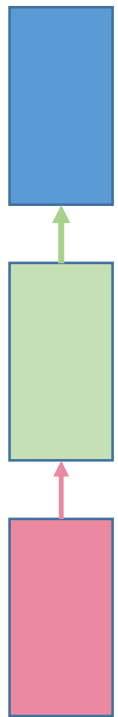


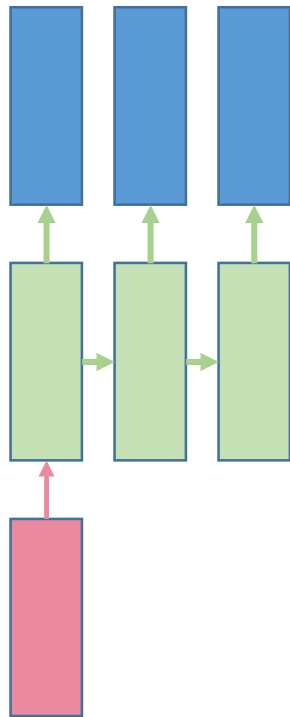
RNN. CTC . Attention.
Seq2Seq.

Рекуррентная нейросеть

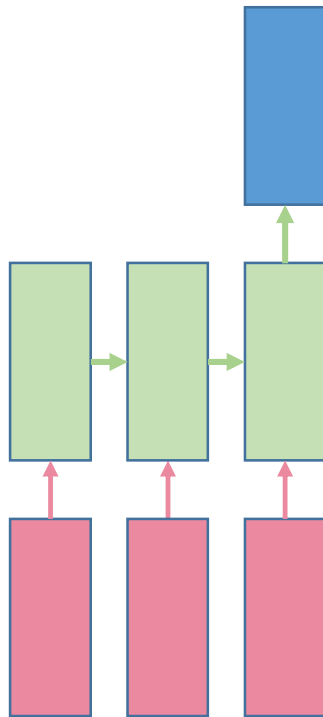
Один к одному



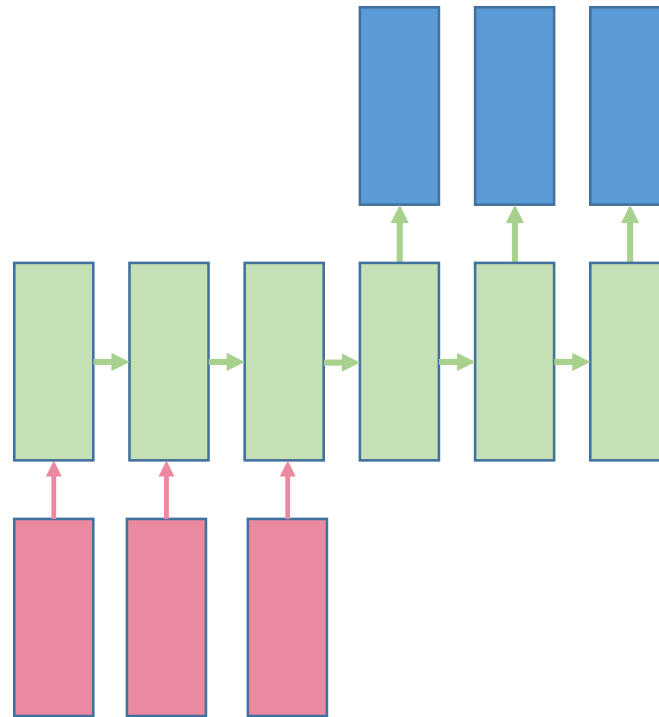
Один ко многим



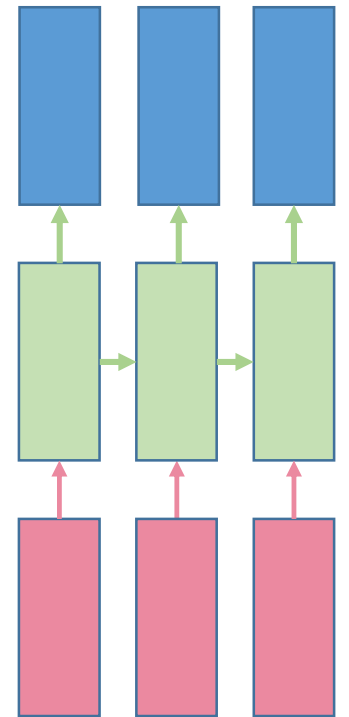
Многие к одному



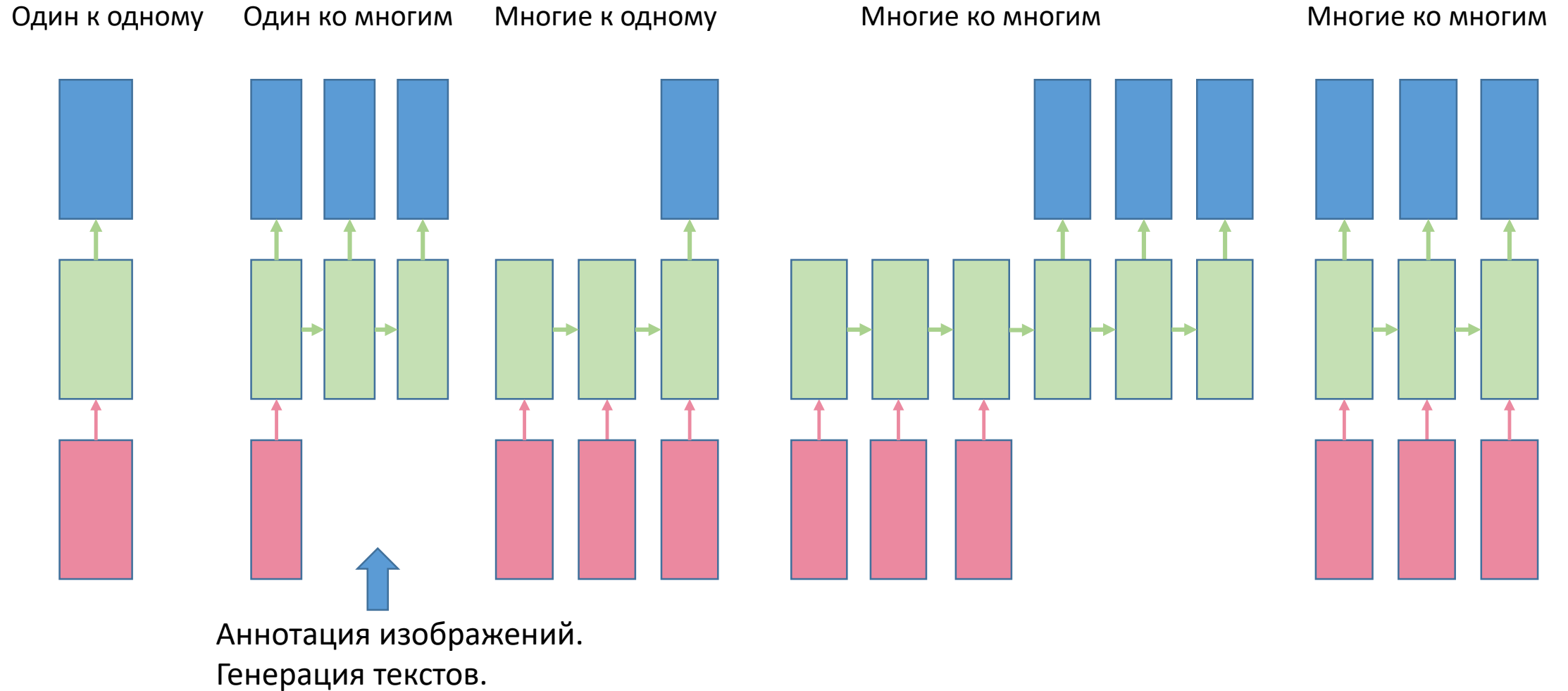
Многие ко многим



Многие ко многим

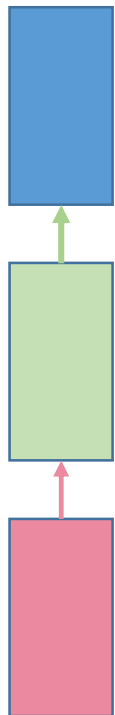


Рекуррентная нейросеть

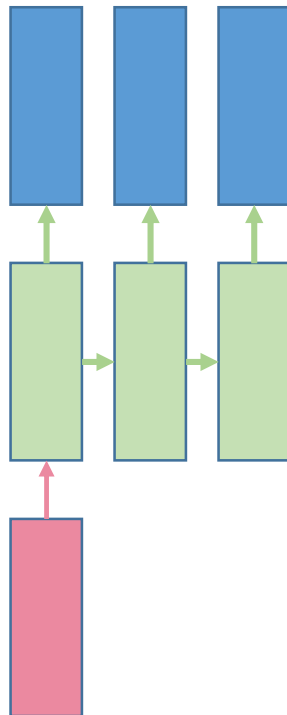


Рекуррентная нейросеть

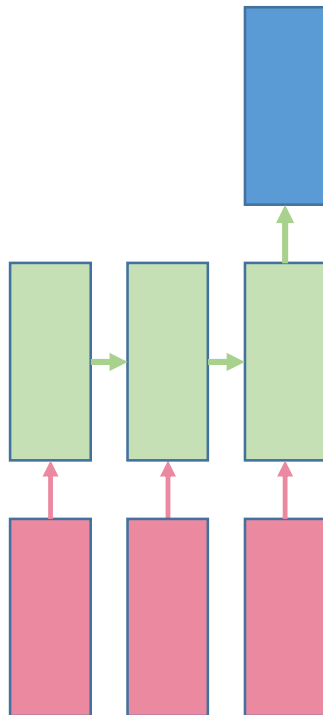
Один к одному



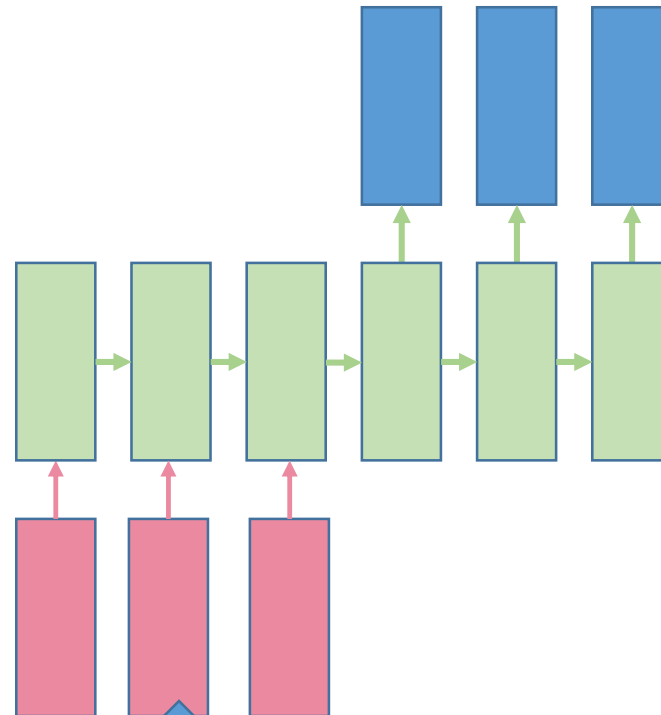
Один ко многим



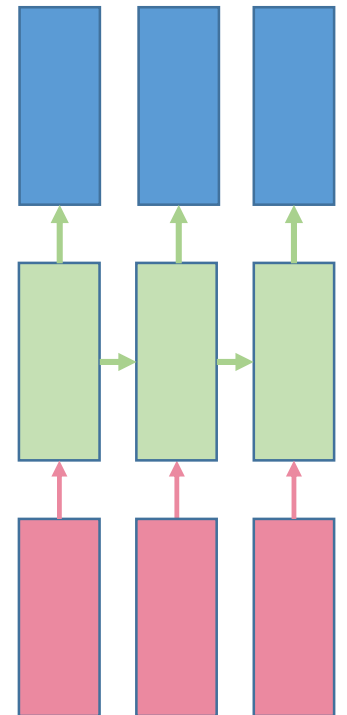
Многие к одному



Многие ко многим



Многие ко многим

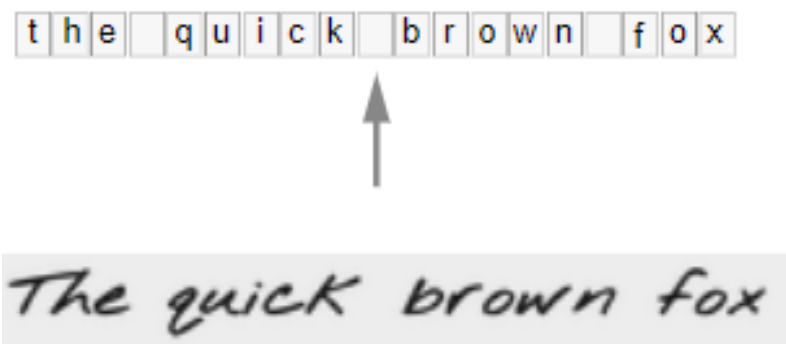


Переводчик, чат боты

Вход – текст. Выход – текст

Системы распознавания речи

Задача

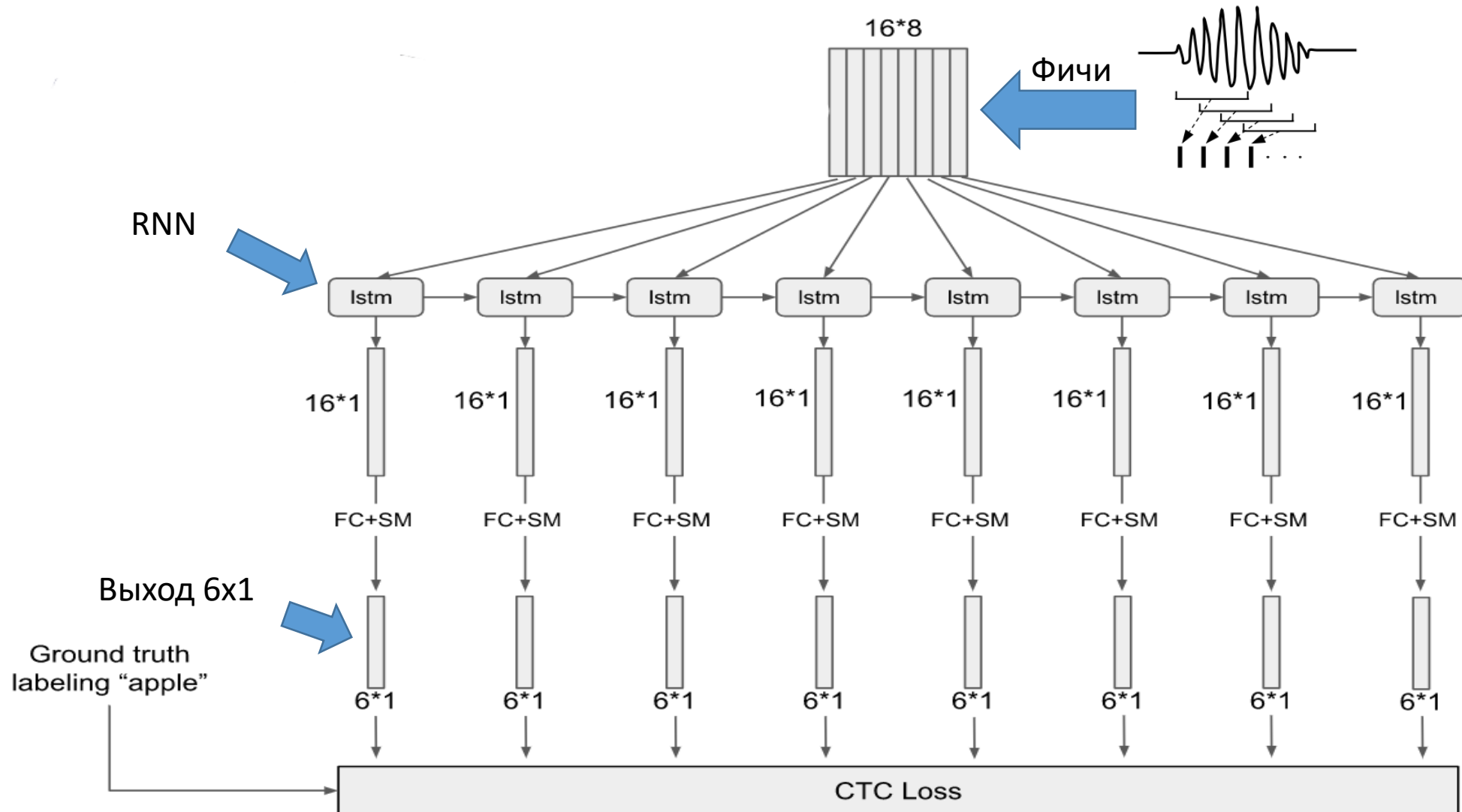


Распознавание рукописного ввода
Вход — координаты пера



Распознавание речи
Вход — спектрограммы сигнала

CTC loss



CTC loss

- CTC loss – это “softmax” слой $p_l = \frac{\exp(x_l)}{\sum_k \exp(x_k)}$
- Количество выходов слоя на 1 больше, чем всего маркеров L
- Активация первых $|L|$ элементов слоя интерпретируется как вероятность
- Активация дополнительного юнита интерпретируется как отсутствие маркера. “blank”

CTC loss

- Для входной последовательности x длиной T
 - Задаем RNN с m входами, n выходами и w – вектор весов как непрерывное отображение $N_w: (R_m)^T \rightarrow (R_n)^T$
 - Тогда $y = N_w(x)$ – последовательность выходов RNN
- y_k^t - активация выходного элемента k в момент времени t
- y_k^t - вероятность пронаблюдать маркер k в момент времени t
 - Определяет распределение по множеству L'^T последовательностей длины T над алфавитом $L' = L \cup \{blank\}$:
 - $p(\pi|x) = \prod_{t=1}^T y_{\pi_t}^t, \forall \pi \in L'^T$.
- Элементы L'^T - это пути π

CTC loss - пути



Path1: "ap-pl-ee" $\xrightarrow{B("ap-pl-ee")}$ Labeling: "apple"

$$p("ap-pl-ee") = y_a^1 \cdot y_p^2 \cdot y_-^3 \cdot y_p^4 \cdot y_l^5 \cdot y_-^6 \cdot y_e^7 \cdot y_e^8$$

CTC loss - пути



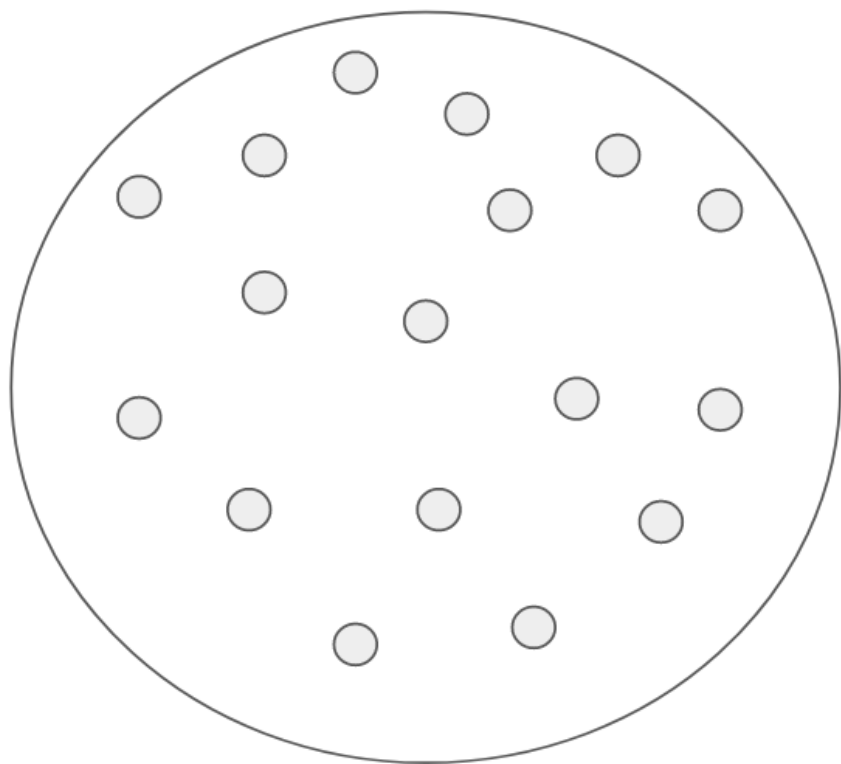
Path1: "ap-pl-ee" $\xrightarrow{B("ap-pl-ee")}$ Labeling: "apple"

$$p("ap-pl-ee") = y_a^1 \cdot y_p^2 \cdot y_-^3 \cdot y_p^4 \cdot y_l^5 \cdot y_-^6 \cdot y_e^7 \cdot y_e^8$$

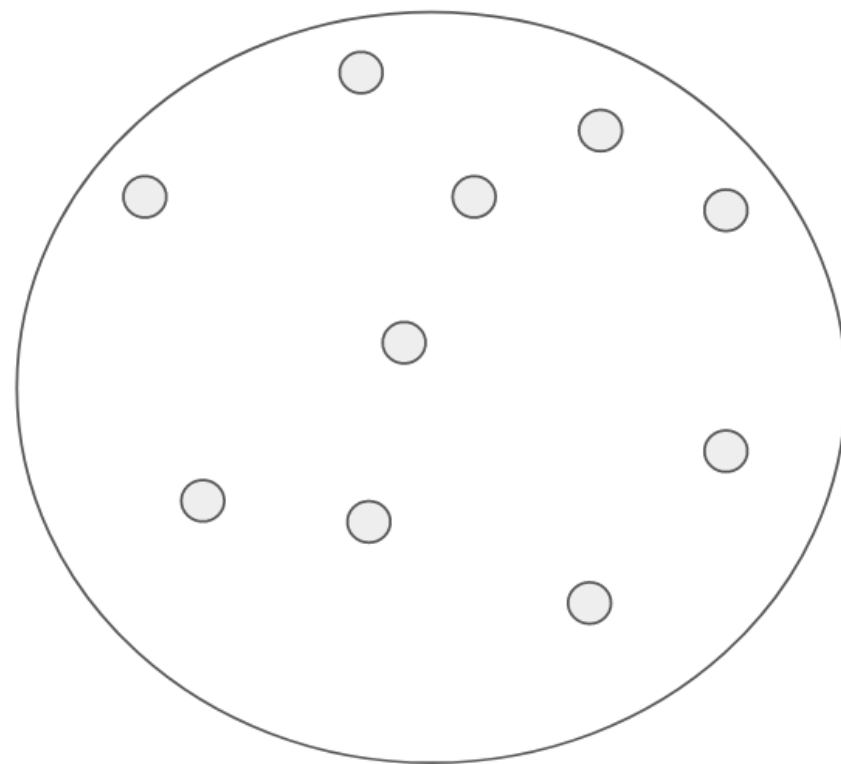
Path2: "aapp--le" $\xrightarrow{B("aapp--le")}$ Labeling: "apple"

$$p("aapp--le") = y_a^1 \cdot y_a^2 \cdot y_p^3 \cdot y_p^4 \cdot y_-^5 \cdot y_-^6 \cdot y_l^7 \cdot y_e^8$$

