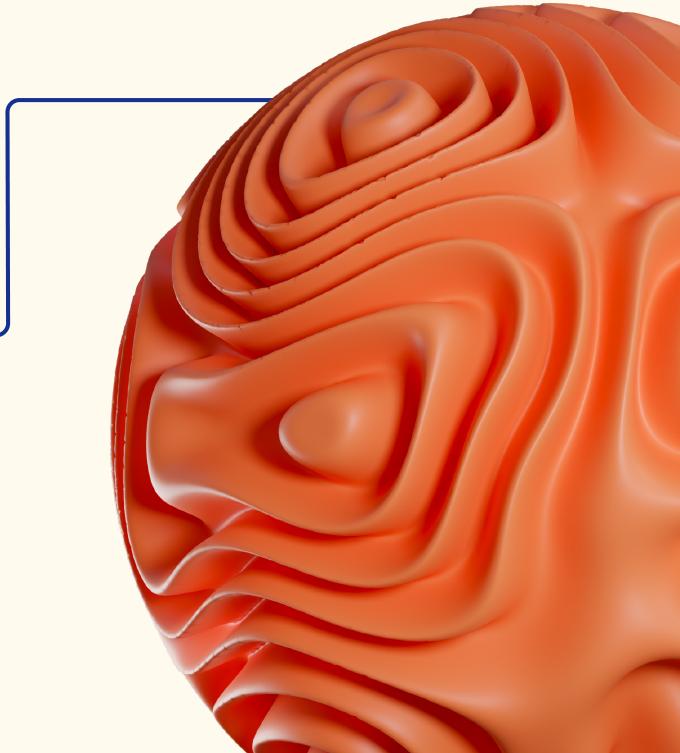


Задачи генерации в NLP

Методы генерации текста и задача машинного перевода

Сидоров Никита, SberDevices



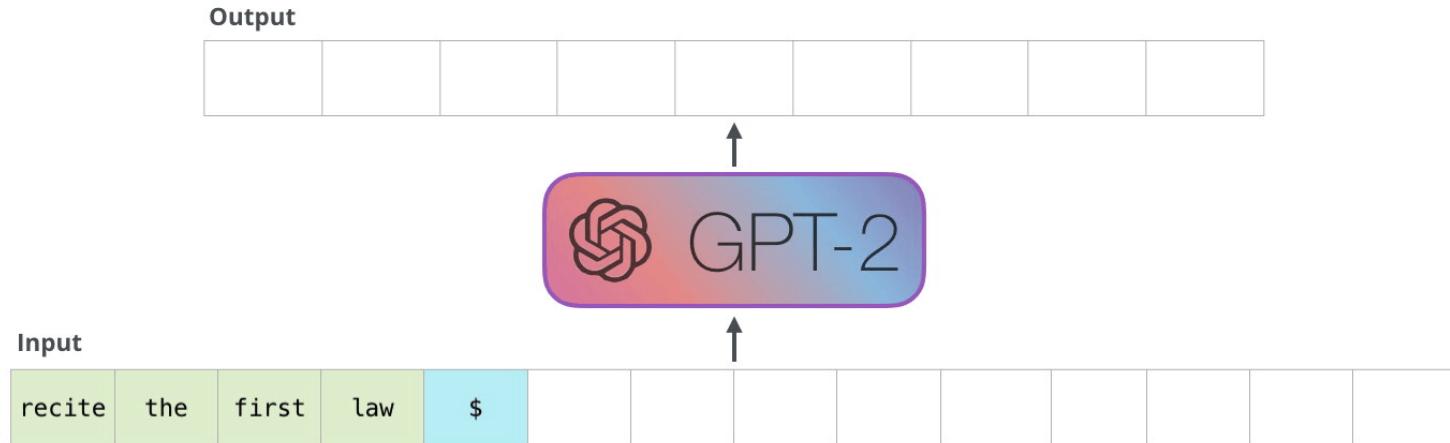
Цели занятия

- Вспомнить особенности decoder-only трансформеров и encoder-decoder трансформеров
- Изучить методы токенизации по подсловам
- Рассмотреть методы сэмплирования текста
- Вспомнить задачу машинного перевода и метрики связанные с ней

План занятия

1. Повторение архитектуры Transformer
2. Методы токенизации по подсловам
3. Методы сэмплирования текста для генерации
4. Отличие задачи генерации от задачи извлечения информации
5. Задача машинного перевода и ее метрики

Архитектура Transformer (Повторение)



Great material for
understanding attention
by [Jay Alammar](#)

Архитектура Transformer (Повторение)



THE TRANSFORMER

DECODER BLOCK

Feed Forward Neural Network

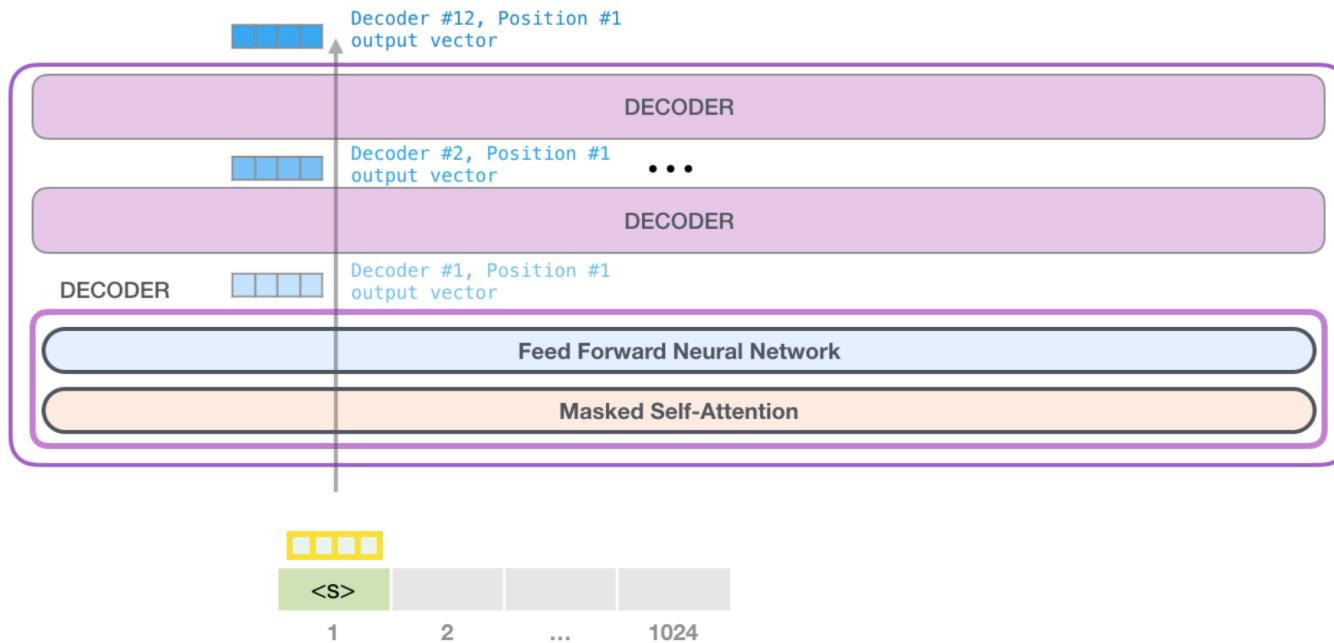
Encoder-Decoder Self-Attention

Masked Self-Attention

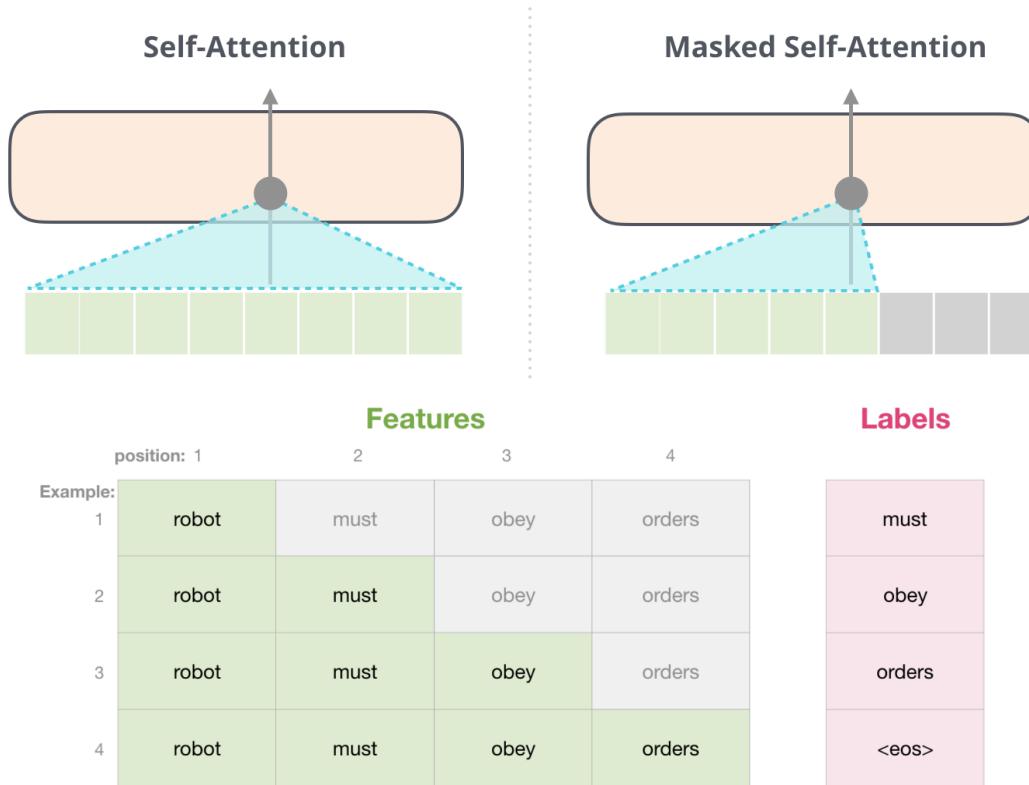
Input

<s>	robot	must	obey					512
1	2	3	4	5	6			

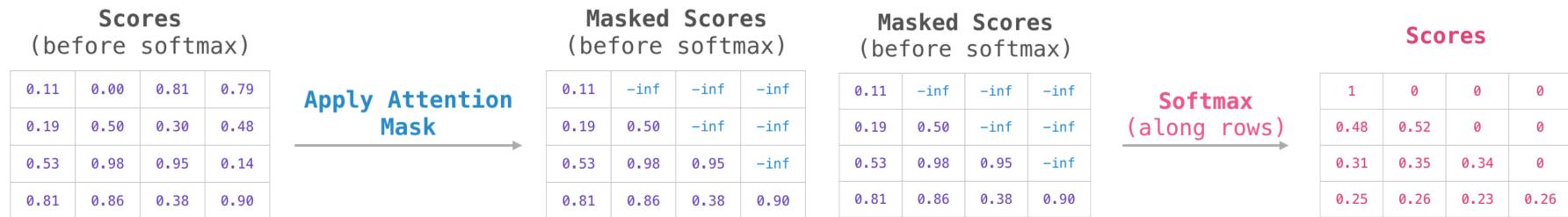
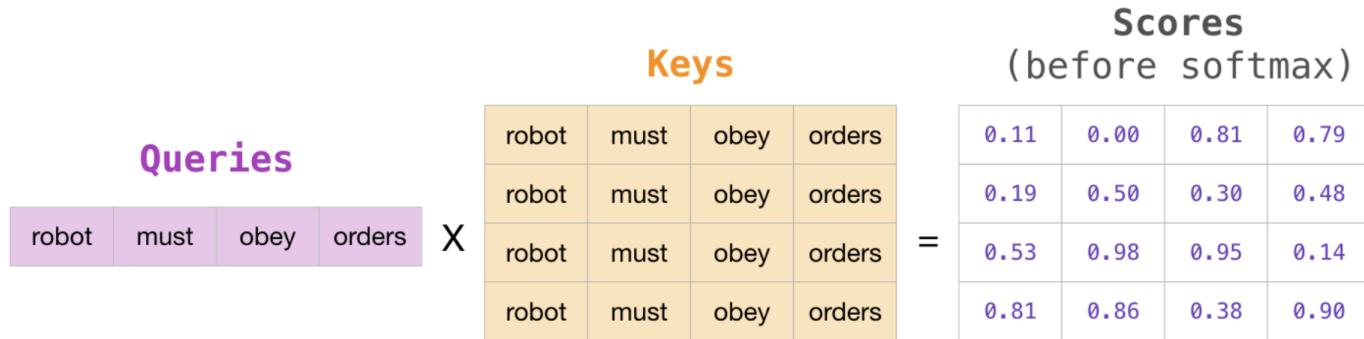
Архитектура Transformer (Повторение)



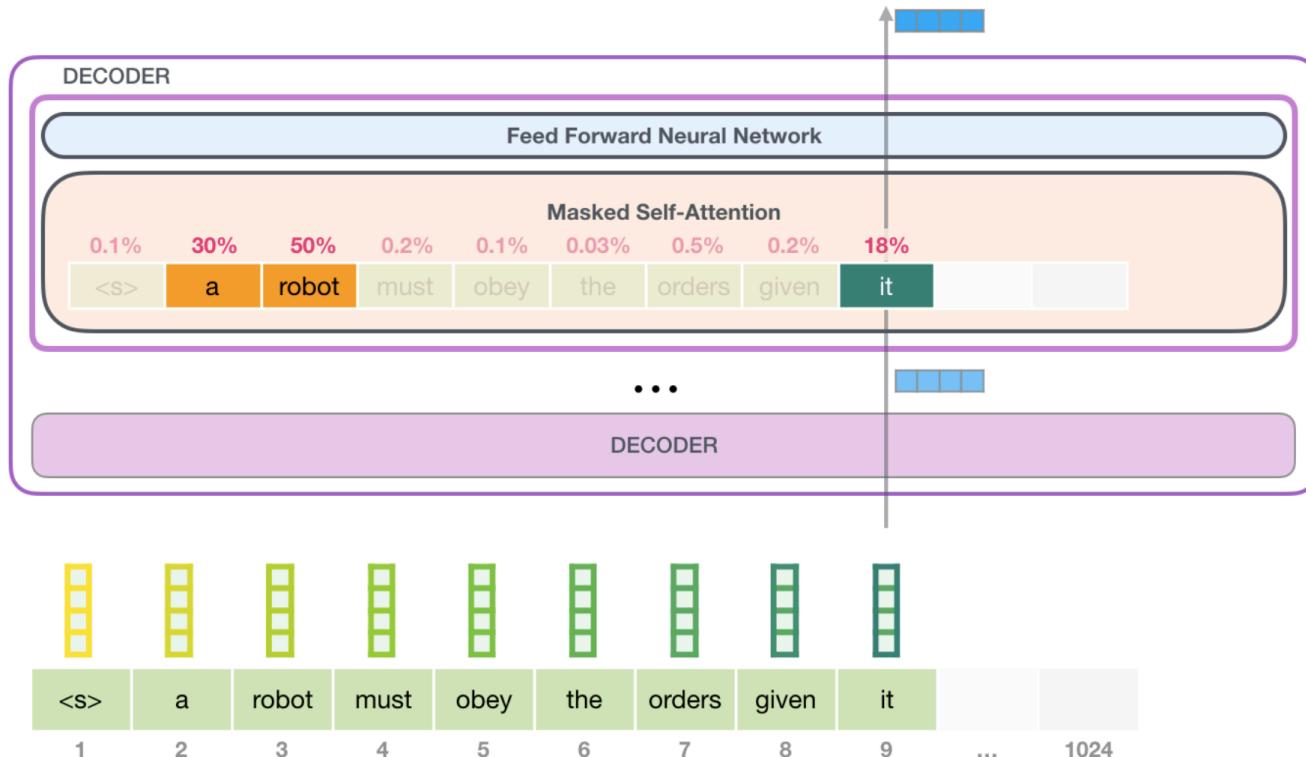
Архитектура Transformer (Повторение)



