Трансформерные модели.

Алексей Андреевич Сорокин

МГУ им. М. В. Ломоносова
весенний семестр 2022–2023 учебного года
Межфакультетский курс “Введение в компьютерную лингвистику” 5 апреля, занятие 7

Механизм внимания: переобозначение

- Текущая формула (переобозначения):

$$
\begin{aligned}
h &= \sum_i \alpha_i h_i^{value}, \\
\alpha_i &\sim \exp\left(\langle h_i^{key}, s\rangle\right), \\
s &- \text{глобальный вектор “запроса” (query)}, \\
h_i^{value} &- \text{“эмбеддинг-значение” (value)}, \\
h_i^{key} &- \text{“эмбеддинг-ключ” (key)}.
\end{aligned}
$$

- Откуда взять h_i^{value}, h_i^{key}.

Механизм внимания: переобозначение

- Текущая формула (переобозначения):

$$
\begin{aligned}
h &= \sum_i \alpha_i h_i^{value}, \\
\alpha_i &\sim \exp\left(\langle h_i^{key}, s\rangle\right), \\
s &- \text{глобальный вектор “запроса” (query)}, \\
h_i^{value} &- \text{“эмбеддинг-значение” (value)}, \\
h_i^{key} &- \text{“эмбеддинг-ключ” (key)}.
\end{aligned}
$$

- Откуда взять h_i^{value}, h_i^{key}.
- Проще всего вставить один слой персептрона:

$$
\begin{aligned}
h_i^{value} &= g(W^{value} h_i), \\
h_i^{key} &= g(W^{key} h_i)
\end{aligned}
$$

Механизм внимания: матричный вид

- Всё можно переписать в матричном виде:

$$\begin{aligned} h &= A_{1\times L} V_{L\times d}, \\ A &= \mathrm{softmax}(q_{1\times d} K^T_{L\times d}), \\ V &= g(H_{L\times d} W^{value}_{d\times d}), \\ K &= g(H_{L\times d} W^{key}_{d\times d}) \end{aligned}$$

- Финальная формула:

$$h = \mathrm{softmax}(QK^T)V$$

Механизм внимания: матричный вид

- Всё можно переписать в матричном виде:

$$\begin{aligned} h &= A_{1\times L} V_{L\times d}, \\ A &= \mathrm{softmax}(q_{1\times d} K^T_{L\times d}), \\ V &= g(H_{L\times d} W^{value}_{d\times d}), \\ K &= g(H_{L\times d} W^{key}_{d\times d}) \end{aligned}$$

- Финальная формула:

$$h = \mathrm{softmax}(QK^T)V$$

- На практике добавляют нормализующий множитель:

$$h = \mathrm{softmax}(\frac{QK^T}{\sqrt{d}})V$$

- Без него обучение более нестабильное.

Механизм самовнимания : матричный вид

- Механизм внимания используется, чтобы посчитать состояние для всего предложения с учётом всех слов.
- А что если так же считать новые состояния для всех слов?
- В этом случае даже удалённые слова будут влиять на текущий вектор (проблема для рекуррентных сетей).

Механизм самовнимания : матричный вид

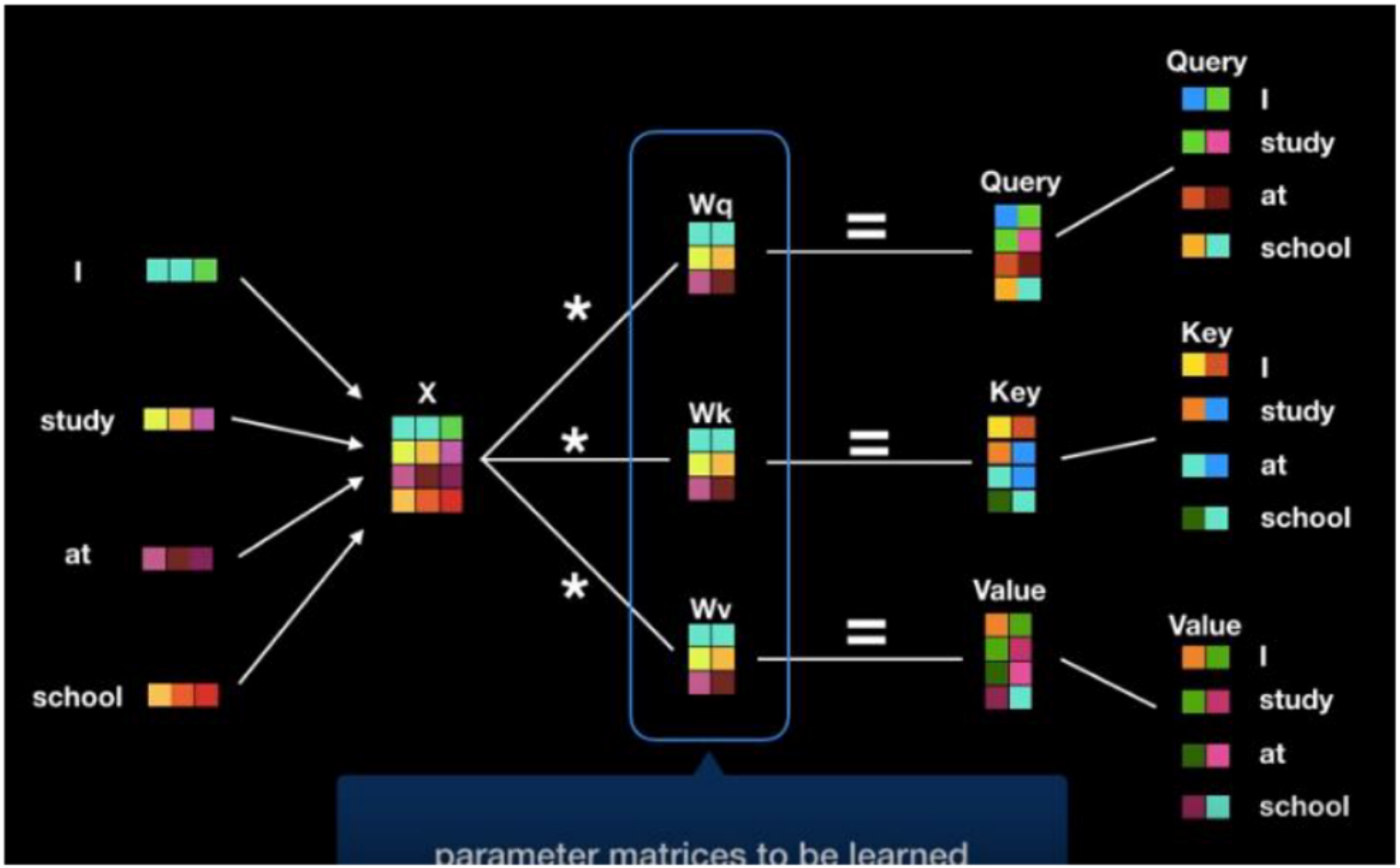
- Механизм внимания используется, чтобы посчитать состояние для всего предложения с учётом всех слов.
- А что если так же считать новые состояния для всех слов?
- В этом случае даже удалённые слова будут влиять на текущий вектор (проблема для рекуррентных сетей).
- Достаточно сделать "запрос" Q матрицей:

$$Q_{L\times d} = g(H_{L\times d} W^{query}_{d\times d})$$

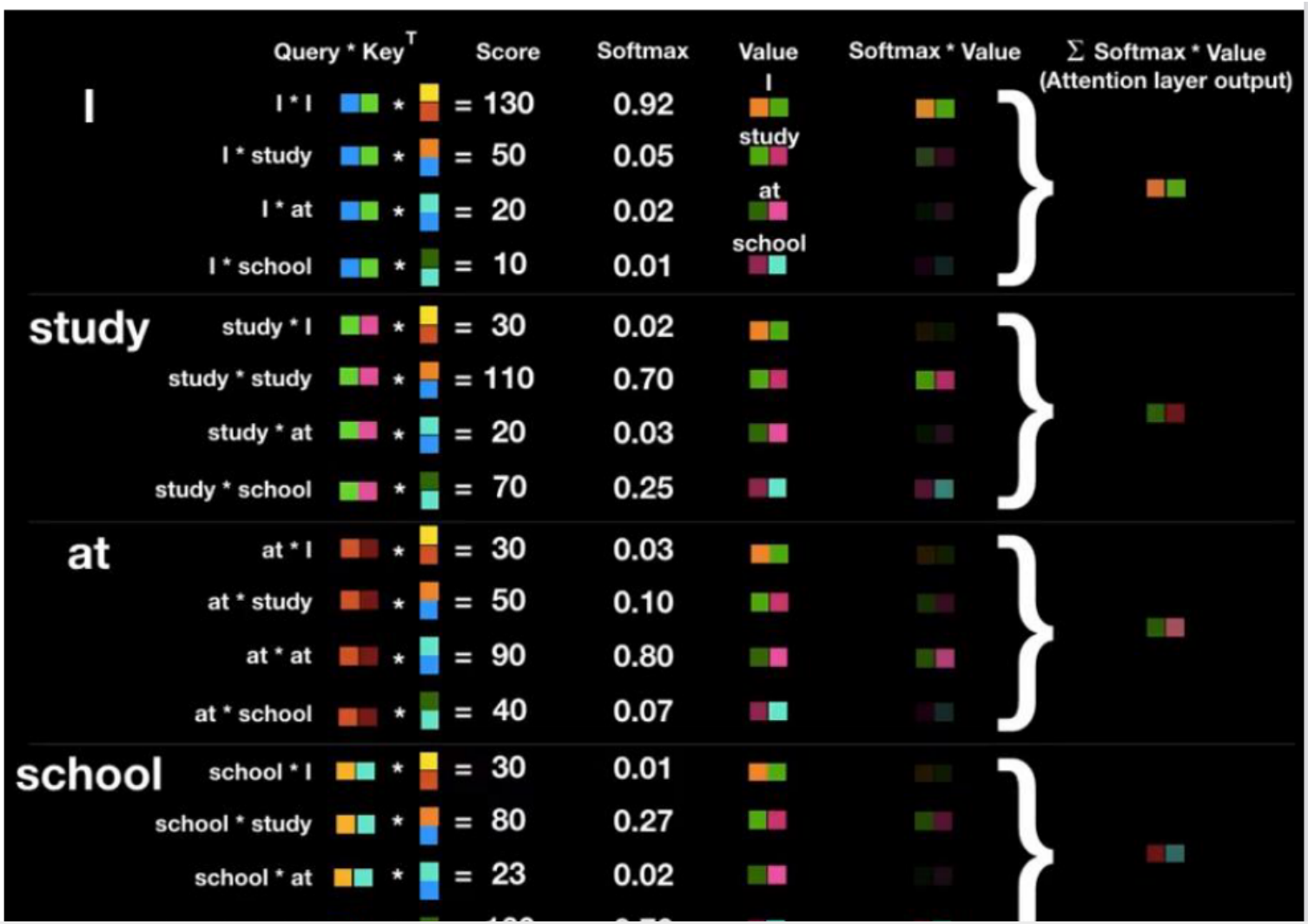
- В итоге получаем:

$$\begin{aligned} H' &= A_{L\times L} V_{L\times d}, \\ A &= \mathrm{softmax}(Q_{L\times d} K^T_{L\times d}), \\ Q &= g(H_{L\times d} W^{query}_{d\times d}), \\ V &= g(H_{L\times d} W^{value}_{d\times d}), \\ K &= g(H_{L\times d} W^{key}_{d\times d}) \end{aligned}$$

Механизм самовнимания: иллюстрация

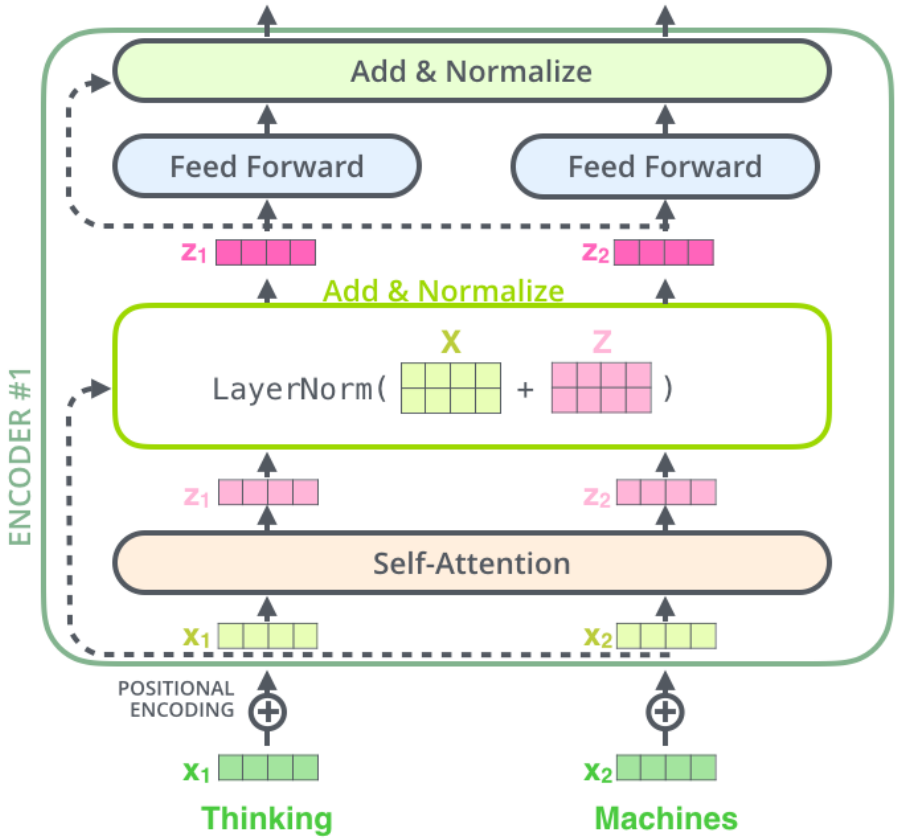


Механизм самовнимания: иллюстрация



Механизм самовнимания: трансформеры

- Механизм внимания – это один слой трансформерной архитектуры.
- Между такими слоями вставляются residual-переходыполносвязные подслои и слой-нормализация (LayerNorm). Параллельно с трансформерным блоком вставляется residual-переход.



Механизм самовнимания: множественное внимание

- Может возникнуть потребность проявлять внимание с точки зрения разных аспектов и к разным словам.

Вася съел большую банку варенья.

банку $\rightarrow$ большую	морфология
банку $\rightarrow$ съел	семантика

Механизм самовнимания: множественное внимание

- Может возникнуть потребность проявлять внимание с точки зрения разных аспектов и к разным словам.

Вася съел большую банку варенья.