CTC loss. Пути

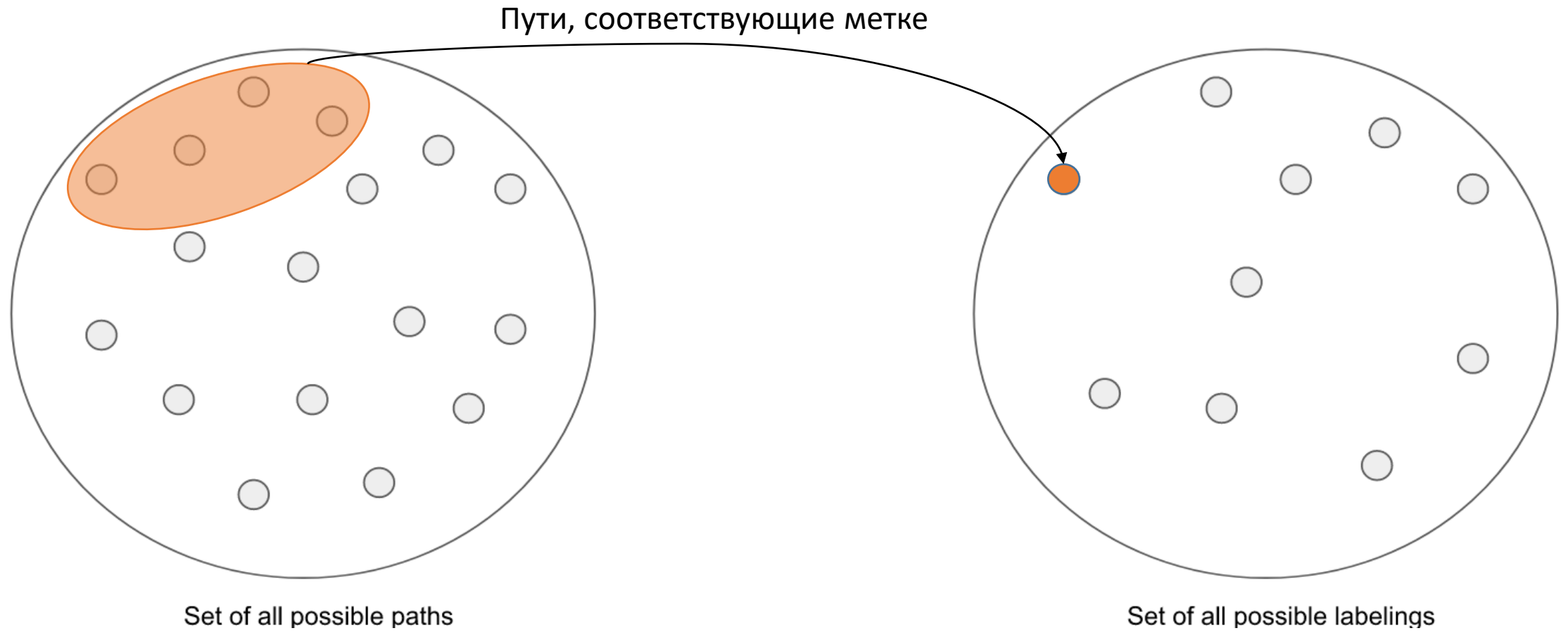


Set of all possible paths



Set of all possible labelings

CTC loss. Пути



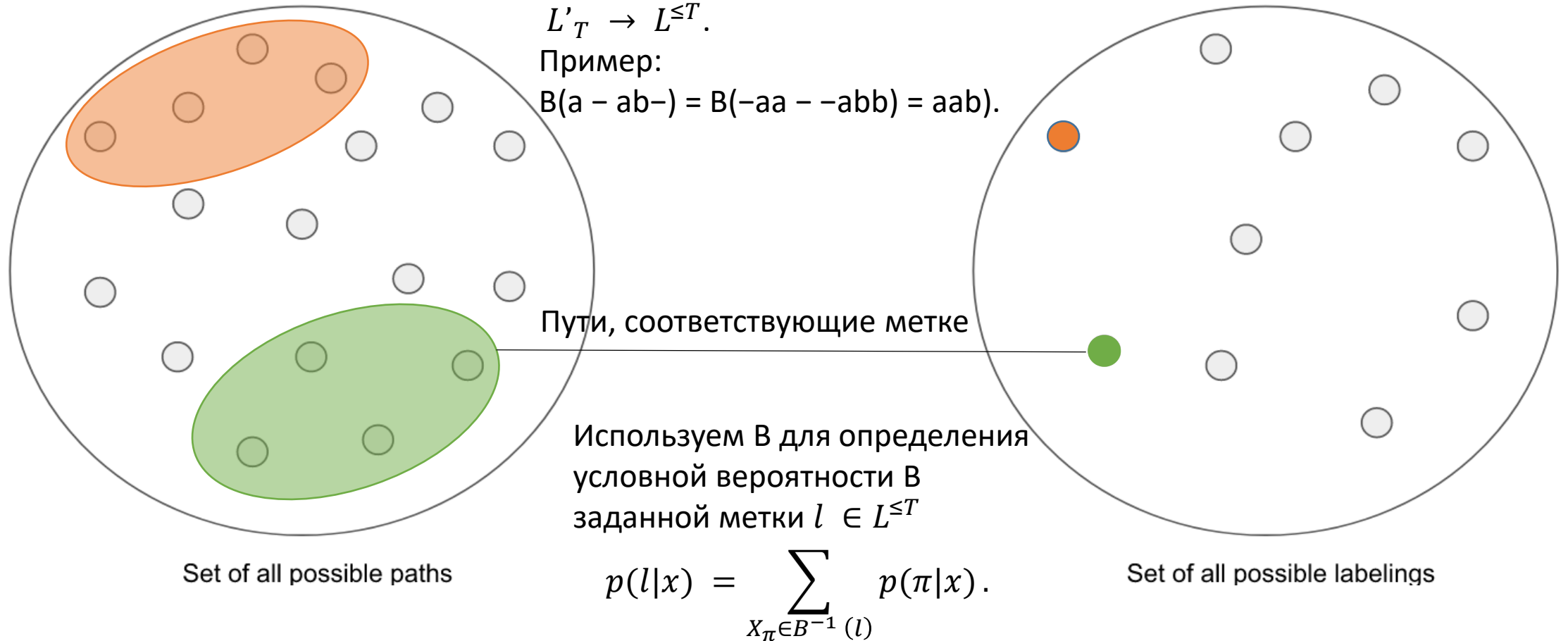
CTC loss. Пути

Мы хотим получить маппинг: $B :$

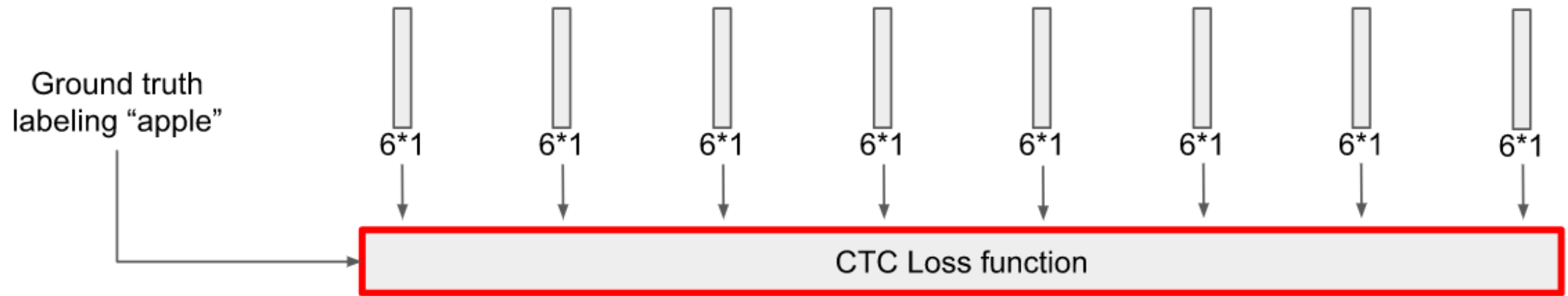
$$L'_T \rightarrow L^{\leq T}.$$

Пример:

$$B(a - ab-) = B(-aa - -abb) = aab).$$



CTC loss. (или как найти градиент)



$$\text{CTC Loss} = -\ln(p(\text{"apple"}))$$

- Вероятность слова – сумма вероятностей по всем возможным путям
- $6^8 = 1\,679\,616$ – возможных путей (случай из примера)
- Используем динамическое программирование для нахождения вероятности целевой последовательности

Нахождение возможных путей

- Аналогично прямому проходу и проходам в НММ мы рассчитываем α и β

$$\alpha_t(s) \stackrel{\text{def}}{=} \sum_{\substack{\pi \in N^T; \\ \mathcal{B}(\pi_{1:t}) = \mathbf{l}_{1:s}}} \prod_{t'=1}^t y_{\pi_{t'}}^{t'}.$$

- Суммарная вероятность всех путей, чей префикс заканчивается символом в позиции s в момент времени t от начала последовательности

$$\beta_t(s) \stackrel{\text{def}}{=} \sum_{\substack{\pi \in N^T; \\ \mathcal{B}(\pi_{t:T}) = \mathbf{l}_{s:|l|}}} \prod_{t'=t}^T y_{\pi_{t'}}^{t'}.$$

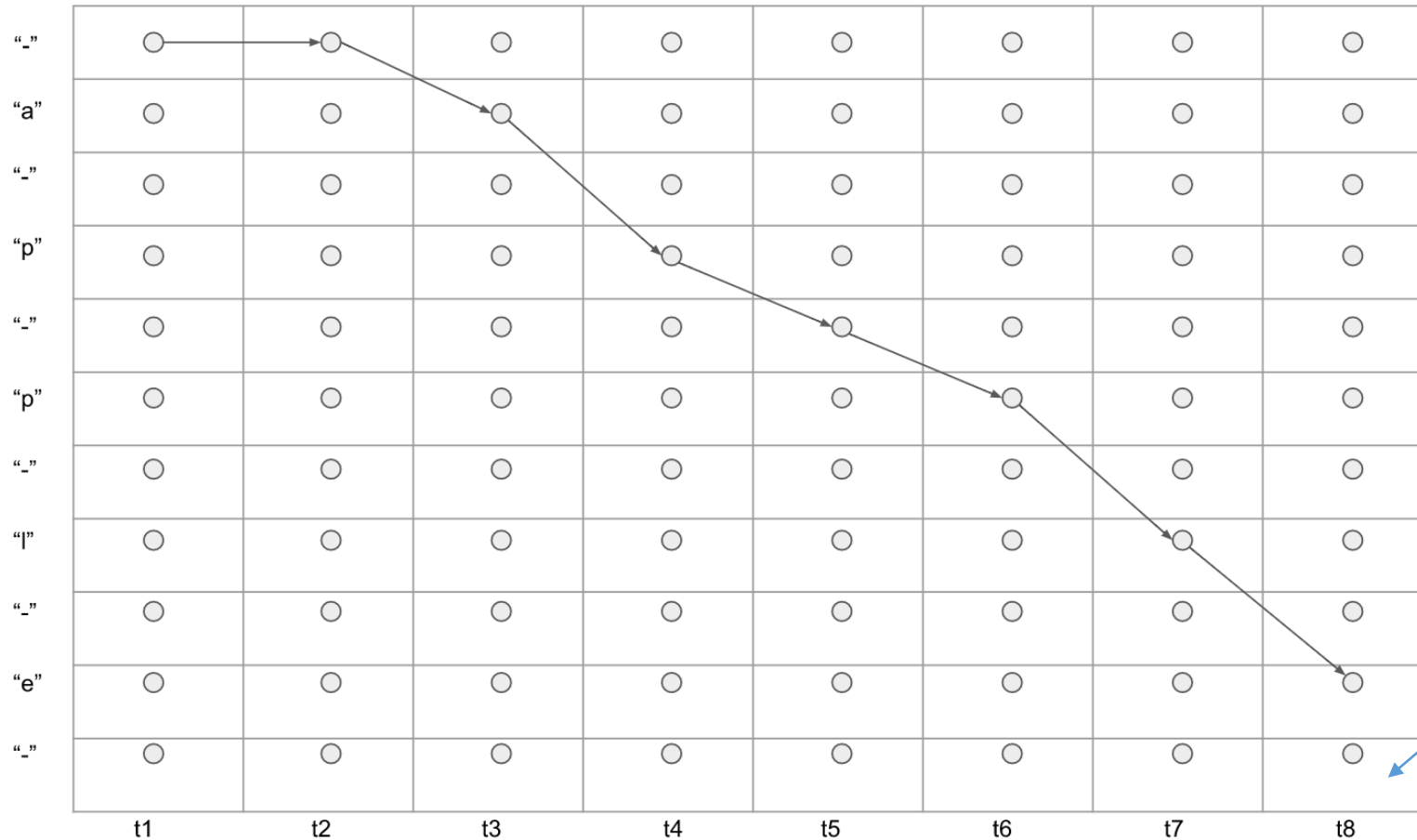
- Суммарная вероятность всех путей, чей суффикс начинается с символом в позиции s в момент времени t

Нахождение возможных путей

"_"	○	○	○	○	○	○	○	○
"a"	○	○	○	○	○	○	○	○
"_"	○	○	○	○	○	○	○	○
"p"	○	○	○	○	○	○	○	○
"_"	○	○	○	○	○	○	○	○
"p"	○	○	○	○	○	○	○	○
"_"	○	○	○	○	○	○	○	○
"l"	○	○	○	○	○	○	○	○
"_"	○	○	○	○	○	○	○	○
"e"	○	○	○	○	○	○	○	○
"_"	○	○	○	○	○	○	○	○
	t1	t2	t3	t4	t5	t6	t7	t8

Нахождение возможных путей

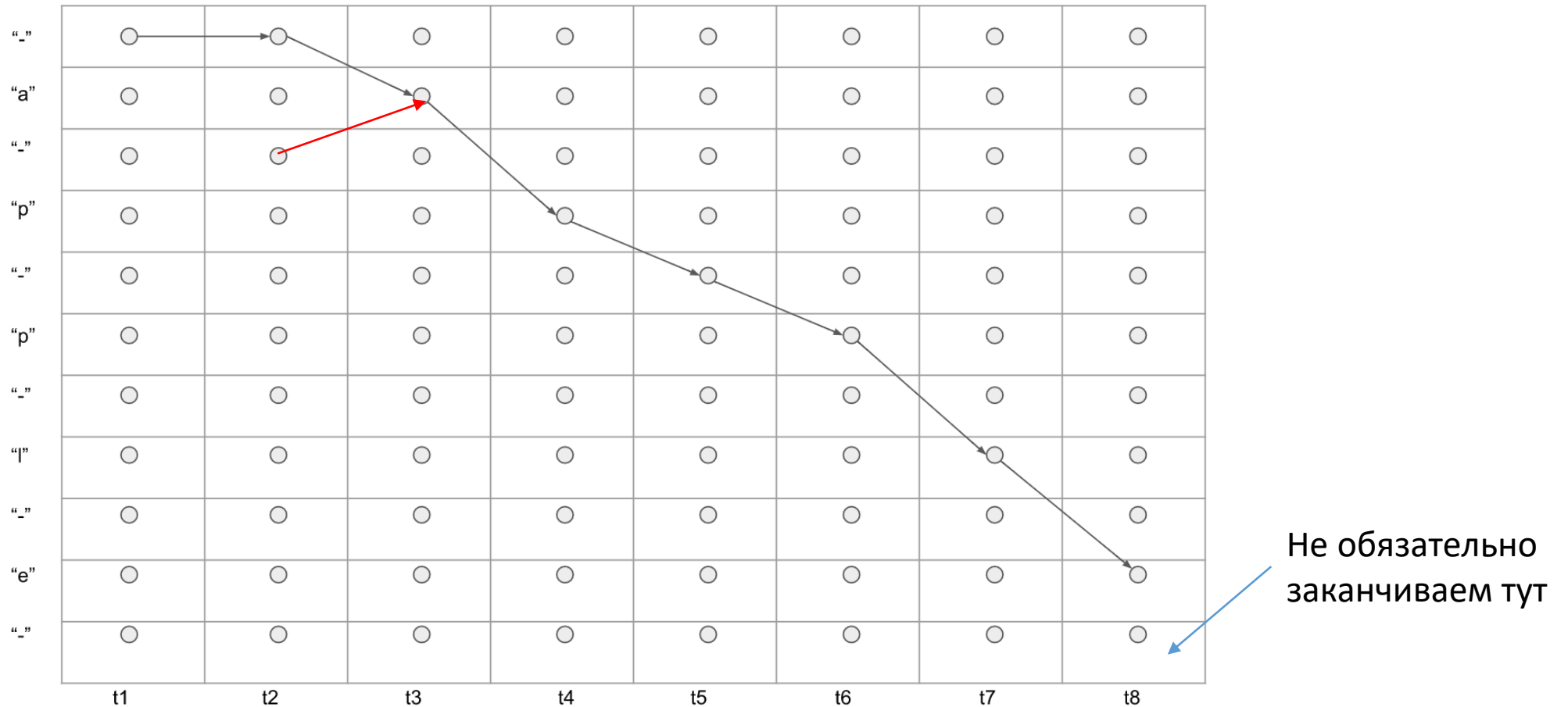
Пример, путь “—ap-ple” может быть отмапирован на маркер “apple” $B(--ap-ple) = \text{“apple”}$



Не обязательно
заканчиваем тут

Нахождение возможных путей

Пример: невозможный переход, нельзя предсказывать предыдущий символ.



Нахождение возможных путей

























































































Начинаем либо b или с первого символа

Инициализация

$$\alpha_1(1) = y_b^1$$

$$\alpha_1(2) = y_{I_1}^1$$

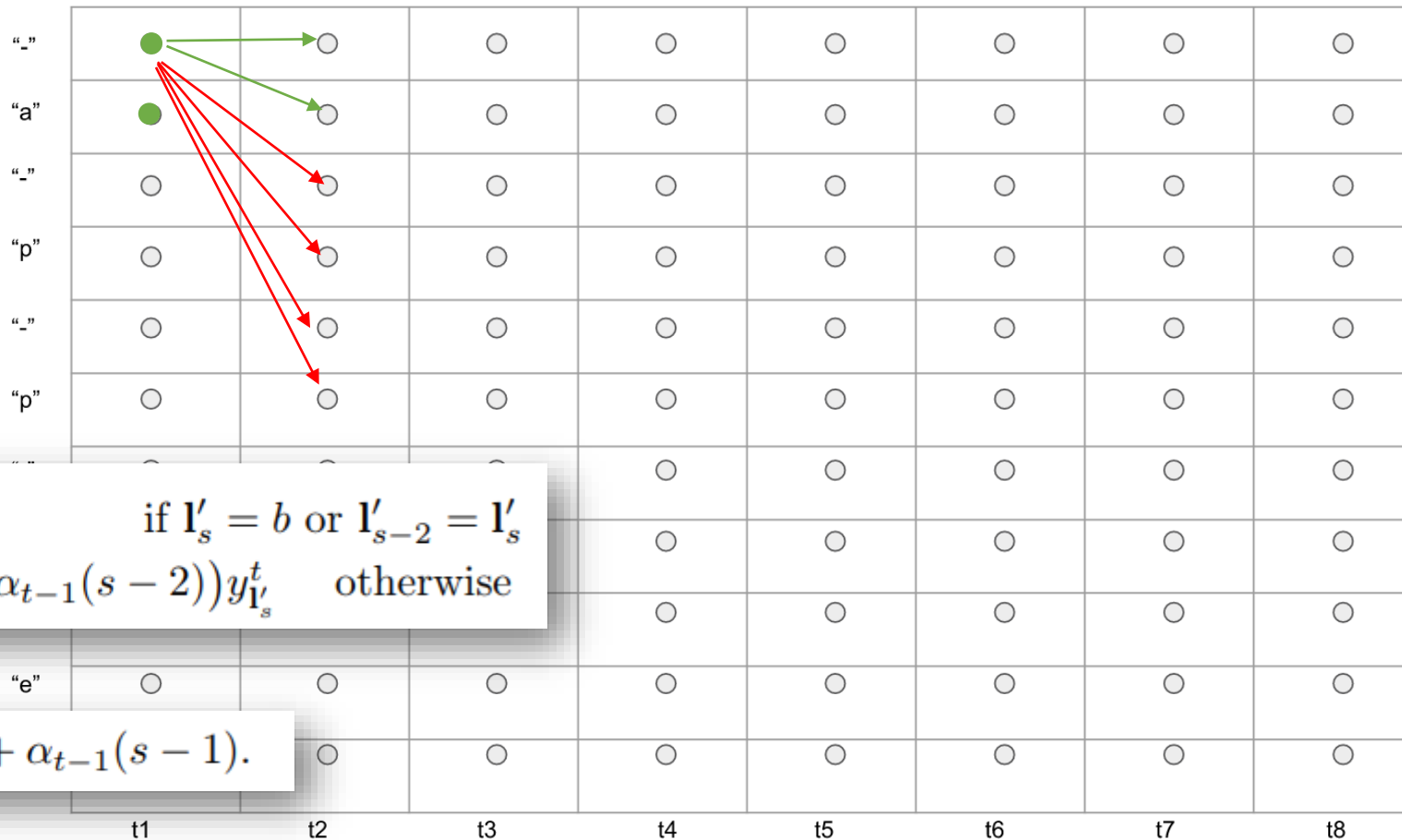
$$\alpha_1(s) = 0, \forall s > 2$$

"_"								
"a"								
"_"								
"p"								
"_"								
"p"								
"_"								
"l"								
"_"								
"e"								
"_"								
	t1	t2	t3	t4	t5	t6	t7	t8

b – blank символ

Нахождение возможных путей

Куда можно двигаться из этих точек

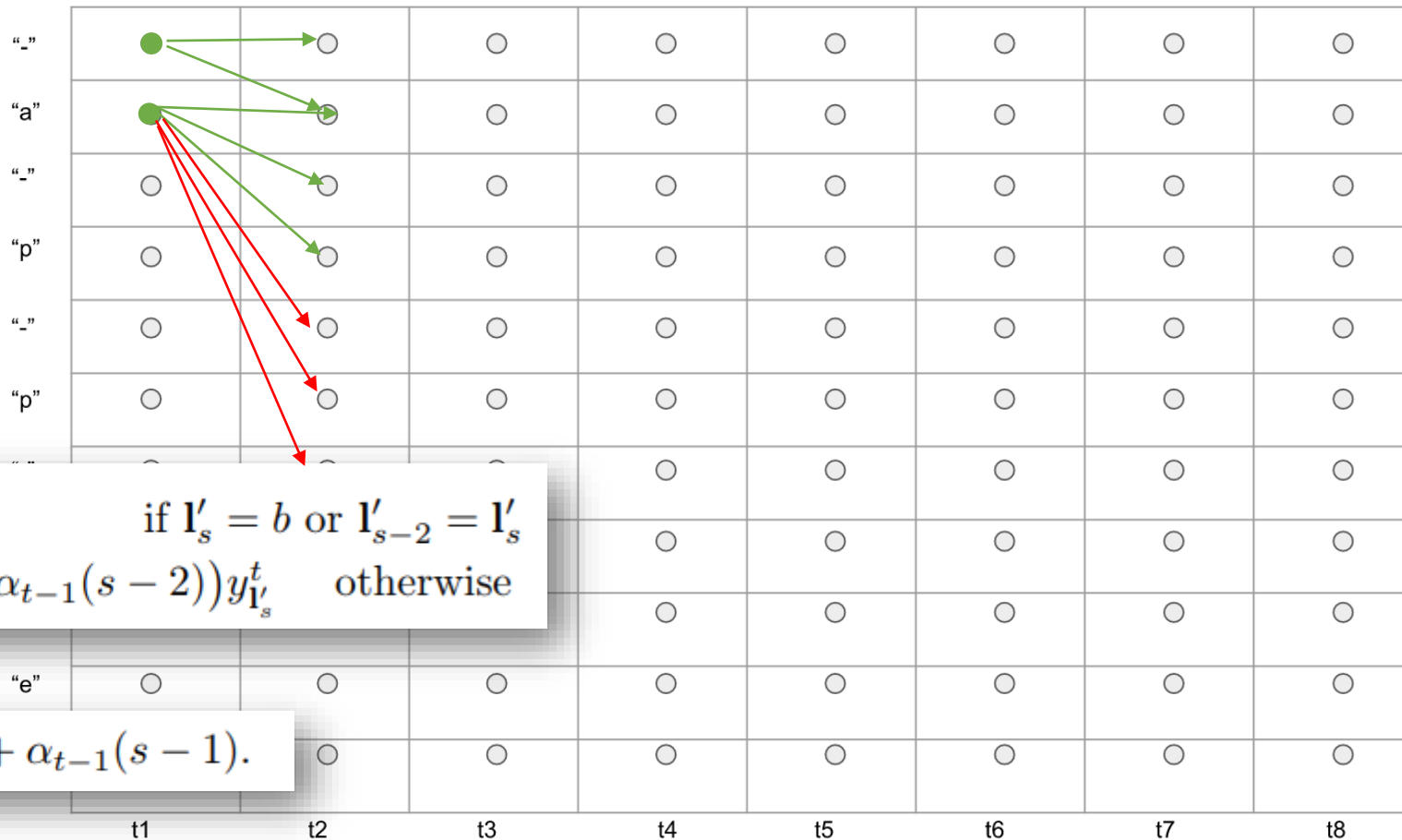


$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y_{l'_s}^t & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y_{l'_s}^t & \text{otherwise} \end{cases}$$

$$\bar{\alpha}_t(s) \stackrel{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1).$$

