



Нейросетевые языковые модели

Евгений Борисов

Нейросетевые языковые модели

Языковая модель

- предсказываем следующее слово на основе предыдущих
- оценка (вероятность) совместимости цепочки слов

Оценка цепочки слов (биграммная модель):

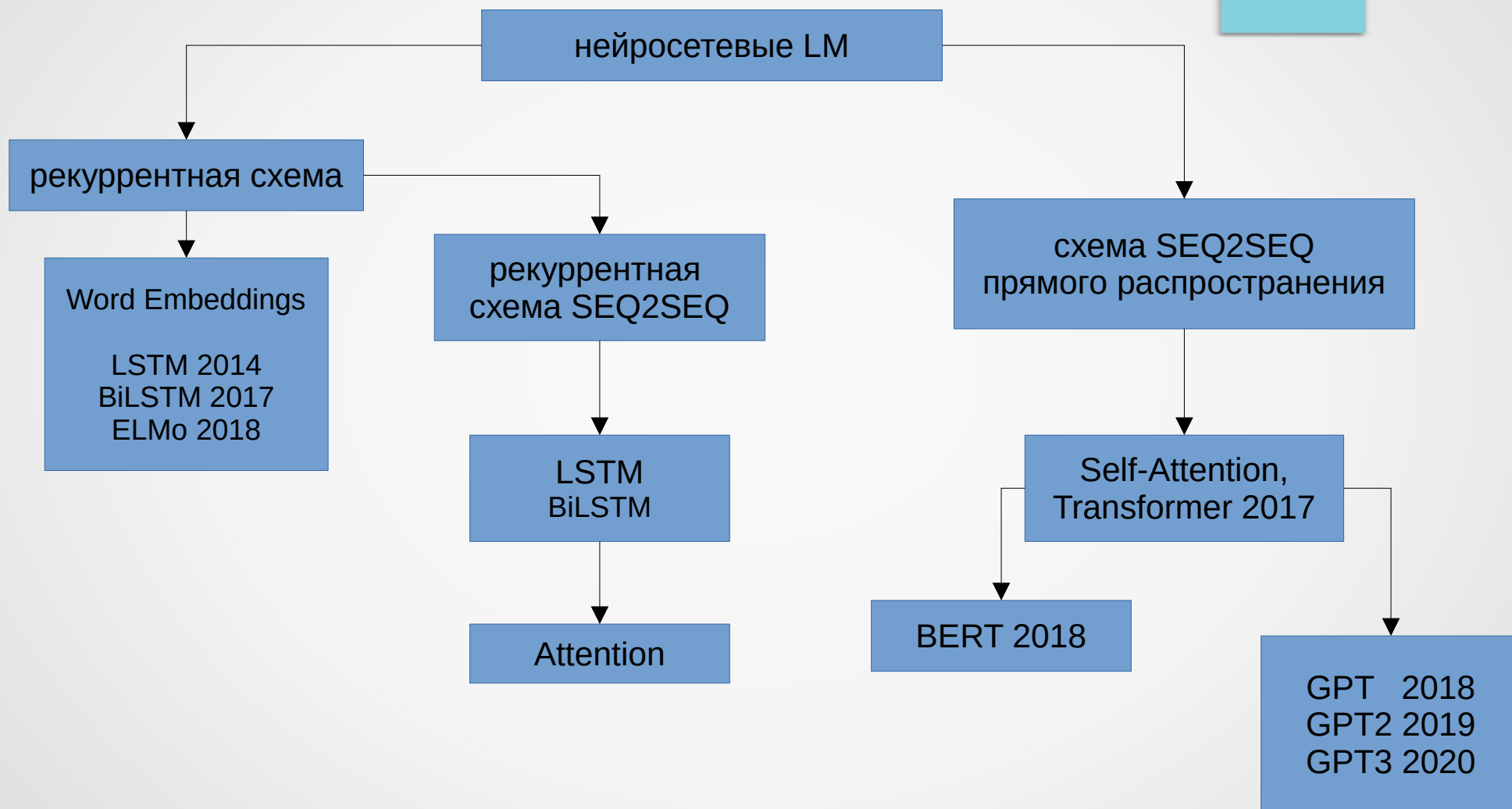
$$p(w_1 \dots w_n) = \prod_{k=1}^n p(w_k | w_{k-1})$$

$$p(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{C(w_{n-1})}$$

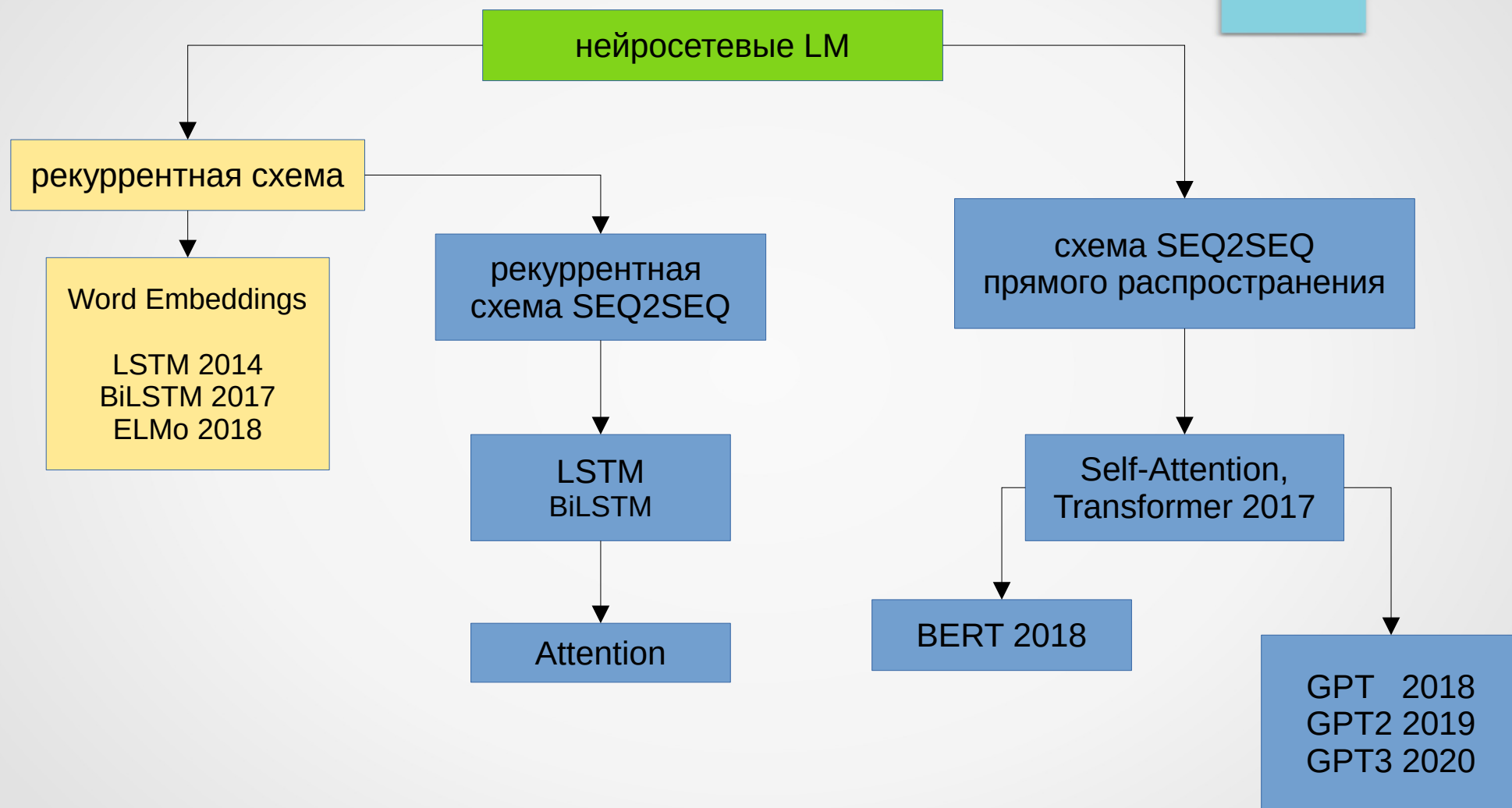
P - вероятность совместного использования слов

C(w) – количество слов w в тексте

Нейросетевые языковые модели



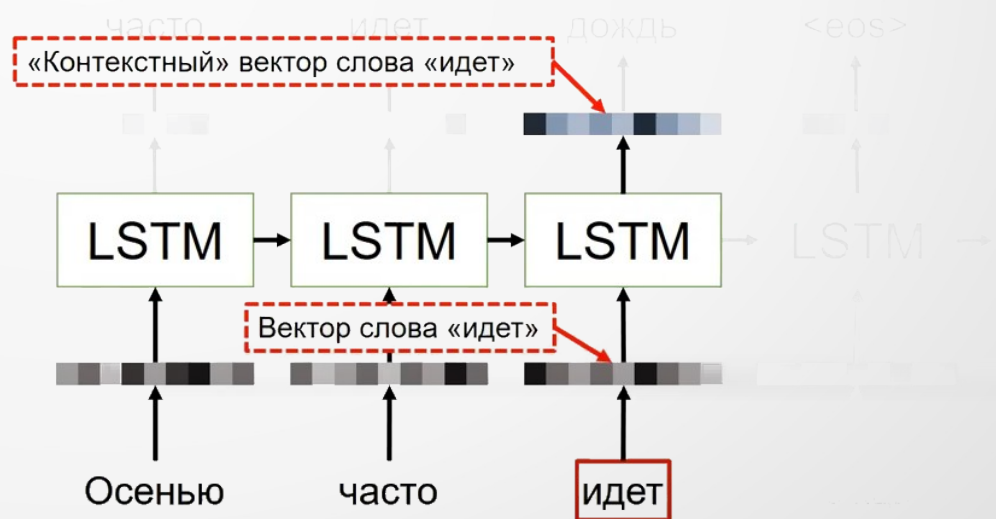
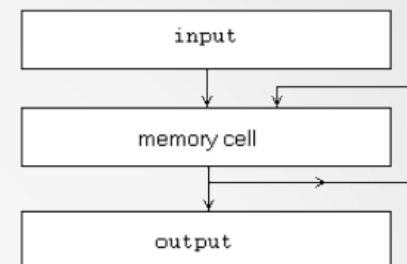
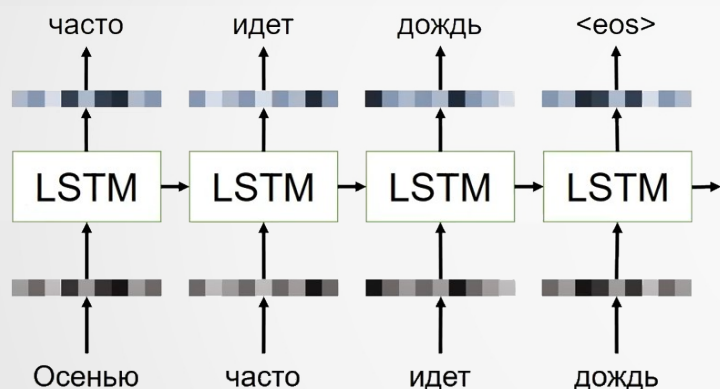
Нейросетевые языковые модели



Нейросетевые языковые модели

Простая схема с рекуррентной сетью

Предсказываем следующее слово по предыдущему контексту
получаем word embedding, который учитывает левый контекст



Нейросетевые языковые модели

Двунаправленная схема

(Peters M. E. et al. Semi-supervised sequence tagging with bidirectional language models - 2017.)