банку $\to$ большую морфология

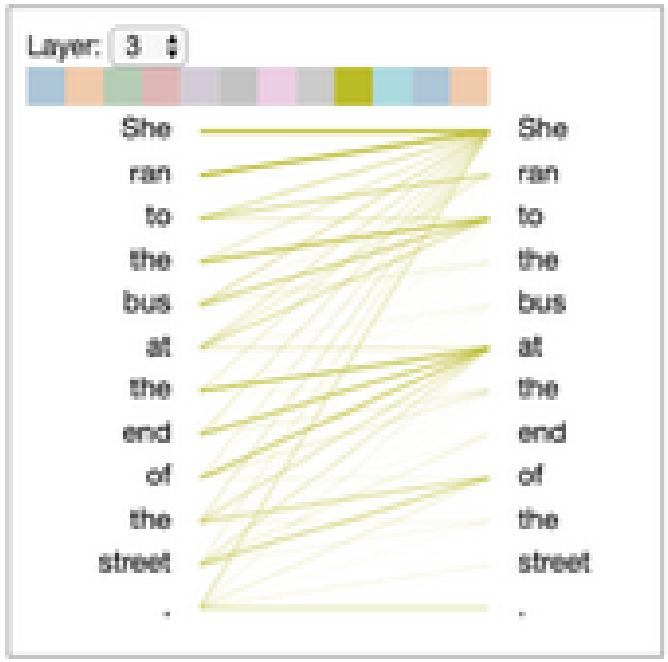
банку $\to$ съел семантика

- Это делается с помощью множественного внимания (multi-head attention):

$$\begin{aligned} H'_i &= \mathrm{Attention}(HW_{q,i}, HW_{k,i}, HW_{v,i}) \\ H' &= \mathrm{Concat}(H'_1, \ldots, H'_m) \\ W_{\star,i} &\in \mathbb{R}^{D \times \frac{D}{m}} \end{aligned}$$

- Все эти операции можно делать параллельно.

Механизм самовнимания: множественное внимание



Механизм самовнимания: энкодер-декодер

- В энкодере механизм внимания нужен, чтобы пересчитывать состояния модели, кодирующие исходное предложение.
- В декодере нужно учитывать не только исходное предложение, но и сгенерированную часть результата.

Механизм самовнимания: энкодер-декодер

- В энкодере механизм внимания нужен, чтобы пересчитывать состояния модели, кодирующие исходное предложение.
- В декодере нужно учитывать не только исходное предложение, но и сгенерированную часть результата.
- Как следствие, есть два подслоя внимания:
  - Внимание состояний энкодера к состояниям декодера α_{ij}:

$$i \quad - \quad \text{позиция в генерируемом тексте,}$$
$$j \quad - \quad \text{позиция в исходном тексте.}$$

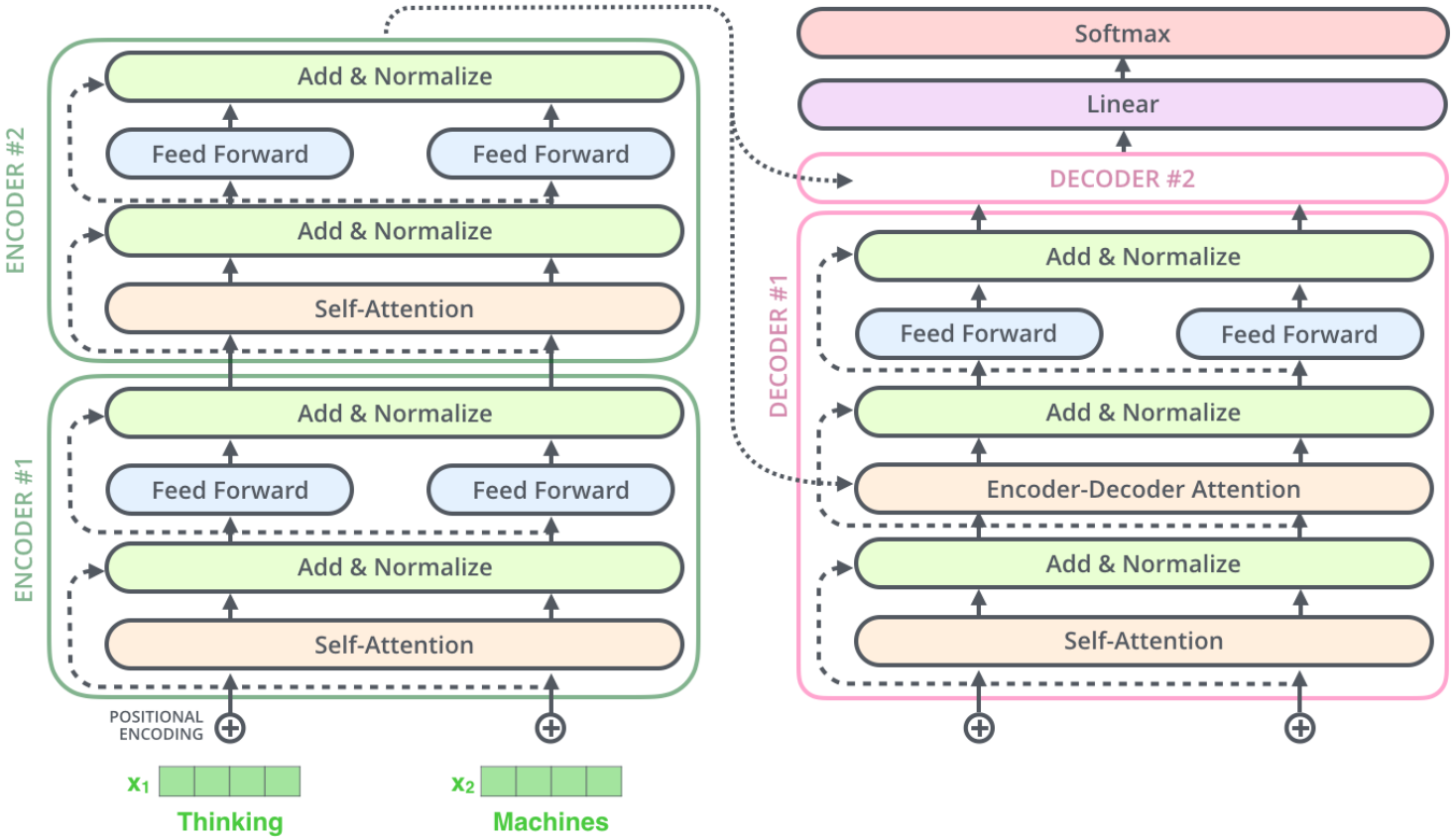
Механизм самовнимания: энкодер-декодер

- В энкодере механизм внимания нужен, чтобы пересчитывать состояния модели, кодирующие исходное предложение.
- В декодере нужно учитывать не только исходное предложение, но и сгенерированную часть результата.
- Как следствие, есть два подслоя внимания:
  - Внимание состояний энкодера к состояниям декодера α_{ij}:

    i – позиция в генерируемом тексте,
    j – позиция в исходном тексте.
  - Внимание состояний энкодера к состояниям декодера β_{ij}:

    i – позиция в генерируемом тексте,
    j – позиция в генерируемом тексте, $j < i$.
  - При обучении считается $\beta_{ij} = 0$ при $j \geqslant i$, чтобы модель не заглядывала в будущее.

Трансформеры: энкодер-декодер



Трансформеры: скорость

- Скорость передачи информации на m шагов:
  - Свёрточные сети: $O(\frac{m}{w})$.
  - Рекуррентные сети: $O(m)$.
  - Трансформеры: $O(1)$.
- Затраты памяти на последовательность длины L:
  - Свёрточные сети: $O(wd^2L)$.
  - Рекуррентные сети: $O(d^2L)$.
  - Трансформеры: $O(L^2d)$.

Трансформеры: позиционное кодирование

- Пока механизм внимания никак не различает одинаковые вектора, стоящие в разных местах.
- Чтобы это исправить, конкатенируют вектора слов с позиционными эмбеддингами:

$$x_i = [emb(word_i), x_{pos}(i)]$$

Трансформеры: позиционное кодирование

- Пока механизм внимания никак не различает одинаковые вектора, стоящие в разных местах.
- Чтобы это исправить, конкатенируют вектора слов с позиционными эмбеддингами:

$$x_i = [emb(word_i), x_{pos}(i)]$$

- x_{pos} иногда задают явно (Vaswani et al., 2017):

$$\begin{aligned} (x_{pos}(i))_{2j} &= \sin \frac{i}{10000^{j/d}}, \\ (x_{pos}(i))_{2j+1} &= \cos \frac{i}{10000^{j/d}}. \end{aligned}$$

- В более поздних подходах их сделали обучаемыми:
  - Это дополнительная матрица размера max_length $\times$ d.
  - Последовательности длиннее max_length не обрабатываются.