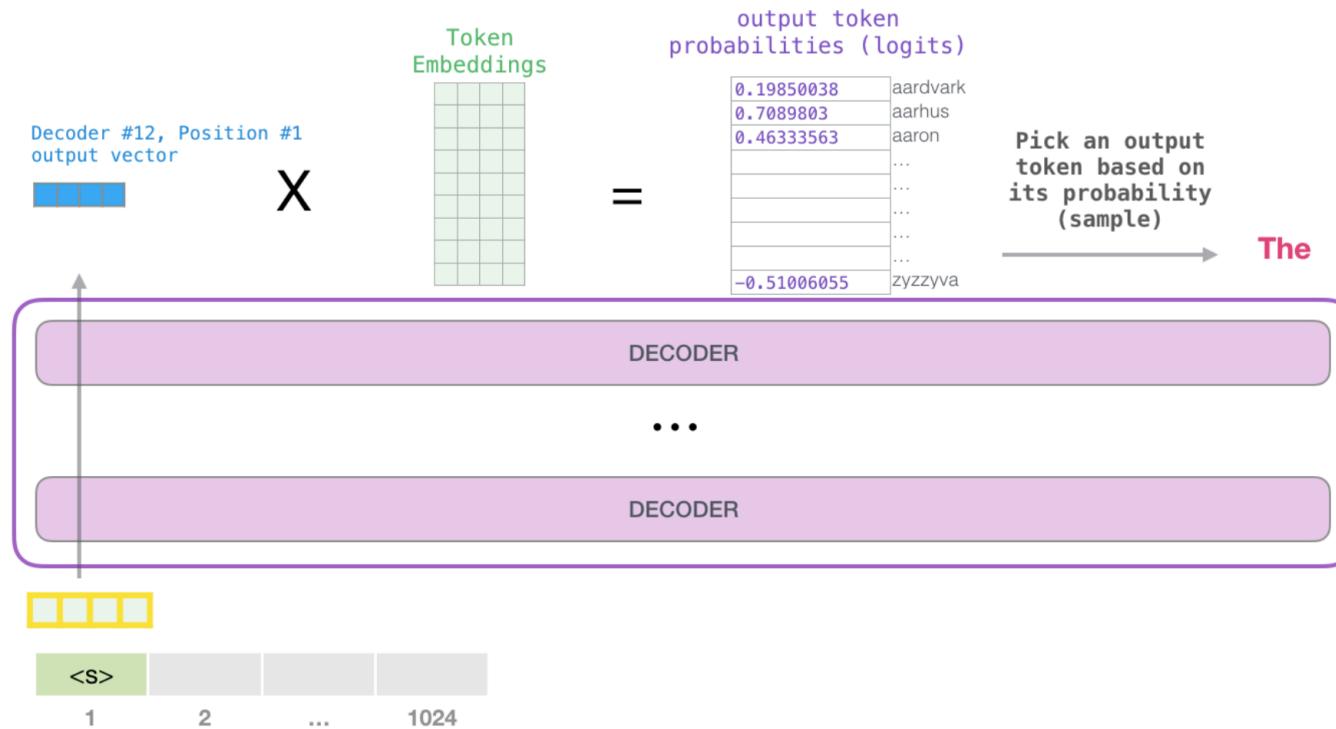
Архитектура Transformer (Повторение)



Архитектура Transformer (Повторение)

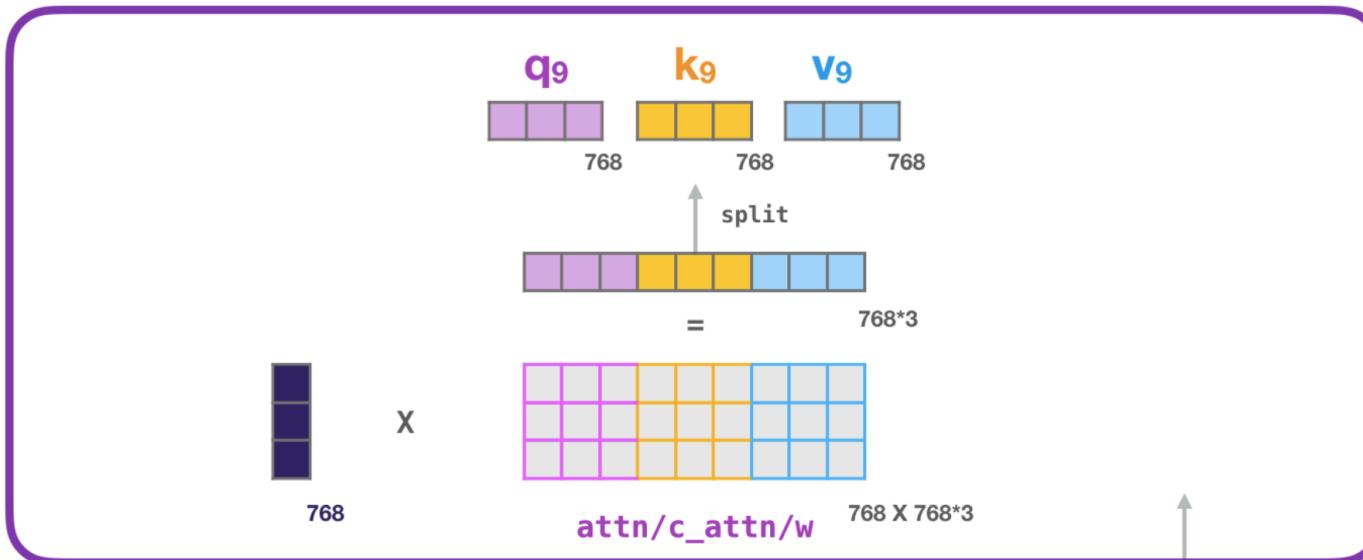


Архитектура Transformer (Повторение)



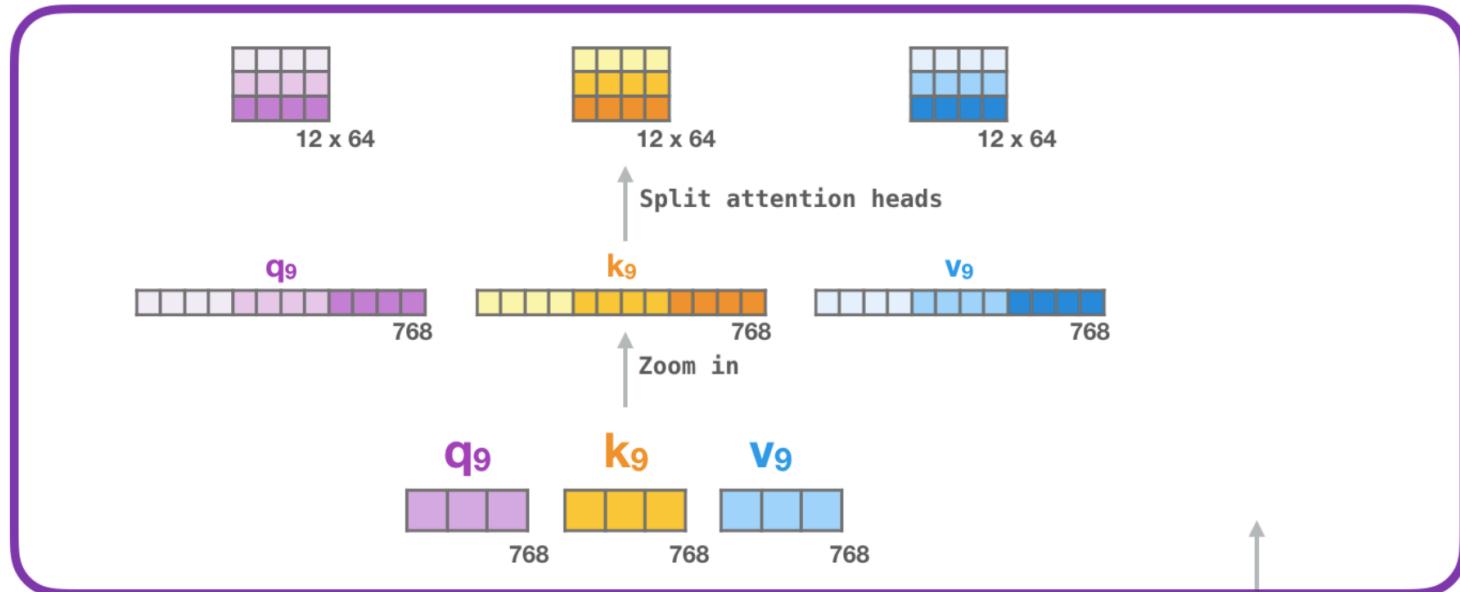
Архитектура Transformer (Повторение)

GPT2 Self-Attention



Архитектура Transformer (Повторение)

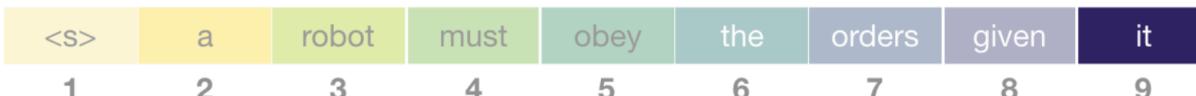
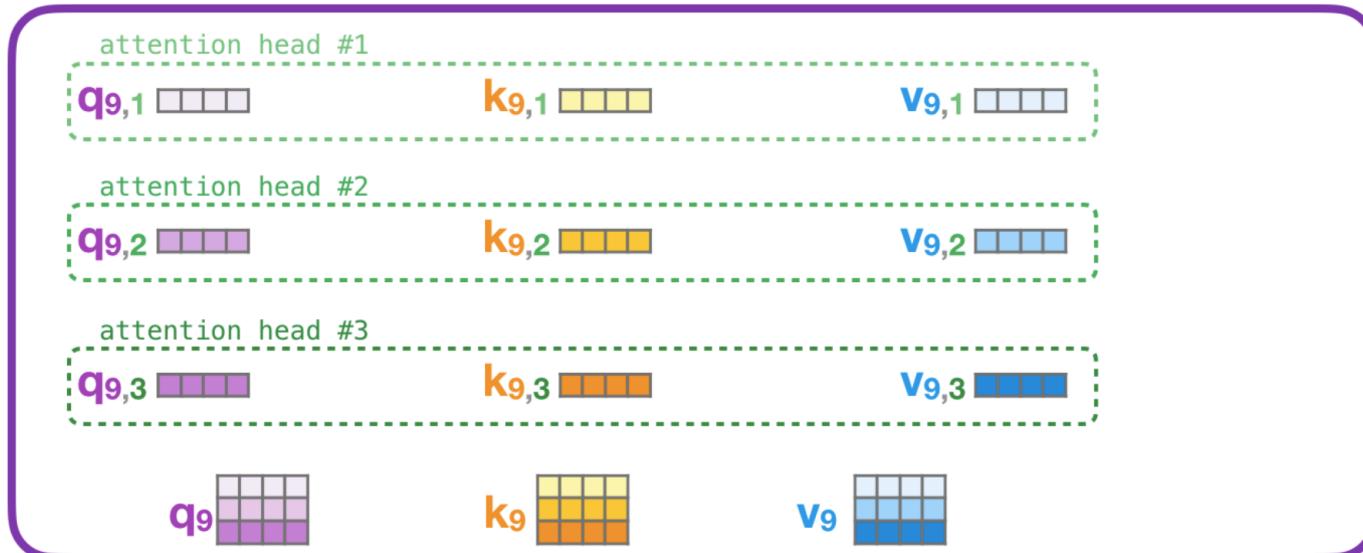
GPT2 Self-Attention



<S>	a	robot	must	obey	the	orders	given	it
1	2	3	4	5	6	7	8	9

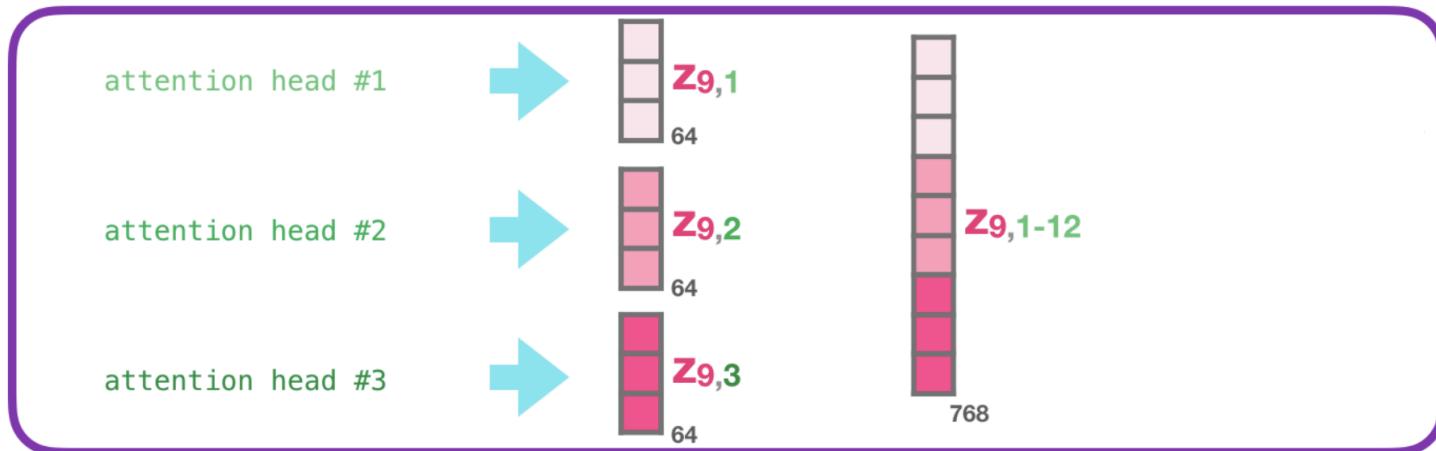
Архитектура Transformer (Повторение)