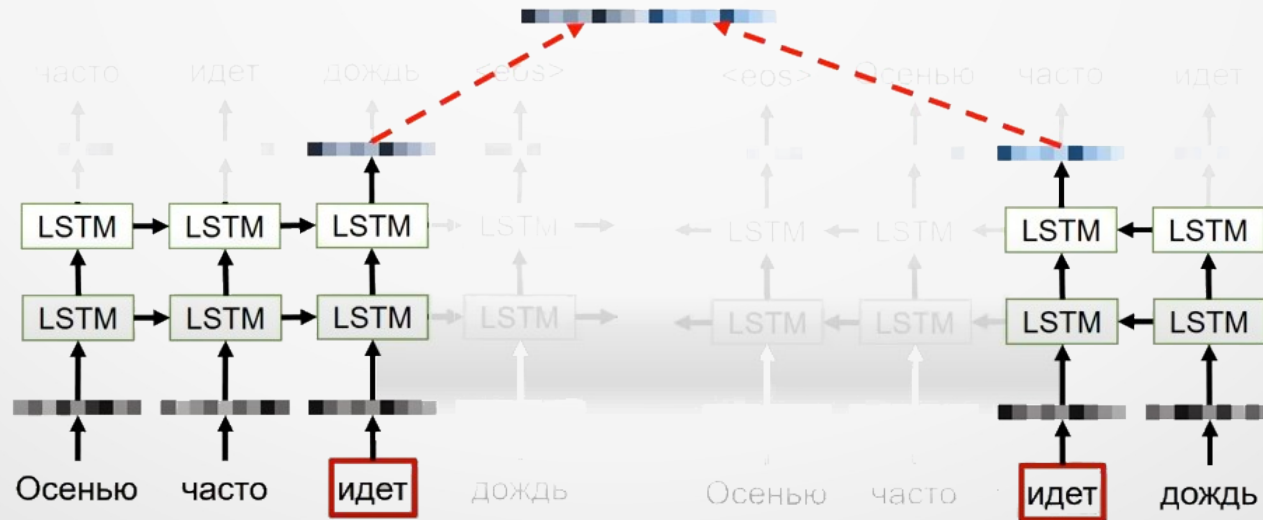
используем две сети (по два слоя) и два контекста - левый и правый

два представления (выхода) агрегируем

получаем word embedding, который учитывает левый и правый контекст

Прямая модель: $p(w_1, \dots, w_N) = \prod_{k=1}^N p(w_k | w_1, \dots, w_{k-1})$

Обратная модель: $p(w_1, \dots, w_N) = \prod_{k=1}^N p(w_k | w_{k+1}, \dots, w_N)$



Нейросетевые языковые модели

ELMo (Embeddings from Language Models)

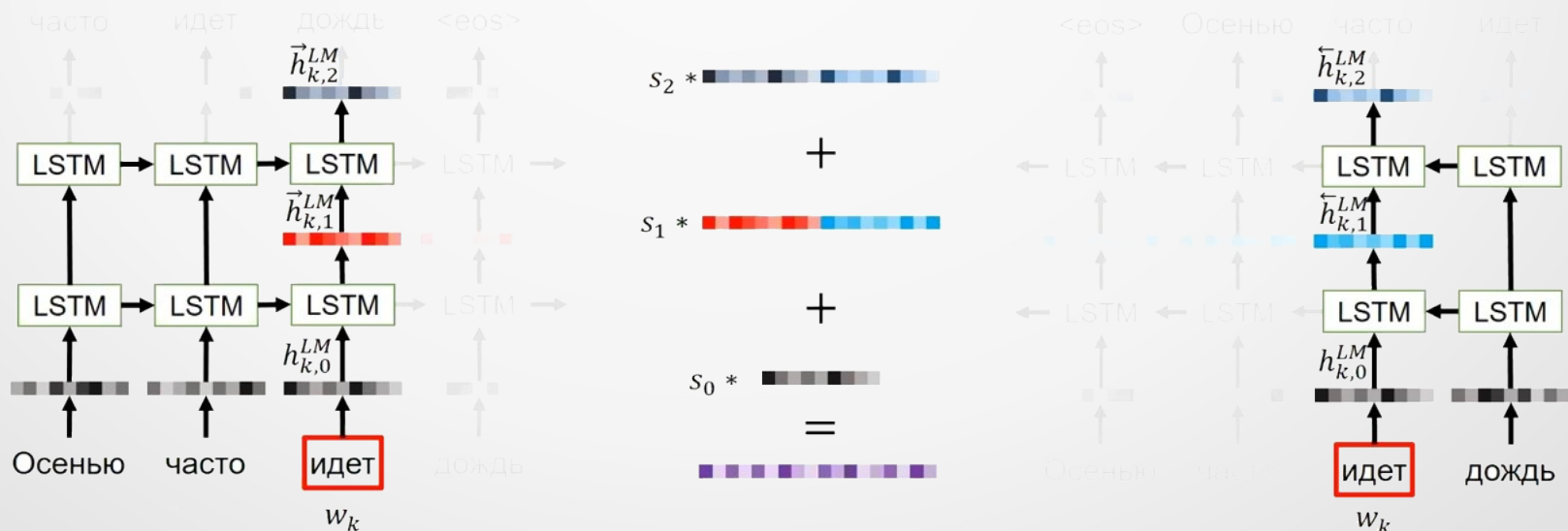
(Peters M. E. et al. Deep contextualized word representations - 2018.)

Обучаем модель на левый и правый контекст одновременно

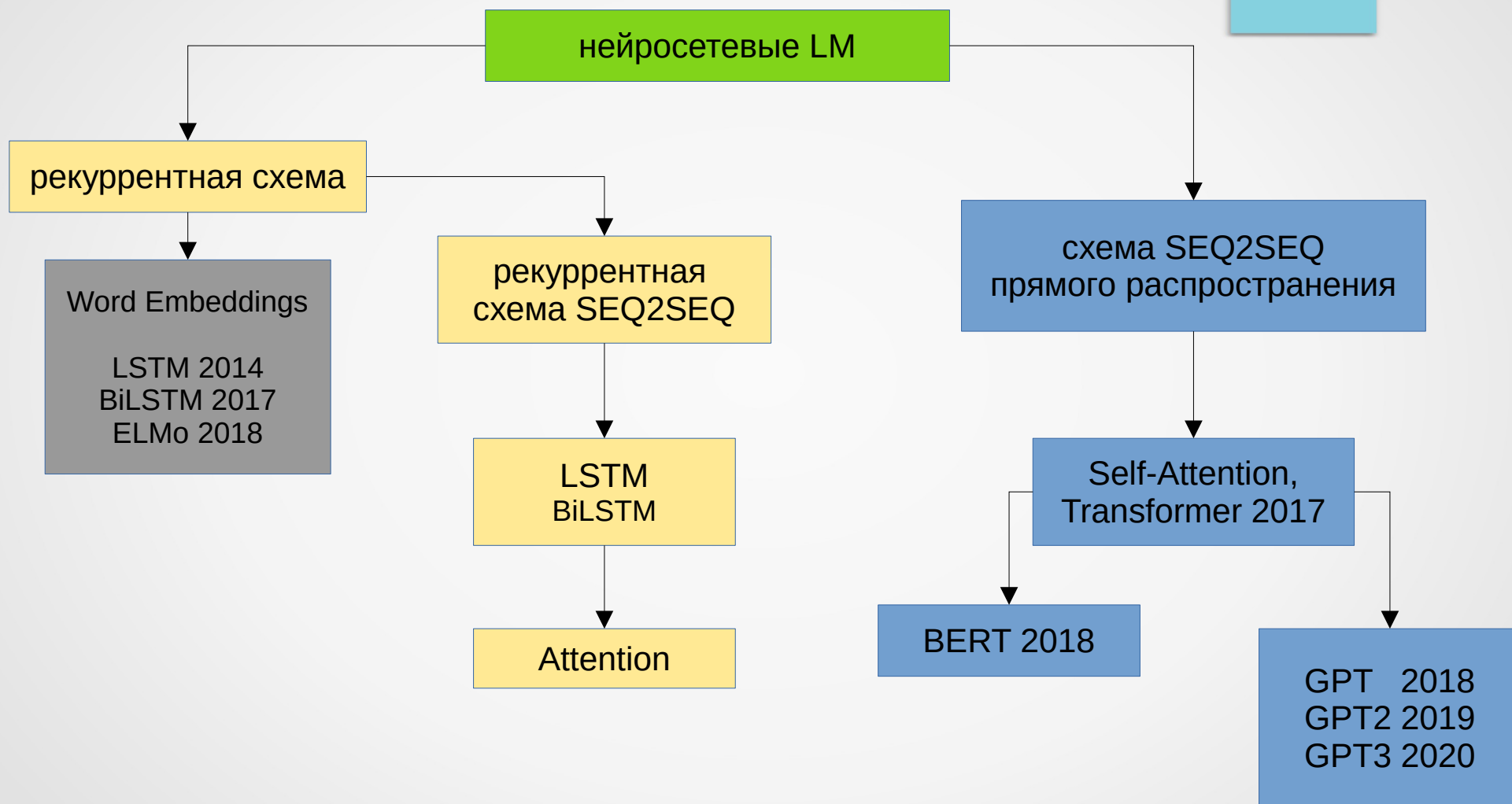
получаем word embedding как агрегированный выход модели -
взвешенная сумма всех промежуточных выходов
на правом и левом контексте

$$J = \sum_{k=1}^N [\log p(w_k | w_1, \dots, w_{k-1}; \Theta_x, \Theta_{\overleftarrow{LSTM}}, \Theta_s) + \log p(w_k | w_{k+1}, \dots, w_N; \Theta_x, \Theta_{\overrightarrow{LSTM}}, \Theta_s)]$$

$$ELMo_k = \gamma^{task} \sum_{j=0}^L s_j^{task} h_{k,j}^{LM}, \text{ где } h_{k,i}^{LM} = [\vec{h}_{k,i}^{LM}; \overleftarrow{h}_{k,i}^{LM}], i = \overline{1, n}$$