Нейронные сети: предобучение на языковом моделировании

- Задача языкового моделирования – предсказание следующего слова в тексте (распределения вероятностей следующего слова).

Я вчера ел вкусную ?

Нейронные сети: предобучение на языковом моделировании

- Задача языкового моделирования – предсказание следующего слова в тексте (распределения вероятностей следующего слова).

Я вчера ел вкусную ?

- Она требует больших знаний о языке:
  - Морфология (если следующее слово существительное, то женского рода).
  - Графика (следующее слово с большой вероятностью кончается на -у).
  - Синтаксис.
  - Семантика (следующее слово “съедобное”).

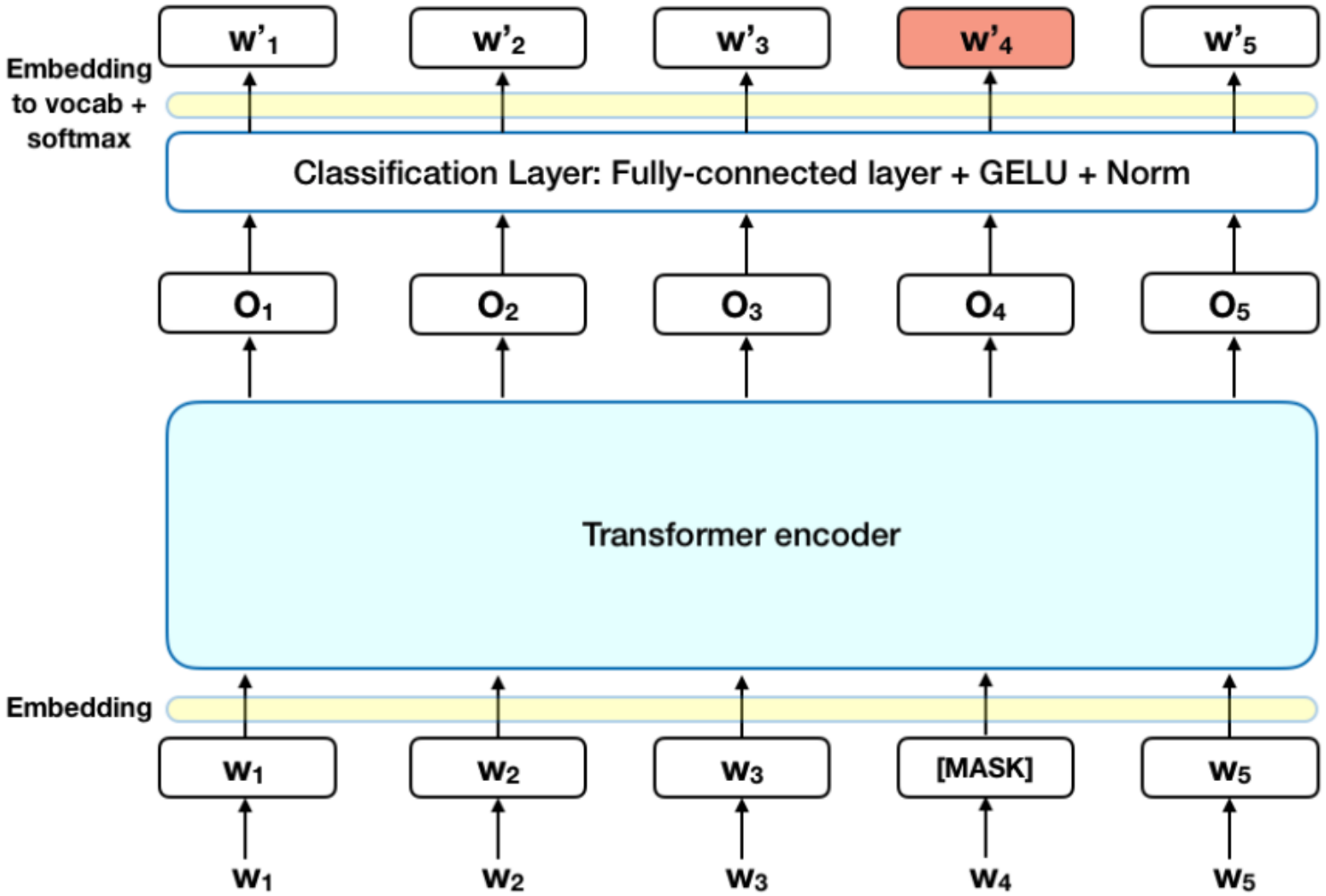
Нейронные сети: предобучение на языковом моделировании

- Задача языкового моделирования – предсказание следующего слова в тексте (распределения вероятностей следующего слова).

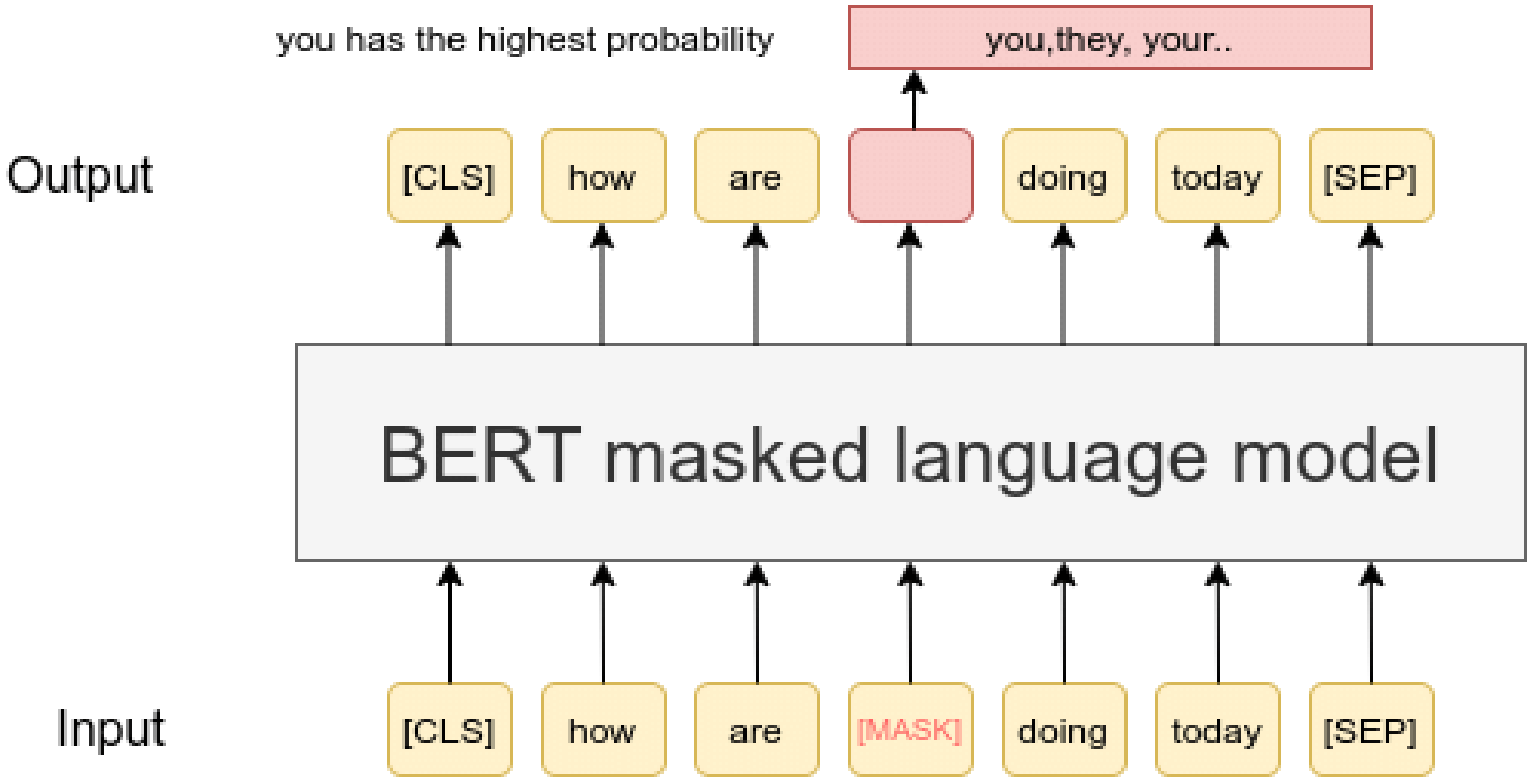
  Я вчера ел вкусную ?