Нахождение возможных путей

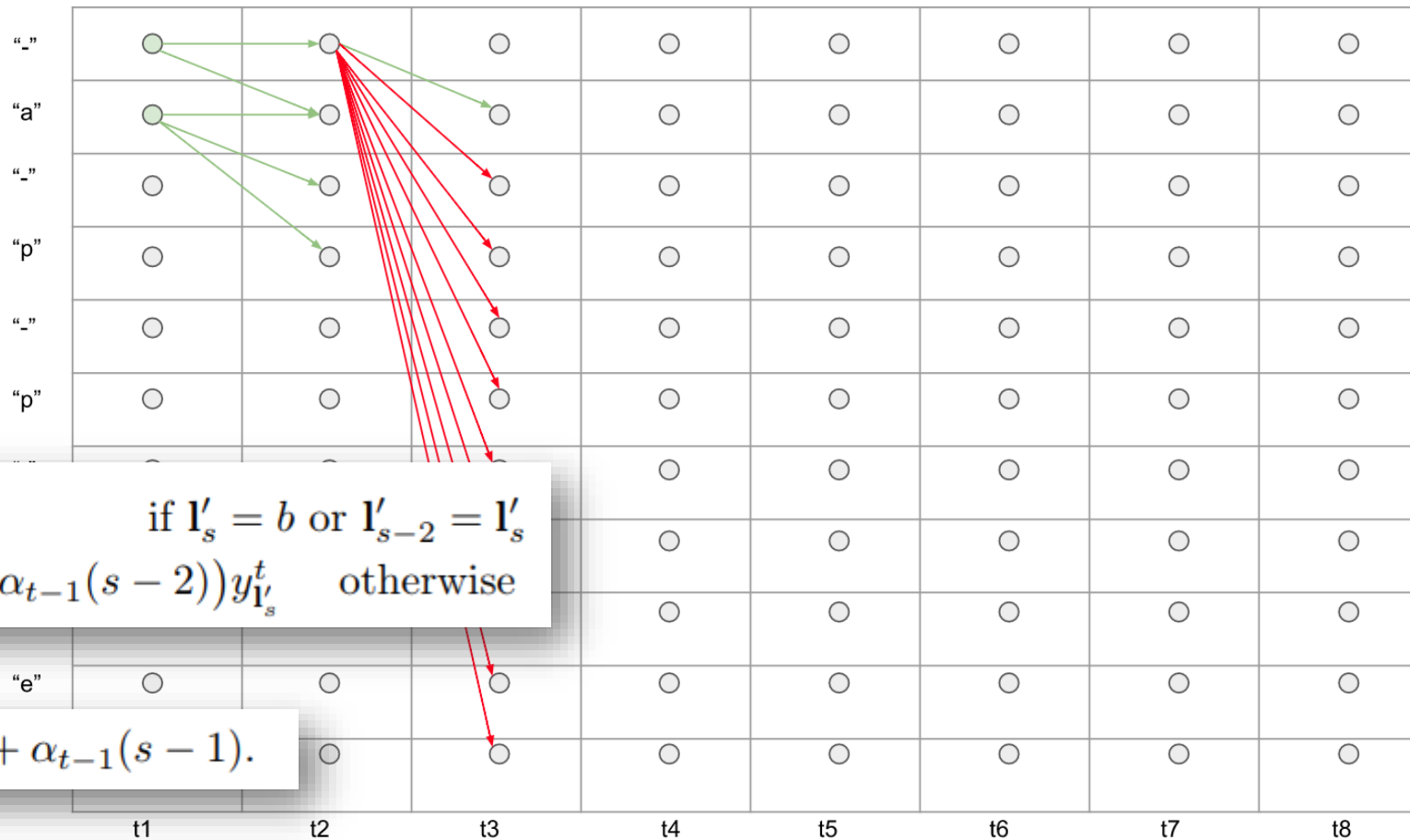
Куда можно двигаться из этих точек



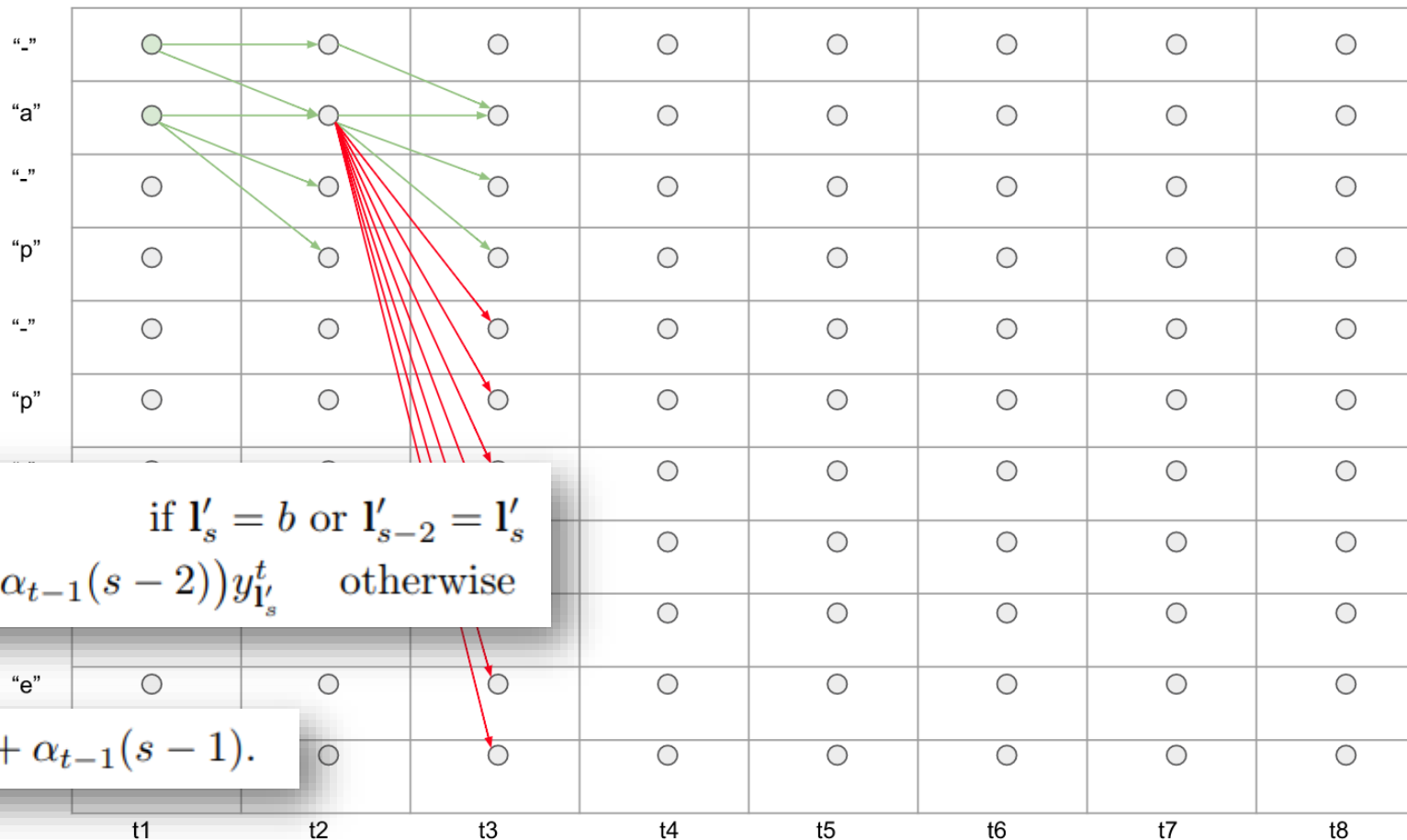
$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y_{l'_s}^t & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y_{l'_s}^t & \text{otherwise} \end{cases}$$

$$\bar{\alpha}_t(s) \stackrel{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1).$$

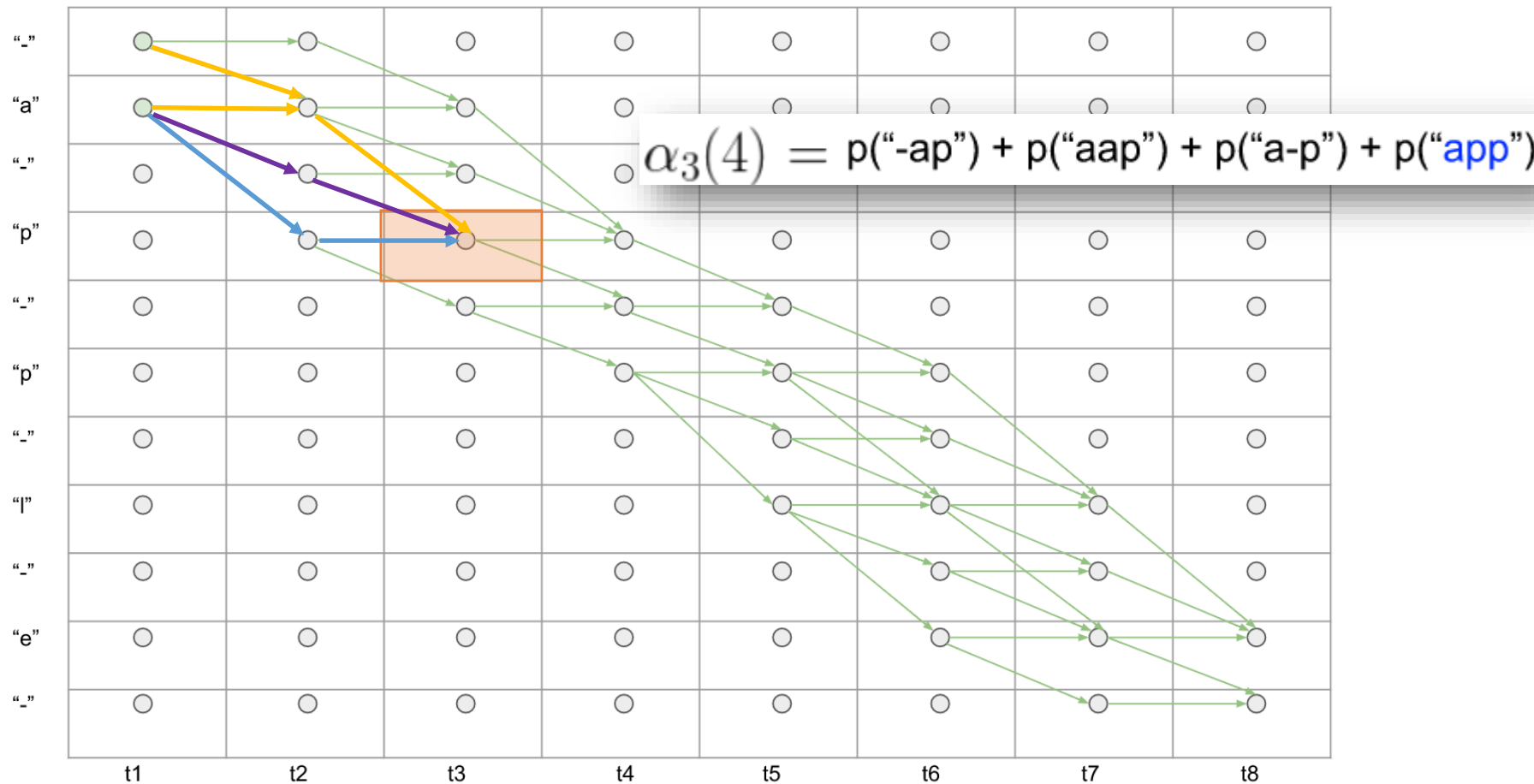
Нахождение возможных путей



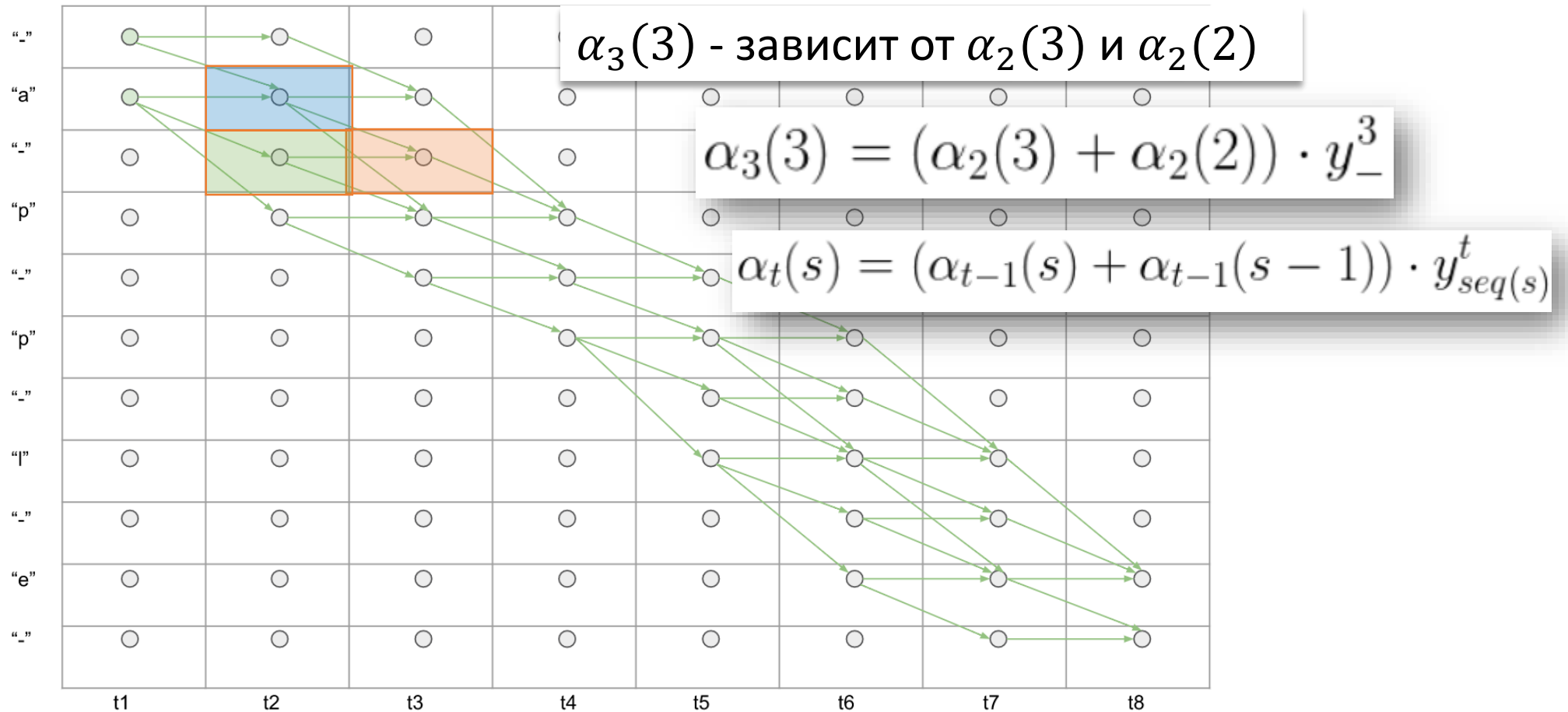
Нахождение возможных путей



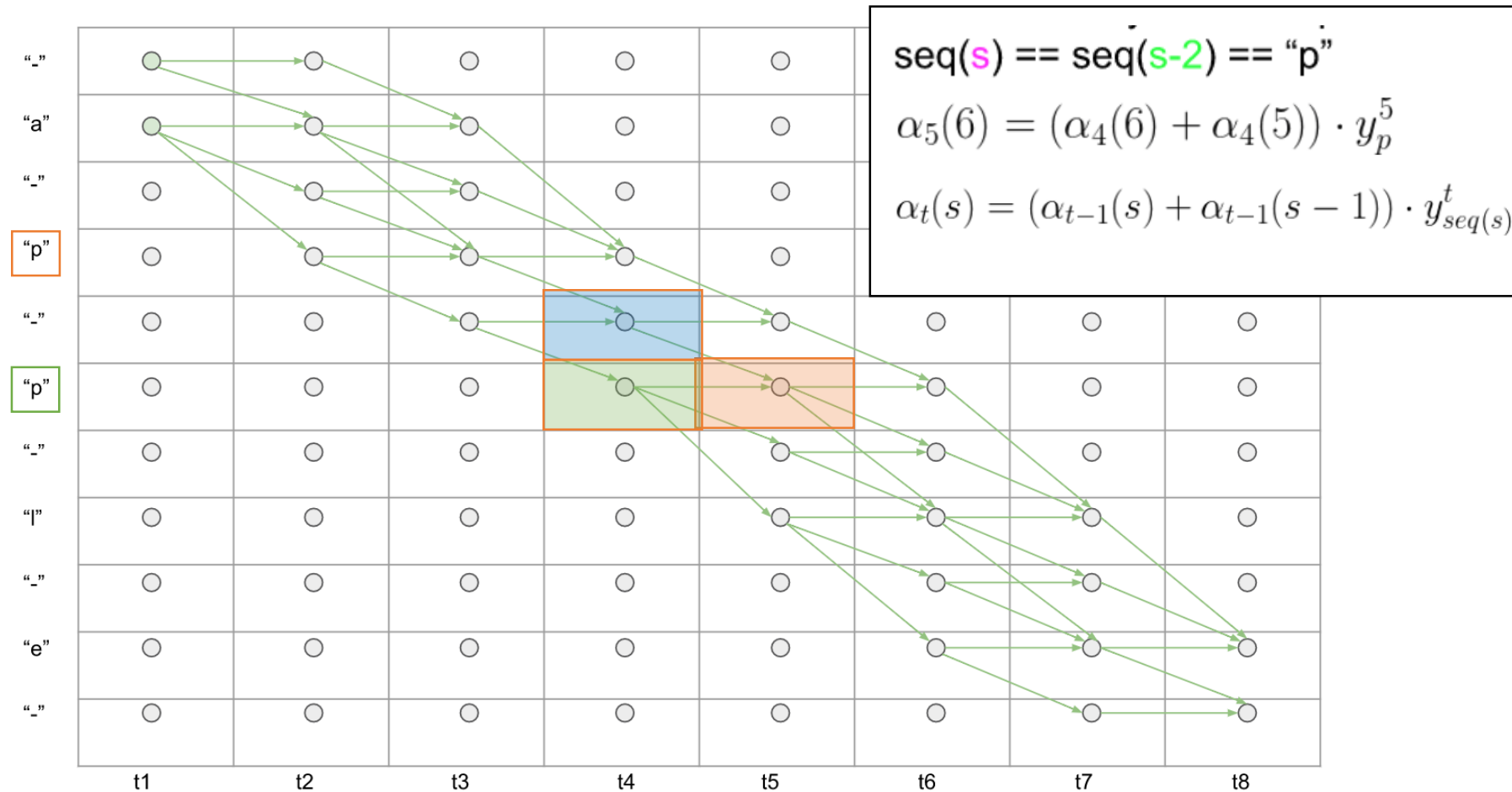
Нахождение возможных путей



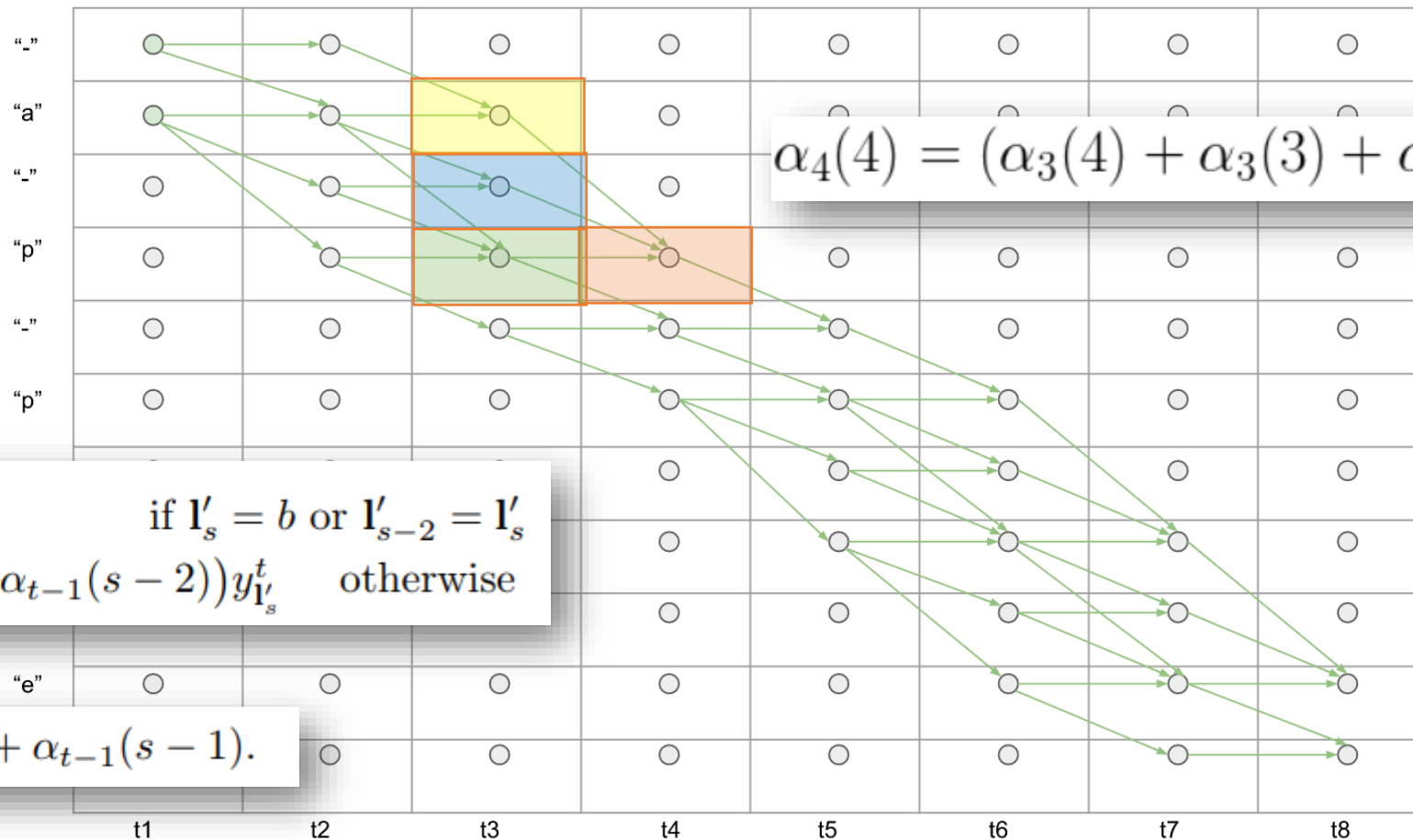
Нахождение возможных путей



Нахождение возможных путей



Нахождение возможных путей



$$\alpha_4(4) = (\alpha_3(4) + \alpha_3(3) + \alpha_3(2)) \cdot y_p^4$$

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y_{l'_s}^t & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y_{l'_s}^t & \text{otherwise} \end{cases}$$

$$\bar{\alpha}_t(s) \stackrel{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1).$$