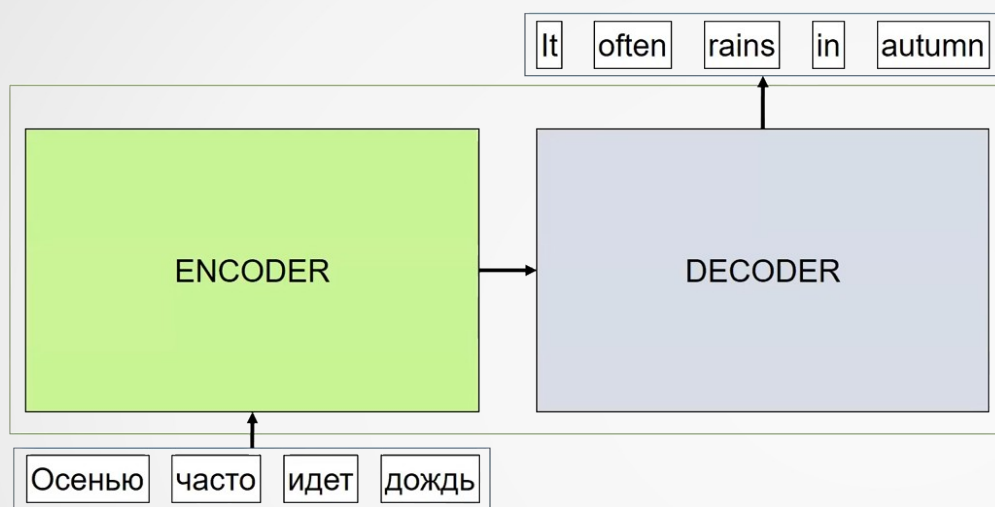


Нейросетевые языковые модели



Нейросетевые языковые модели

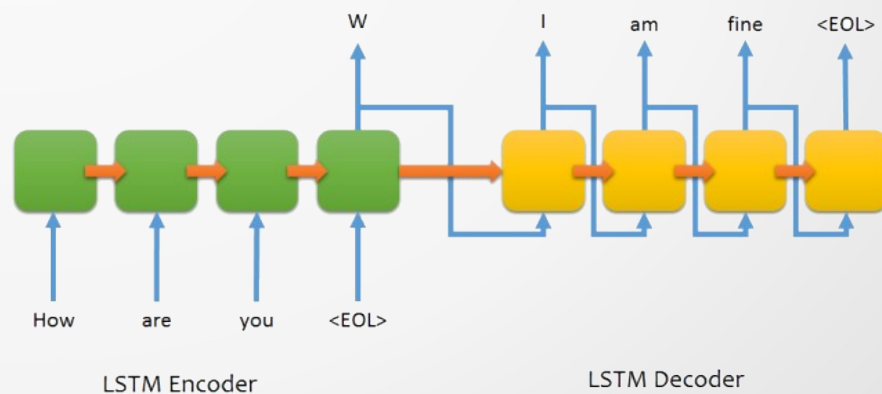
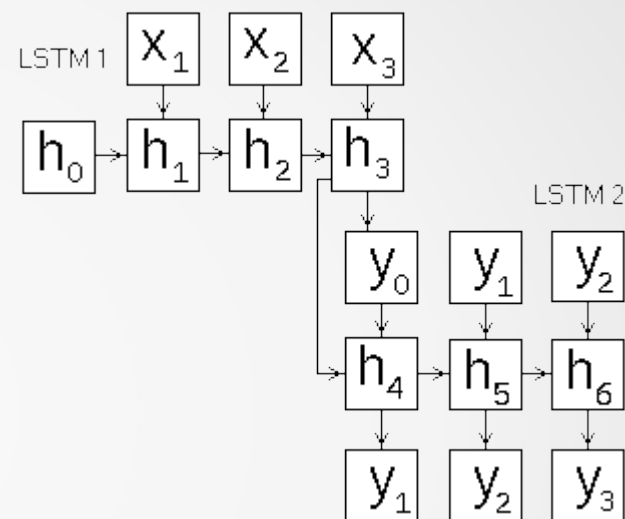
Рекуррентная схема SEQ2SEQ



пара рекуррентных неросетей

- кодировщик,
формирует внутреннее представление

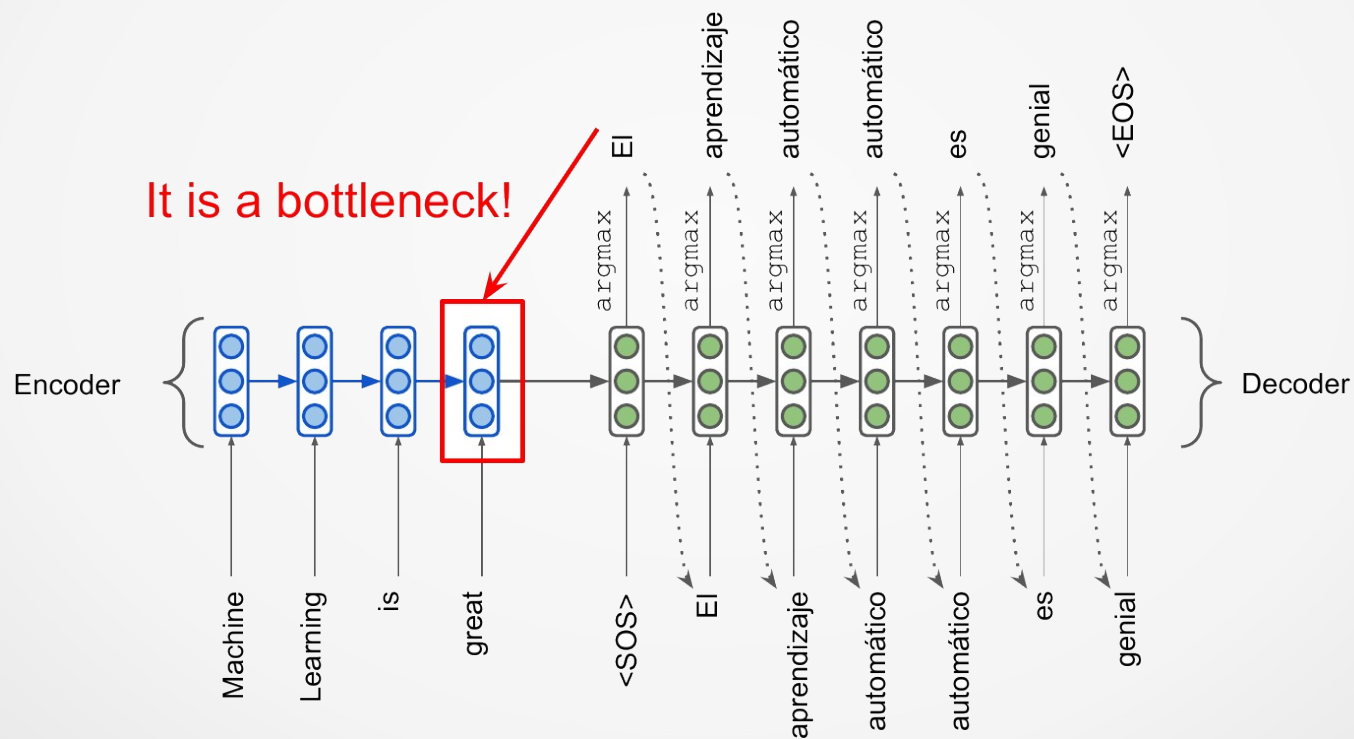
- декодировщик,
авторегрессионная модель,
разворачивает состояние энкодера



Нейросетевые языковые модели

Рекуррентная схема SEQ2SEQ

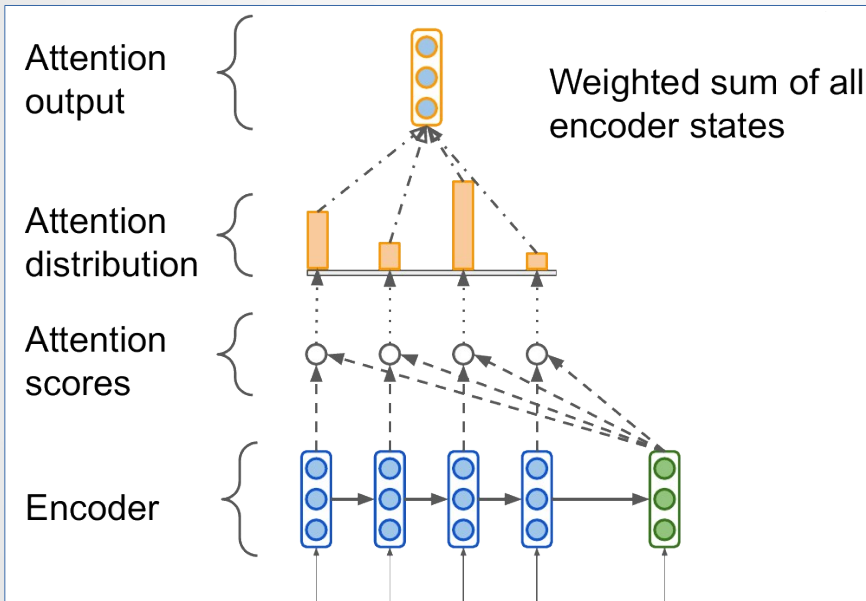
проблема: модель может "забывать" начальный контекст



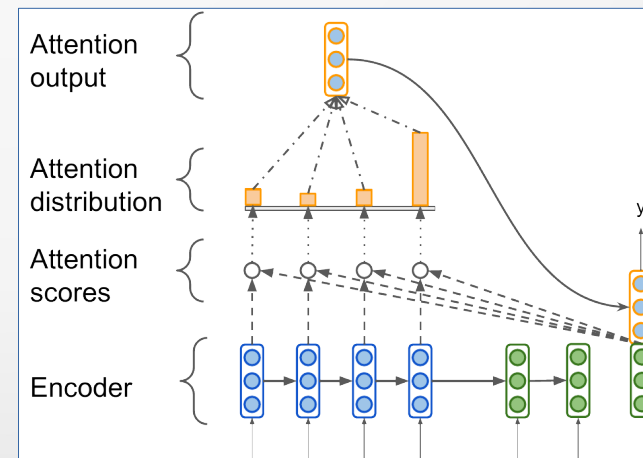
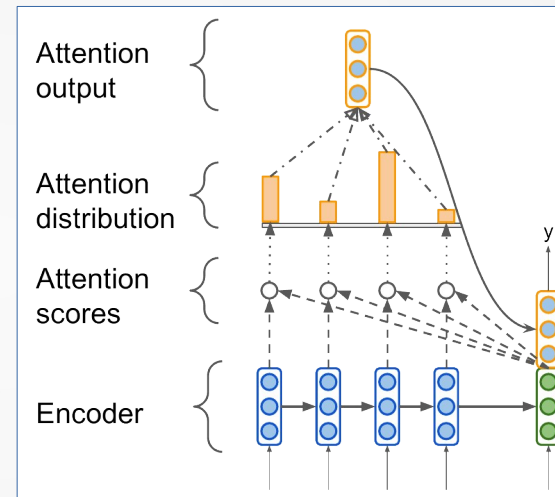
Нейросетевые языковые модели

Рекуррентная схема SEQ2SEQ и механизм внимания (Attention)