GPT2 Self-Attention

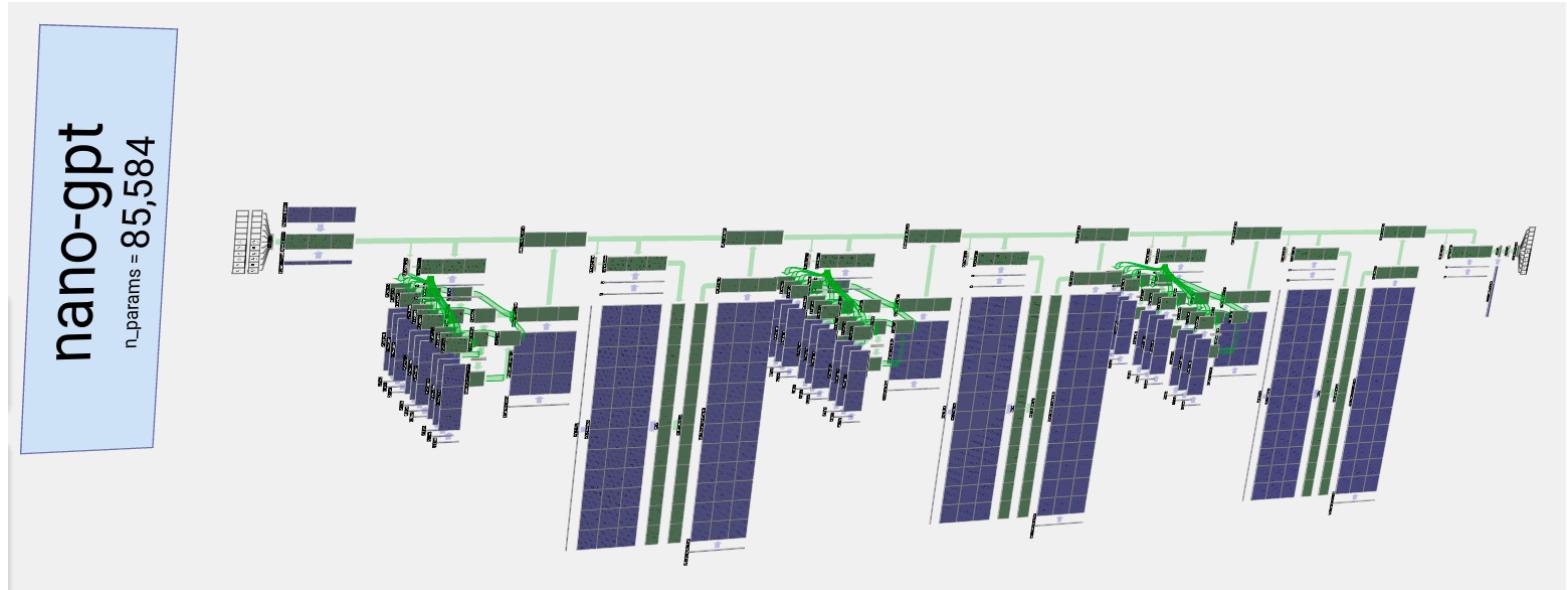


Архитектура Transformer (Повторение)

GPT2 Self-Attention

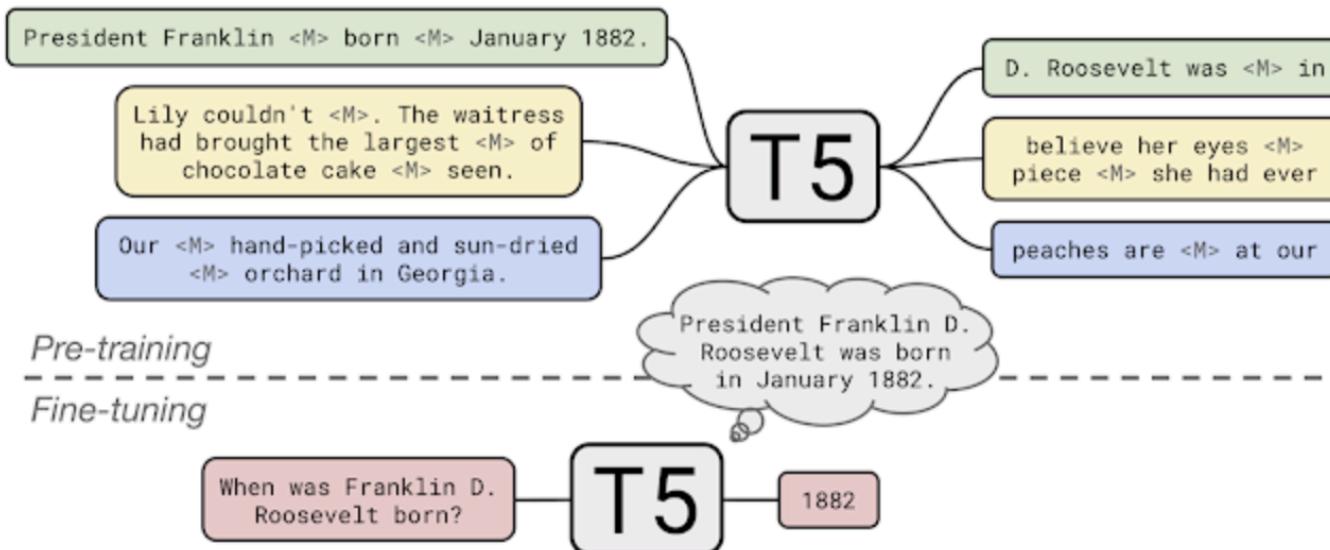


3-D визуализация GPT



<https://bbycroft.net/llm>

Encoder-decoder модель T5



Токенизация

Токенизация – это процесс разбиения текста на меньшие единицы, называемые токенами. Токены могут быть словами, фразами, знаками препинания и другими элементами текста.

Токенизация

Цель токенизации — упростить последующую обработку текста путем его структурирования на стандартизованные единицы и привести к формату, пригодному для последующего использования.

Требования к токенизации

- Согласованность
- Универсальность
- Скорость работы
- Помехоустойчивость

Виды токенизации

Sentence: “It is raining.”

Character level tokenization



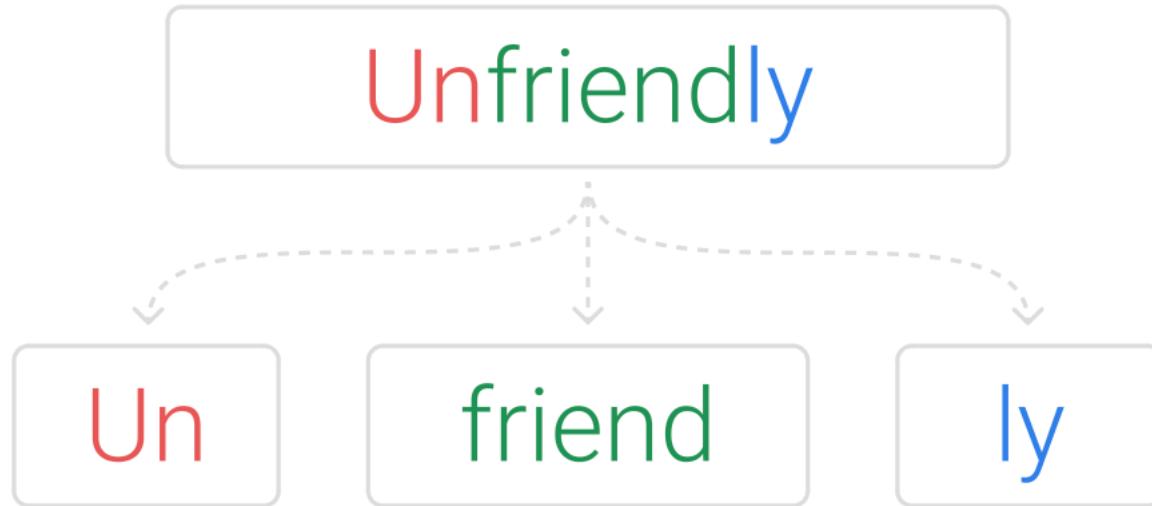
Word level tokenization



Sub-word level tokenization



Токенизация по подсловам



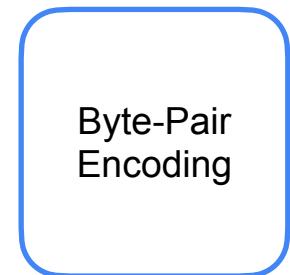
Токенизация по подсловам



BERT, DistilBERT

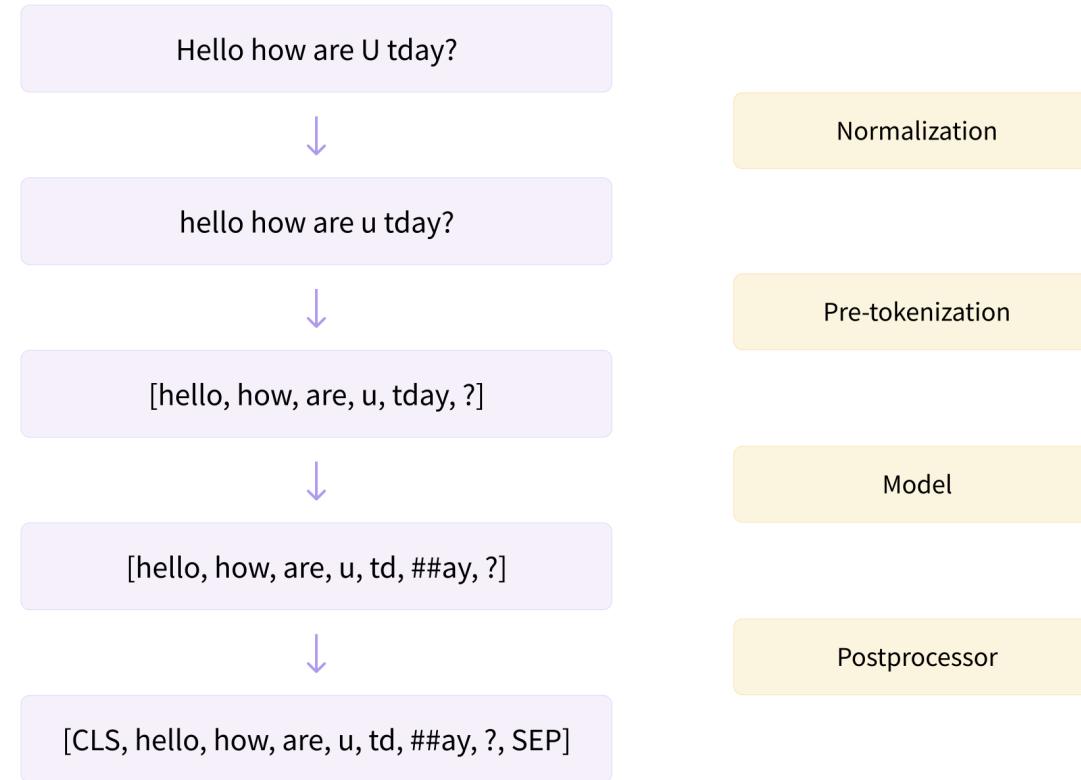


XLNet, ALBERT



GPT-like, RoBERTa

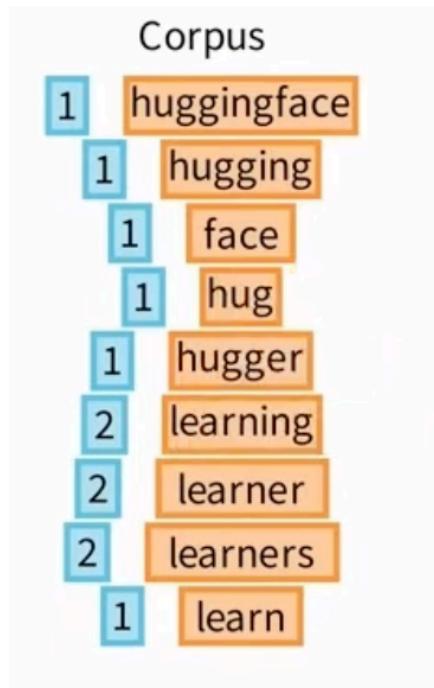
Токенизация по подсловам



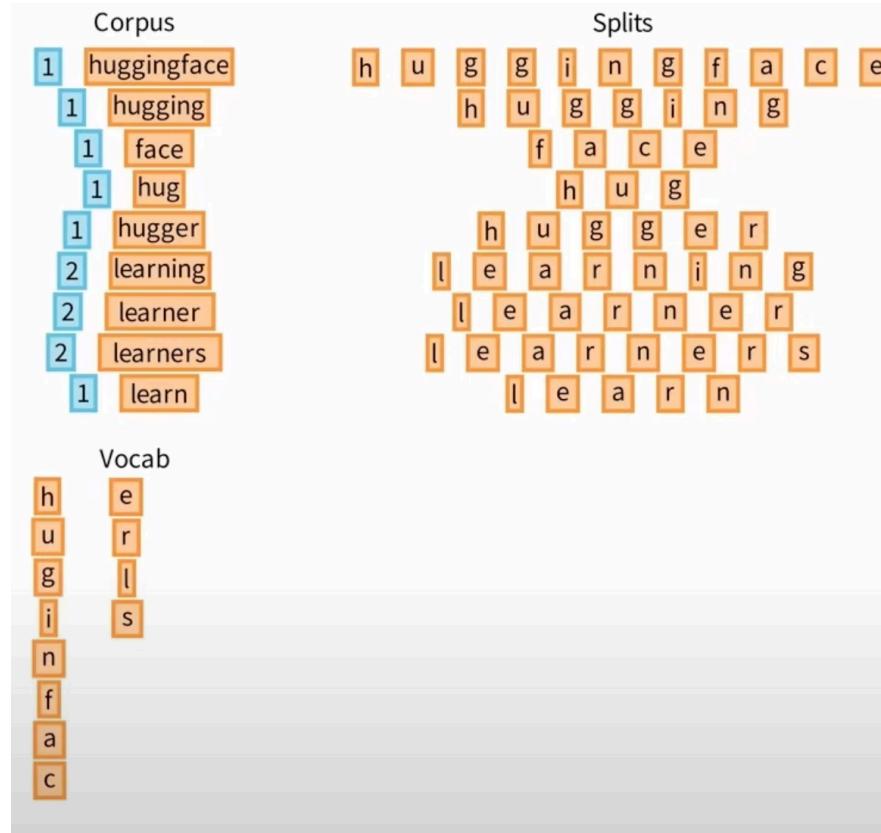
Byte-Pair Encoding

1. Считаем все уникальные слова в тексте
2. Создаем базовый словарь, который включает в себя все символы, которые используются в словах
3. Итеративно расширяем словарь за счет объединения двух последовательно идущих токенов словаря, которые встречаются чаще всего. Расширяем до тех пор, пока не достигнем заданного размера словаря.
4. Готово 