- Она требует больших знаний о языке:
  - Морфология (если следующее слово существительное, то женского рода).
  - Графика (следующее слово с большой вероятностью кончается на -у).
  - Синтаксис.
  - Семантика (следующее слово “съедобное”).
- Кроме того, для получения этой информации не нужны размеченные данные.

Нейронные сети: BERT



BERT: иллюстрация



Нейронные сети: BERT

- Глубокая (12 слоёв) архитектура на основе трансформера (механизма внимания).
- Преобразует последовательность индексов токенов в последовательность векторов.
- Дополнительный вектор для токена начала предложения (CLS-токен).

Нейронные сети: BERT

- Глубокая (12 слоёв) архитектура на основе трансформера (механизма внимания).
- Преобразует последовательность индексов токенов в последовательность векторов.
- Дополнительный вектор для токена начала предложения (CLS-токен).
- Предобучается на двух задачах:
  - Восстановлении пропущенного токена.
  - Проверке, идут ли два предложения друг за другом.

Нейронные сети: BERT

- Глубокая (12 слоёв) архитектура на основе трансформера (механизма внимания).
- Преобразует последовательность индексов токенов в последовательность векторов.
- Дополнительный вектор для токена начала предложения (CLS-токен).
- Предобучается на двух задачах:
  - Восстановлении пропущенного токена.
  - Проверке, идут ли два предложения друг за другом.
- Восстановление пропущенных слов – информация на уровне слов.
- Проверка следования предложений – задачи на уровне предложений / пар предложений.

Восстановление пропущенного токена

- Задача восстановления решается для 15% токенов.
  - 80% заменяются на специальный токен $\langle \text{MASK} \rangle$.
  - 10% на произвольное слово.
  - 10% остаются неизменными.
- Это эффективнее, чем предобучать модель слева направо.

BERT: входные данные

- Входное представление:

$$
\begin{aligned}
x_i &= x_i^{token} + x_i^{pos} + x_i^{type}, \\
x_i^{token} &- \text{эмбеддинг текущего токена}, \\
x_i^{pos} &- \text{эмбеддинг позиции } i, \\
x_i^{type} &- \text{эмбеддинг типа токена}.
\end{aligned}
$$

BERT: входные данные

- Входное представление:

$$
\begin{aligned}
x_i &= x_i^{token} + x_i^{pos} + x_i^{type}, \\
x_i^{token} &- \text{эмбеддинг текущего токена}, \\
x_i^{pos} &- \text{эмбеддинг позиции } i, \\
x_i^{type} &- \text{эмбеддинг типа токена.}
\end{aligned}
$$

- Тип токена – 0/1, нужно в задачах для пары предложений.
- Эмбеддинг позиции - i-ый элемент обучаемой матрицы $max\_length \times d_{hidden}$. Обычно $max\_length = 512, d_{hidden} = 768$.

BERT: токенизация

- BERT рассматривает слова на уровне BPE-токенов (Byte-Pair Encoding).
- BPE-словарь строится по следующим правилам:

BERT: токенизация

- BERT рассматривает слова на уровне BPE-токенов (Byte-Pair Encoding).
- BPE-словарь строится по следующим правилам:
- В начале элементы BPE-словаря – отдельные символы.
- На каждом шаге объединяется самая частая пара:

$$\begin{aligned} \mathrm{t} + \mathrm{h} &\mapsto \mathrm{th}, \\ \mathrm{i} + \mathrm{t} &\mapsto \mathrm{it}, \\ \ldots &\ldots \ldots, \\ \mathrm{th} + \mathrm{e} &\mapsto \mathrm{the}, \\ \ldots &\ldots \ldots \end{aligned}$$

BERT: токенизация

- BERT рассматривает слова на уровне BPE-токенов (Byte-Pair Encoding).
- BPE-словарь строится по следующим правилам:
- В начале элементы BPE-словаря – отдельные символы.
- На каждом шаге объединяется самая частая пара:

$$\begin{aligned} \mathrm{t} + \mathrm{h} &\mapsto \mathrm{th}, \\ \mathrm{i} + \mathrm{t} &\mapsto \mathrm{it}, \\ \ldots \quad \ldots &\quad \ldots, \\ \mathrm{th} + \mathrm{e} &\mapsto \mathrm{the}, \\ \ldots \quad \ldots &\quad \ldots \end{aligned}$$

- Так делается, пока не будет достигнут заранее заданный размер (~ 30000 в английской модели).

BERT: токенизация

- При токенизации BERT жадно пытается выделить самый длинный токен из словаря, начиная с конца слова.
- Так делается, пока не удастся дойти до начала слова.

BERT: токенизация

- При токенизации BERT жадно пытается выделить самый длинный токен из словаря, начиная с конца слова.
- Так делается, пока не удастся дойти до начала слова.
- При этом различаются токены в начале слова и не в начале:

$$\begin{aligned} \text{playing} &\to \text{play} + \#\#\text{ing}, \\ \text{replay} &\to \text{re} + \#\#\text{play}. \end{aligned}$$

- Наиболее частотные слова состоят из одного токена.

BERT: дообучение

- Задачи, на которых предобучается BERT, это классификация:
  - Языковое моделирование – классификация токенов.
  - Проверка следования – бинарная классификация CLS-токена.
- Непосредственно за классификацию отвечает верхний полносвязный слой с softmax-активацией.

BERT: дообучение

- Задачи, на которых предобучается BERT, это классификация:
  - Языковое моделирование – классификация токенов.
  - Проверка следования – бинарная классификация CLS-токена.
- Непосредственно за классификацию отвечает верхний полносвязный слой с softmax-активацией.
- Архитектура сети практически не изменится, если будет решаться другая задача:
  - Любая задача классификации предложений (или пар предложений).
  - Любая задача классификации отдельных слов.

BERT: дообучение

- При дообучении BERT на новую классификационную задачу заменяется только финальный слой:

$$\begin{aligned} \mathbf{p} &= \mathrm{softmax}(W\mathbf{h}), \\ W \in \mathbb{R}^{K*H} &- \text{обучаемая матрица}, \\ K &- \text{число классов}, \\ H &- \text{выходная размерность BERT (обычно 768)}. \end{aligned}$$

- Всего с нуля учится только несколько тысяч параметров (матрица W).