Нахождение возможных путей



При помощи динамического
программирования рассчитываем
 $\alpha_8(10)$ и $\alpha_8(11)$

$$\text{CTC Loss} = -\ln(p(\text{"apple"})) = -\ln(\alpha_8(10) + \alpha_8(11))$$

Нахождение возможных путей. Обратный проход

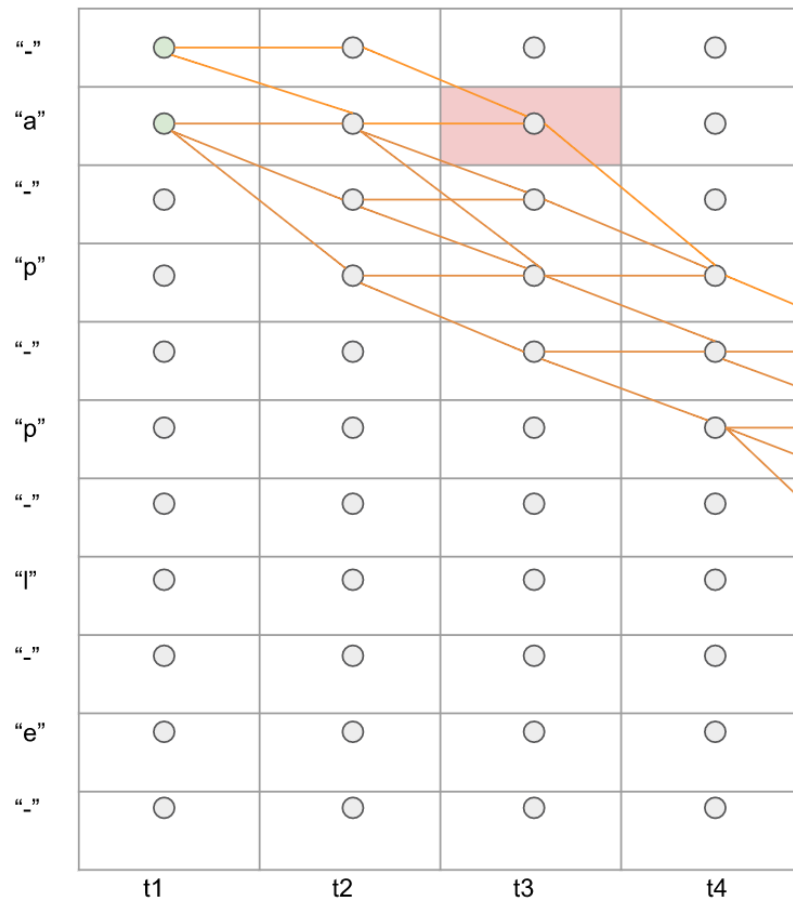
- Делается аналогично прямому проходу для каждого момента времени и метки рассчитываем вероятность

$$\beta_t(s) \stackrel{\text{def}}{=} \sum_{\substack{\pi \in N^T: \\ \mathcal{B}(\pi_{t:T}) = \mathbf{l}_{s:|I|}}} \prod_{t'=t}^T y_{\pi_{t'}}^{t'}$$

$$\begin{aligned} \beta_T(|I'|) &= y_b^T \\ \beta_T(|I'| - 1) &= y_{\mathbf{l}_{|I|}}^T \\ \beta_T(s) &= 0, \quad \forall s < |I'| - 1 \end{aligned}$$

$$\beta_t(s) = \begin{cases} \bar{\beta}_t(s) y_{\mathbf{l}'_s}^t & \text{if } \mathbf{l}'_s = b \text{ or } \mathbf{l}'_{s+2} = \mathbf{l}'_s \\ (\bar{\beta}_t(s) + \beta_{t+1}(s+2)) y_{\mathbf{l}'_s}^t & \text{otherwise} \end{cases} \quad \bar{\beta}_t(s) \stackrel{\text{def}}{=} \beta_{t+1}(s) + \beta_{t+1}(s+1).$$

Вероятность путей для отдельной метки



$$\alpha_3(2) = p("--a") + p("-aa") + p("aaa") =$$

$$= y_{-}^1 \cdot y_{-}^2 \cdot y_a^3 + y_{-}^1 \cdot y_a^2 \cdot y_a^3 + y_a^1 \cdot y_a^2 \cdot y_a^3$$

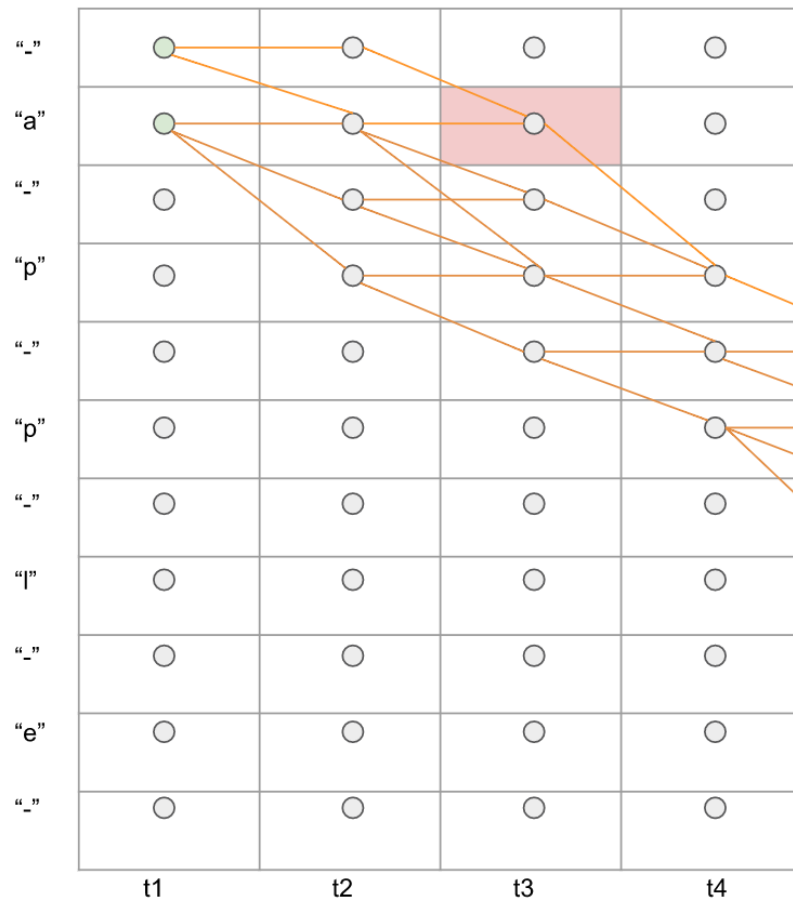
$$\beta_3(2) = p("ap-le") = y_a^3 \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8$$

$$\Rightarrow \alpha_3(2) \cdot \beta_3(2) = y_{-}^1 \cdot y_{-}^2 \cdot y_a^3 \cdot y_a^3 \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8 +$$

$$+ y_{-}^1 \cdot y_a^2 \cdot y_a^3 \cdot y_a^3 \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8$$

$$+ y_a^1 \cdot y_a^2 \cdot y_a^3 \cdot y_a^3 \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8$$

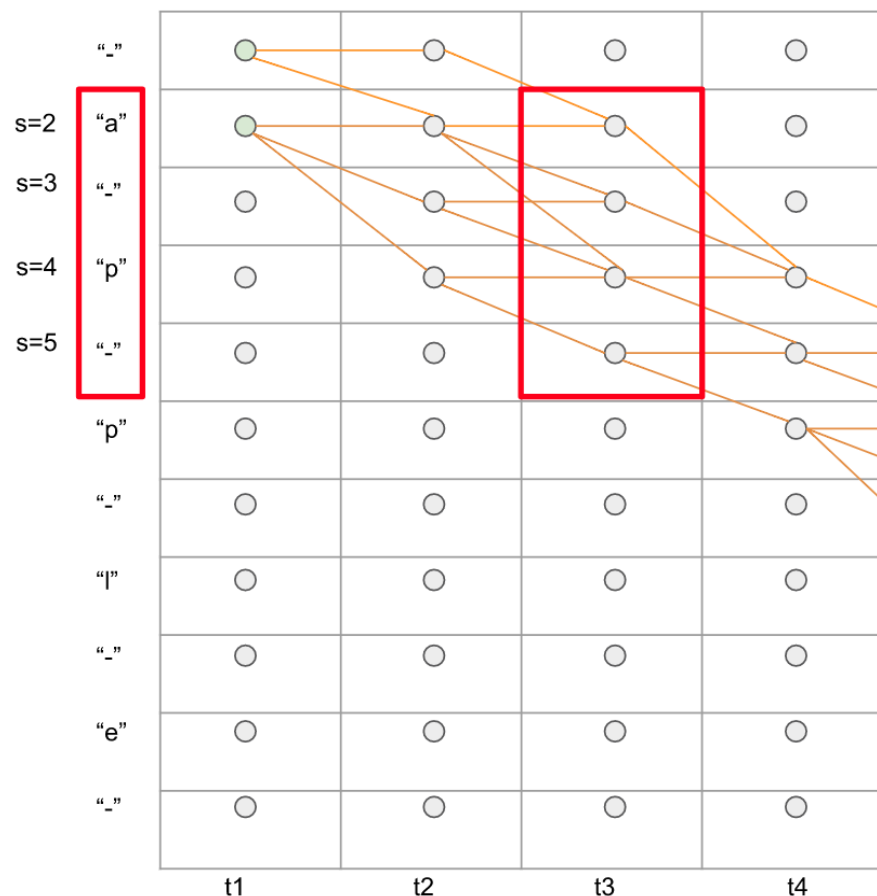
Вероятность путей для отдельной метки



$$\begin{aligned}
 \alpha_3(2) \cdot \beta_3(2) &= y_{-}^1 \cdot y_{-}^2 \cdot y_a^3 \cdot \boxed{y_a^3} \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8 + \\
 &+ y_{-}^1 \cdot y_a^2 \cdot y_a^3 \cdot \boxed{y_a^3} \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8 \\
 &+ y_a^1 \cdot y_a^2 \cdot y_a^3 \cdot \boxed{y_a^3} \cdot y_p^4 \cdot y_{-}^5 \cdot y_p^6 \cdot y_l^7 \cdot y_e^8 = \\
 &= (p("--aap-ple") + p("-aap-ple") + p("aaap-ple")) * y_a^3
 \end{aligned}$$

$$\frac{\alpha_3(2) \cdot \beta_3(2)}{y_a^3} = (p("--aap-ple") + p("-aap-ple") + p("aaap-ple"))$$

Вероятность слова для времени



Суммарная вероятность
всех путей на шаге 3

$$p("apple") = \sum_{s=2}^5 \frac{\alpha_3(s) \cdot \beta_3(s)}{y_{seq(s)}^3}$$

Вероятность в любой
момент времени

$$p("apple") = \sum_{s=1}^{|seq|} \frac{\alpha_t(s) \cdot \beta_t(s)}{y_{seq(s)}^t}$$

$$\text{CTC Loss} = -\ln(p(\text{"apple"}))$$

Градиент

$$p(\text{"apple"}) = \sum_{s=1}^{|\text{seq}|} \frac{\alpha_t(s) \cdot \beta_t(s)}{y_{\text{seq}(s)}^t}$$

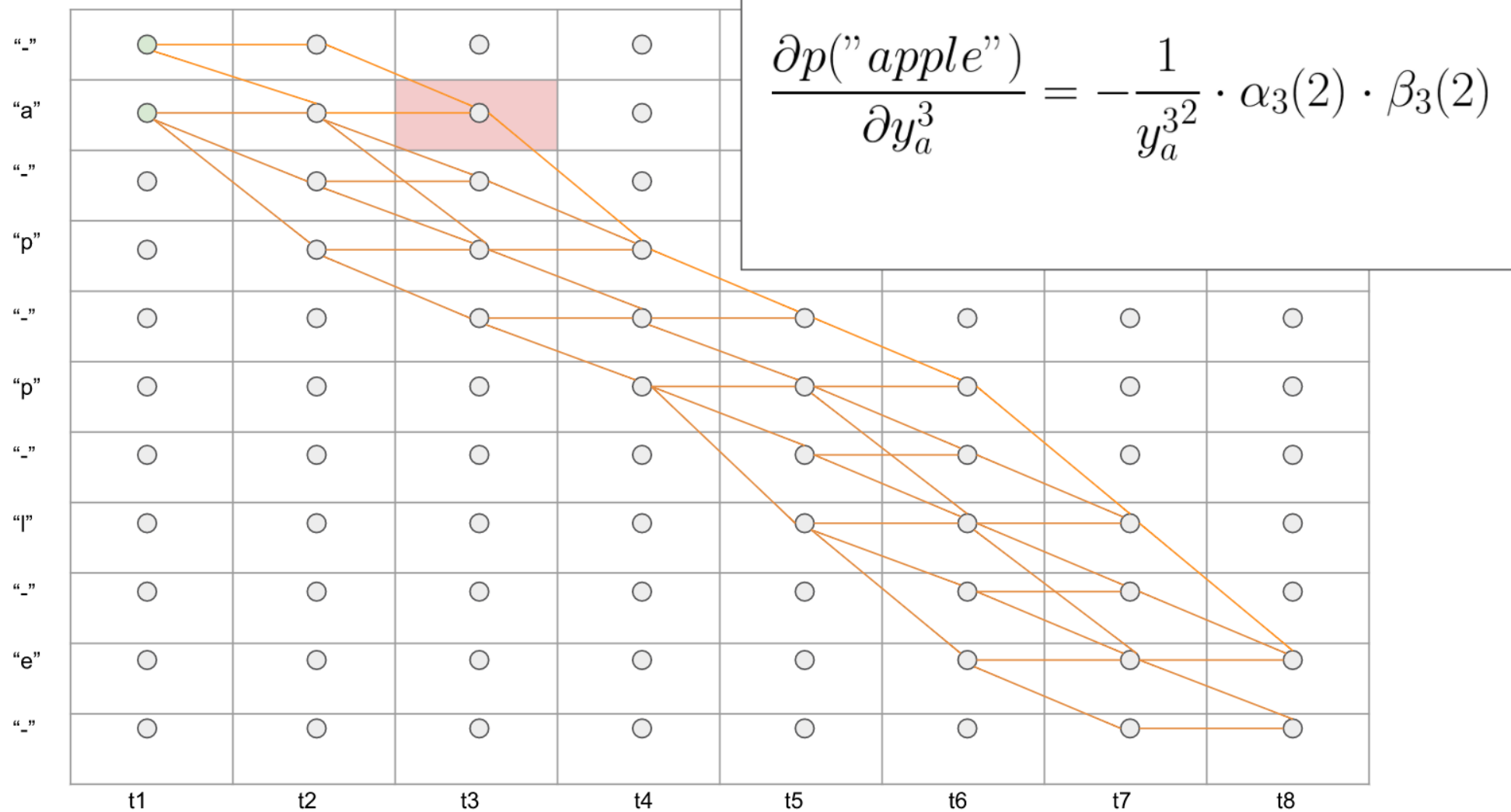
- Принимаем во внимание, что

$$\frac{\partial(-\ln(p(\text{"apple"})))}{\partial y_k^t} = -\frac{1}{p(\text{"apple"})} \cdot \boxed{\frac{\partial p(\text{"apple"})}{\partial y_k^t}}$$

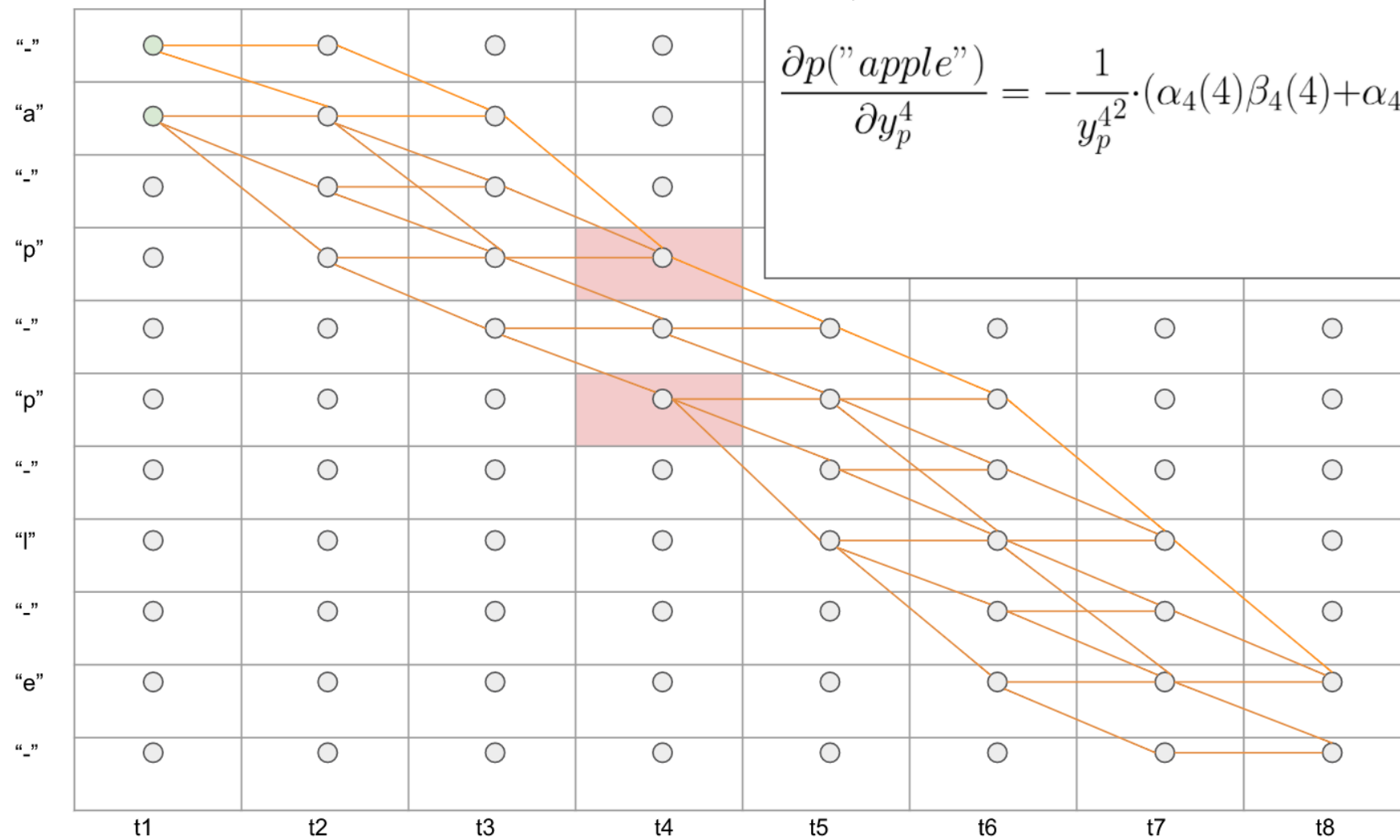
- Чтобы обучить сеть нам нужно продифференцировать функцию ошибки по всем выходам сети y_k^t
- Рассматриваем только пути идущие через символ k в момент t

$$\frac{\partial p(\text{"apple"})}{\partial y_k^t} = -\frac{1}{y_k^{t^2}} \cdot \sum_{s:\text{seq}(s)=k} \alpha_t(s) \cdot \beta_t(s)$$

Градиент



Градиент



Example2:

$$\frac{\partial p("apple")}{\partial y_p^4} = -\frac{1}{y_p^{42}} \cdot (\alpha_4(4)\beta_4(4) + \alpha_4(6)\beta_4(6))$$

Deep Speech

- Вход – транскрибированное аудио сэмплы (частота дискретизации 8K)

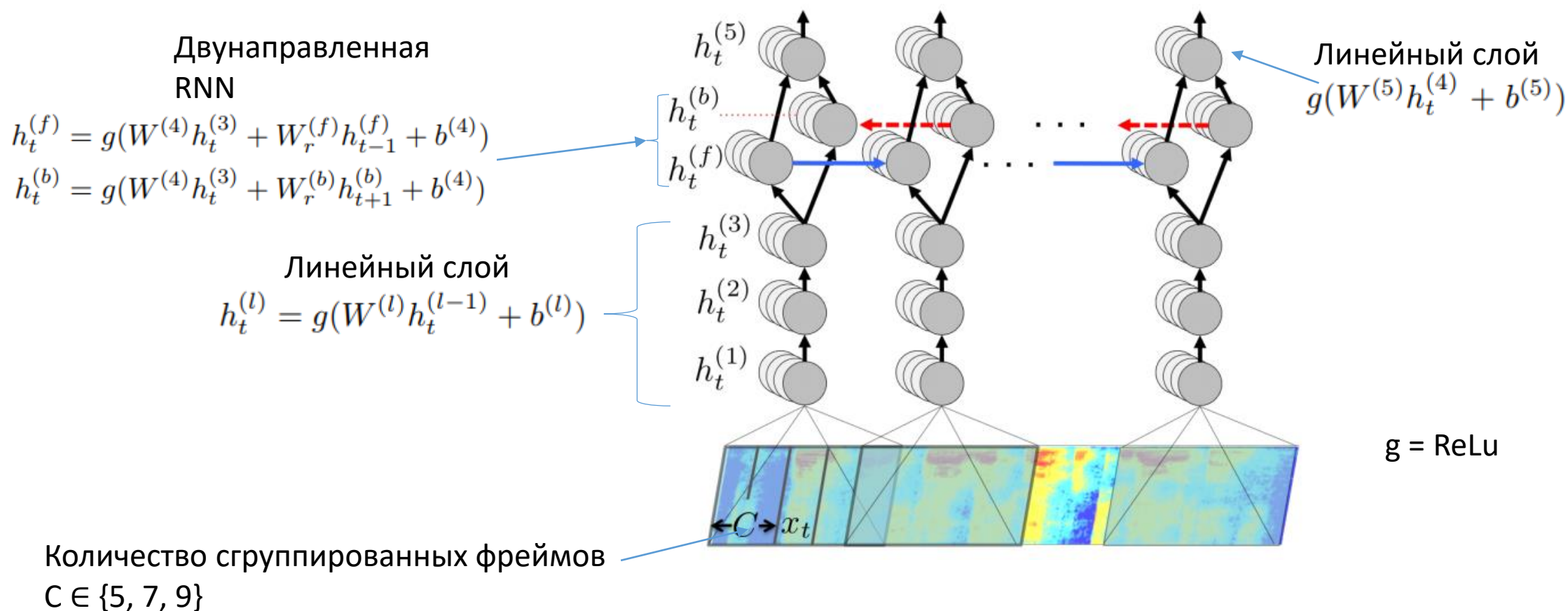
$$\mathcal{X} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots\}.$$

- Фичи – спектрограммы. 80 линейно распределенных логарифмических фильтр-банков.

