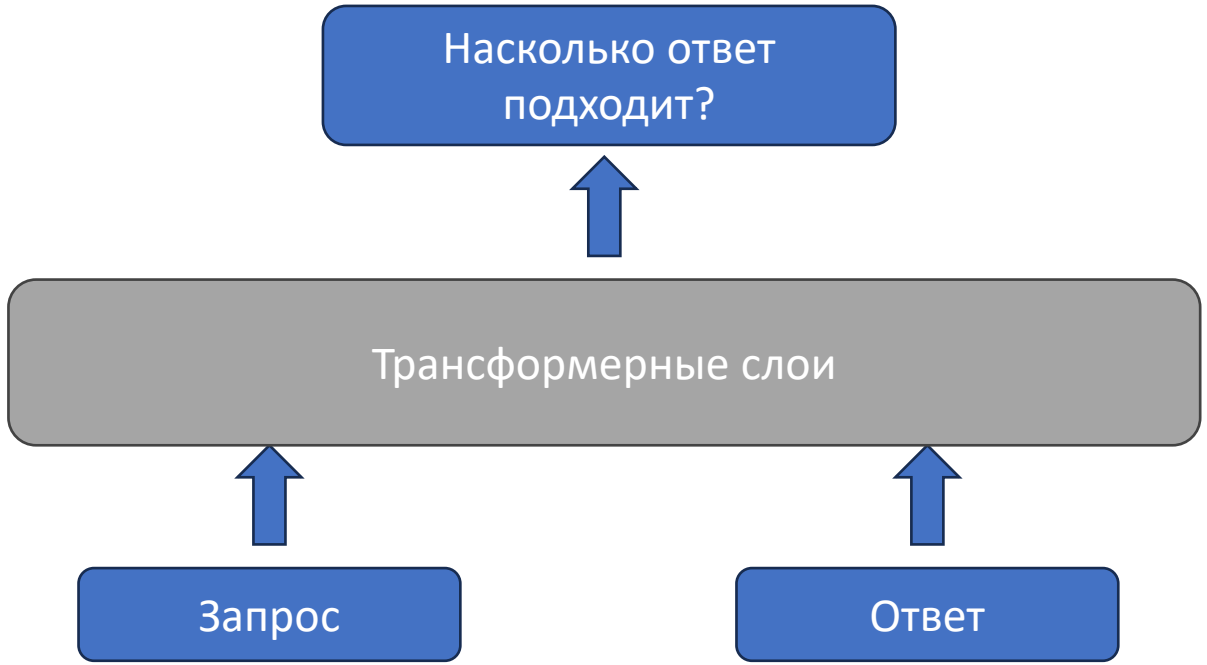
Оценивающая модель

Задача: предсказать качество ответа на запрос

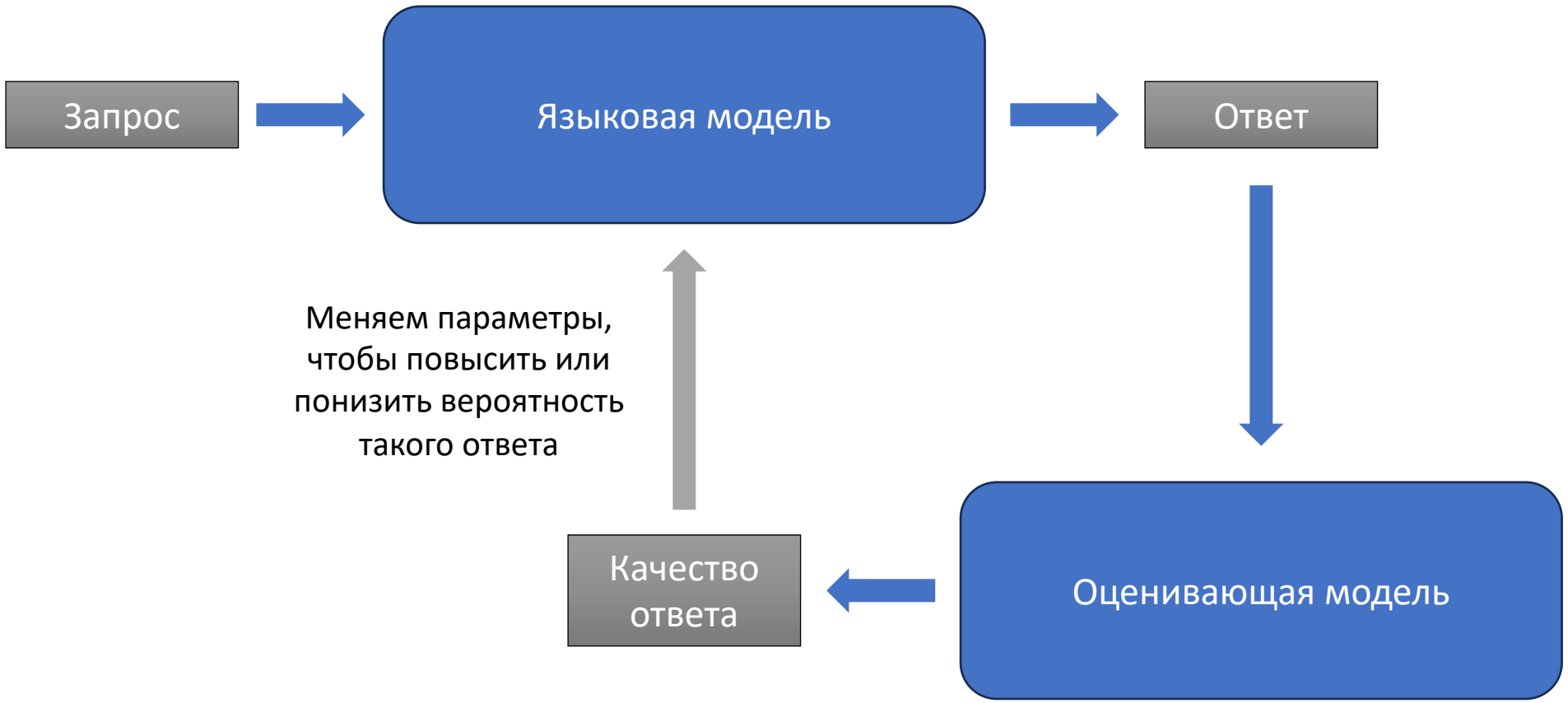
Модель: дообученный трансформер

Данные: ранжированные ассесорами ответы

Оценивающая модель



RLHF


Запрос
Языковая модель
Ответ
Меняем параметры, чтобы повысить или понизить вероятность такого ответа
Качество ответа
Оценивающая модель

Работа с GPT

- Дообучать тяжело
- Первое направление улучшений — качественное составление запросов
- Второе направление — доступ модели ко внешним данным