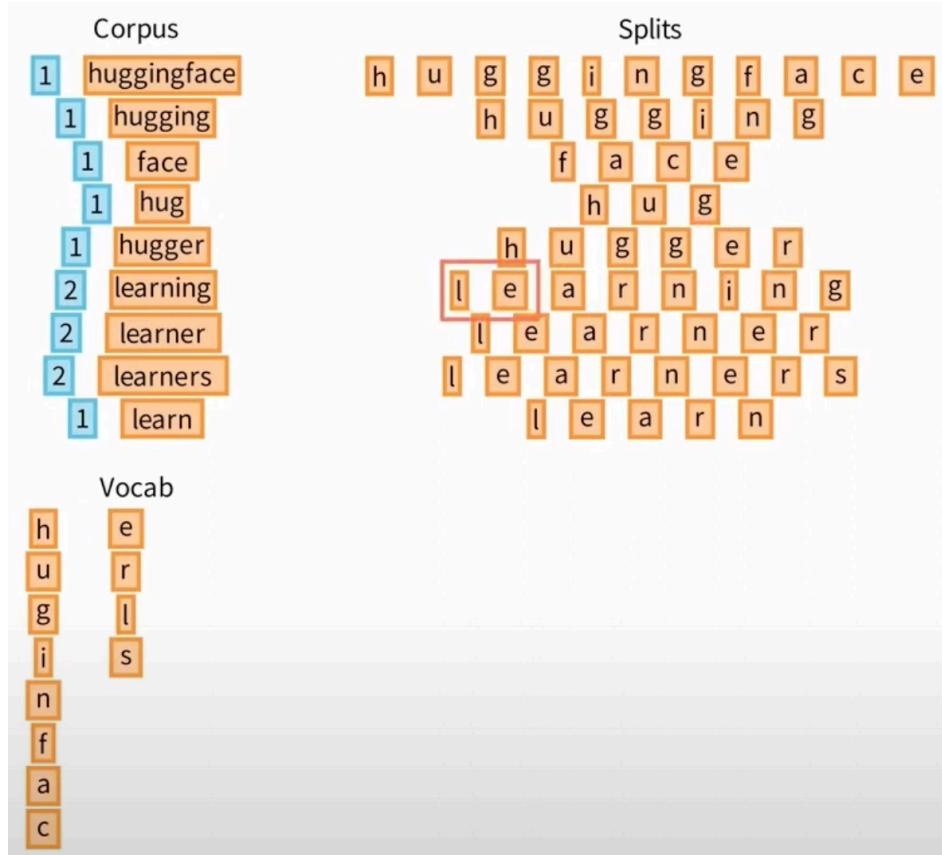
Byte-Pair Encoding



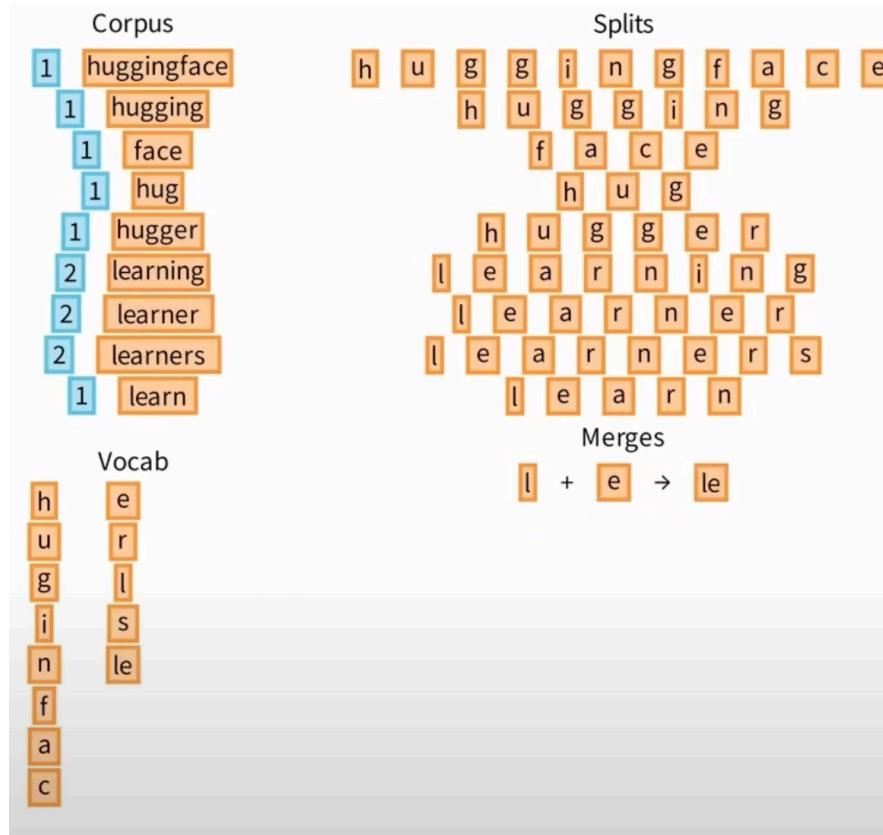
Byte-Pair Encoding



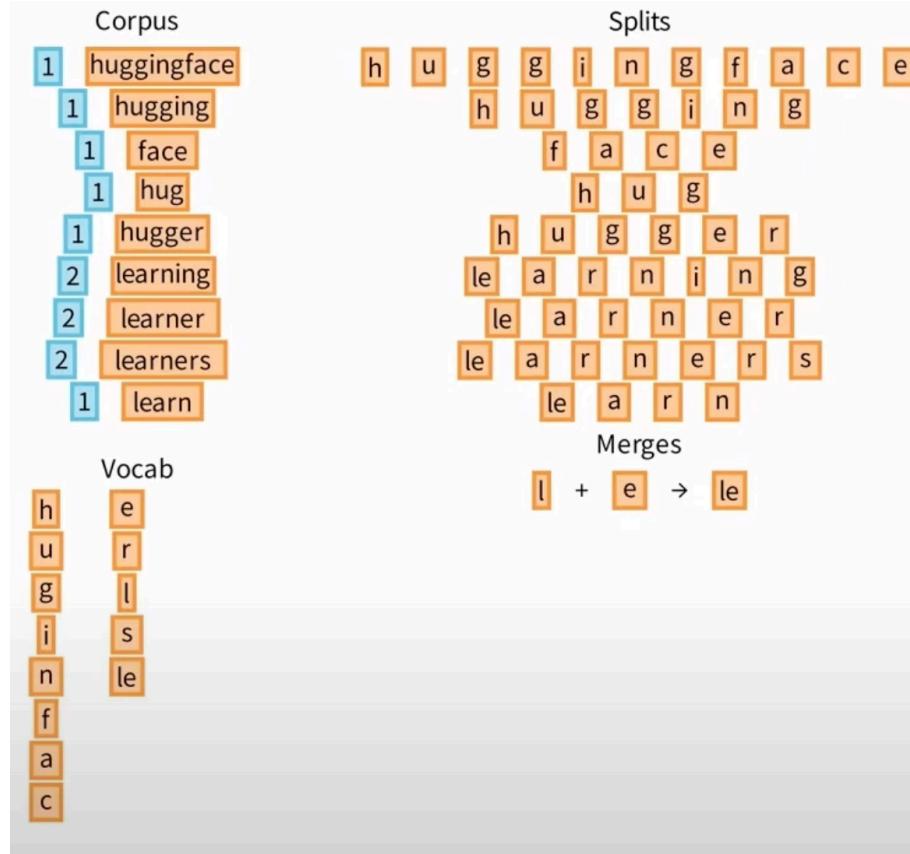
Byte-Pair Encoding



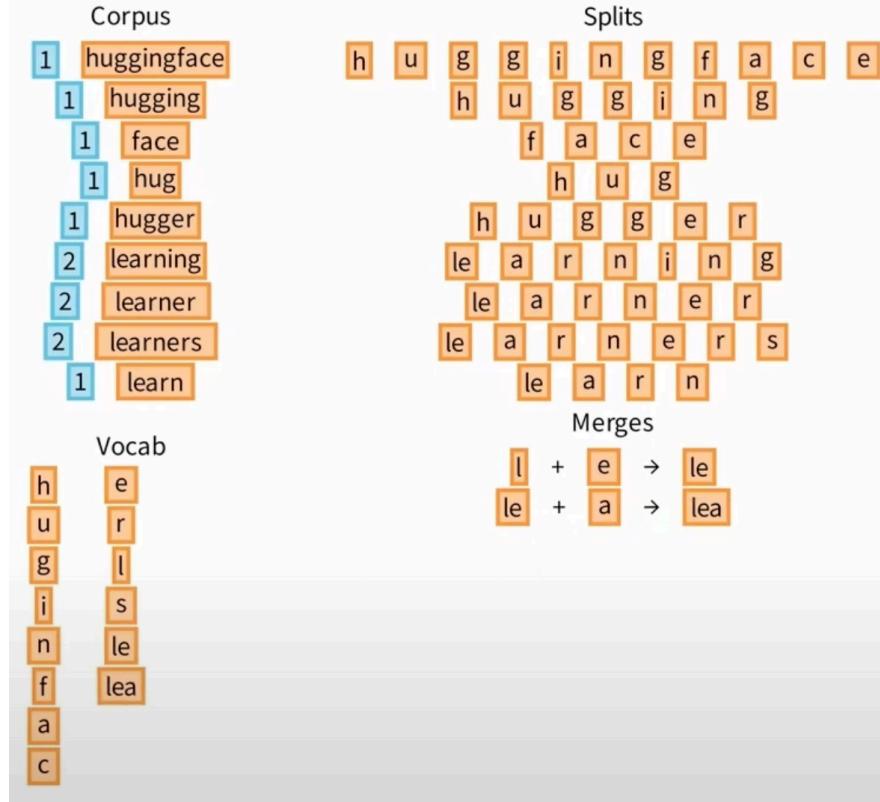
Byte-Pair Encoding



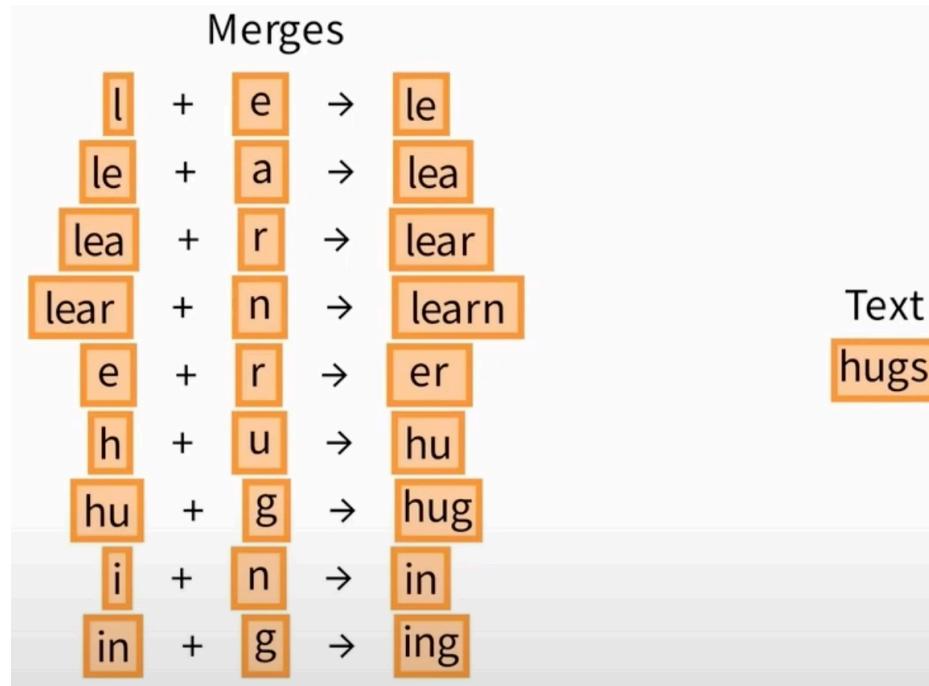
Byte-Pair Encoding



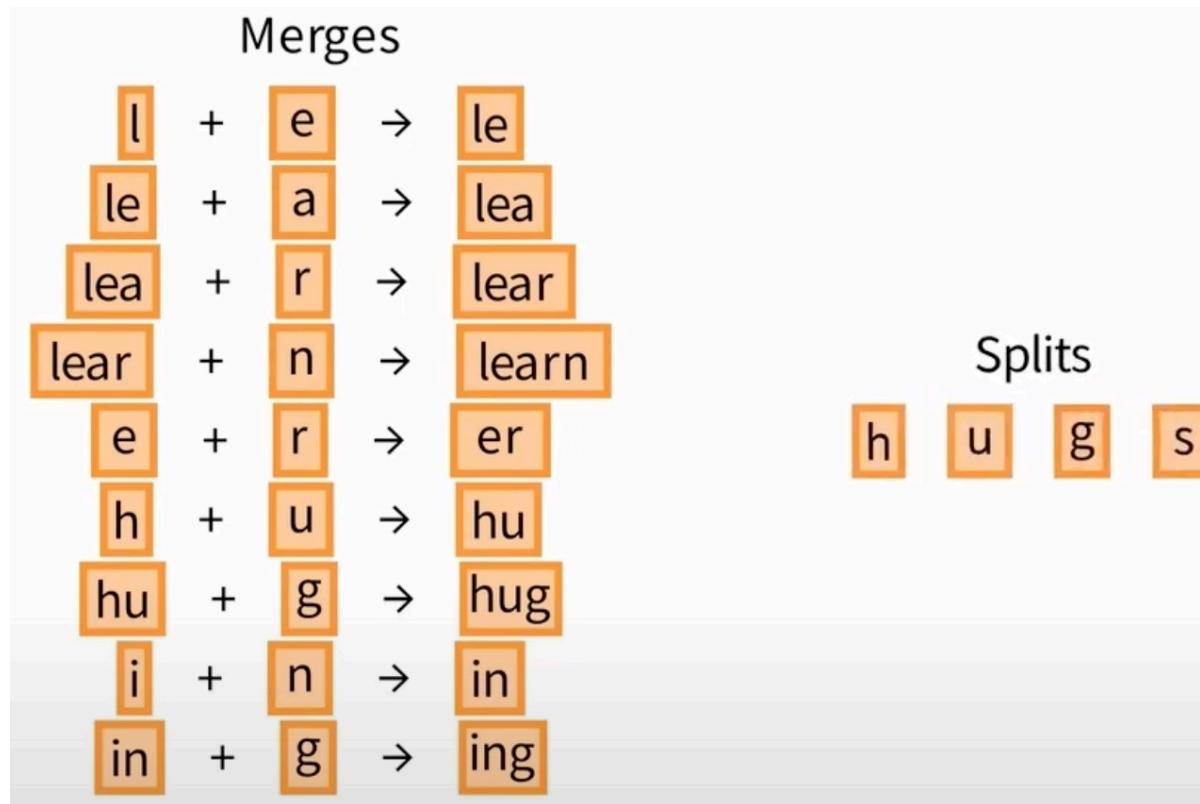
Byte-Pair Encoding



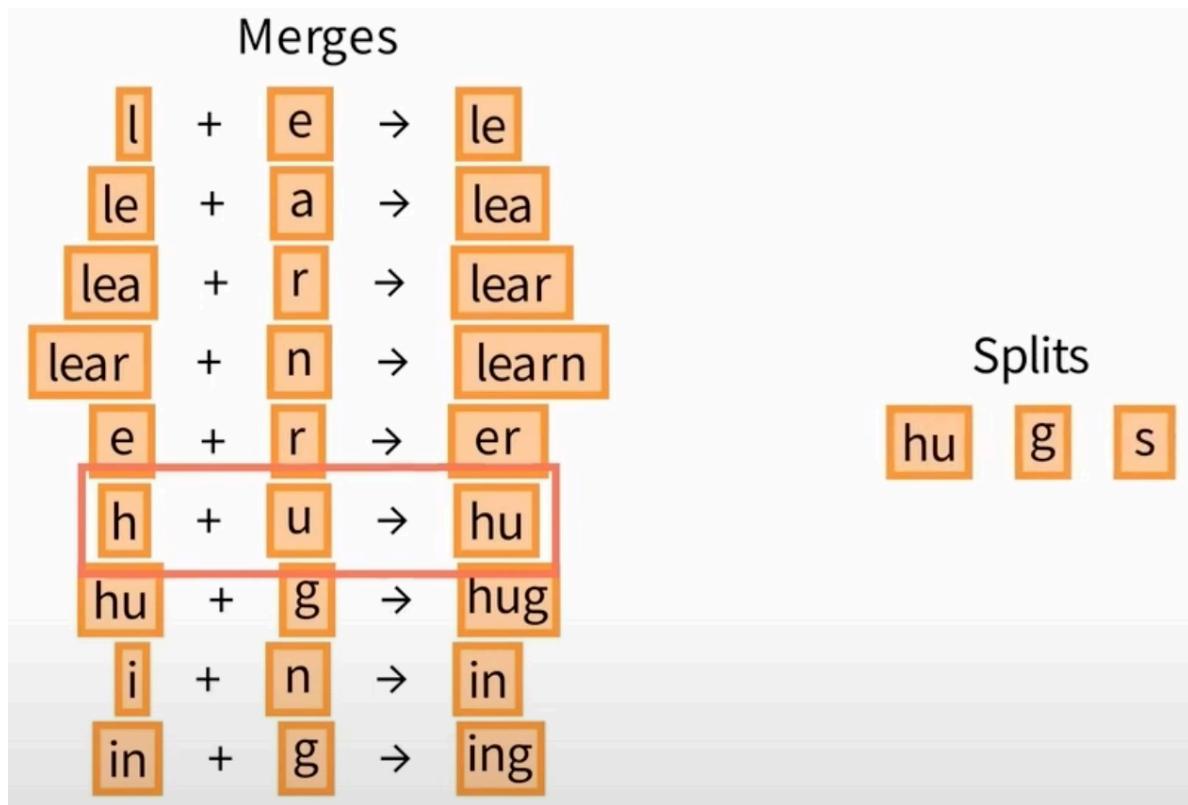
Byte-Pair Encoding, токенизация слов



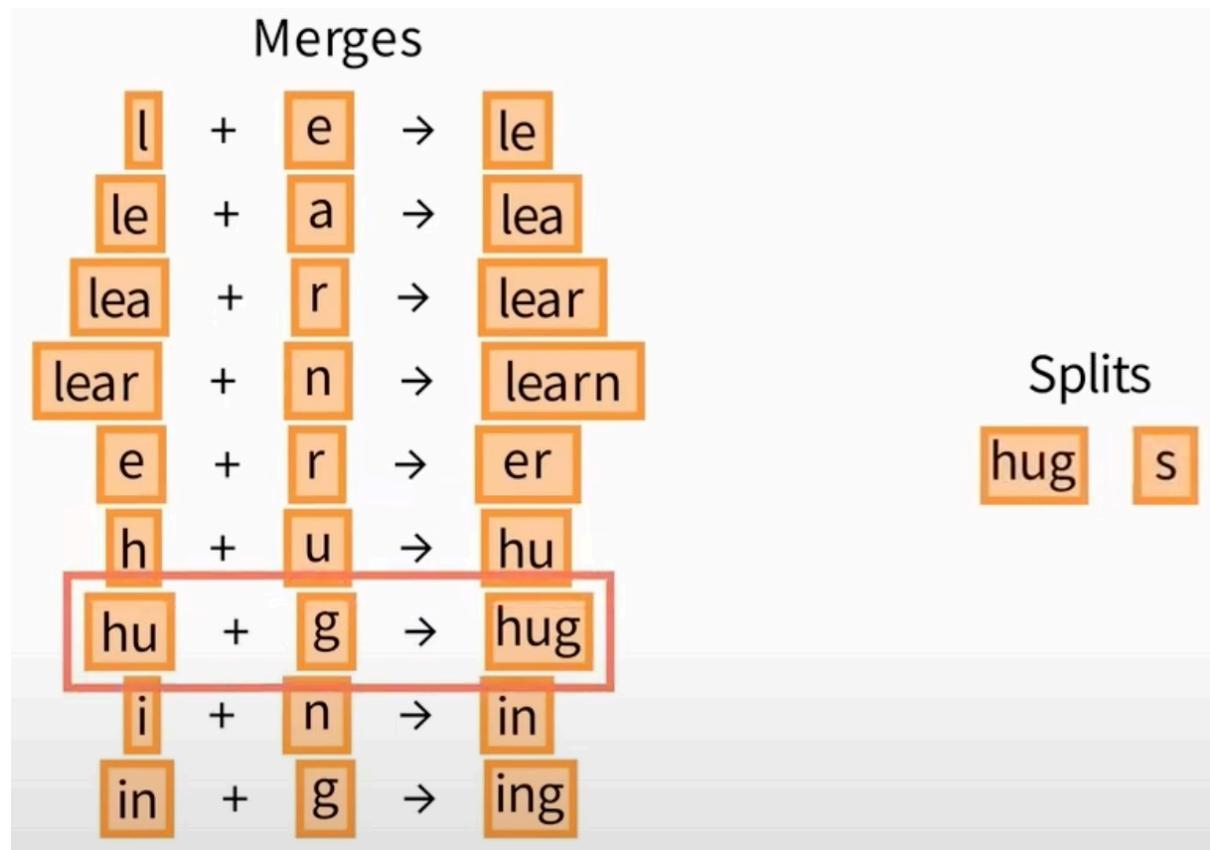
Byte-Pair Encoding, токенизация слов



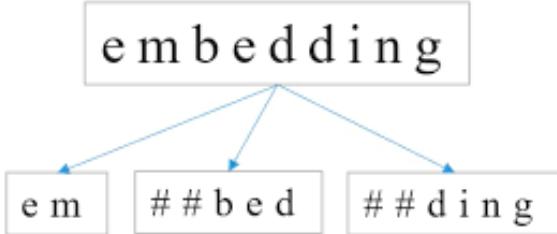
Byte-Pair Encoding, токенизация слов



Byte-Pair Encoding, токенизация слов



WordPiece



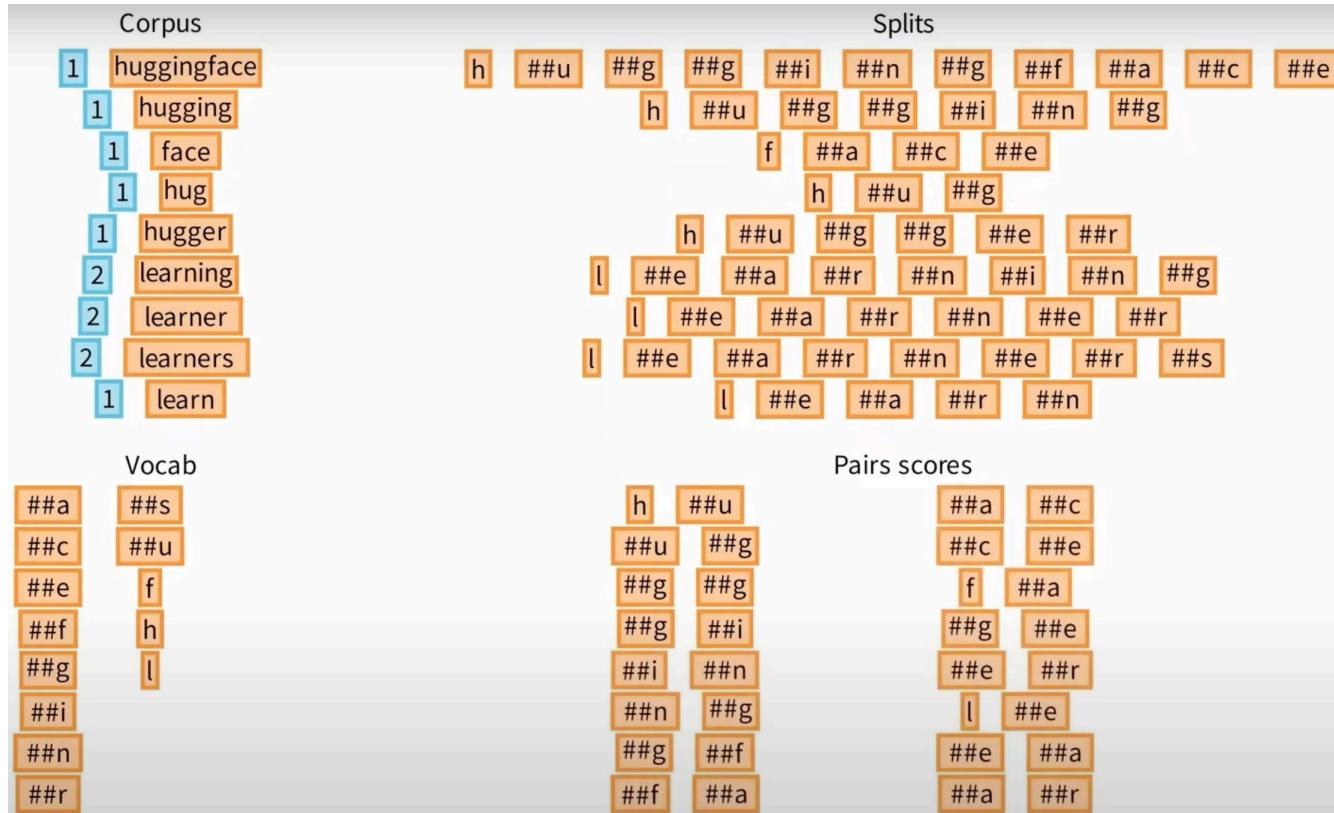
1. Считаем все уникальные слова в тексте
2. Создаем базовый словарь, который включает в себя все символы, которые встречаются в словах, символы которые встречаются не в начале слова дополняем двумя #
3. Итеративно расширяем словарь за счет объединения двух последовательно идущих токенов словаря, у которых максимальная оценка. Расширяем до тех пор, пока не достигнем заданного размера словаря.
4. Готово 😊

WordPiece (Score)