Deep Speech

- Архитектура сети

$$h_{t,k}^{(6)} = \hat{y}_{t,k} \equiv \mathbb{P}(c_t = k|x) = \frac{\exp(W_k^{(6)} h_t^{(5)} + b_k^{(6)})}{\sum_j \exp(W_j^{(6)} h_t^{(5)} + b_j^{(6)})}.$$



Deep Speech

- Предсказывает последовательность символов
- Loss = CTC
- Для предсказания использует языковую модель

$$Q(c) = \log(\mathbb{P}(c|x)) + \alpha \log(\mathbb{P}_{lm}(c)) + \beta \text{word_count}(c)$$

Вероятность
последователь-
ности RNN

Вероятность последовательности языковой модели

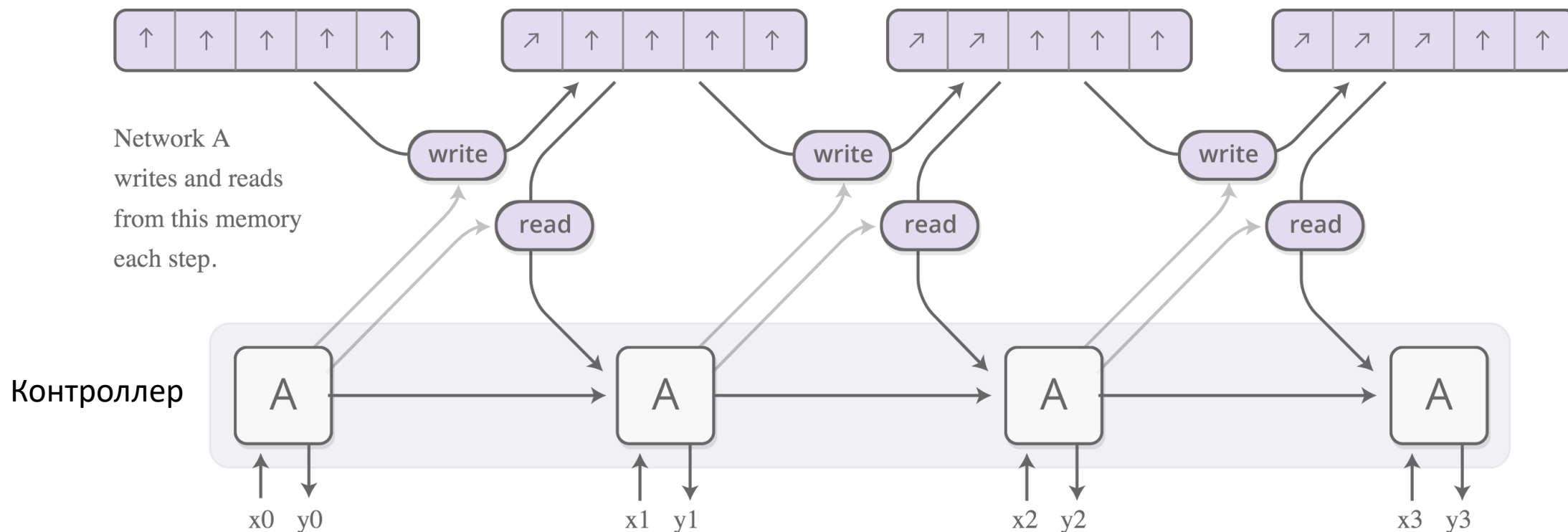
RNN output	Decoded Transcription
what is the weather like in bostin right now prime miniter nerenr modi arther n tickets for the game	what is the weather like in boston right now prime minister narendra modi are there any tickets for the game

$$F(x, y) = x + y = \text{NTM}$$

- Можно ли с помощью нейросети решать задачи сложения или умножения?
 - Рассмотрим выражение как последовательность
 - Поместить в память x
 - Поместить в память $+$
 - Извлечь из памяти $x +$
 - Выполнить сложение
 - Нужно решить задачу записи и считывания из памяти
 - Память
 - Механизм адресации

Attention. NTM

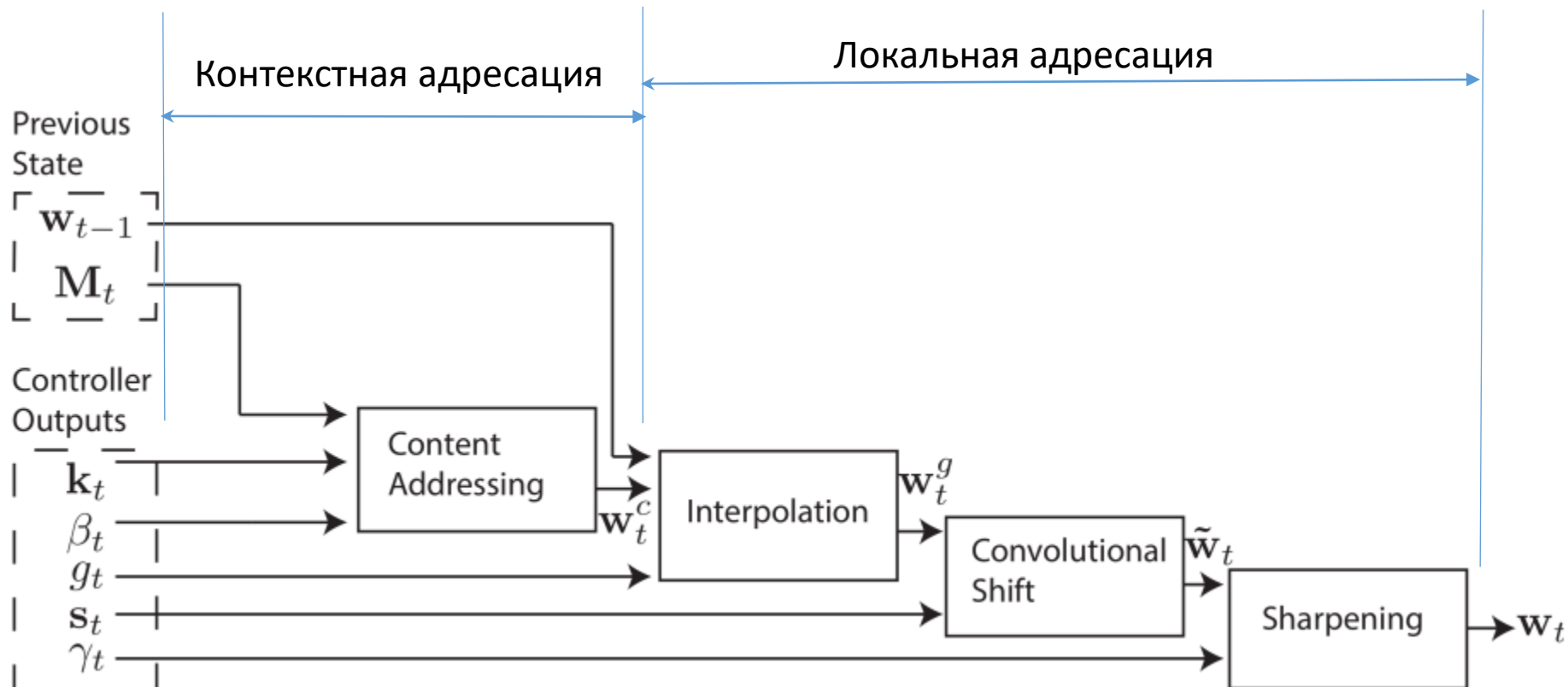
Memory is an array of vectors.



NTM – рекуррентная нейронная сеть, умеющая работать с внешней памятью. Является полной по Тьюрингу.

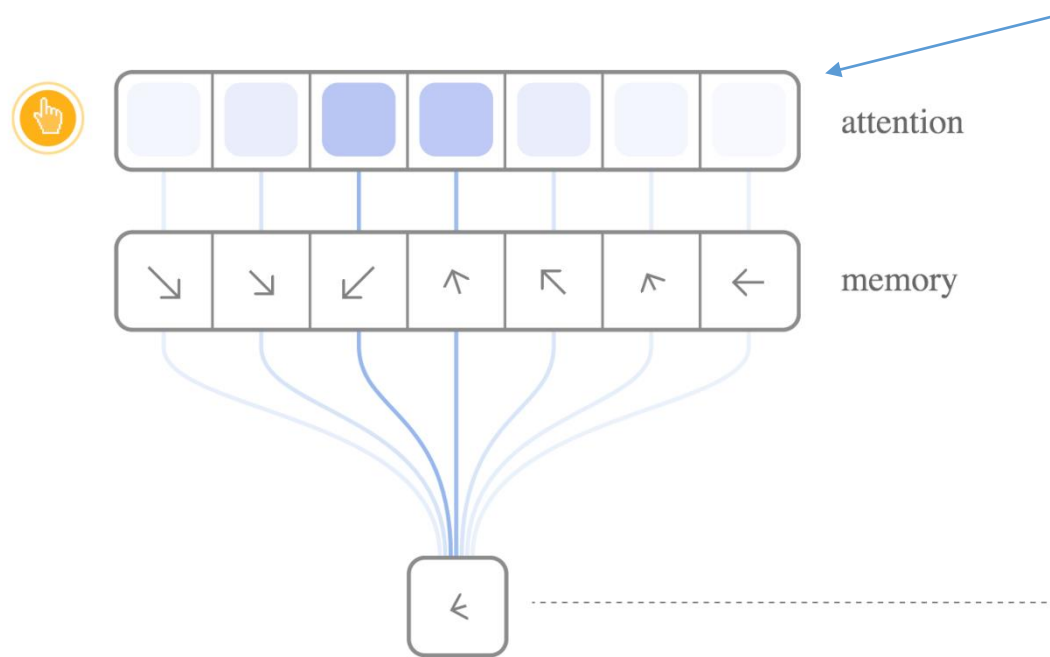
<https://distill.pub/2016/augmented-rnns/>

Схема механизма адресации



Контекстная адресация – находим смещение в памяти, в соответствии с контекстом запроса
Локальная адресация – находим смещение, внутри контекста

Attention. NTM. Read



Распределение Softmax

The RNN gives an attention distribution which describe how we spread out the amount we care about different memory positions.

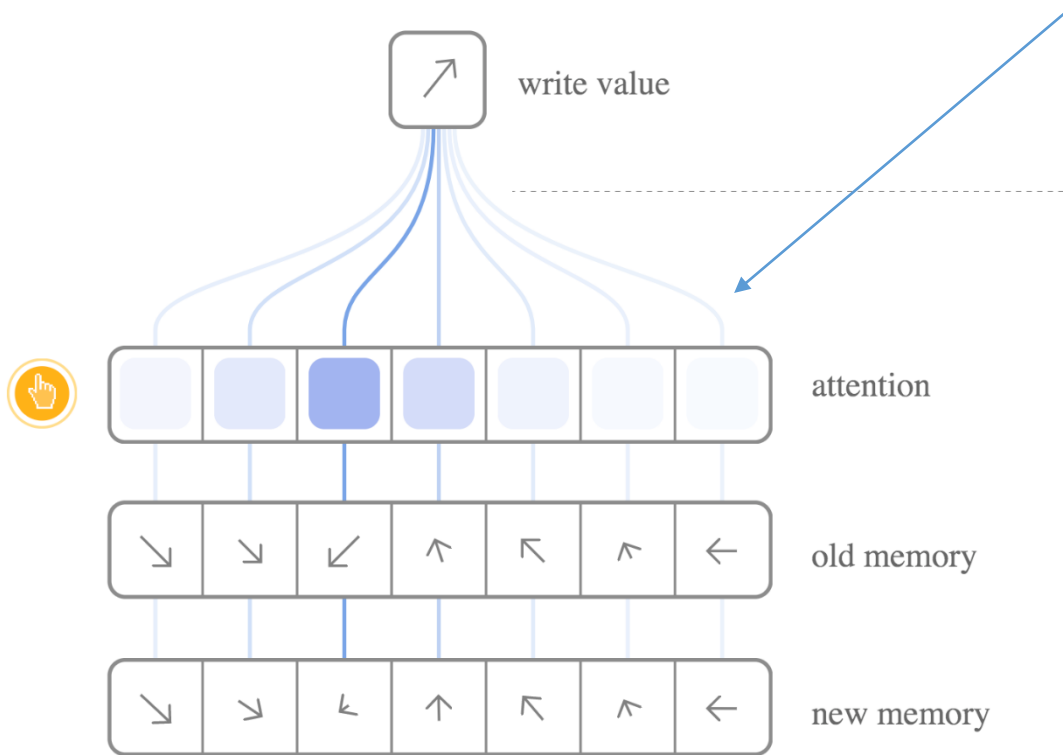
The read result is a weighted sum.

$$r \leftarrow \sum_i a_i M_i$$

Чтение выполняется из всех ячеек сразу в виде линейной комбинации. За адресацию отвечает скрытое состояние.

<https://distill.pub/2016/augmented-rnns/>

Attention. NTM. Write



Распределение Softmax

Instead of writing to one location, we write everywhere, just to different extents.

The RNN gives an attention distribution, describing how much we should change each memory position towards the write value.

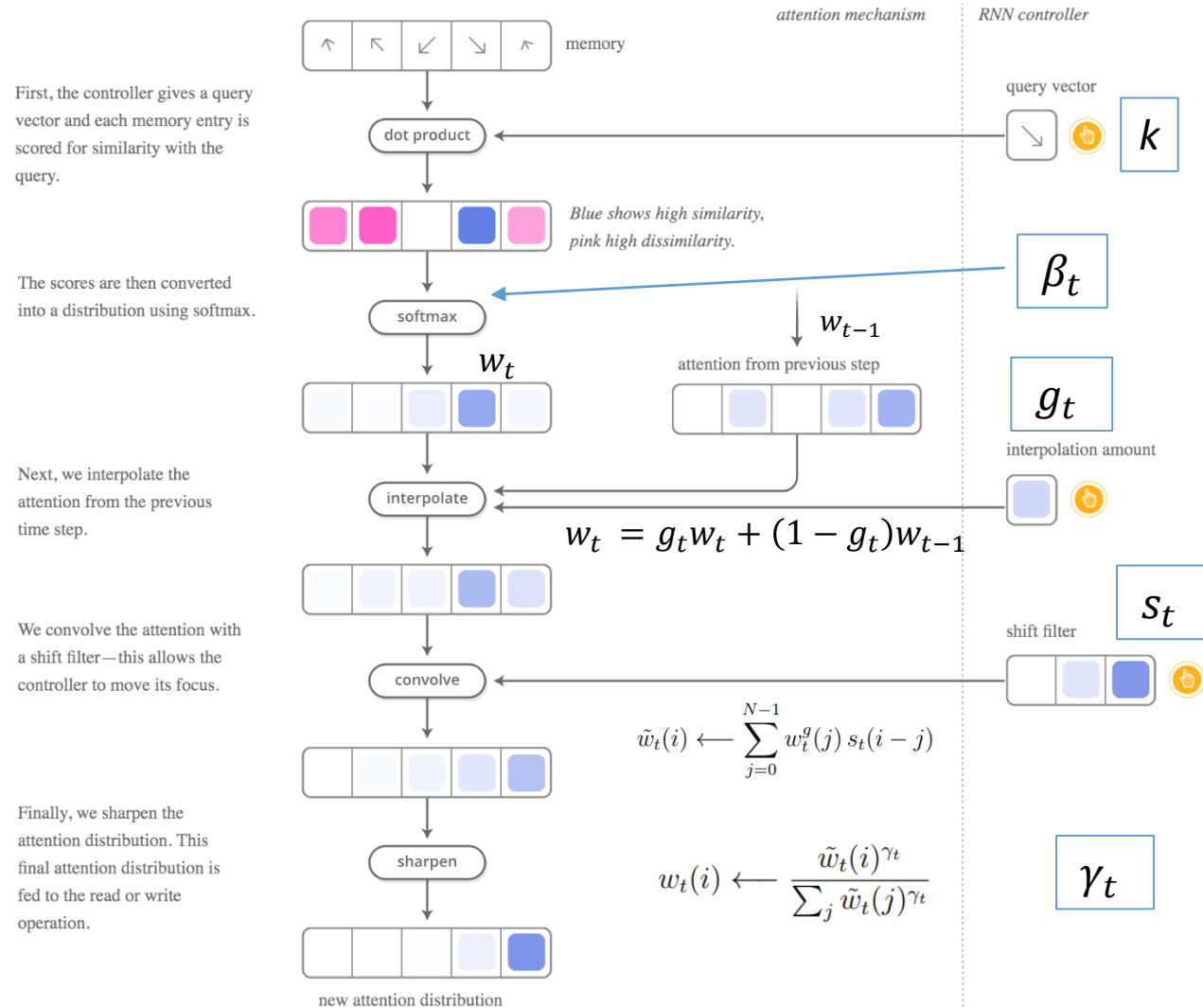
$$M_i \leftarrow a_i w + (1 - a_i) M_i$$

Пишем так же во все ячейки но с разным «акцентом/вниманием». Веса для записи выбираются по такому же принципу, как при чтении.

<https://distill.pub/2016/augmented-rnns/>

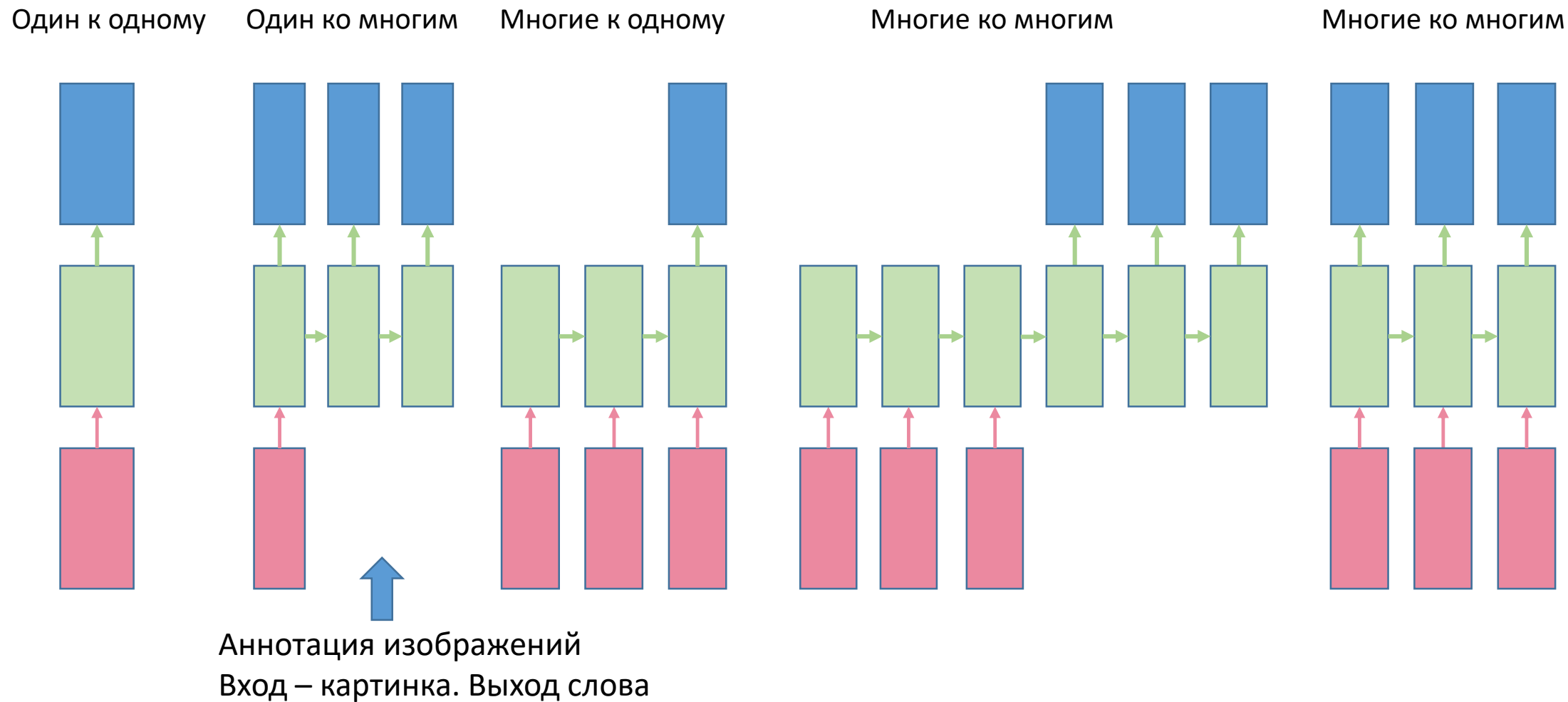
Attention. NTM. Attention

RNN выдает параметры k, β, g, s, γ

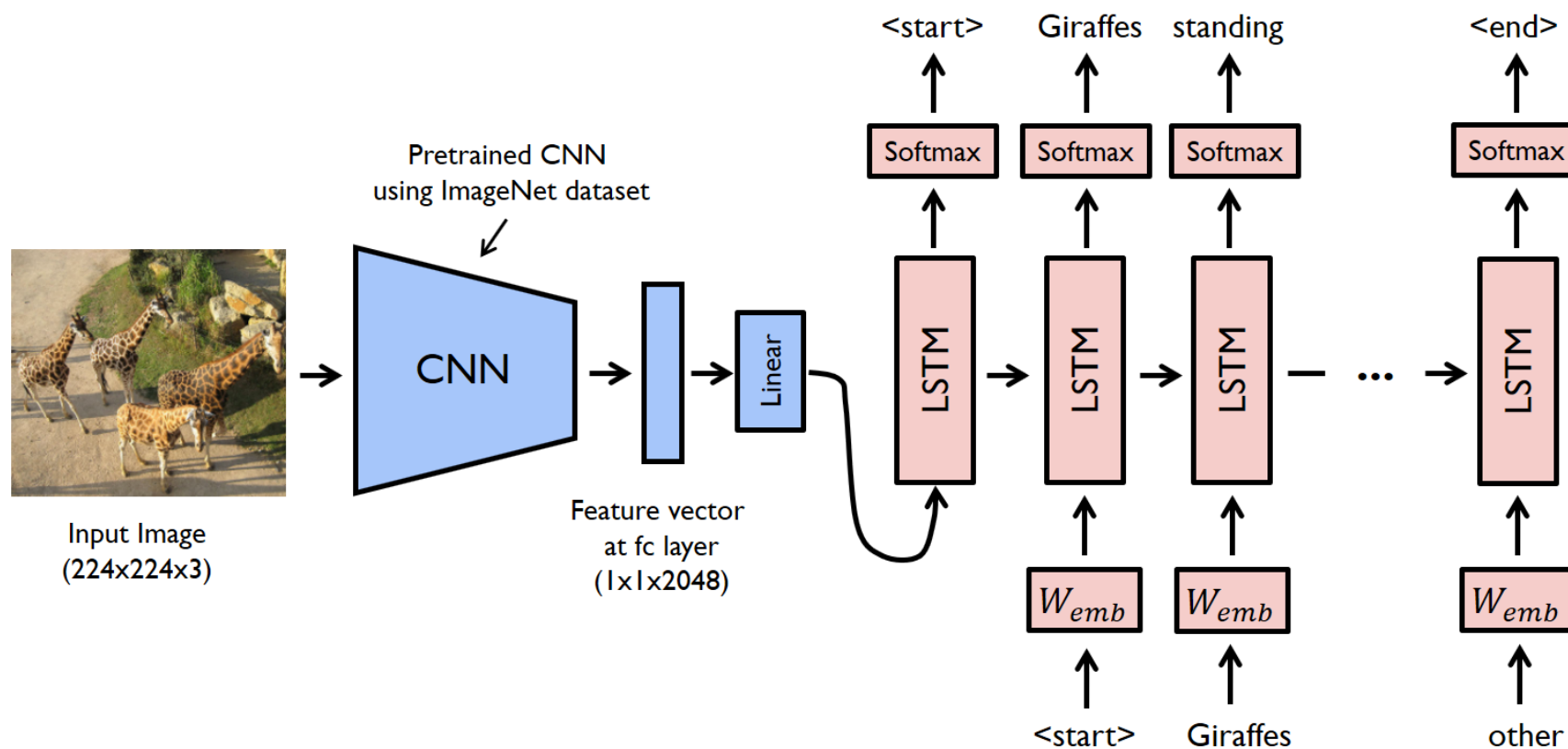


1. По скрытому состоянию генерируем запрос.
2. Сравниваем его со всеми имеющимися векторами в памяти по близости
3. Пропускаем получившиеся оценки близости через softmax и получаем attention
4. Складываем его с некоторым весом со старым attention
5. Проходимся по полученному attention конволюцией для смещения attention в «пространстве» памяти
6. Огрубляем attention для точности