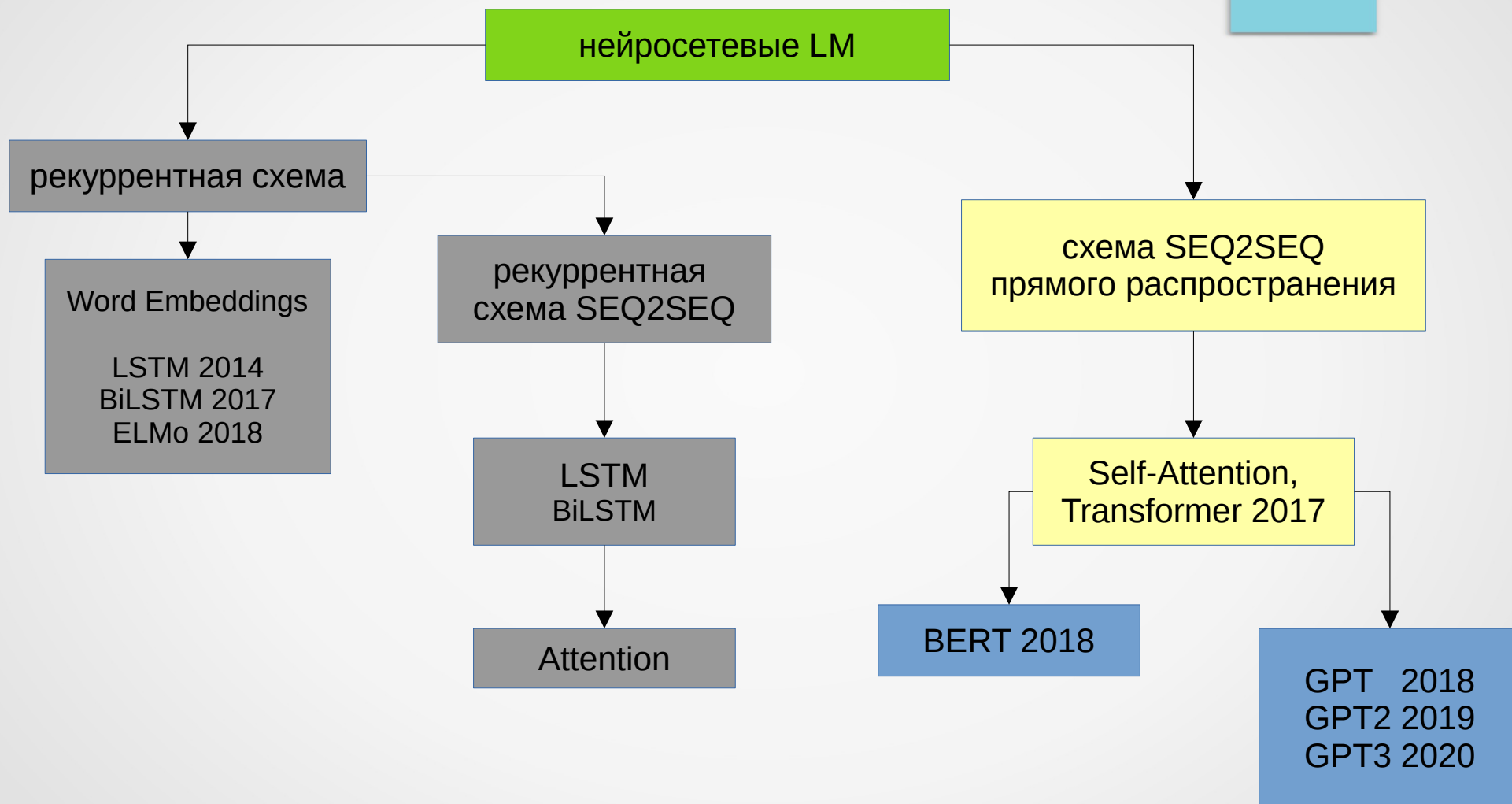
- считаем коэффициенты внимания
- добавляем к состоянию декодера взвешенную сумму входов



все слова подаём в модель последовательно,
схему вычислений трудно распараллелить



Нейросетевые языковые модели

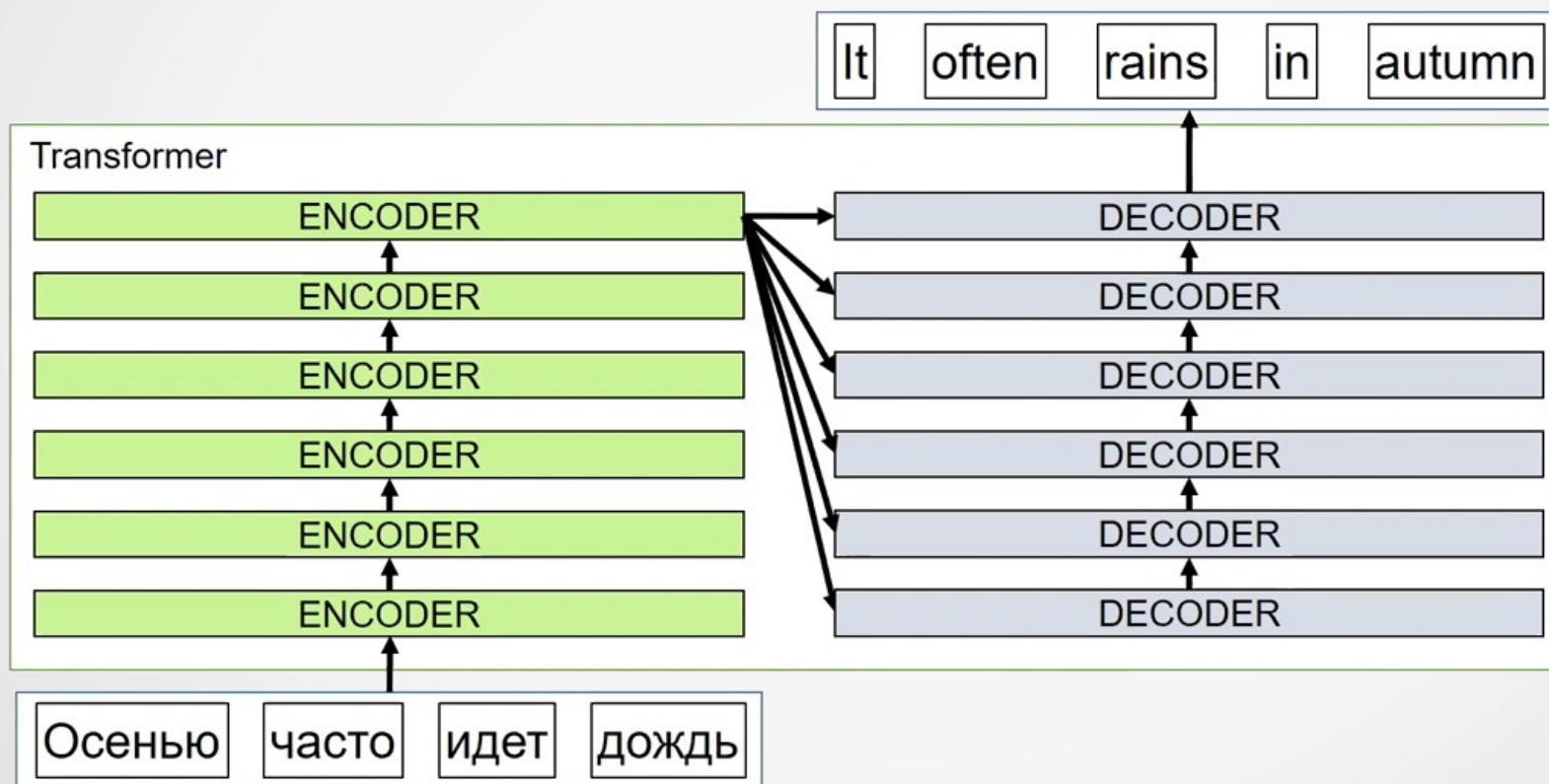


Нейросетевые языковые модели

Схема SEQ2SEQ прямого распространения

Модель Transformer и механизм Self-Attention

Attention Is All You Need (2017) <https://arxiv.org/abs/1706.03762>



токенизация BPE (Byte Pair Encoding)

Sennrich R., et al. Neural machine translation of rare words with subword units- 2015.

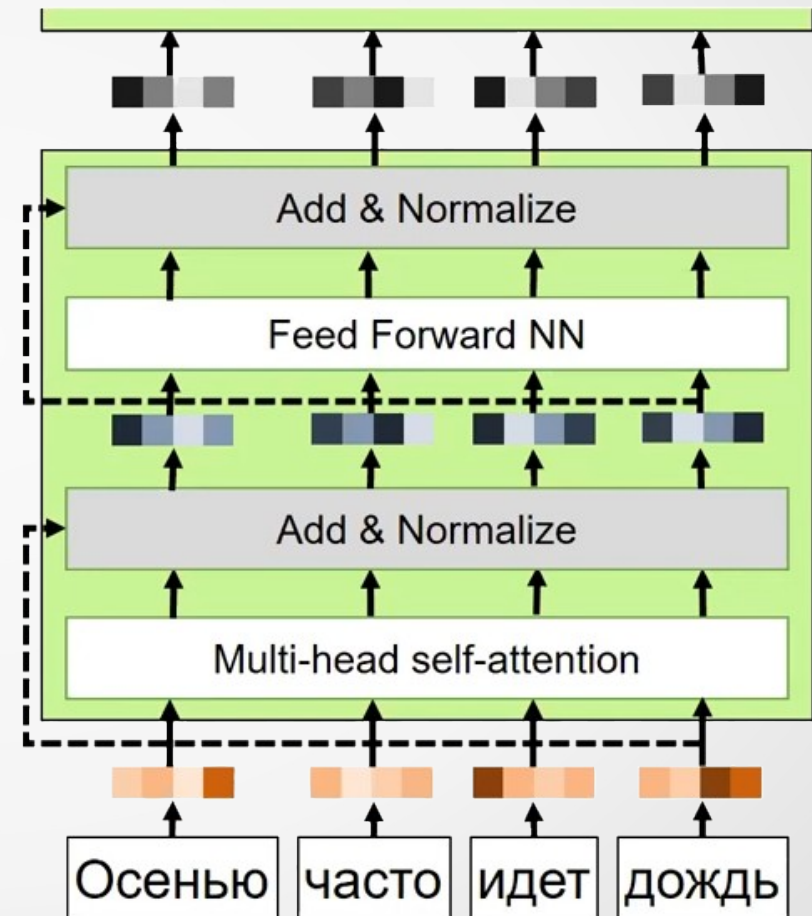
Нейросетевые языковые модели

Transformer: Энкодер

- блок внимания MHSA
- skip connection, normalization
- сеть прямого распространения

все слова подаём в модель одновременно,
они обрабатываются совместно,

схема вычислений хорошо распараллеливается



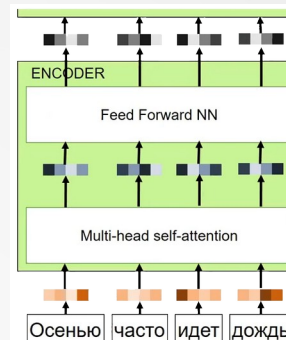
Нейросетевые языковые модели

Transformer: Encoder : Self-Attention

query - откуда смотрим (из какого слова)

key - куда смотрим (на какое слово)

value - смысл (условно) слова



Вход:	Осенью	часто	идет	дождь
Векторы:	x_1	x_2	x_3	x_4
Запросы:	q_1	q_2	q_3	q_4
Ключи:	k_1	k_2	k_3	k_4
Оценки:	$q_3 k_1 / \sqrt{d}$	$q_3 k_2 / \sqrt{d}$	$q_3 k_3 / \sqrt{d}$	$q_3 k_4 / \sqrt{d}$
softmax:	0.1	0.08	0.7	0.12
Значения:	v_1	v_2	v_3	v_4
Сумма:	z_1	z_2	z_3	z_4

Осенью	часто	идет	дождь
x_1	x_2	x_3	x_4
$Q = X * W^Q$			
$K = X * W^K$			
$V = X * W^V$			
$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V$			
z_1	z_2	z_3	z_4