BERT: дообучение

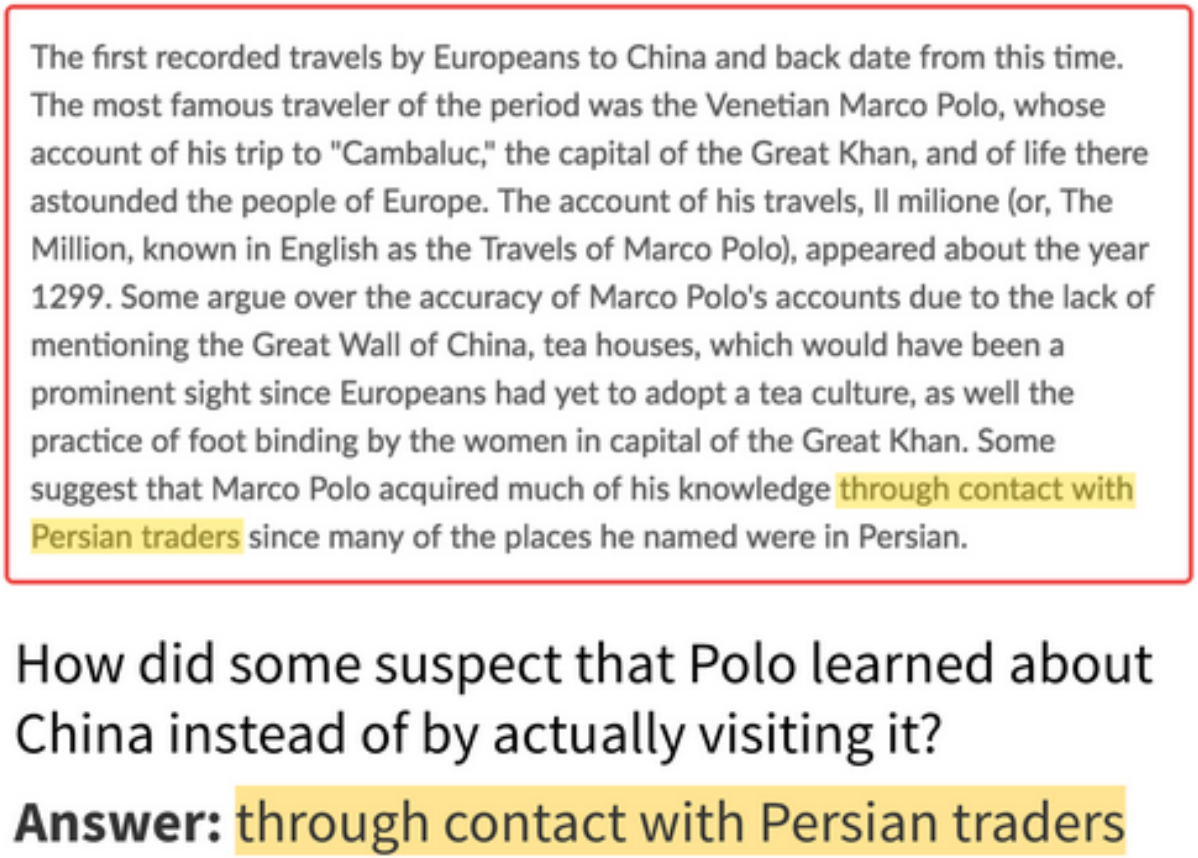
- При дообучении BERT на новую классификационную задачу заменяется только финальный слой:

$$\begin{aligned} \mathbf{p} &= \mathrm{softmax}(W\mathbf{h}), \\ W \in \mathbb{R}^{K*H} &- \text{обучаемая матрица}, \\ K &- \text{число классов}, \\ H &- \text{выходная размерность BERT (обычно 768)}. \end{aligned}$$

- Всего с нуля учится только несколько тысяч параметров (матрица W).
- Это значительно меньше, чем основной энкодер BERT ($\sim 2 * 10^8$ параметров).
- Эта часть сети выучится гораздо быстрее и на небольшом количестве данных.
- При этом веса основной части сети тоже доучивают (обычно они изменятся не так сильно).

BERT: дообучение

- Одна из популярных задач, где BERT сильно улучшил качество – SQuAD:



SQuAD: постановка задачи

- Одна из популярных задач, где BERT сильно улучшил качество – SQuAD.
- В ней требуется выделить в абзаце текста фрагмент, являющийся ответом на вопрос.
- Абзацы взяты из Википедии, вопросы составлены вручную.
- В SQuAD 2.0 появились вопросы, не содержащие ответа.

SQuAD: постановка задачи

- При составлении вопросов рекомендовалось использовать перефразировку, синонимы, гипонимы и гиперонимы:

Passage Segment

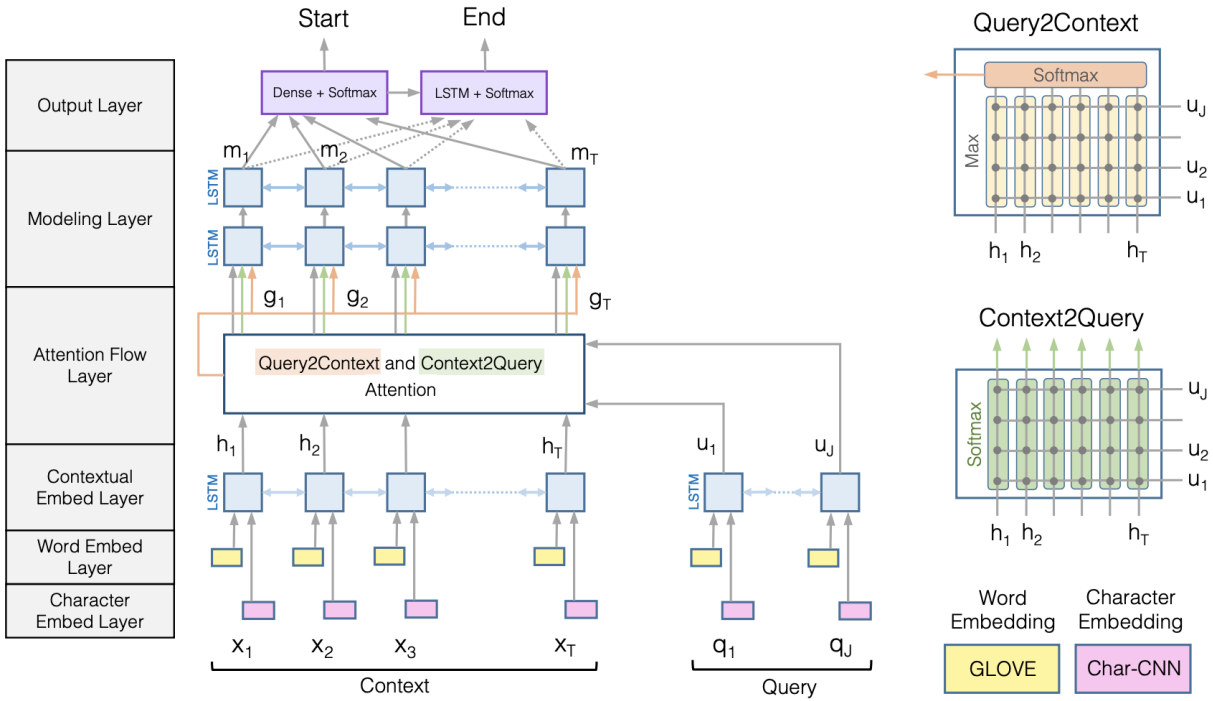
...The European Parliament and the Council of the European Union have powers of amendment and veto during the legislative process...

Question

Which governing bodies have veto power?