Attention. Image Captioning

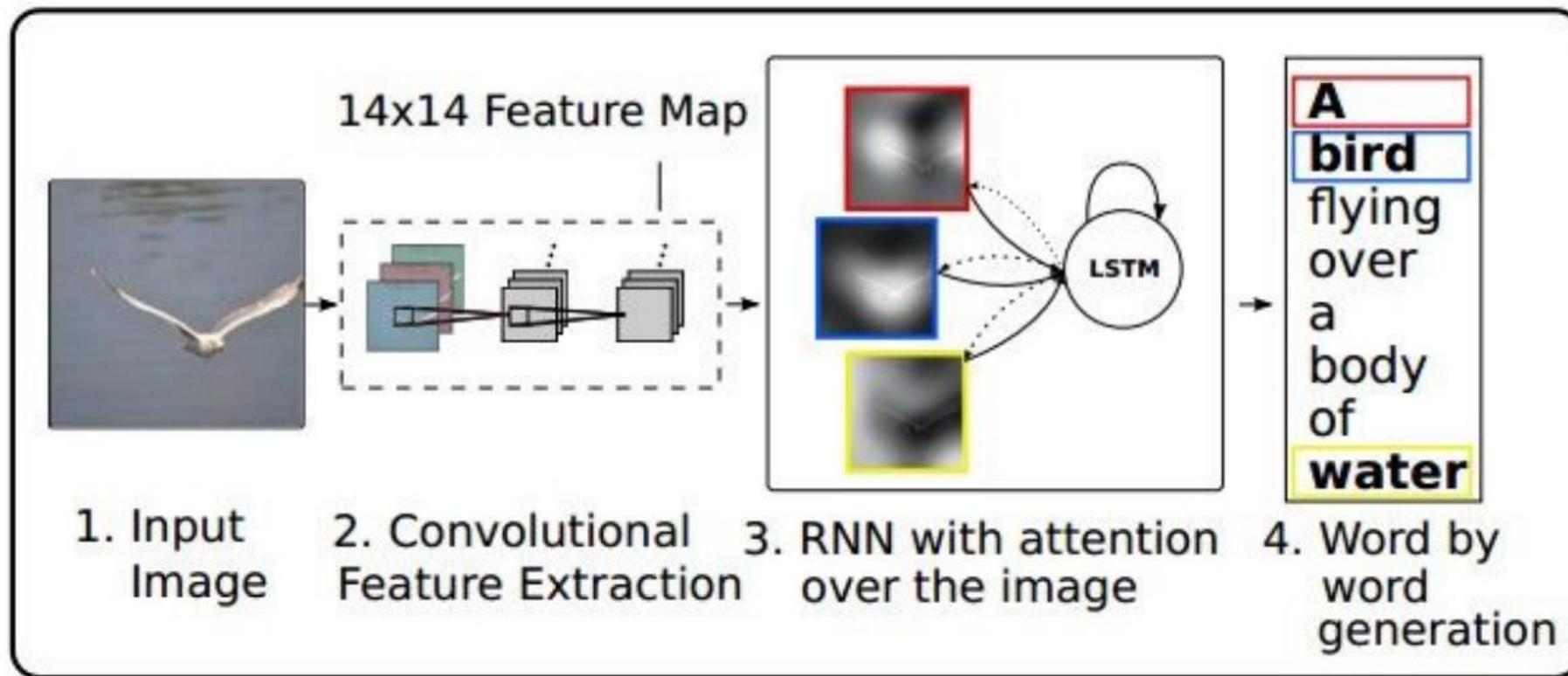


Attention. Image Captioning. Схема работы



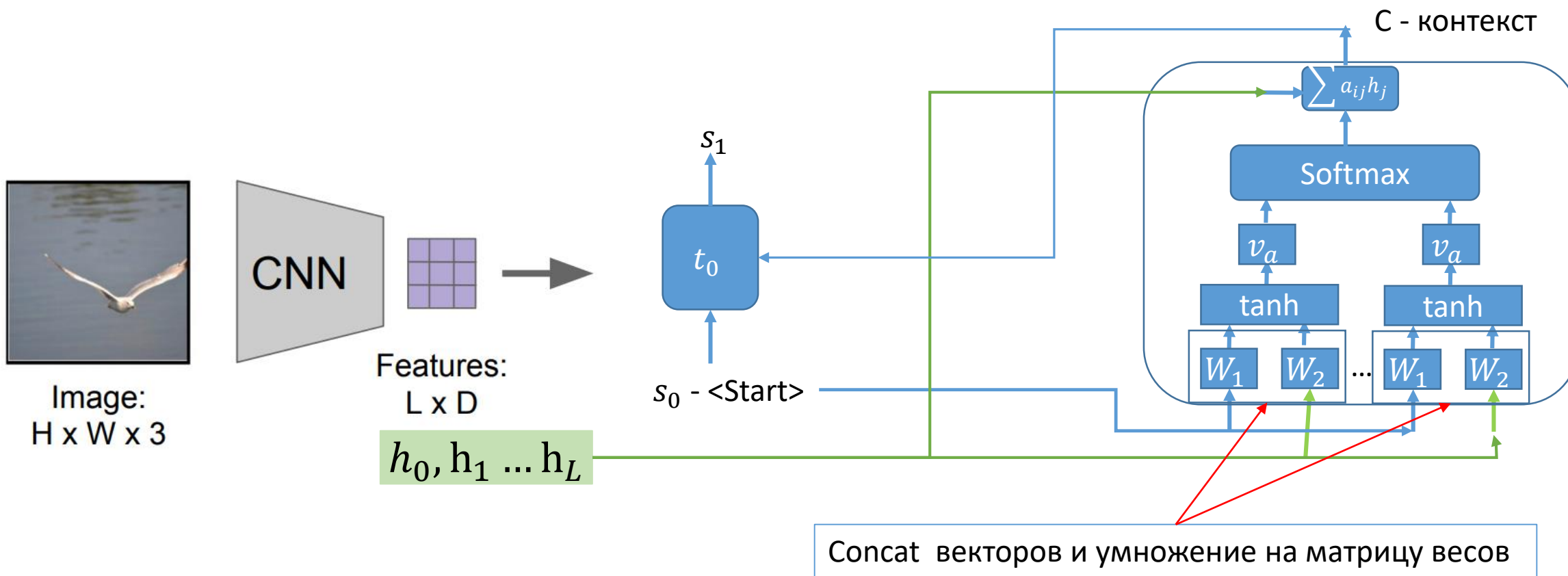
Attention. Image Captioning. Improve

RNN фокусирует свое внимание на различных областях изображения когда генерирует следующее слово

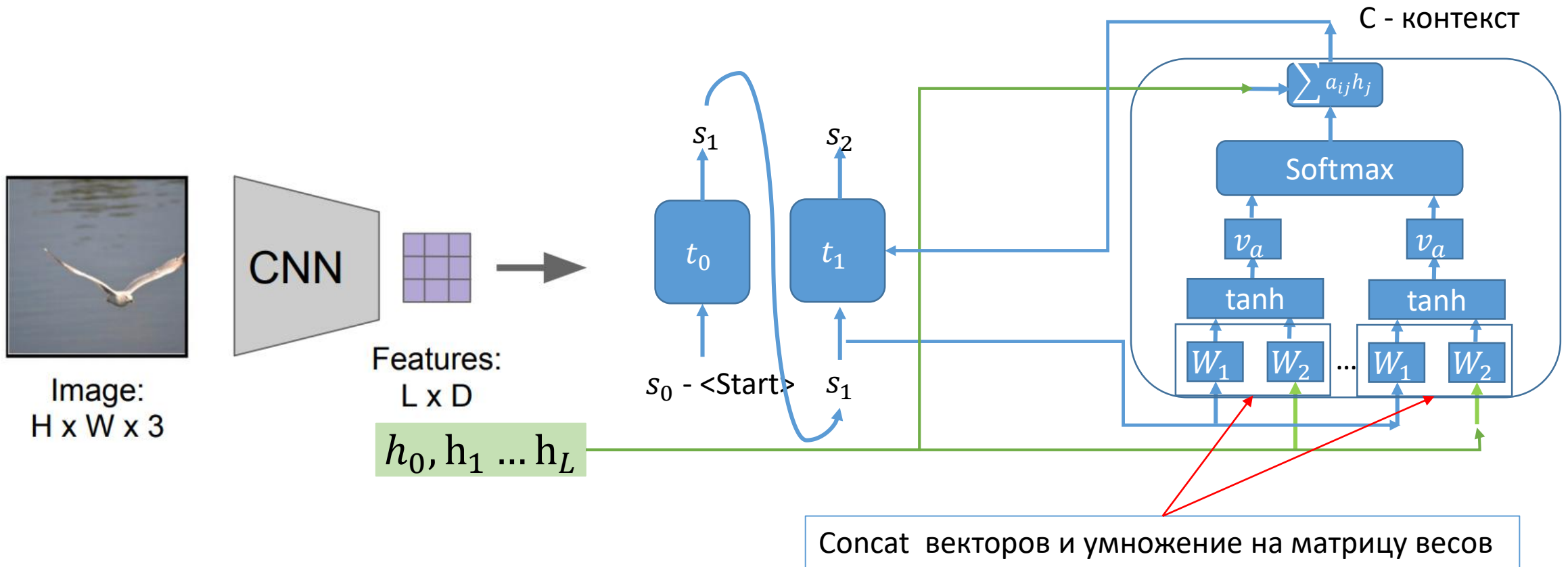


Xu et al, "Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

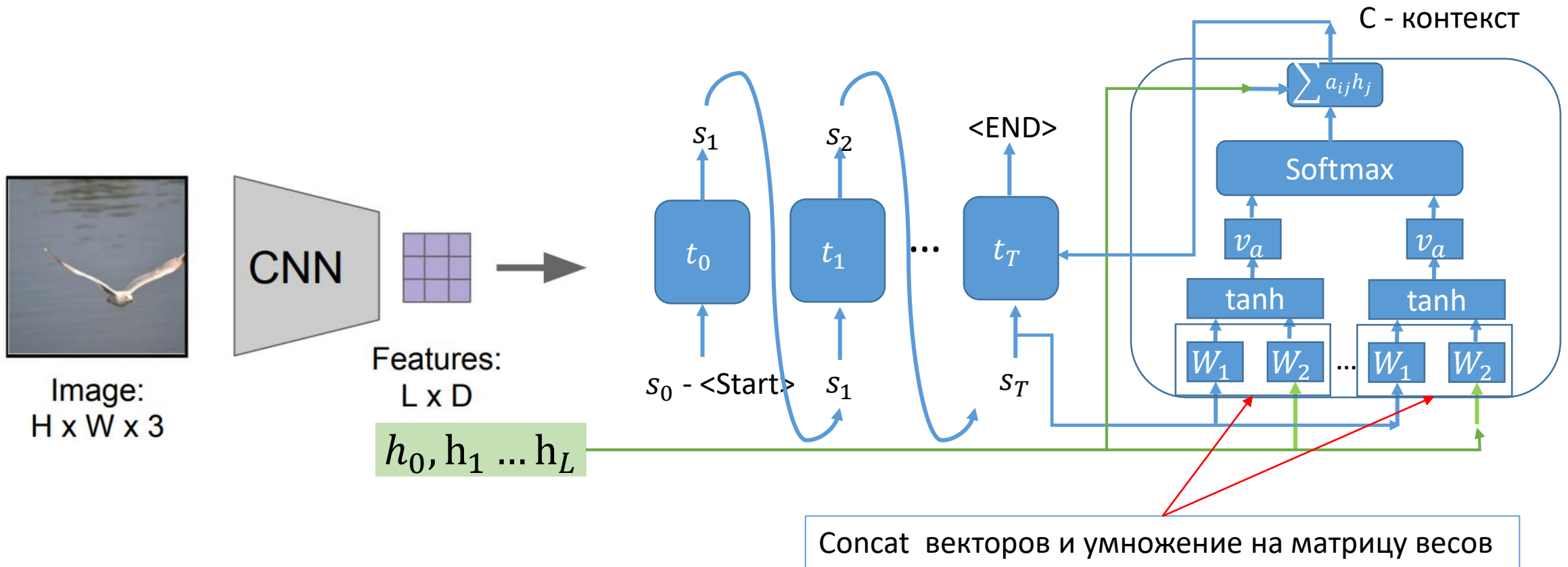
Attention. Image Captioning



Attention. Image Captioning



Attention. Image Captioning



Attention. Image Captioning. Results



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



A stop sign is on a road with a mountain in the background.



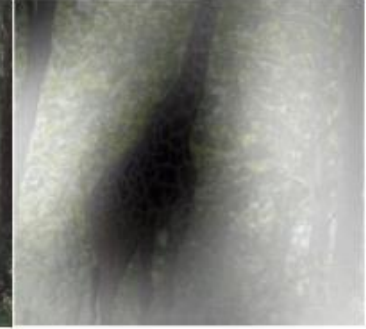
A little girl sitting on a bed with a teddy bear.



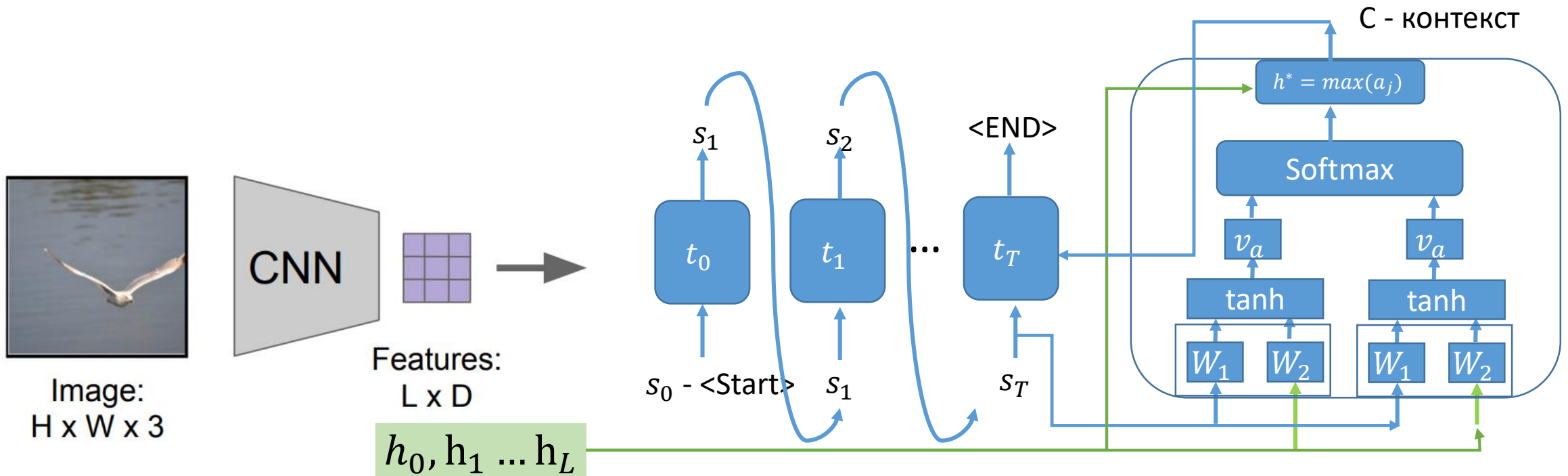
A group of people sitting on a boat in the water.



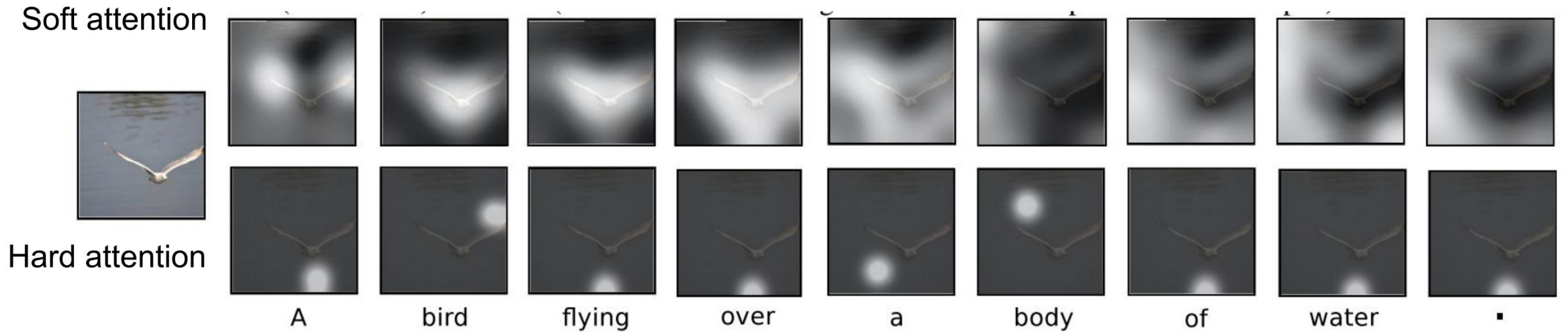
A giraffe standing in a forest with trees in the background.



Attention. Image Captioning. Hard Attention



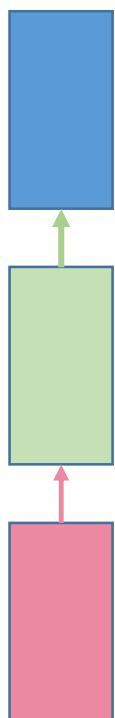
Attention. Image Captioning. Hard Attention



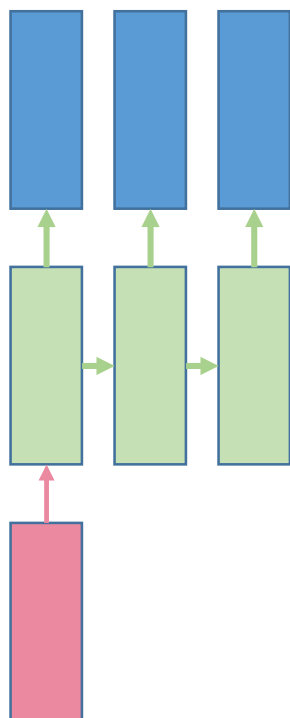
Xu et al, "Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Seq2Seq