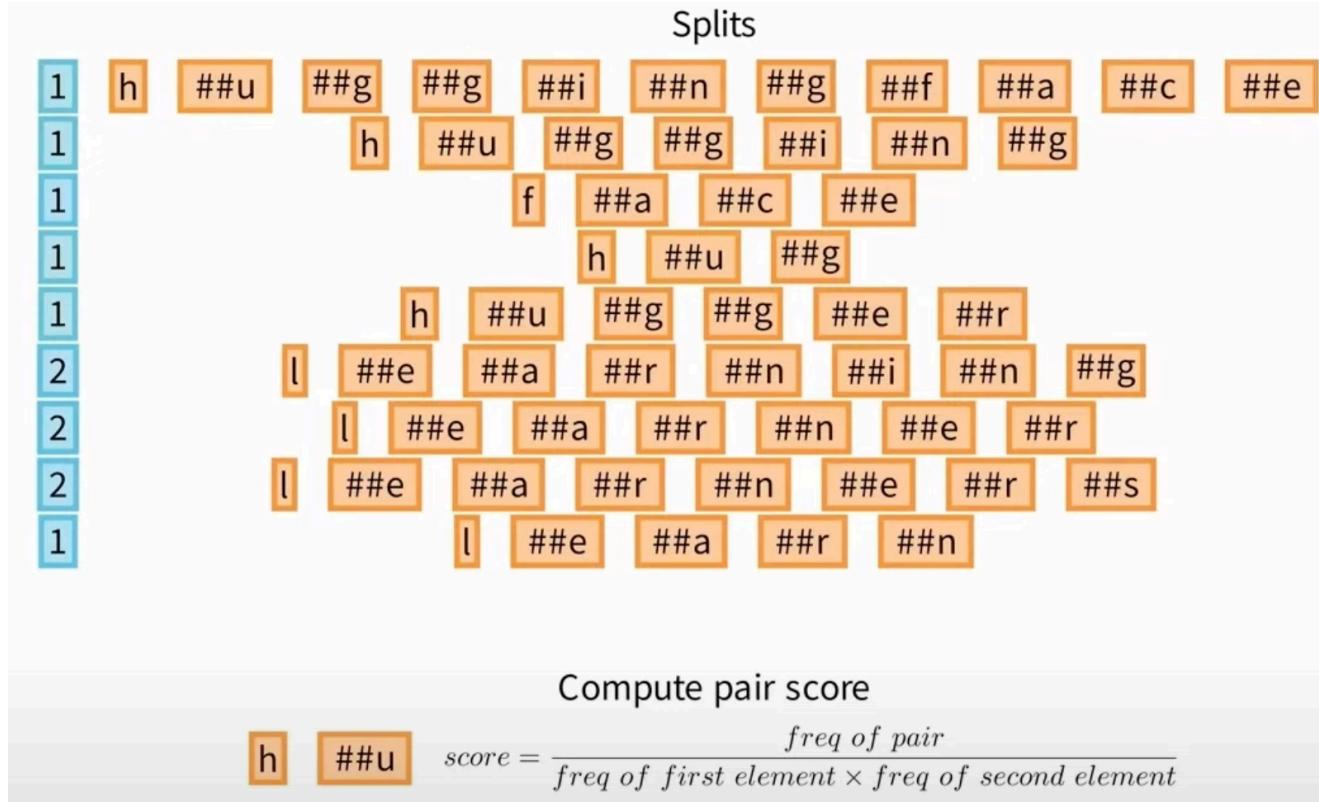
Вместо объединения самой популярной пары соседних токенов, WordPiece приоритезирует объединение пар токенов, у которых токены по отдельности встречаются в словаре реже всего

$$\text{score} = \frac{\text{Freq of pair}}{\text{Freq of first element} \times \text{Freq of second element}}$$