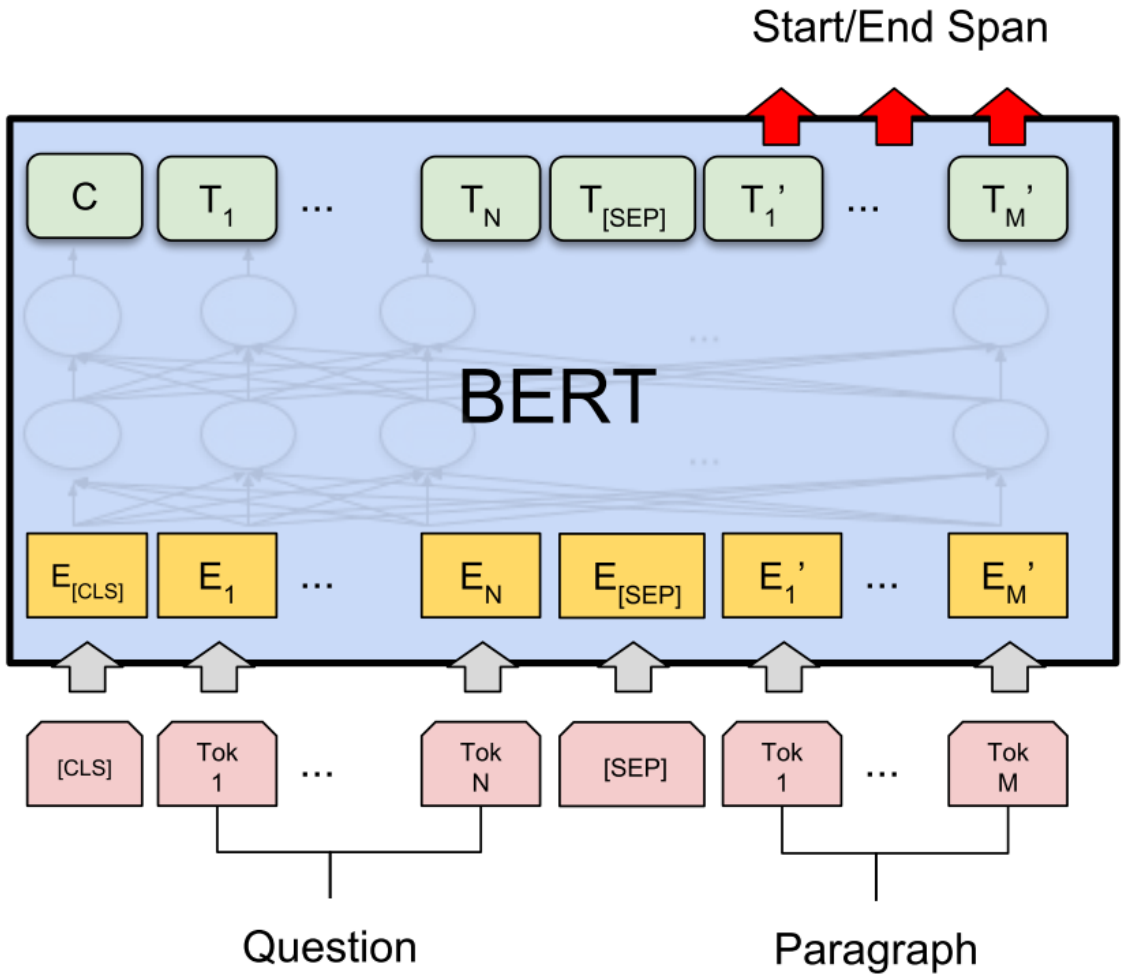
SQuAD: способы решения

- До появления BERT решали с помощью сопоставления между вопросом и контекстом.
- Например, с помощью разных вариантов внимания.
- Одна из архитектур – BiDAF (Bidirectional Attention Flow):



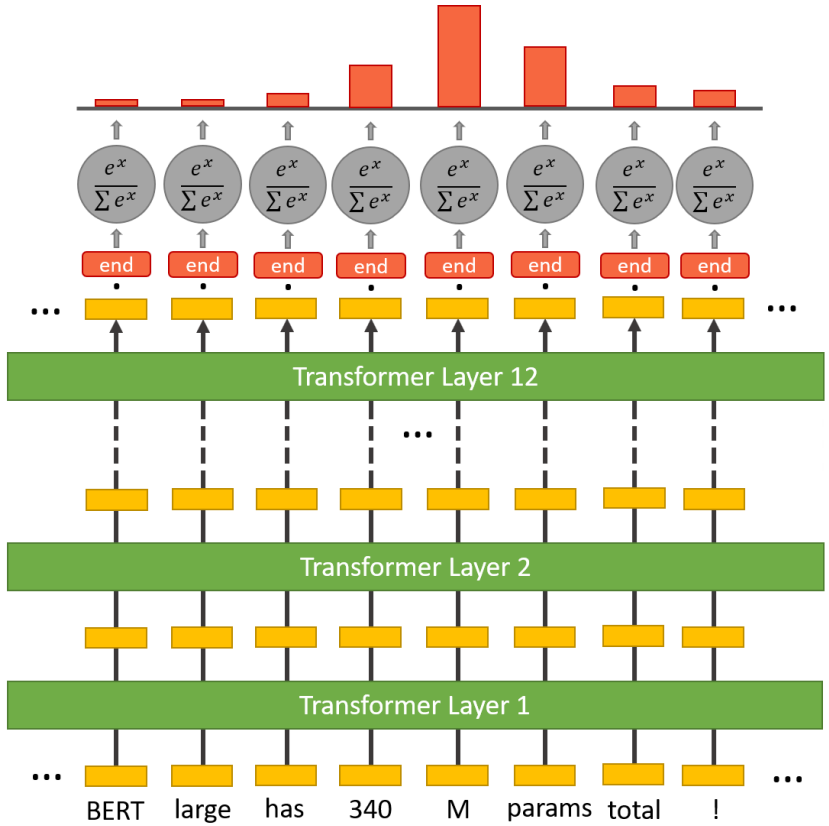
SQuAD: решение с помощью BERT

- Предсказываются позиции начала и конца фрагмента:



SQuAD: решение с помощью BERT

- Для каждой из границ ответ – распределение по токенам:



SQuAD: решение с помощью BERT

- Формальная архитектура (для позиции конца – аналогично):

$$\begin{aligned} [h_1, \ldots, h_n] &= \mathrm{BERT}([x_1, \ldots, x_n]), \\ a_i &= \langle w_S, x_i \rangle, \\ [p_1, \ldots, p_n] &= \mathrm{softmax}([a_1, \ldots, a_n]) \\ w_S &- \text{обучаемый вектор весов.} \end{aligned}$$

- Границы фрагмента выбираются как самая вероятная пара, где начало идёт раньше конца.

Мультиязычная BERT-модель

- Исходный BERT обучался на английской Википедии и корпусе художественной литературы.
- Мультиязычный BERT – на конкатенации 103 основных Википедий.
- Текст из более коротких Википедий сэмплировался с большим весом.

Мультиязычная BERT-модель

- Исходный BERT обучался на английской Википедии и корпусе художественной литературы.
- Мультиязычный BERT – на конкатенации 103 основных Википедий.
- Текст из более коротких Википедий сэмплировался с большим весом.
- Токенизация обучалась на тех же данных с тем же сэмплированием.
- В текст никак не включалась информация о языке.

Мультиязычная BERT-модель

- Исходный BERT обучался на английской Википедии и корпусе художественной литературы.
- Мультиязычный BERT – на конкатенации 103 основных Википедий.
- Текст из более коротких Википедий сэмплировался с большим весом.
- Токенизация обучалась на тех же данных с тем же сэмплированием.
- В текст никак не включалась информация о языке.
- Основное применение мультиязычной модели – настройка моделей для решения задач на конкретном языке.