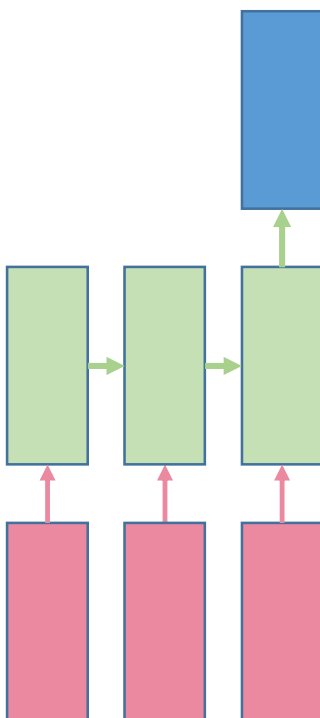
Один к одному



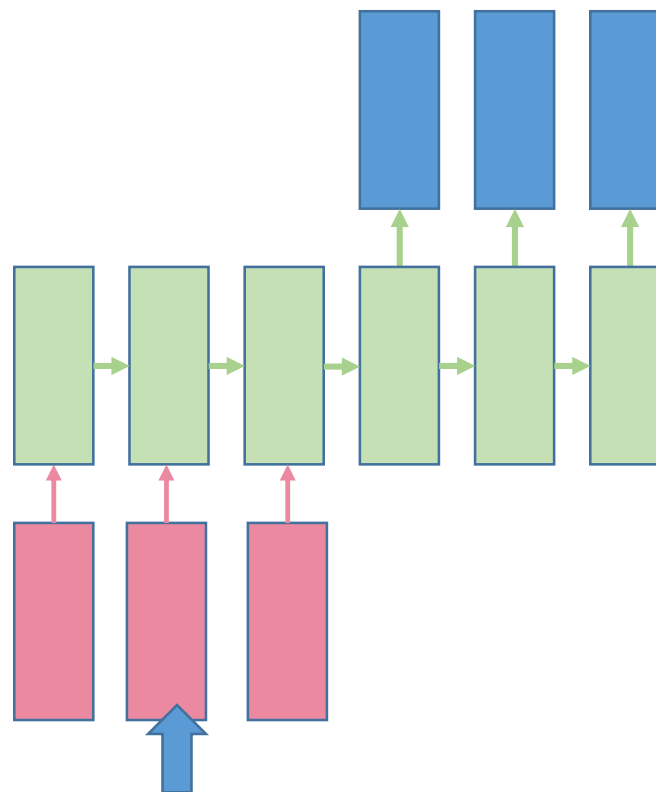
Один ко многим



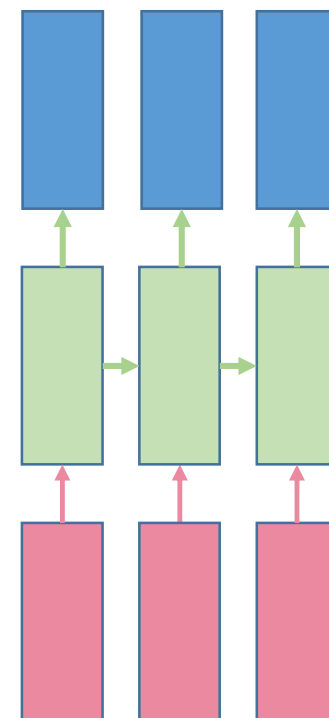
Многие к одному



Многие ко многим

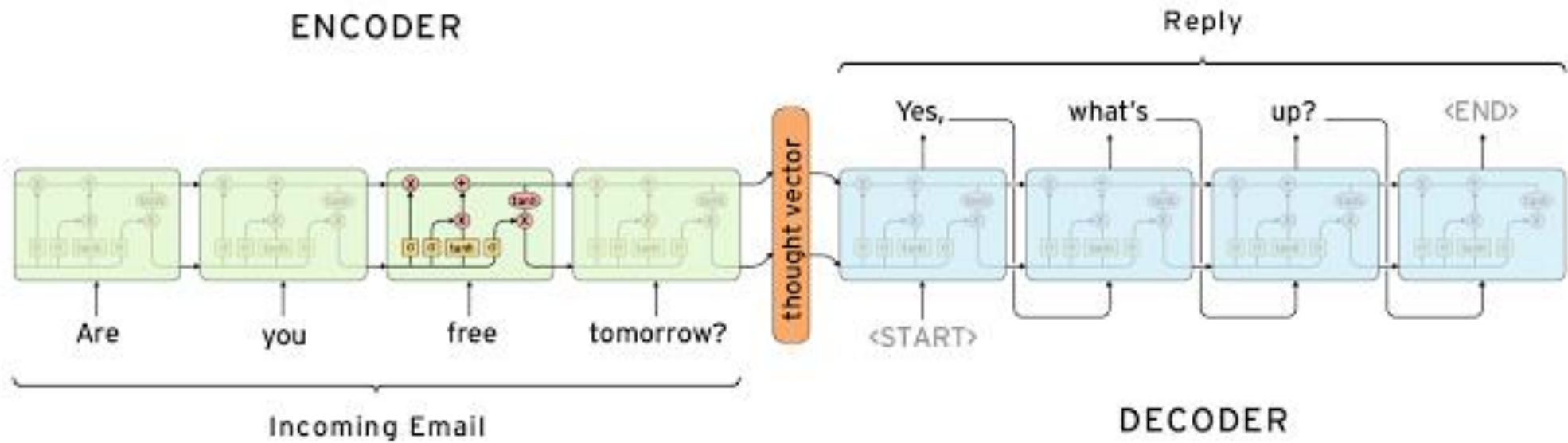


Многие ко многим

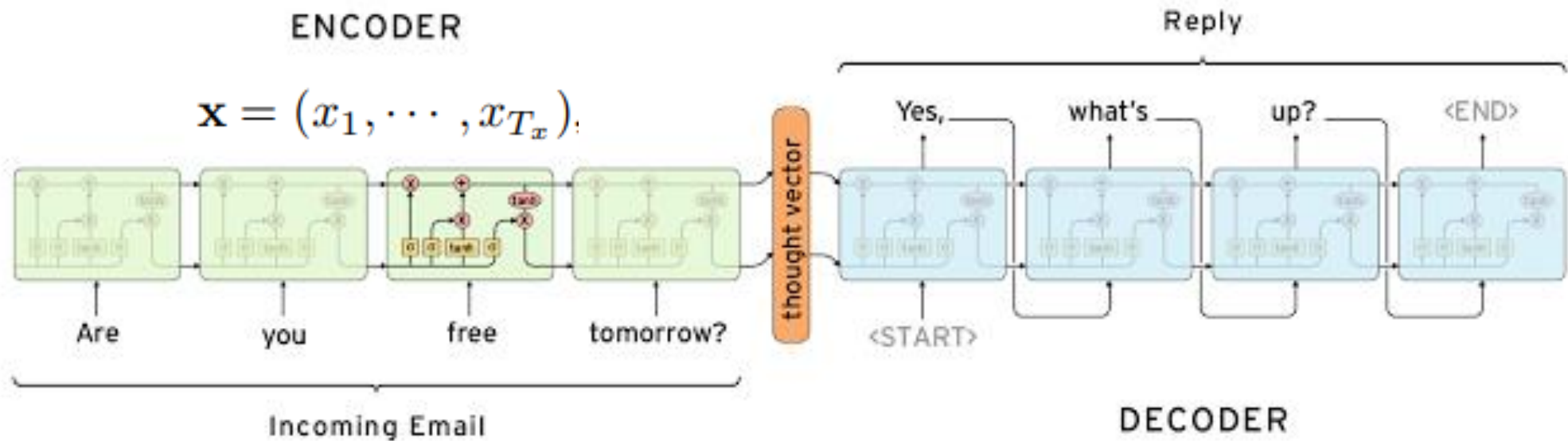


Переводчик, чат боты
Вход – текст. Выход - текст

Seq2Seq

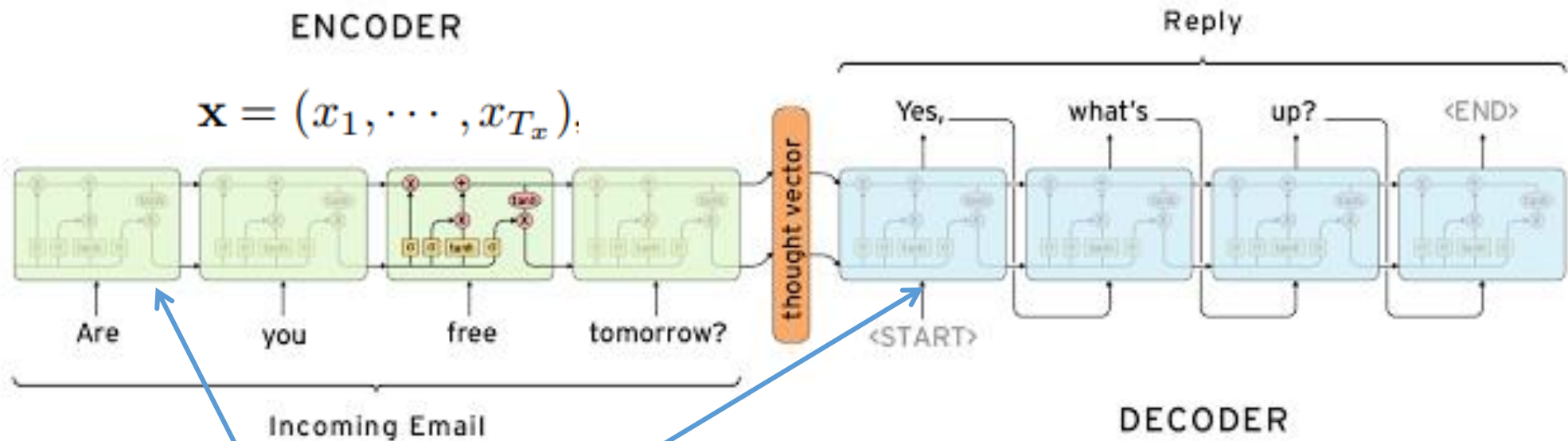


Seq2Seq



Входная последовательность,
развернутая во времени

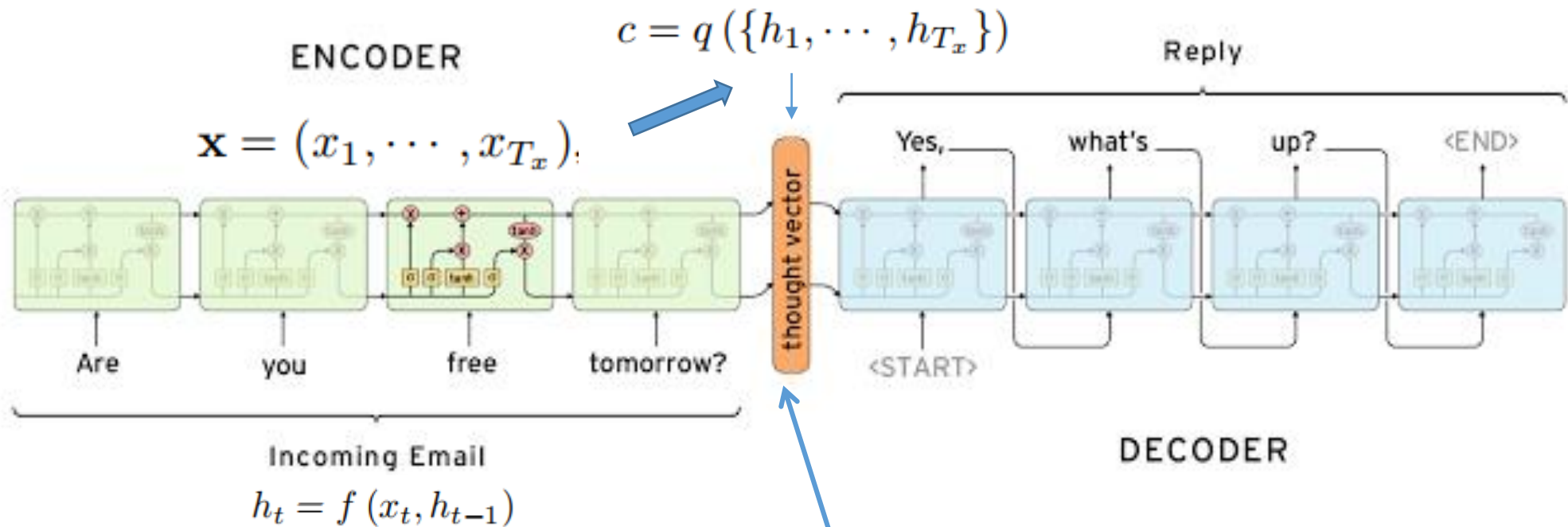
Seq2Seq



RNN (LSTM) - блок

$$h_t = f(x_t, h_{t-1})$$

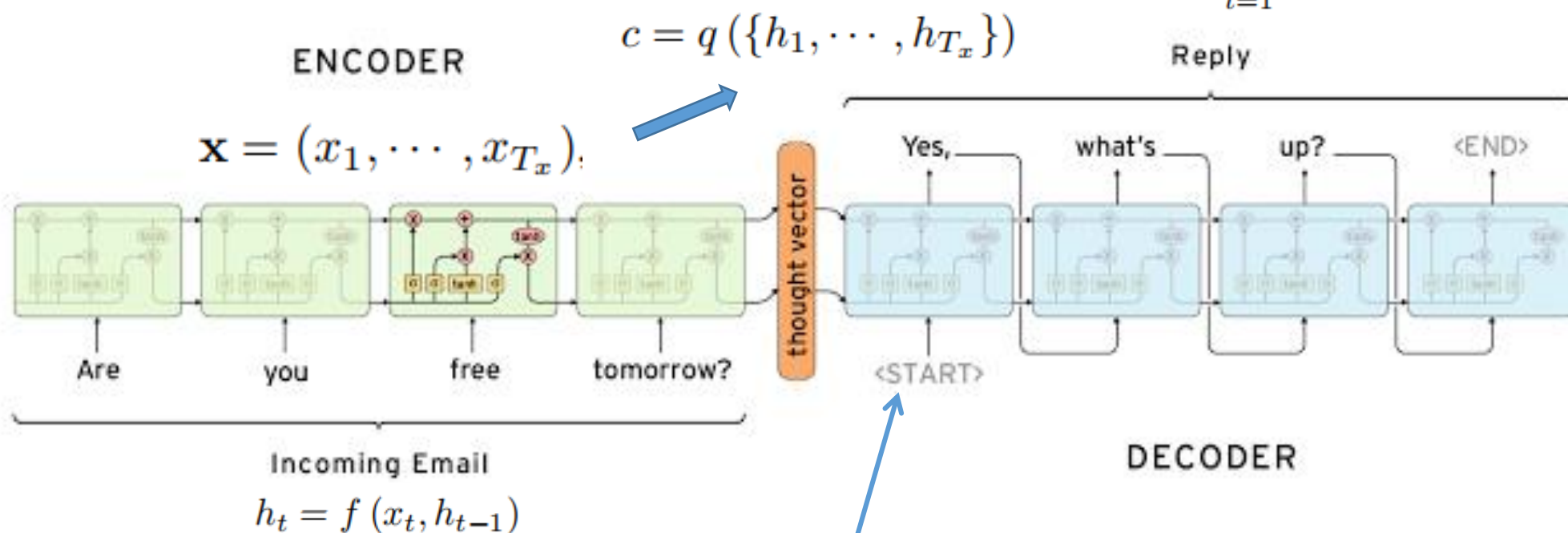
Seq2Seq



Последнее состояние RNN агрегирует
информацию о входной
последовательности

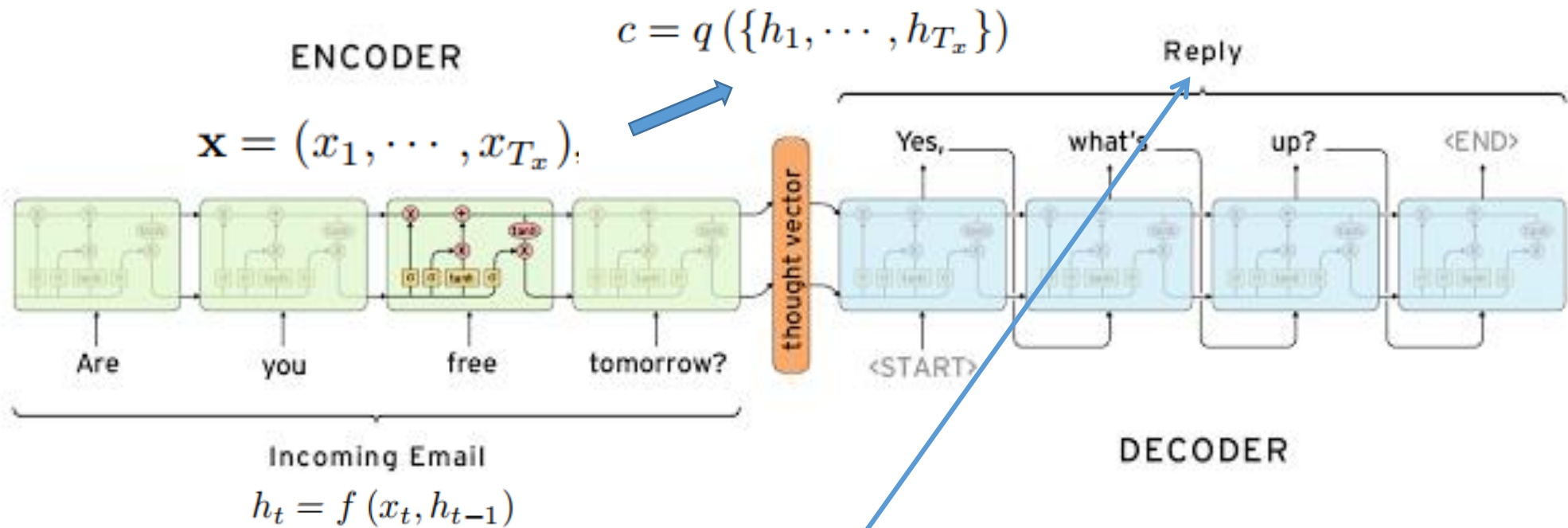
Seq2Seq

$$p(\mathbf{y}) = \prod_{t=1}^T p(y_t \mid \{y_1, \dots, y_{t-1}\}, c),$$



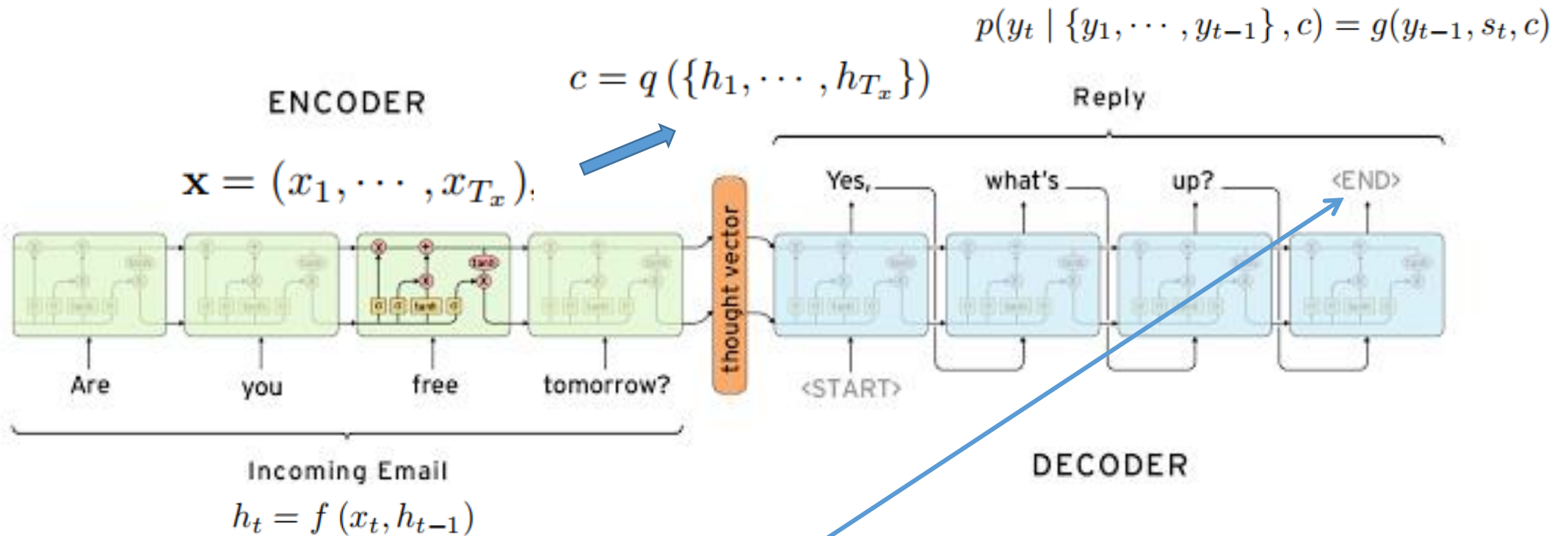
Подаем начальный символ генерации
ответа

Seq2Seq



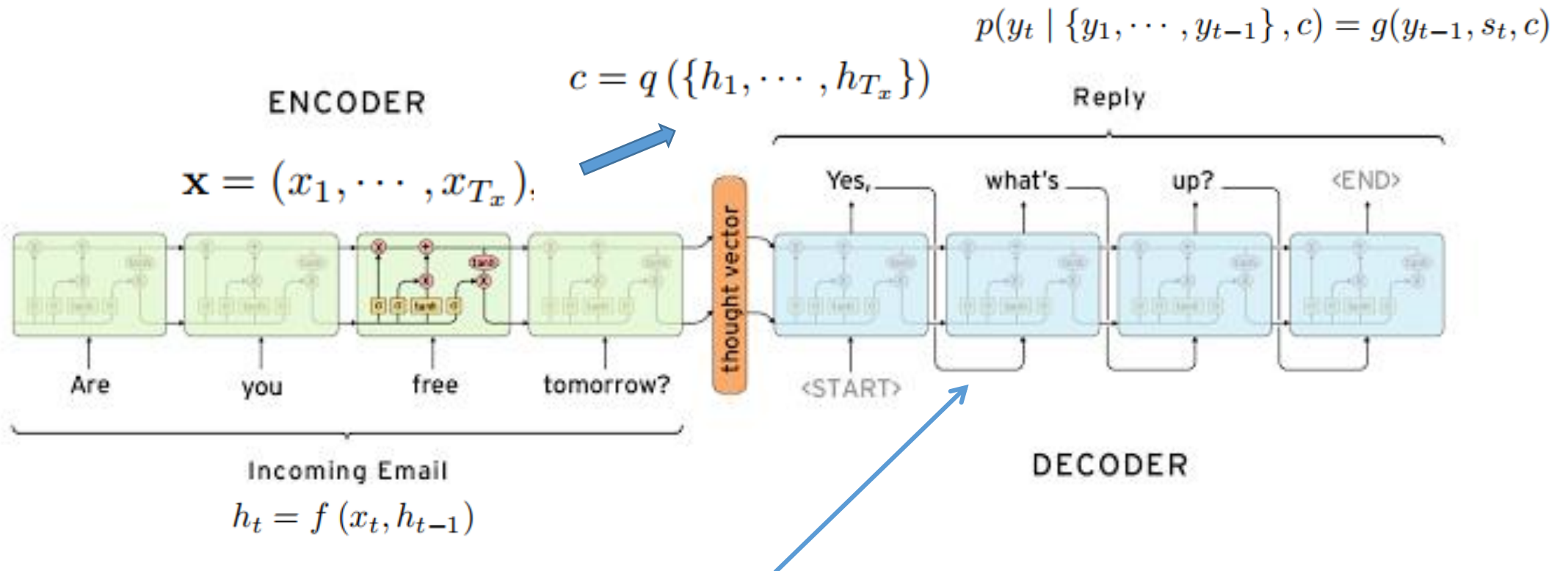
Получаем последовательность ответа

Seq2Seq



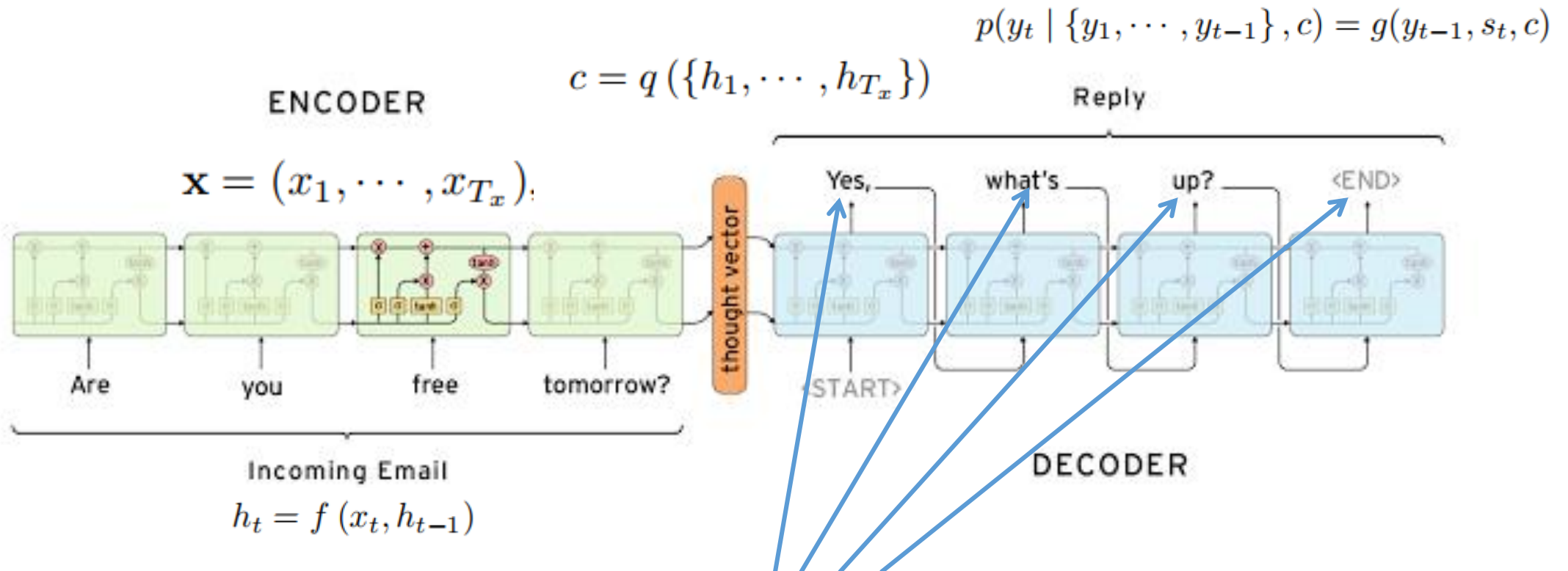
Декодер должен сгенерировать состояние
"Конец последовательности"