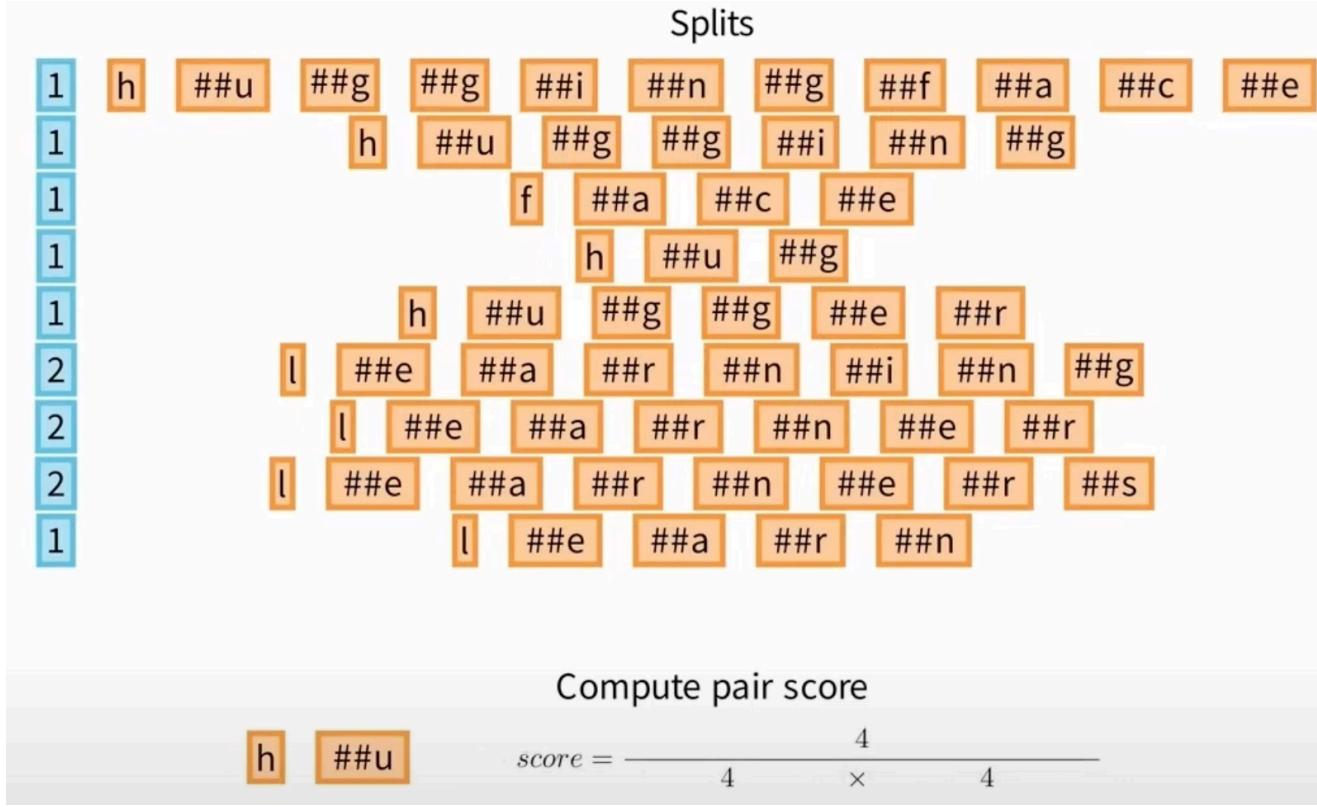
WordPiece



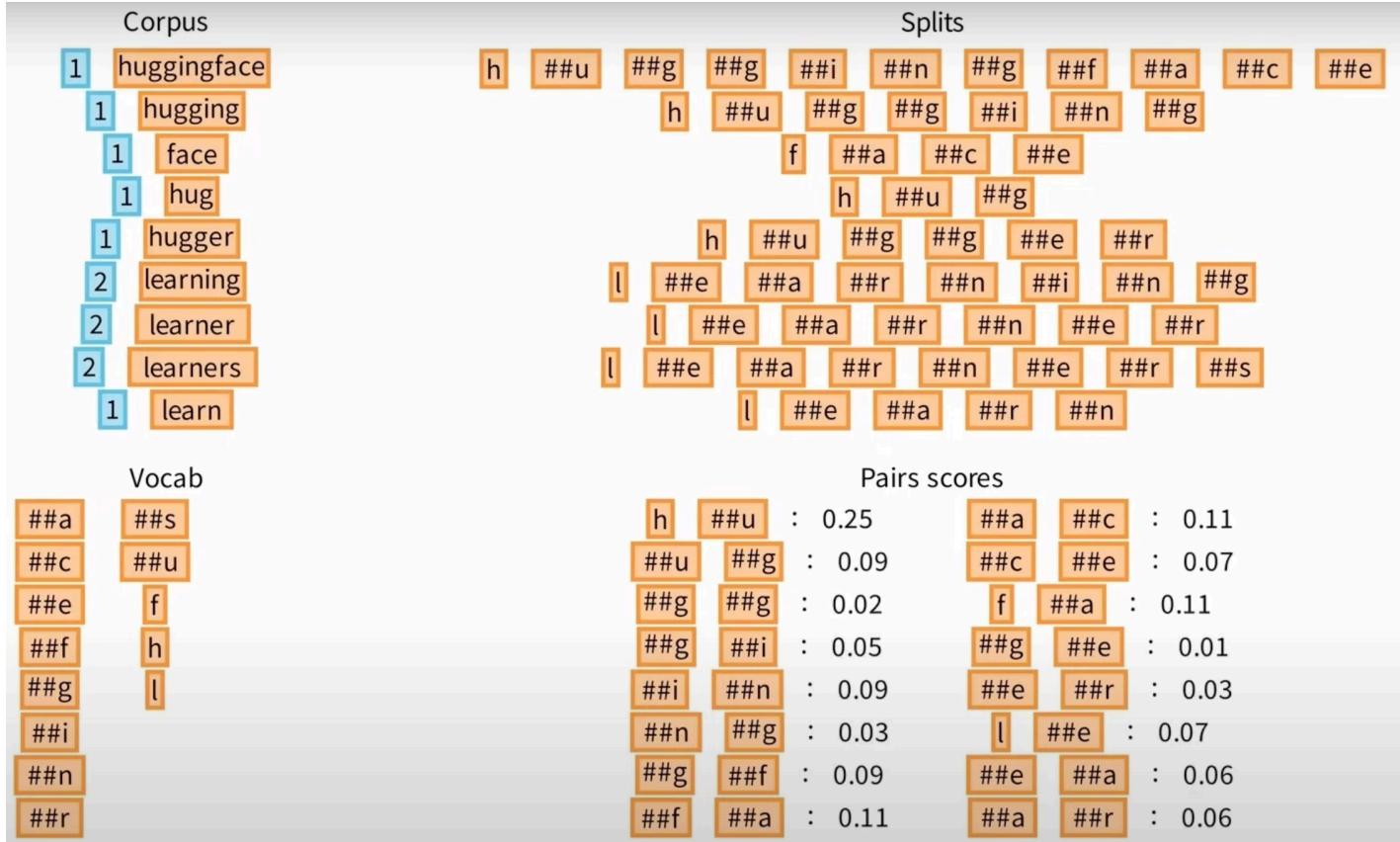
WordPiece



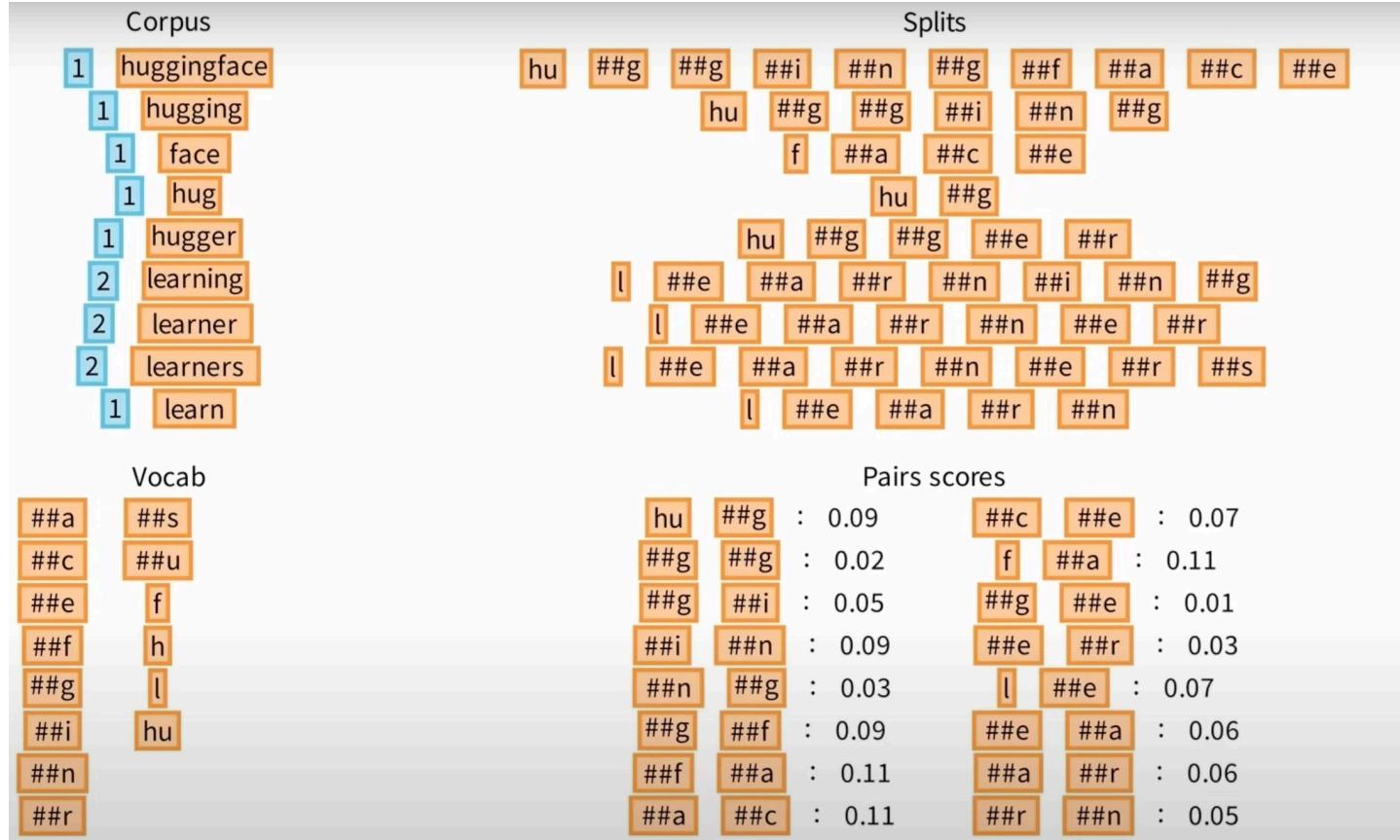
WordPiece



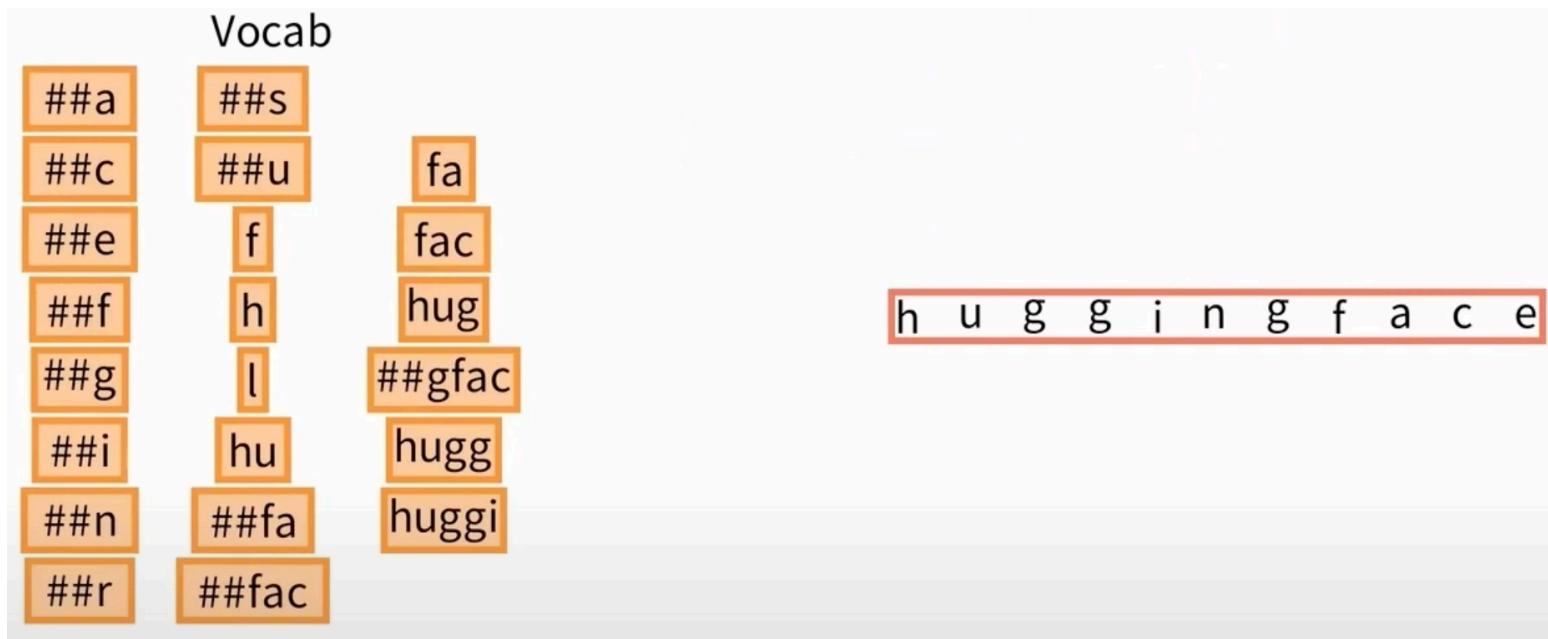
WordPiece



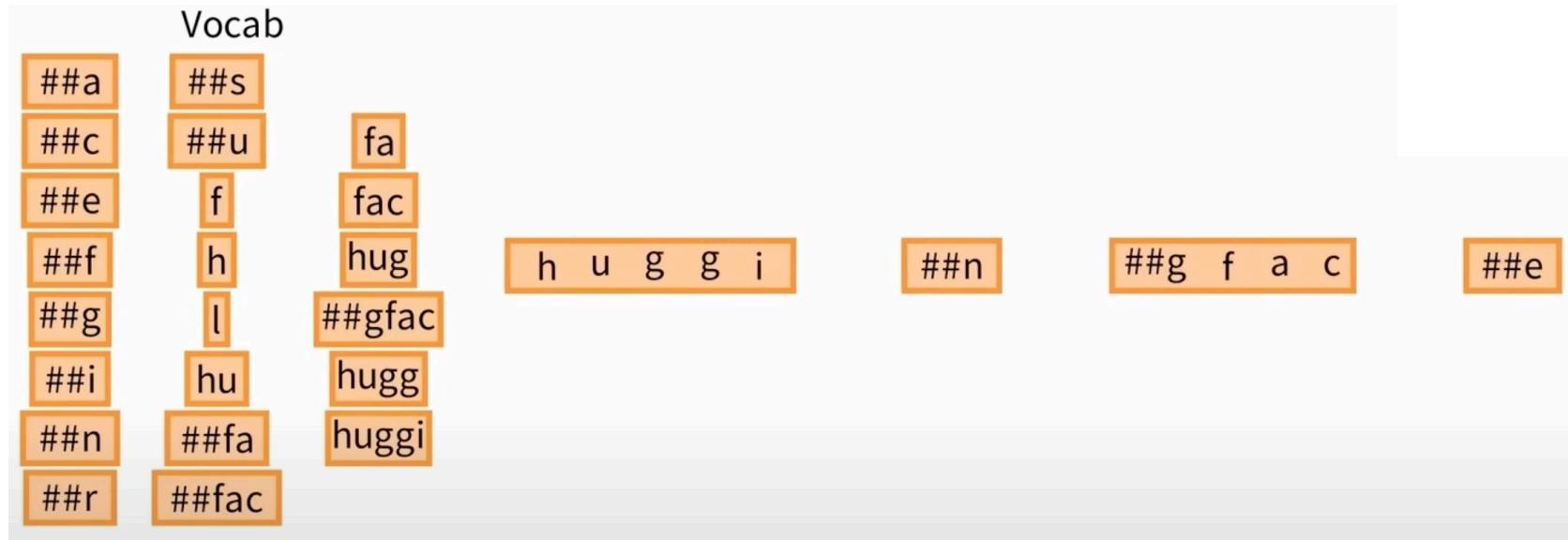
WordPiece



WordPiece, токенизация слов



WordPiece, токенизация слов



Unigram

Unigram модель - это языковая модель, которая рассматривает каждый токен независимо от предшествующих ему токенов. Это простейшая статистическая языковая модель для которой вероятность появления токена X , обусловленного контекстом равна простой вероятности появления токена X .

Получается, если бы мы использовали языковую модель Unigram для генерации текста, мы бы всегда предсказывали наиболее распространенный токен.

Unigram

Вероятность любого токена - это его частота (количество раз, когда мы ее находим) в исходном корпусе, деленная на сумму всех частот всех лексем в словаре.

Unigram

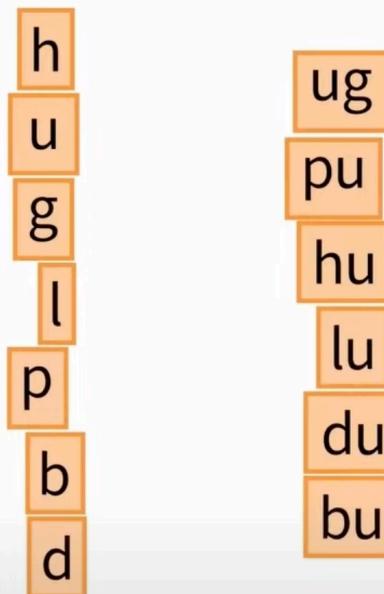
1. Создаем большой базовый словарь, например с помощью алгоритма ВРЕ задав большое значение размера словаря или любым другим способом.
2. На каждом шаге считается функция ошибок Unigram`ной языковой модели, при условии выкидывания каждого отдельно взятого токена. Токен для которого функция ошибок наибольшая будет удаляться из словаря.
3. Итеративно сокращаем словарь на фиксированную пропорцию словаря за счет удаления токенов, выбранных на предыдущем шаге. Когда размер словаря достигает заданных размеров алгоритм останавливается.
4. Готово 😊

Unigram

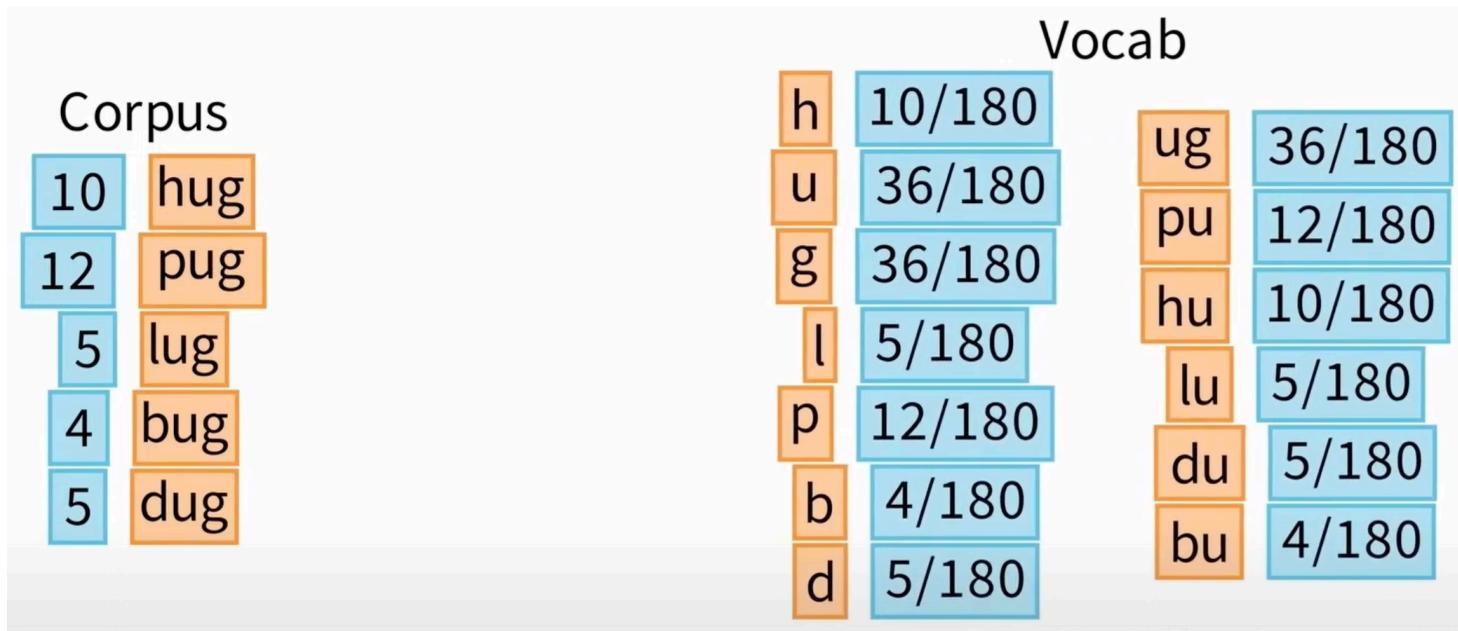
Corpus

10	hug
12	pug
5	lug
4	bug
5	dug

Vocab



Unigram



Unigram

Посчитаем вероятность для каждого возможного разбиения слова «hug»