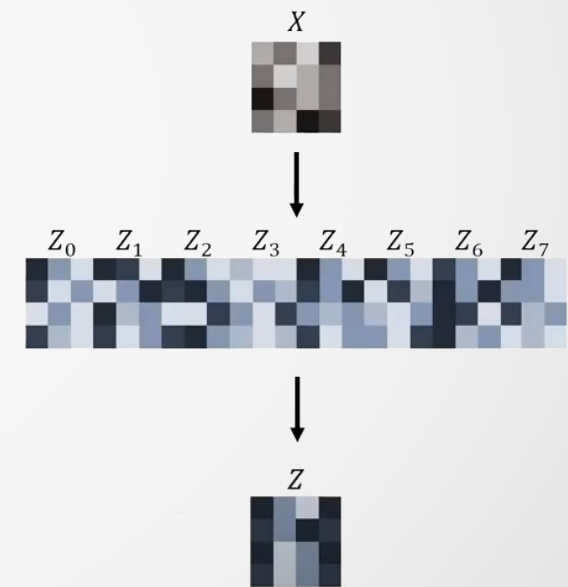
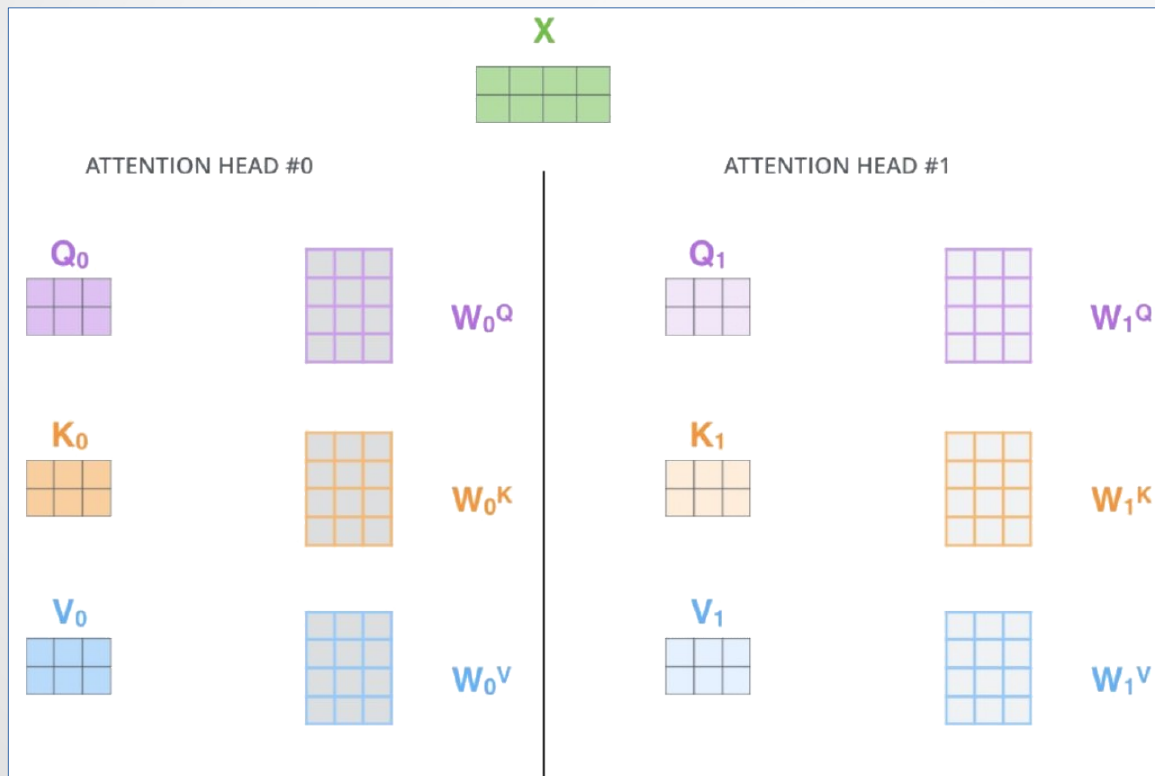
все слова подаём в модель одновременно,
они обрабатываются совместно,
схема вычислений хорошо распараллеливается

Нейросетевые языковые модели

Transformer: Encoder : Multi-Head-Self-Attention

Используем параллельно несколько блоков Self-Attention с разными весами

Результаты агрегируются в размер входа X для стекирования блоков энкодера



Нейросетевые языковые модели

Transformer: Encoder : Multi-Head-Self-Attention

1) This is our input sentence*

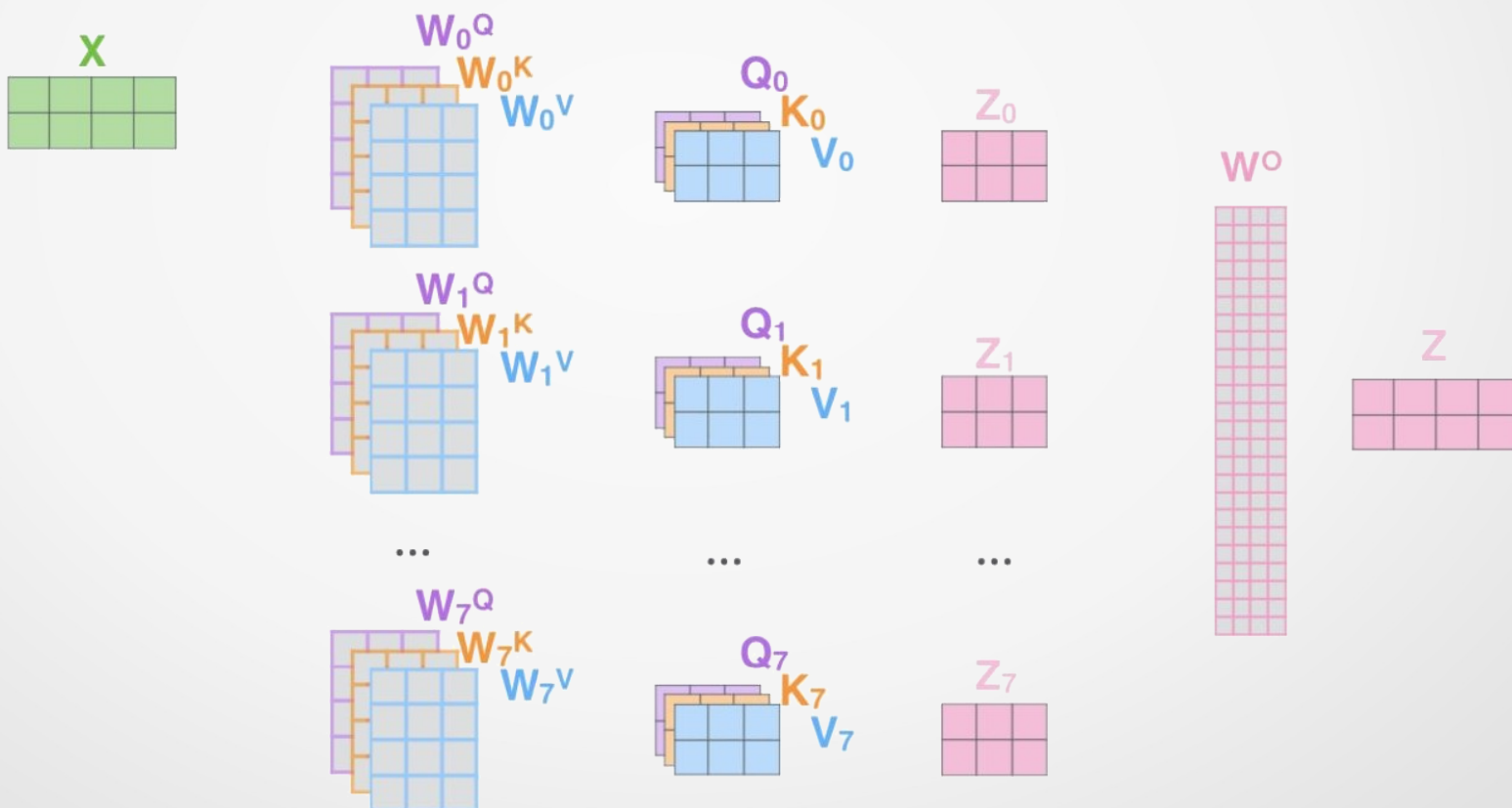
2) We embed each word*

3) Split into 8 heads.
We multiply X with weight matrices

4) Calculate attention using the resulting $Q/K/V$ matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

Thinking
Machines

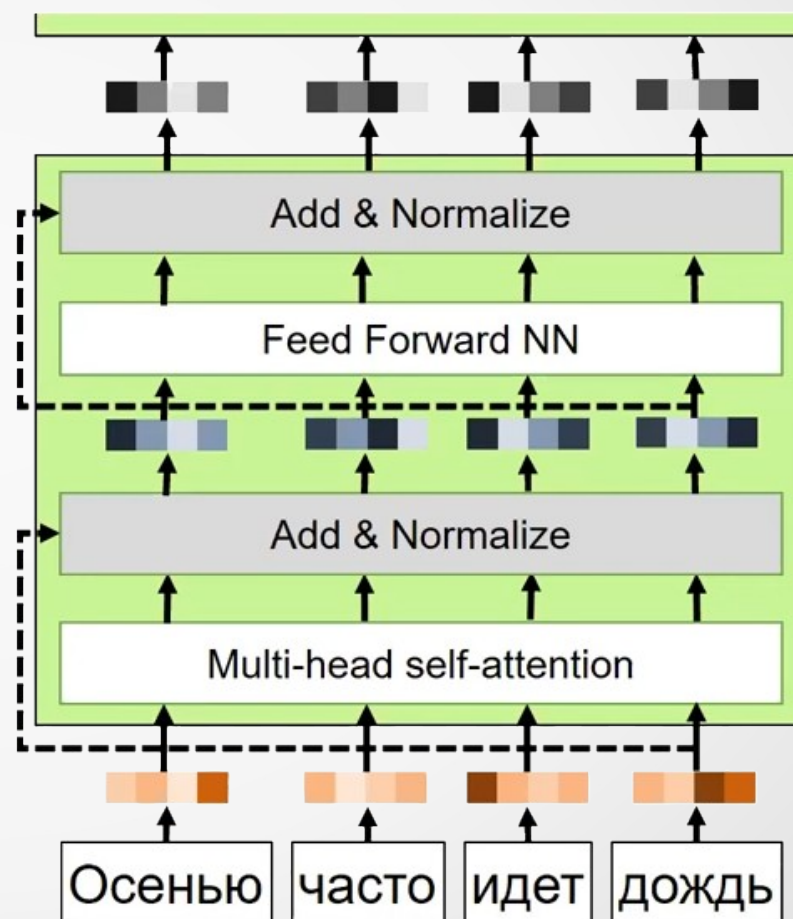


Нейросетевые языковые модели

Transformer: Encoder

Проблема: не учитывается порядок слов

Решение: positional encoding



Нейросетевые языковые модели

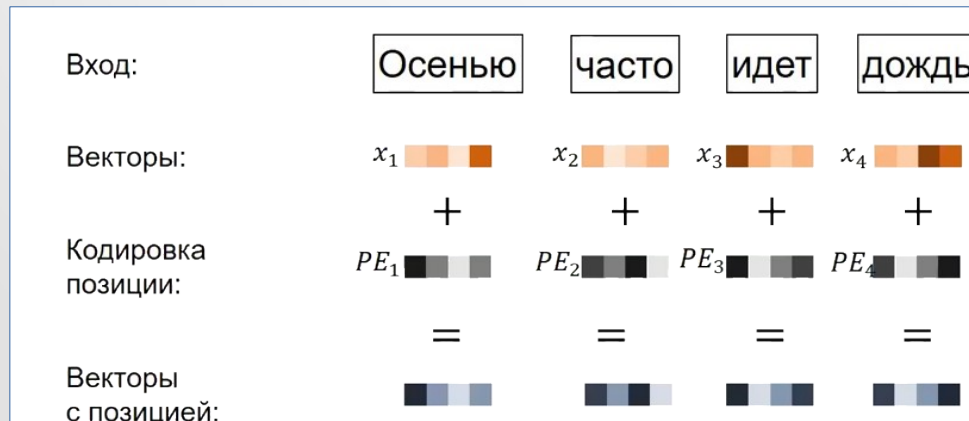
Transformer: Encoder : Positional encoding