Seq2Seq



В процессе обучения на каждой итерации
подаем на вход RNN правильное слово на
шаге t-1

Seq2Seq



Считаем ошибку как сумму ошибок на
каждом шаге генерации
последовательности и пропускаем градиент

Seq2Seq

- Проблема: При генерации ответа нужна не только скрытая информация, но и контекстная информация запроса.
- Выход: при генерации ответа смотреть на слова, которые необходимы для создания ответа.
 - Куда мы пойдем завтра -> нужно смотреть на слово “завтра”
 -
- Можно заставить сеть выучить вероятностное распределение над входной последовательностью и использовать скрытые состояния энкодера через взвешенную сумму

Seq2Seq. Attention

- Расширенная модель внимания представляет декодер как:

$$p(y_i | y_1, \dots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_i, c_i)$$

где s_i - это скрытое состояние RNN $s_i = f(s_{i-1}, y_{i-1}, c_i)$

- В отличие от модели энкодер-декодер вероятность обуславливается контекстным вектором c_i для каждого целевого слова y_{i-1}
- Контекстный вектор c_i зависит от аннотаций $(h_1 \dots h_T)$ на которые энкодер мапирует входную последовательность

Seq2Seq. Attention

- Каждая аннотация h_i содержит информацию о всей последовательности и фокус на часть слов, окружающих i -е слово
- Контекст-вектор считается как взвешенная сумма аннотаций

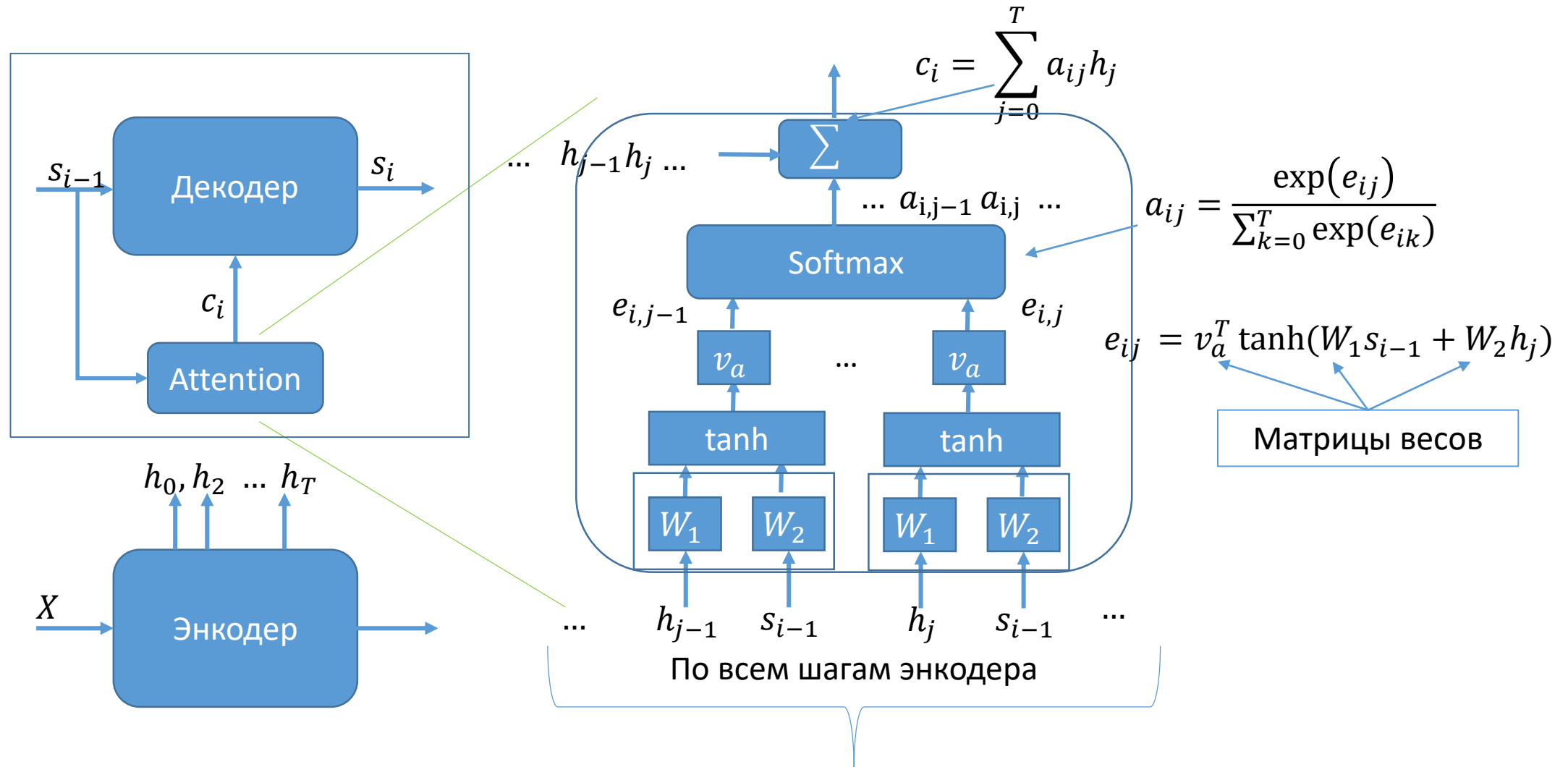
$$c_i = \sum_{j=0}^T a_{ij} h_j$$

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=0}^T \exp(e_{ik})} - \text{вес}$$

$$e_{ij} = a(s_{i-1}, h_j) - \text{модель выравнивания}$$

- Выравнивание показывает на сколько подходит вход с позиции j к выходу позиции i .

Seq2Seq. Attention



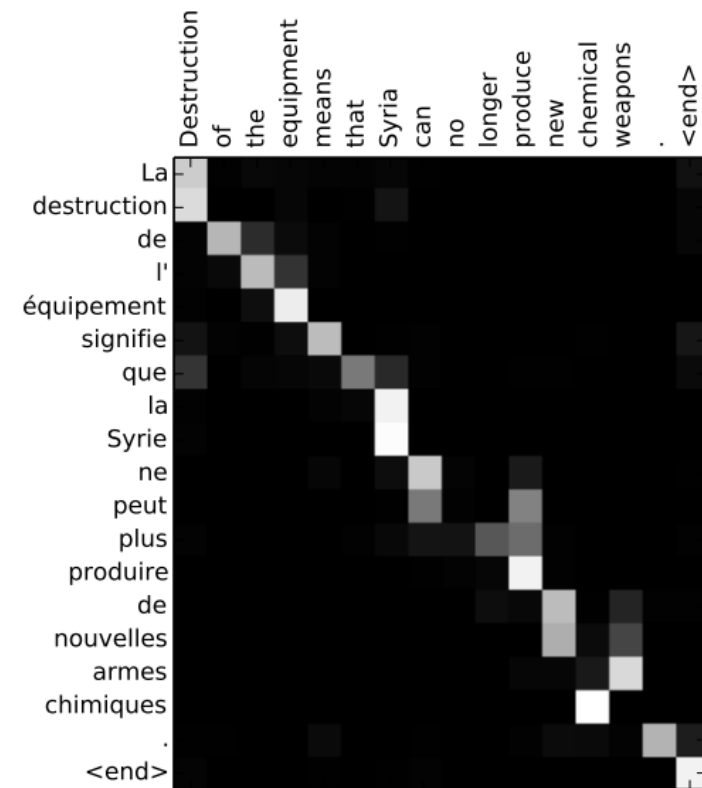
Seq2Seq. Attention

- Выходы $\{a\}$ softmax можно объединить в матрицу A размером $T_x \times T_y$

• $A = \begin{pmatrix} a_{00} & \cdots & a_{0T_y} \\ \vdots & \ddots & \vdots \\ a_{0T_x} & \cdots & a_{T_y T_x} \end{pmatrix}$

Декодер $\xrightarrow{\hspace{10em}}$

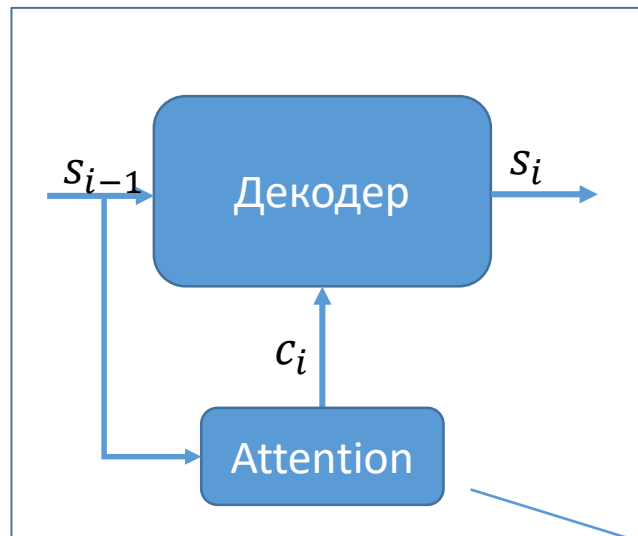
\downarrow Энкодер



Seq2Seq. Attention

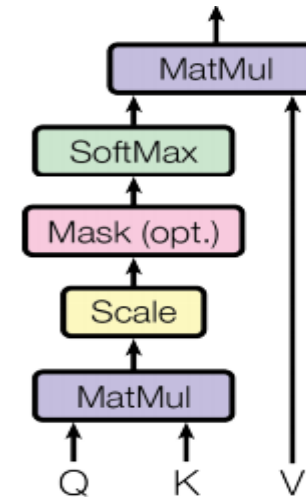
- Bahdanau Attention – достаточно громоздкий в вычислениях
- На каждый шаг декодера требуется прогон через матрицу весов Attention RNN всей последовательности энкодера
- Можно ли упростить?

Seq2Seq. Attention Is All You Need



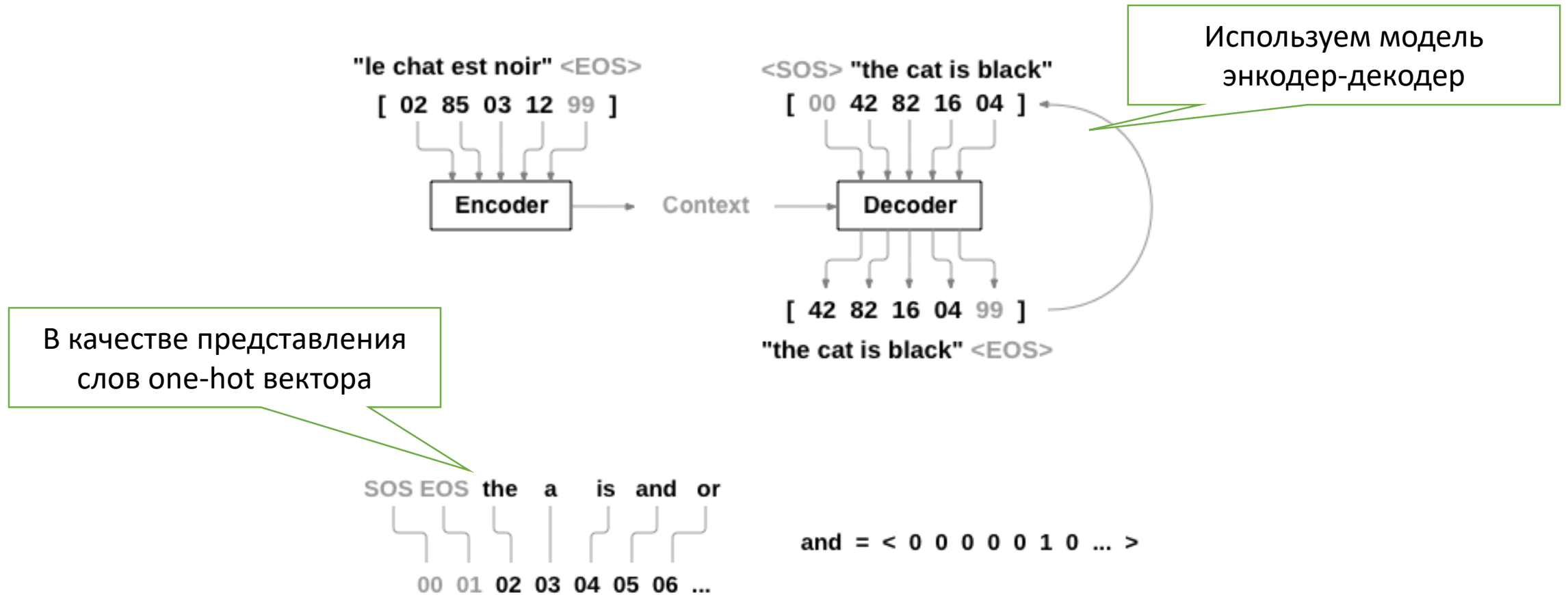
s_{i-1} — запрос к механизму внимания Q
 h_i — представляет пару ключ K значение V

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



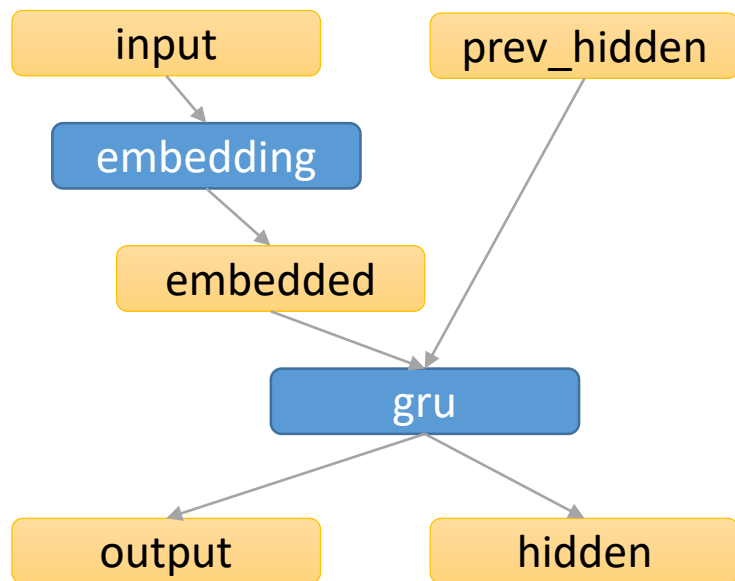
[Attention Is All You Need]

Seq2Seq. Практика

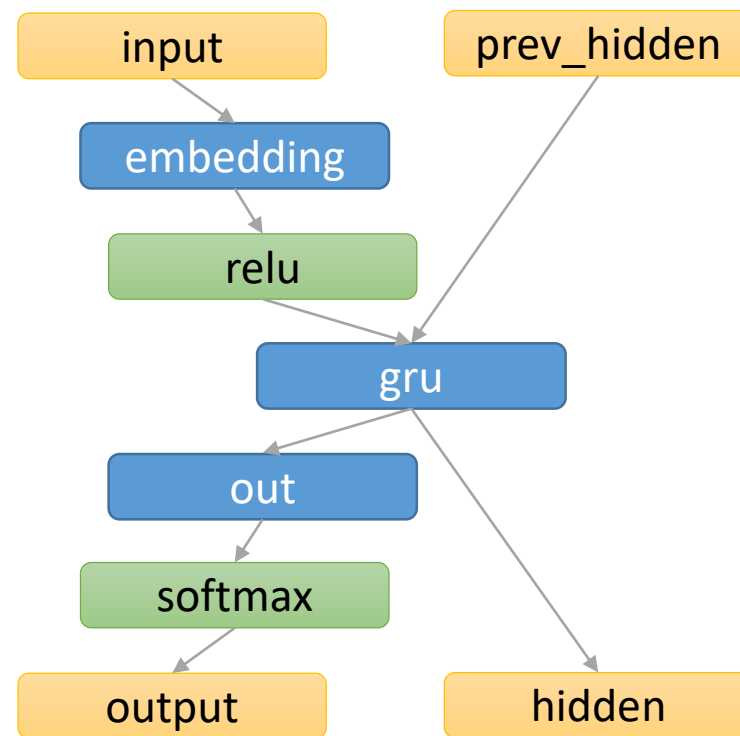


Seq2Seq. Практика

Энкодер

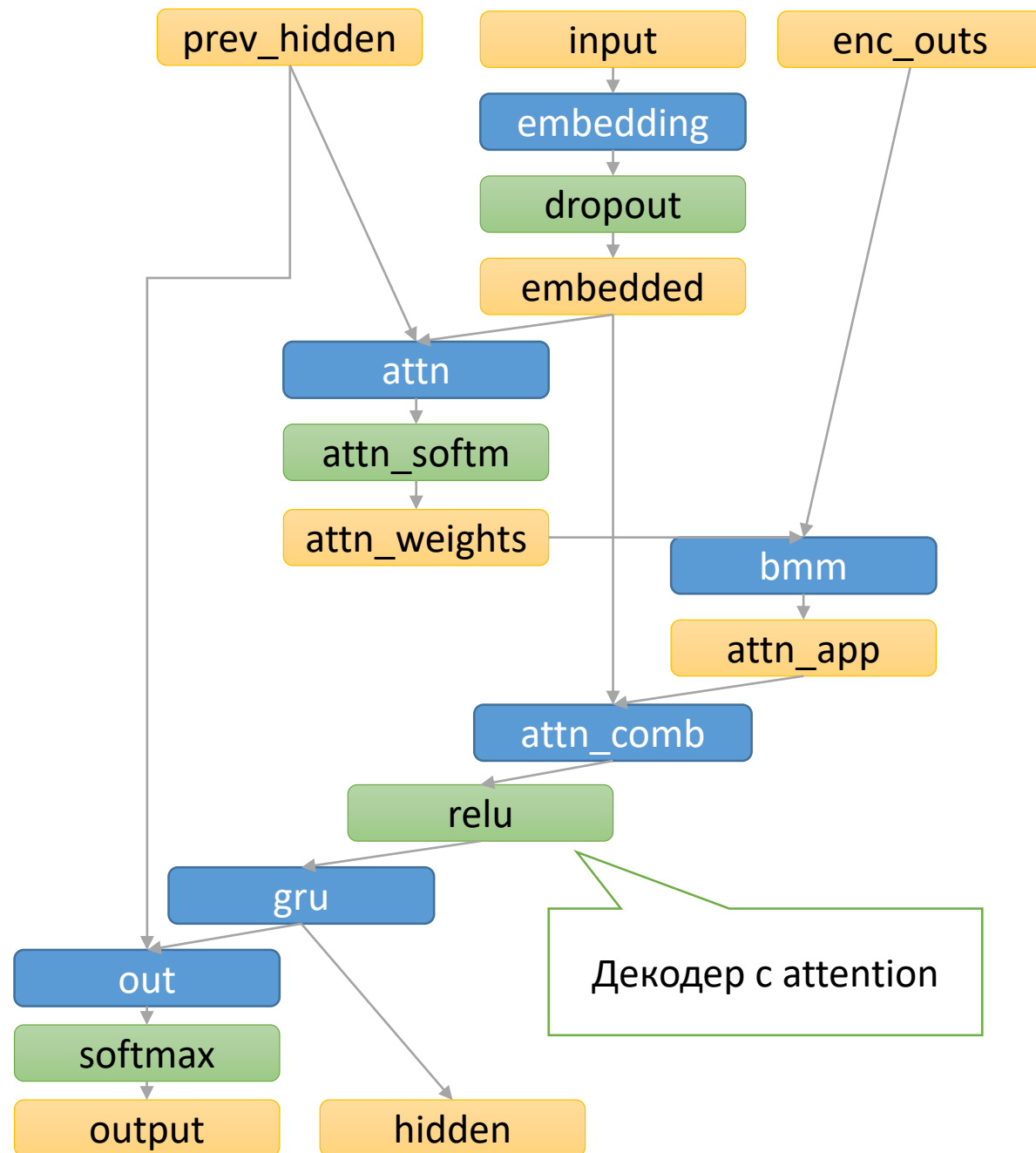
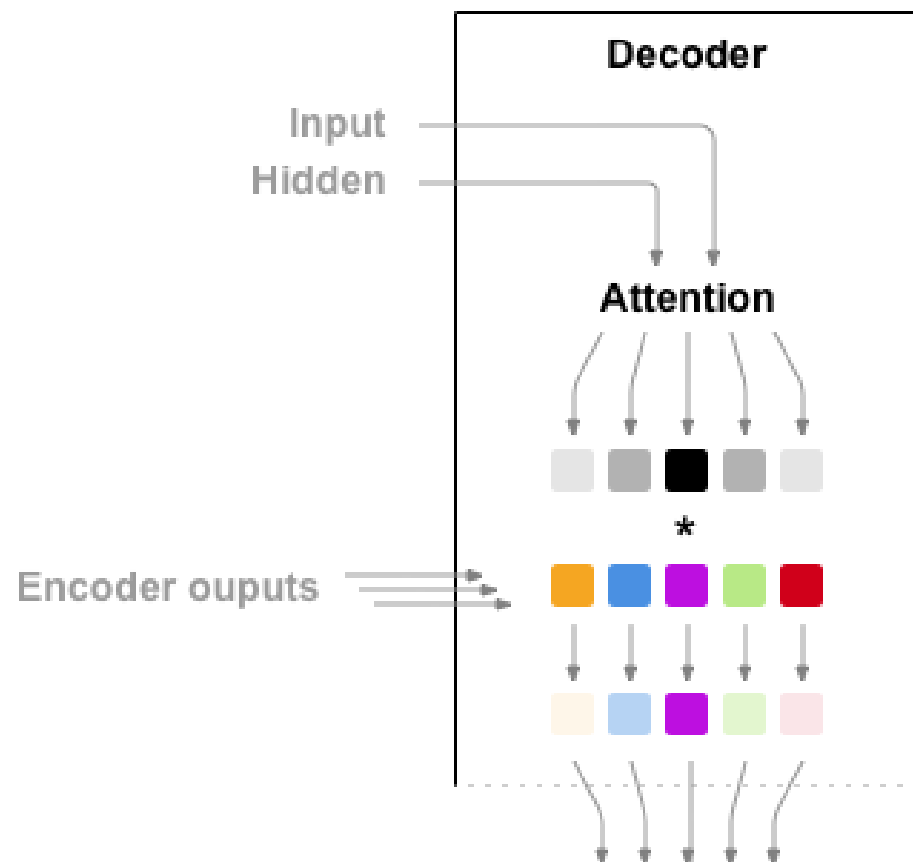


Декодер



Seq2Seq. Практика

Схема работы attention



Seq2Seq. NMT. ДЗ

1. Берем тушку из ноутбука практики Seq2Seq.ipynb
2. Вместо заменяем attention на любой из статьи : Effective Approaches to Attention-based Neural Machine Translation
<https://arxiv.org/pdf/1508.04025.pdf>