Мультиязычная BERT-модель: недостатки

- Не все языки одинаково хорошо представлены в мультиязычной модели (точность языковой модели из Rönnquist et al., 2019):

	Mono	Multi
English	**45.92**	33.94
German	**43.93**	28.10
Swedish		22.30
Finnish		14.56
Danish		25.07
Norwegian (Bokmål)		25.21
Norwegian (Nynorsk)		22.28

Мультиязычная BERT-модель: недостатки

- Не все языки одинаково хорошо представлены в мультиязычной модели (точность языковой модели из Rönnquist et al., 2019):

	Mono	Multi
English	**45.92**	33.94
German	**43.93**	28.10
Swedish		22.30
Finnish		14.56
Danish		25.07
Norwegian (Bokmål)		25.21
Norwegian (Nynorsk)		22.28

- Токенизатор тоже может брать частотные фрагменты из другого языка:
  - године
  - року
  - було
  - ##лар.

Обучение BERT для языка

- BERT для любого языка можно было бы обучать так же, как английский.
- Однако обучение с нуля займёт много времени.
- С нуля надо обучать только BPE-токенизацию.

Обучение BERT для языка

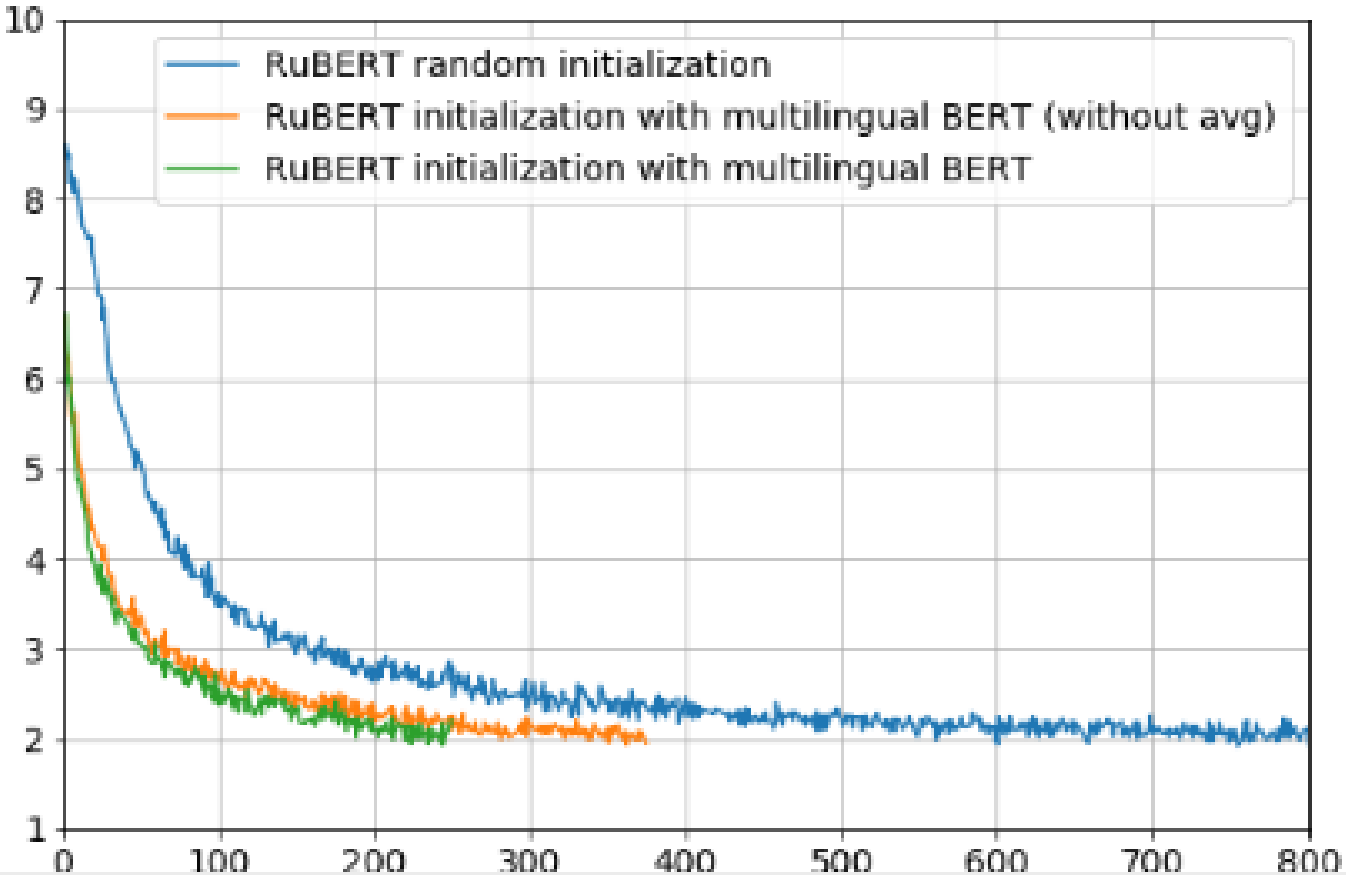
- BERT для любого языка можно было бы обучать так же, как английский.
- Однако обучение с нуля займёт много времени.
- С нуля надо обучать только BPE-токенизацию.
- Можно инициализировать веса слоёв и эмбеддинги сабтокенов весами мультиязычной модели.
- Проблема в том, что набор токенов в словаре поменялся.

Обучение BERT для языка

- BERT для любого языка можно было бы обучать так же, как английский.
- Однако обучение с нуля займёт много времени.
- С нуля надо обучать только BPE-токенизацию.
- Можно инициализировать веса слоёв и эмбеддинги сабтокенов весами мультиязычной модели.
- Проблема в том, что набор токенов в словаре поменялся.
- Решение: эмбеддинг токена инициализируется усреднением его мультиязычных эмбеддингов.

Обучение BERT для русского языка

- За счёт инициализации обучение существенно ускоряется (Kuratov and Arkhipov, 2019):



Обучение BERT для русского языка

- Улучшается качество на парафразе и ответах на вопросы:

model	F-1	Accuracy
Neural networks [11]	79.82	76.65
Classifier + linguistic features [11]	81.10	77.39
Machine Translation + Semantic similarity [6]	78.51	81.41
BERT multilingual	85.48 ± 0.19	81.66 ± 0.38
RuBERT	$\mathbf{87.73} \pm 0.26$	$\mathbf{84.99} \pm 0.35$

Table 1: ParaPhraser. We compare BERT based models with models in non-standard run setting, when all resources were allowed.

model	F-1 (dev)	EM (dev)
R-Net from DeepPavlov [2]	80.04	60.62
BERT multilingual	83.39 ± 0.08	64.35 ± 0.39
RuBERT	$\mathbf{84.60} \pm 0.11$	$\mathbf{66.30} \pm 0.24$

Обучение BERT для нескольких языков

- Можно обучать BERT и для нескольких родственных языков одновременно:

Model	Span F_1	RPM	REM	SM
Bi-LSTM-CRF (Lample et al., 2016)	75.8	73.9	72.1	72.3
Multilingual BERT [5]	79.6	77.8	76.1	77.2
Multilingual BERT-CRF	81.4	80.9	79.2	79.6
Slavic BERT	83.5	83.8	82.0	82.2
Slavic BERT-CRF	87.9	85.7 (90.9)	84.3 (86.4)	84.1 (85.7)

Table 1: Metrics for BSNLP on validation set (Asia Bibi documents). Metrics on the test set are in the brackets.

Обучение BERT для нескольких языков

- Побочный недостаток дообучения BERT: падает качество на родственных языках (пример для задачи морфосинтаксического анализа, Sorokin, 2019)

Training data	BERT	Tag	Tag sent	LAS	Sent LAS	UAS	Sent UAS
be	multlingual	85,09	10,29	76,34	14,71	83,72	17,65
be	Russian	80,75	4,41	45,66	1,47	57,45	4,41
be+ru+uk	multlingual	**88,57**	**19,12**	**84,8**	**16,18**	**90,74**	**33,82**
be+ru+uk	Russian	83,79	7,35	59,3	1,47	68,74	4,41

- То есть модель очень сильно переобучается под конкретный язык.