Необходимо обозначить позицию слова, выполняя условия

- уникальность для каждого слова
- не зависит от длины предложения
- детерминирован (не стохастический)

$$\vec{p}_t^{(i)} = f(t)^{(i)} = \begin{cases} \sin(\omega_k t), & \text{if } i = 2k \\ \cos(\omega_k t), & \text{if } i = 2k + 1 \end{cases}$$
$$\omega_k = \frac{1}{10000^{2k/d}}$$
$$\vec{p}_t = \begin{bmatrix} \sin(\omega_1.t) \\ \cos(\omega_1.t) \\ \sin(\omega_2.t) \\ \cos(\omega_2.t) \\ \vdots \\ \sin(\omega_{d/2}.t) \\ \cos(\omega_{d/2}.t) \end{bmatrix}_{d \times 1}$$

t - номер слова в строке
d - размерность входа модели
k - номер элемента в векторе PE



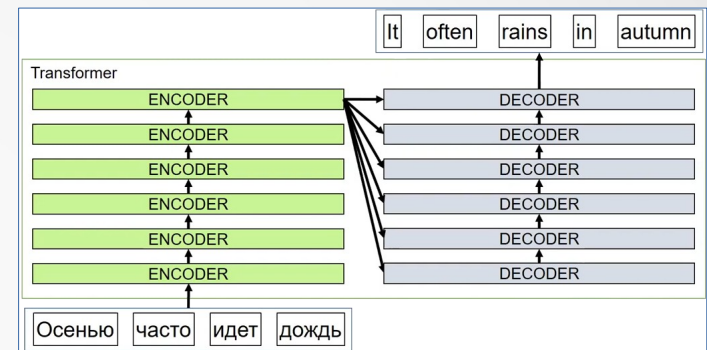
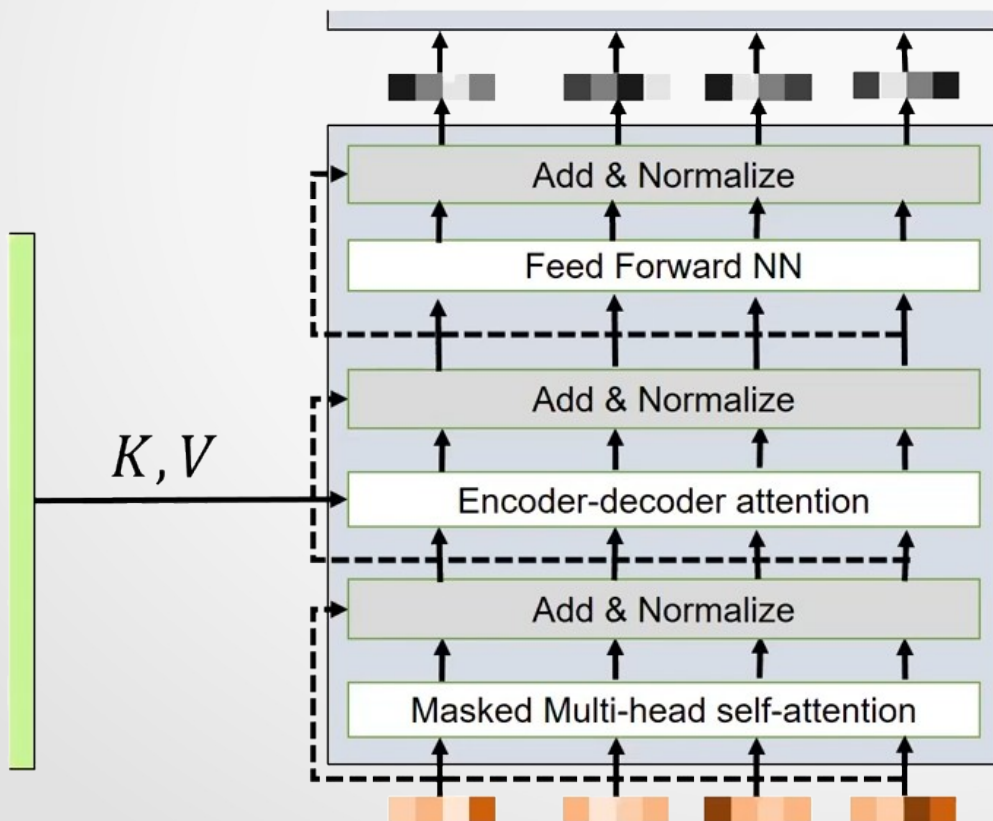
PE - не обучается вместе с моделью,
но вычисляется по формуле

длина последовательности ограничена

Нейросетевые языковые модели

Transformer: Decoder

авторегрессионная модель - выход подаём на вход



- encoder-decoder attention, используем $[K, V]$ из последнего блока encoder

- masked MHA
при расчёте self-attention используем только левый контекст

Нейросетевые языковые модели

Transformer: Decoder : Masked Multi-Head-Self-Attention

в процессе обучения модели,
при расчёте self-attention используем только левый контекст

Вход:	Осенью	часто	идет	дождь
Векторы:	x_1	x_2	x_3	x_4
Запросы:	q_1	q_2	q_3	q_4
Ключи:	k_1	k_2	k_3	k_4
Оценки:	$q_3 k_1 / \sqrt{d}$	$q_3 k_2 / \sqrt{d}$	$q_3 k_3 / \sqrt{d}$	$-\infty$
softmax:	0.14	0.12	0.74	0
Значения:	v_1	v_2	v_3	
Сумма:	z_1	z_2	z_3	z_4

Осенью	часто	идет	дождь
x_1	x_2	x_3	x_4
$Q = X * W^Q$			
$K = X * W^K$			
$V = X * W^V$			
$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V$			
z_1	z_2	z_3	z_4

Нейросетевые языковые модели

Transformer: Decoder : Encoder-Decoder Attention

используем [K,V] берём из последнего блока encoder

похоже на Attention из рекуррентных SEQ2SEQ

Вход:	Осенью	часто	идет	дождь
Векторы:	x_1	x_2	x_3	x_4
Запросы:	q_1	q_2	q_3	q_4
Ключи:	k_1	k_2	k_3	k_4
Оценки:	$q_3 k_1 / \sqrt{d}$	$q_3 k_2 / \sqrt{d}$	$q_3 k_3 / \sqrt{d}$	$-\infty$
softmax:	0.14	0.12	0.74	0
Значения:	v_1	v_2	v_3	
Сумма:	z_1	z_2	z_3	z_4

Осенью	часто	идет	дождь
x_1	x_2	x_3	x_4
$Q = X * W^Q$			
K			
V			
$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V$			
z_1	z_2	z_3	z_4

Нейросетевые языковые модели

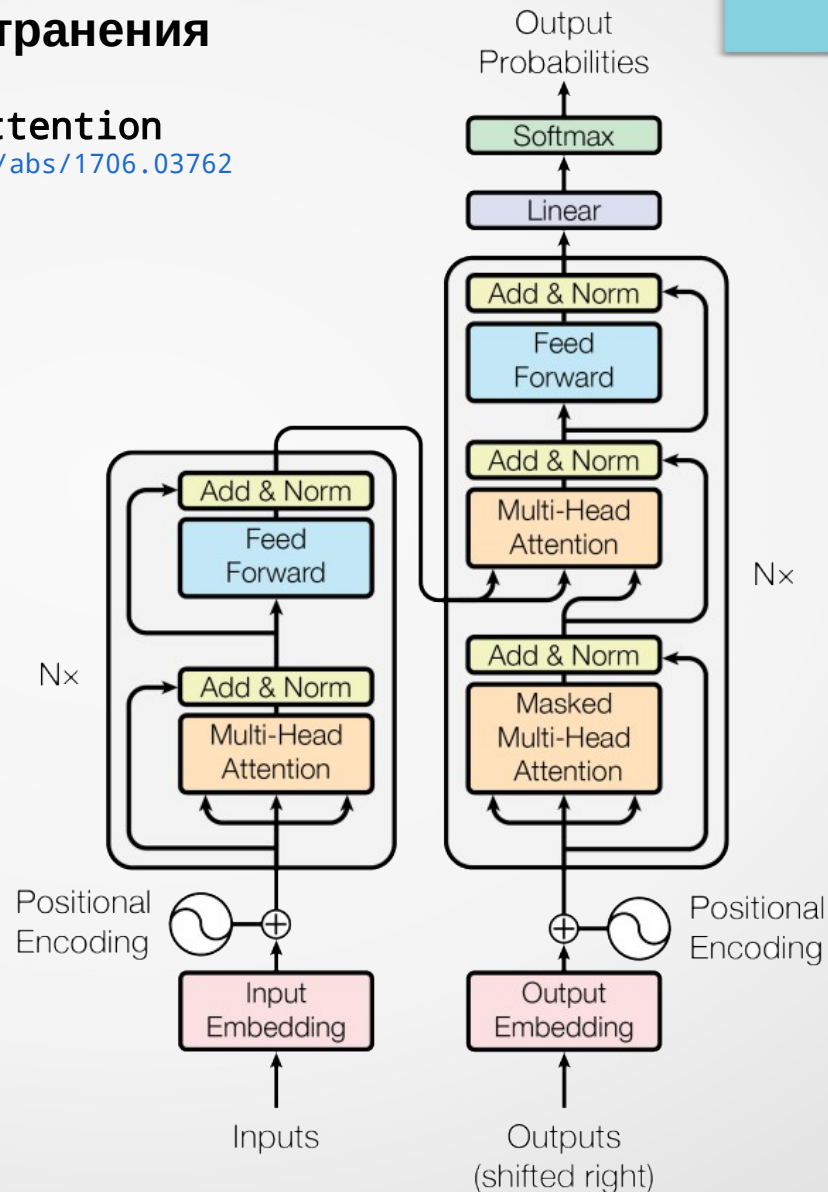
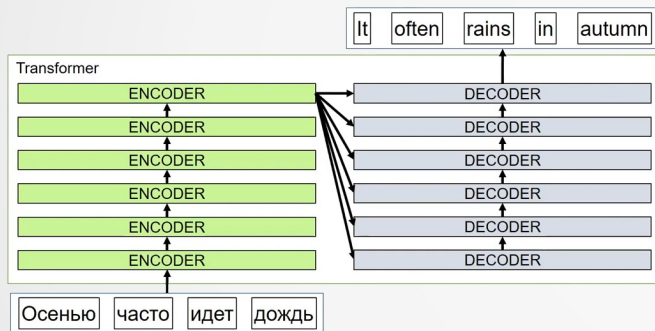
Схема SEQ2SEQ прямого распространения

Модель Transformer и механизм Self-Attention

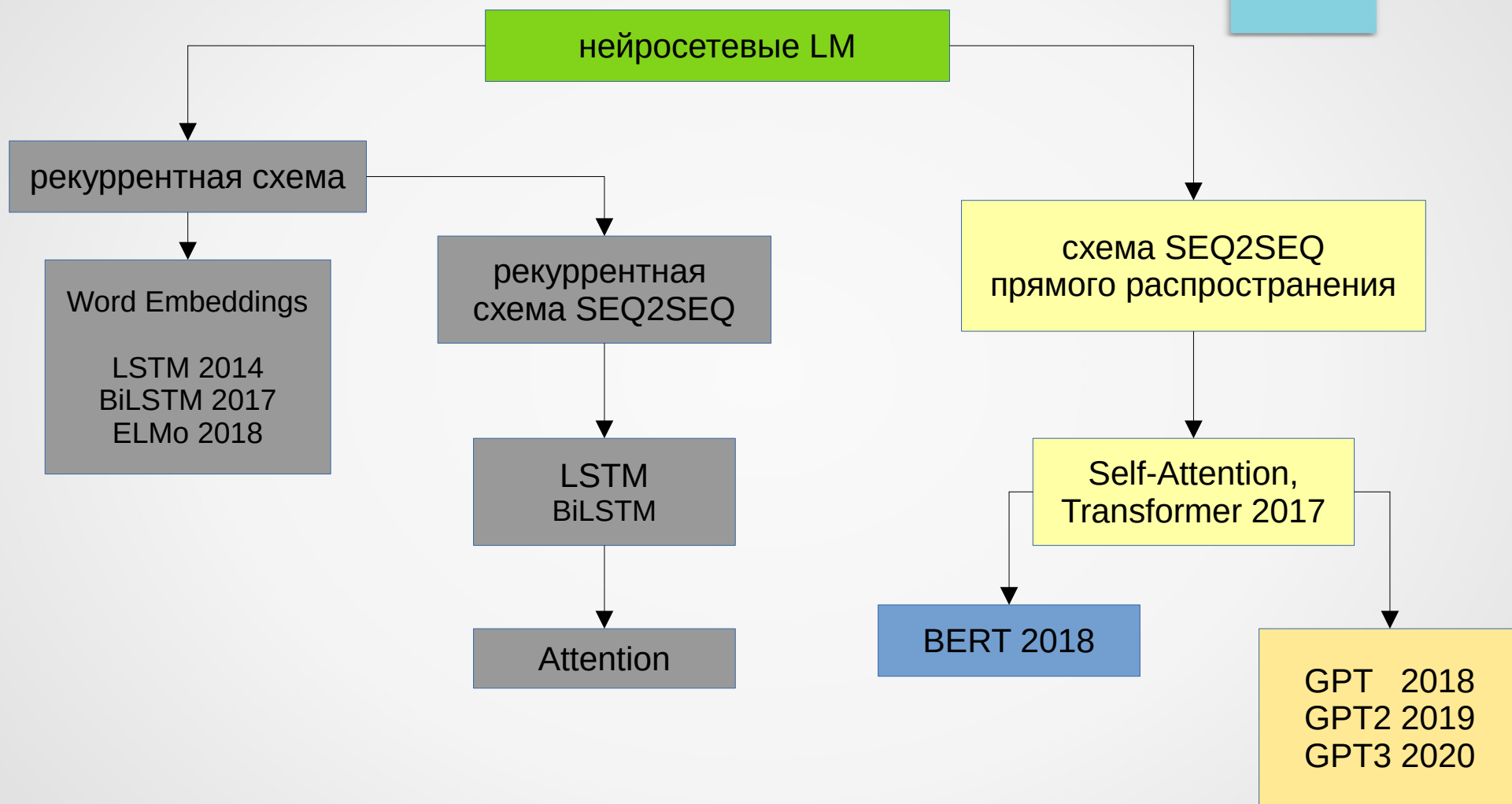
Attention Is All You Need (2017) <https://arxiv.org/abs/1706.03762>

токенизация BPE (Byte Pair Encoding)

Sennrich R., et al. Neural machine translation of rare words with subword units- 2015.



Нейросетевые языковые модели

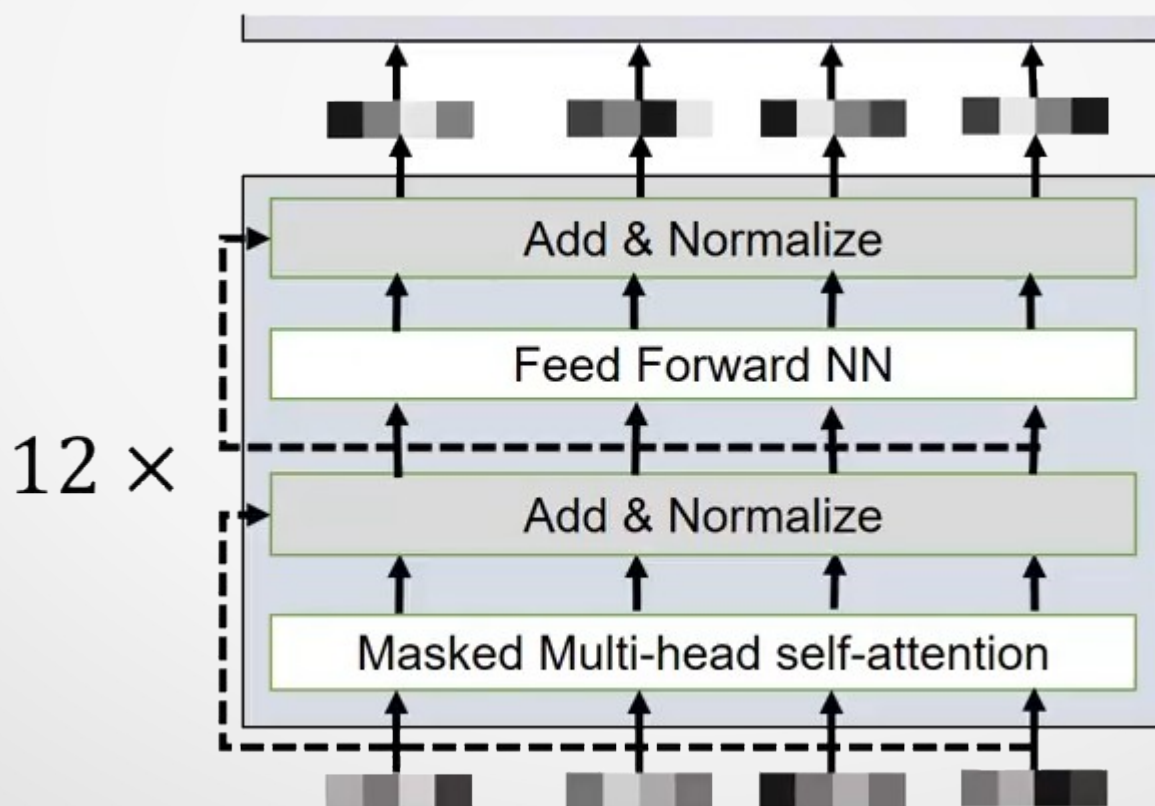


Нейросетевые языковые модели

Модели GPT (Generative Pre-Training)

Radford A. et al. Improving language understanding by generative pre-training. - 2018

Языковая модель, основанная на блоке Decoder модели Transformer, блок Encoder-Decoder-Attention выкидываем.



Нейросетевые языковые модели

Модели GPT (Generative Pre-Training)

Radford A. et al. Improving language understanding by generative pre-training. - 2018

Схема обучения модели на основе GPT

- Предобучение на корпусе \mathcal{U} (без учителя)

$$\begin{aligned}h_0 &= UW_e + W_p; \\h_i &= \text{transformer}(h_{i-1}), i = \overline{1, n} \\P(u) &= \text{softmax}(h_n W_e^T)\end{aligned}$$

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \dots, u_{i-1}; \Theta)$$

- Донастройка под целевую задачу на корпусе \mathcal{C} (с учителем)

$$P(y | x^1, \dots, x^m) = \text{softmax}(h_l^m W_y)$$

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y | x^1, \dots, x^m)$$

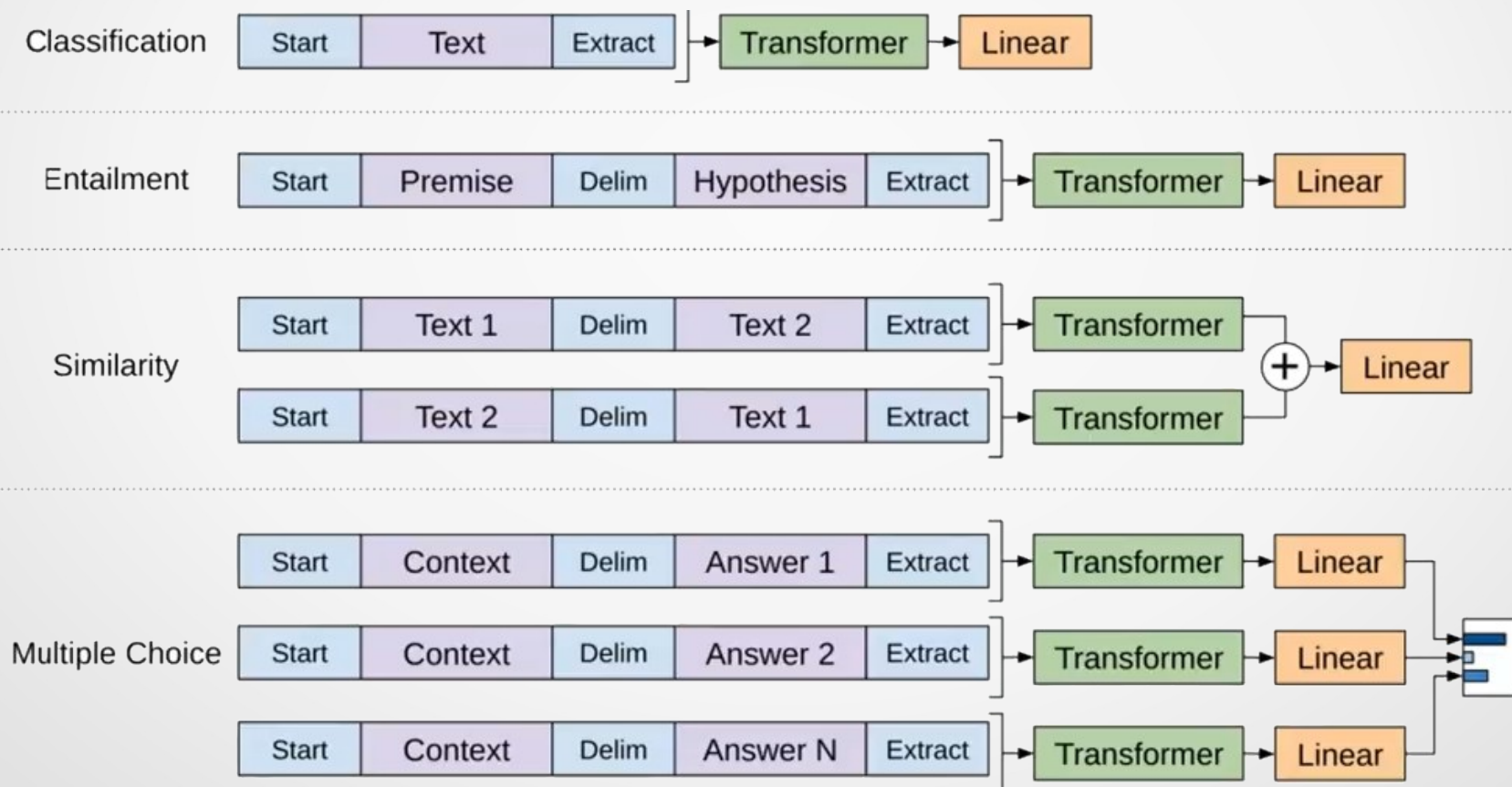
$$L(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C});$$

Нейросетевые языковые модели

Модели GPT (Generative Pre-Training)

Radford A. et al. Improving language understanding by generative pre-training. - 2018

Различные варианты применения GPT



Нейросетевые языковые модели

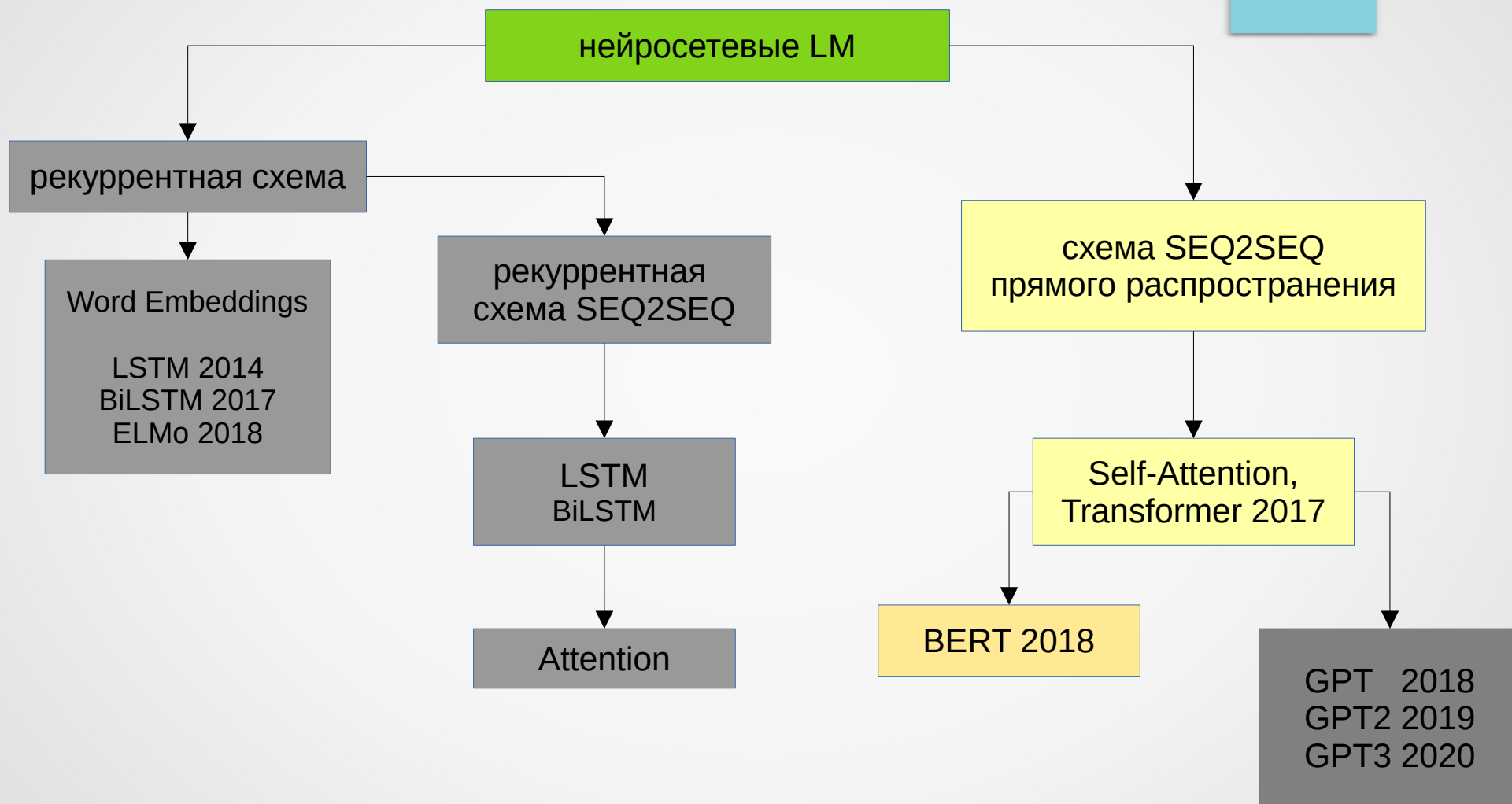
Модели GPT (Generative Pre-Training)

GPT: Radford A. et al. Improving language understanding by generative pre-training. - 2018
- 12 слоев трансформера

GPT-2: Radford A. et al. Language models are unsupervised multitask learners - 2019
- 48 слоев трансформера
- Больше корпус для обучения (40 GB текста)
- Другая токенизация: BPE по байтам, а не по символам
- передвинут Layer normalization
- изменена инициализация

GPT-3: Brown T. B. et al. Language models are few-shot learners - 2020.
(почти не отличается от GPT-2)
- 96 слоев трансформера
- Еще больше корпус для обучения (570 GB текста)
- Еще больше контекст (2048 токенов)

Нейросетевые языковые модели



Нейросетевые языковые модели

BERT (Bidirectional Encoder Representation Transformers)

Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding - 2018.

Языковая модель, основанная на Transformer Encoder

BERT Base – 12 блоков

BERT Large - 24 блока

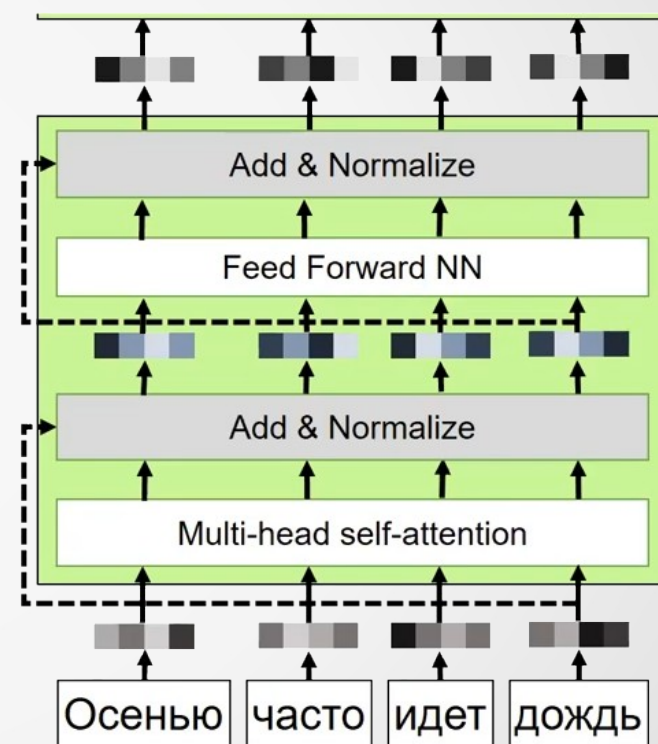
применяем обучаемый Positional Encoding

данные для обучение разбиты на пары предложений

обучаем модель определять связаны ли два предложения

маскируем слова и обучаем модель их предсказывать

$12(24) \times$

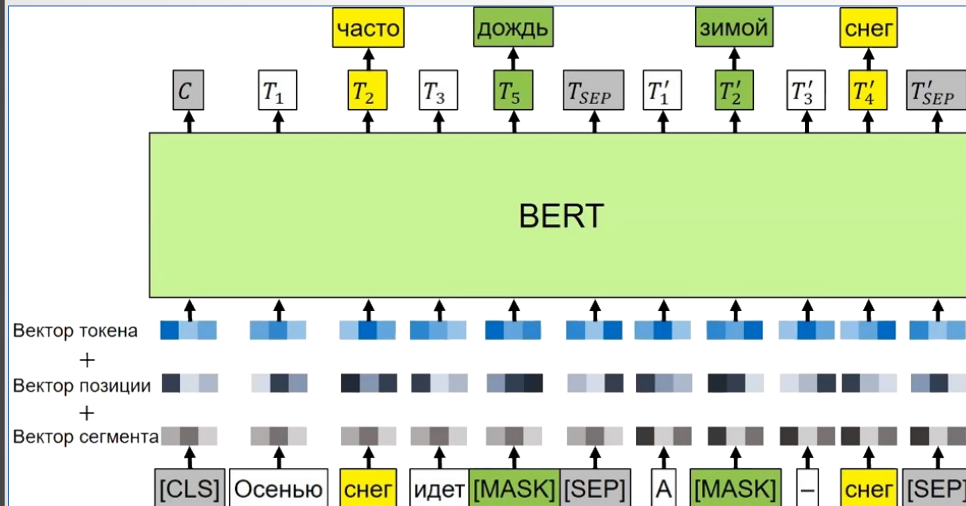


Нейросетевые языковые модели

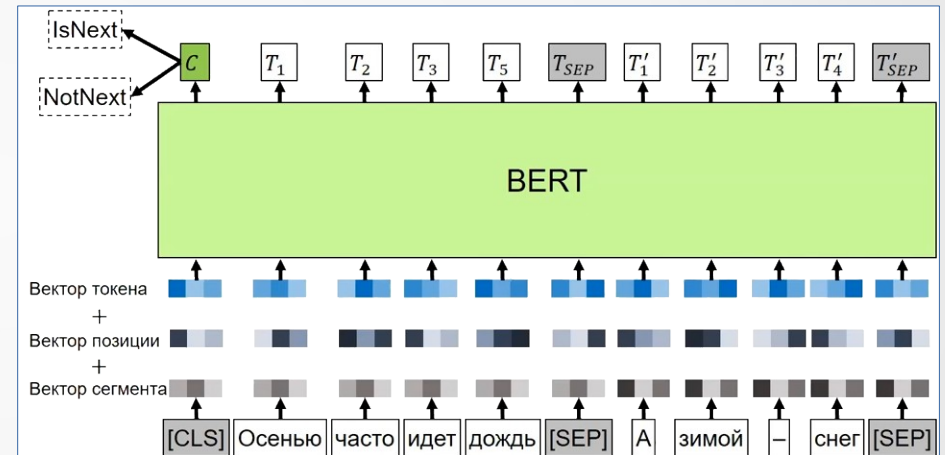
BERT (Bidirectional Encoder Representation Transformers)

Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding – 2018.

- обучаемый Positional Encoding + дополнительный вектор номера сегмента



- маскируем слова и обучаем модель их предсказывать
- 15% токенов могут маскироваться, из них:
 - 80% - [MASK];
 - 10% - подменяем на случайные;
 - 10% - оставляем;



- обучаем модель определять связаны ли сегменты (опционально)

Нейросетевые языковые модели

git clone *https://github.com/mechanoid5/ml_nlp.git*

Майоров В.Д.

Основы обработки текстов. Лекция 10. Языковые модели. ИСП РАН, 2021

https://www.youtube.com/watch?v=_8MGdpt4I9M

Нейчев Радослав

Прикладное машинное обучение 3. Machine translation. Лекторий ФПМИ, 2020

<https://www.youtube.com/watch?v=6HibilFua-U>