$$\begin{array}{c} \boxed{h} \quad \boxed{u} \quad \boxed{g} \quad \frac{10}{180} \times \frac{36}{180} \times \frac{36}{180} = 2.22e - 03 \\ \boxed{hu} \quad \boxed{g} \quad \frac{10}{180} \times \frac{36}{180} = 1.11e - 02 \\ \boxed{h} \quad \boxed{ug} \quad \frac{10}{180} \times \frac{36}{180} = 1.11e - 02 \\ \boxed{hug} \quad 0 \end{array}$$

Для двух разбиений получили одинаковое значение, поэтому можем выбрать любое из них

Unigram

Corpus	Splits	Scores	Loss
10 hug	→ hu g	1.11e-02	$\sum freq \times (-\log(P(word)))$
12 pug	→ pu g	1.33e-02	$10 \times (-\log(1.11e - 02))$
5 lug	→ lu g	5.56e-03	$+12 \times (-\log(1.33e - 02))$
4 bug	→ bu g	4.44e-03	$+5 \times (-\log(5.56e - 03))$
5 dug	→ du g	5.56e-03	$+4 \times (-\log(4.44e - 03))$
			$+5 \times (-\log(5.56e - 03))$

Unigram

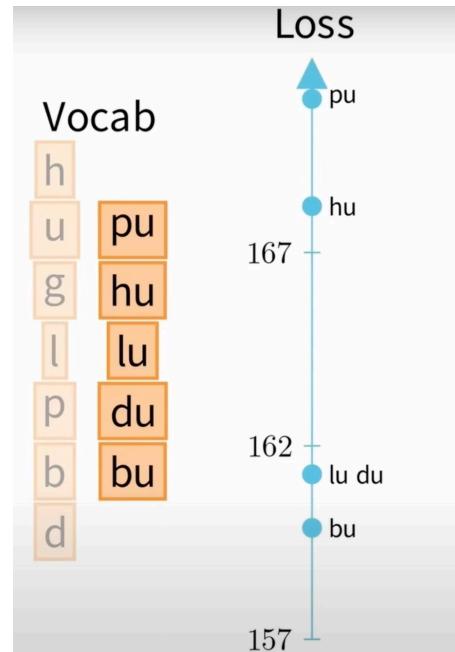
Попробуем посчитать loss при условии исключения одного выбранного токена

	Loss
With all vocabulary	170.4
Without ug	170.4
Without pu	170.4
Without hu	170.4
Without lu	170.4
Without du	170.4
Without bu	170.4

Так как значения одинаковые, то можно удалить любой токен и продолжить итеративное сокращение словаря. Выберем токен «иг» в качестве жертвы на удаление

Unigram

Попробуем посчитать loss при условии исключения одного выбранного токена



Теперь значения разные и удалять будем токен с минимальным значением loss (удаляем наименее важные токены) и продолжим итеративное сокращение словаря. Выберем токен «bu» в качестве жертвы на удаление

Unigram, токенизация слов

При токенизации слова каждый раз считается наиболее вероятная токенизация из всех возможных, исходя из unigram модели, которую мы обучили

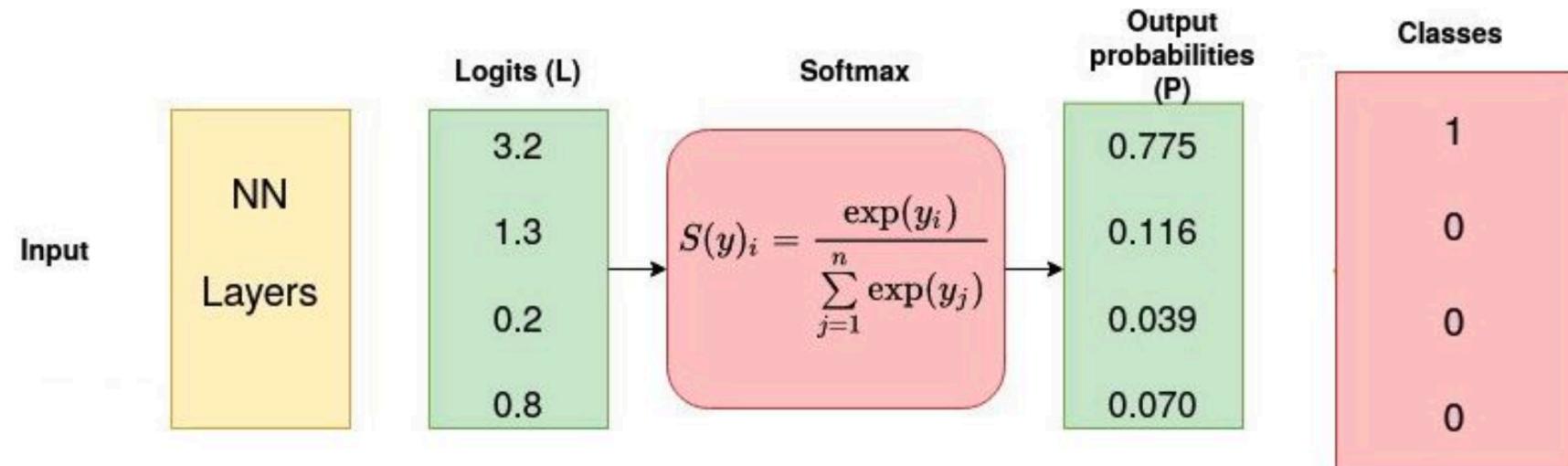
Стратегии генерации текста

Мы хотим, чтобы сгенерированные тексты были:

- связанными и согласованными
- разнообразными и интересными

Стратегии генерации текста

Для определенности, можно использовать простой softmax над логитами



Стратегии генерации текста

Чтобы изменить распределение вероятностей можно видоизменить формулу, добавив температуру

$$\frac{\exp(h^T w)}{\sum_{w_i \in V} \exp(h^T w_i)} \rightarrow \frac{\exp\left(\frac{h^T w}{\tau}\right)}{\sum_{w_i \in V} \exp\left(\frac{h^T w_i}{\tau}\right)}$$

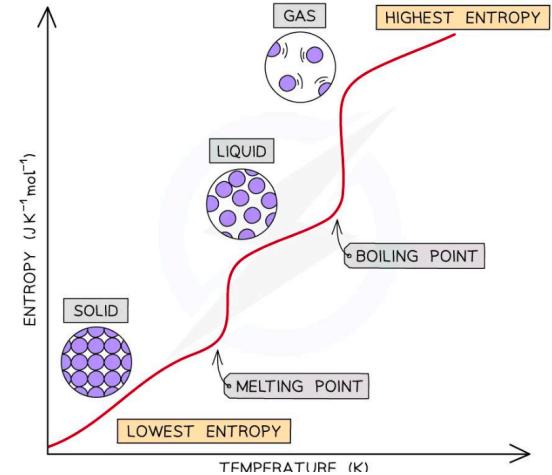
τ - softmax temperature

Интуиция - разделить на значение температуры, чтобы изменить общую энтропию системы

Стратегии генерации текста

Чтобы изменить распределение вероятностей можно видоизменить формулу, добавив температуру

$$\frac{\exp(h^T w)}{\sum_{w_i \in V} \exp(h^T w_i)} \rightarrow \frac{\exp\left(\frac{h^T w}{\tau}\right)}{\sum_{w_i \in V} \exp\left(\frac{h^T w_i}{\tau}\right)}$$

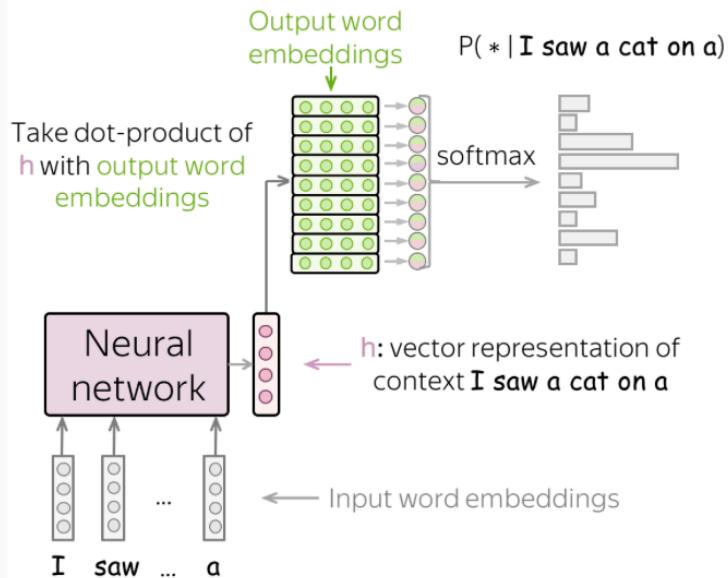


τ - softmax temperature

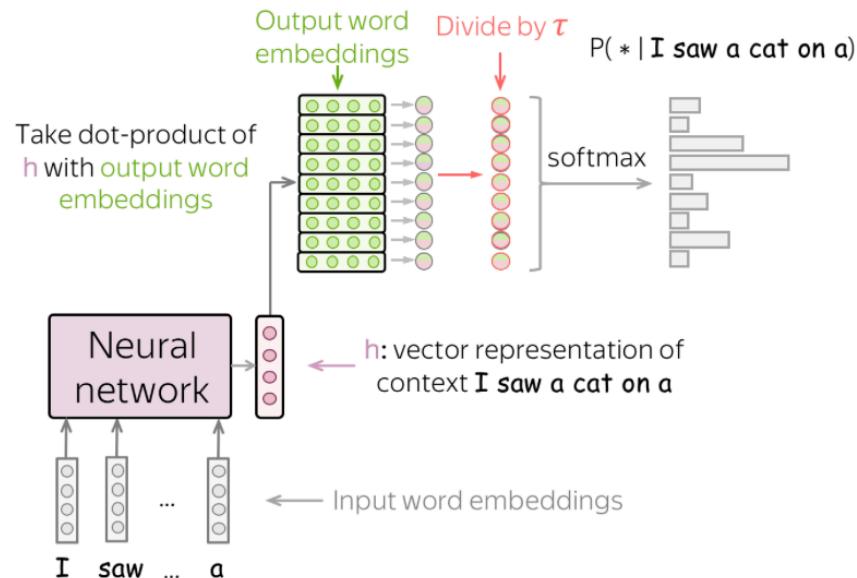
Интуиция - разделить на значение температуры, чтобы изменить общую энтропию системы

Стратегии генерации текста

Before



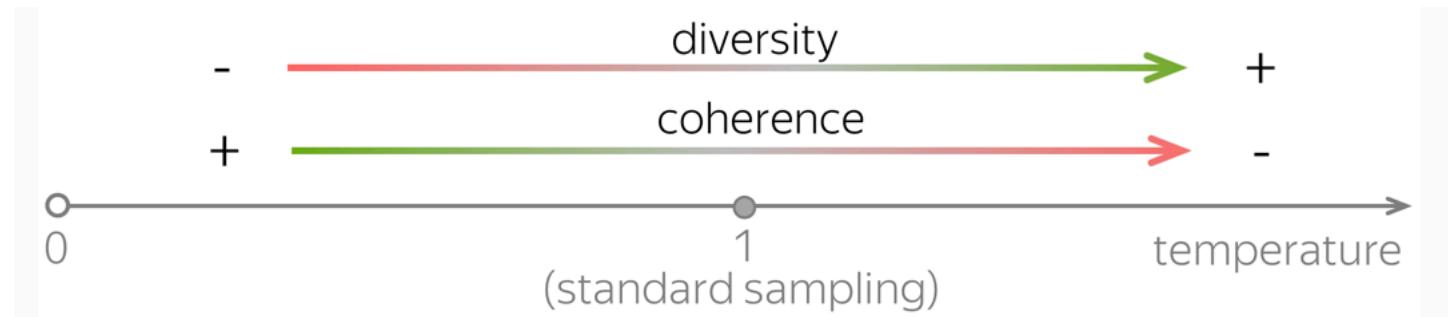
After



Стратегии генерации текста

Мы хотим, чтобы сгенерированные тексты были:

- связанными и согласованными (coherence)
- разнообразными и интересными (diversity)

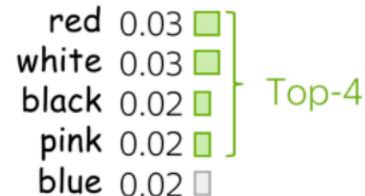


Top-k сэмплинг

Top-K for a flat distribution: not enough

The dress color was _____

$P(*) | \text{The dress color was} \dots)$



violet 0.02

olive 0.02

...

Top-K for a peaky distribution: too many

The light was _____

$P(*) | \text{The light was} \dots)$



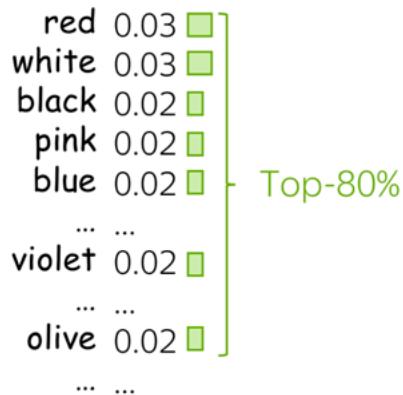
...

...

Топ-р сэмплинг / Nucleus sampling

The dress color was _____

$P(* | \text{The dress color was})$



The light was _____

$P(* | \text{The light was})$

get probability distribution



Beam-search

Beam-search - итеративный метод семплирования, для которого на каждом шаге мы выбираем не один самый вероятный токен, а сразу несколько (beam-size), и дальше продолжаем поиск для каждого из выбранных токенов.

Таким образом мы разветвляем пути генерации, получая несколько вариантов сгенерированного текста. В итоге можно выбрать тот вариант, у которого наилучшая перплексия

Beam-search, пример

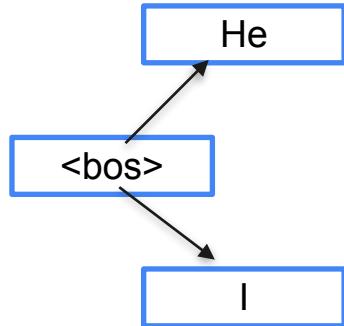
На каждом шаге мы храним несколько лучших гипотез

<bos>

Начнем со специального токена BeginOfSentence

Beam-search, пример

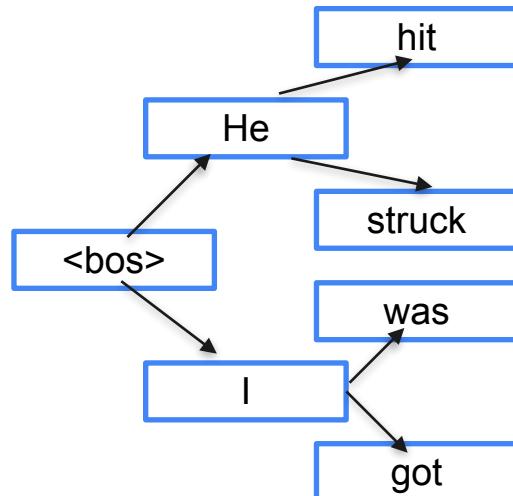
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

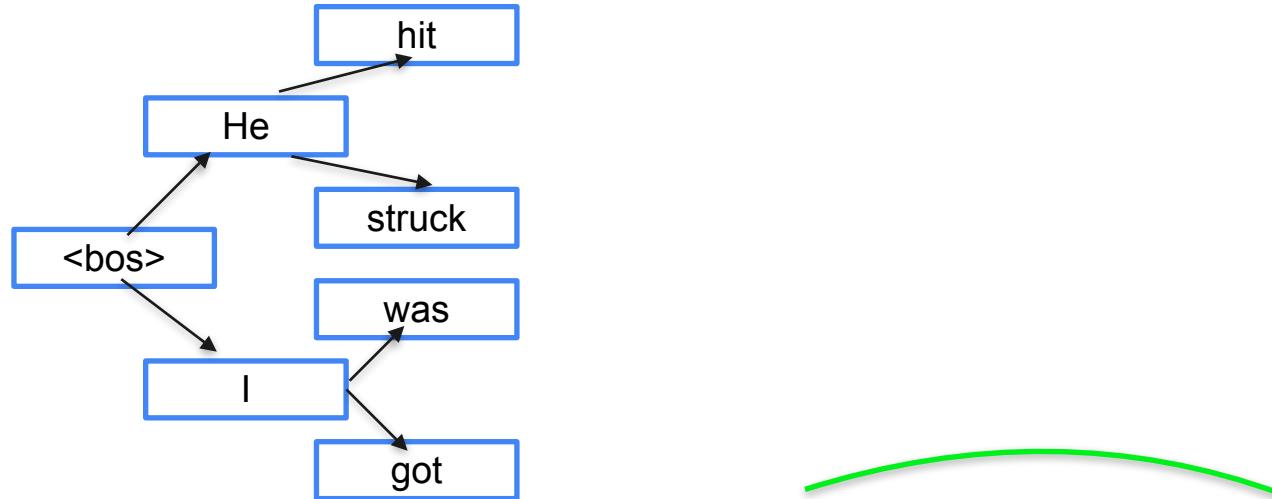
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

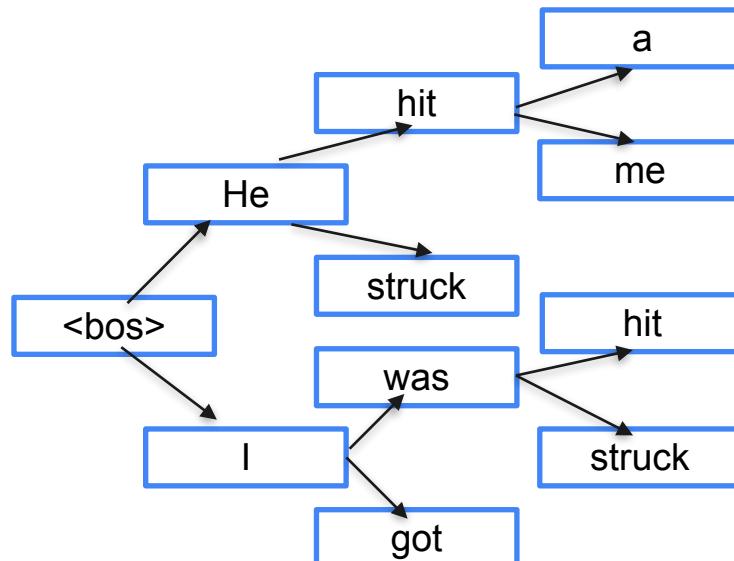
На каждом шаге мы храним несколько лучших гипотез



Посмотрим на оценки для генераций: $\text{score}(\text{hit}| \text{he}) > \text{score}(\text{was}| \text{I}) > \text{score}(\text{struck}| \text{he}) > \text{score}(\text{got}| \text{I})$

Beam-search, пример

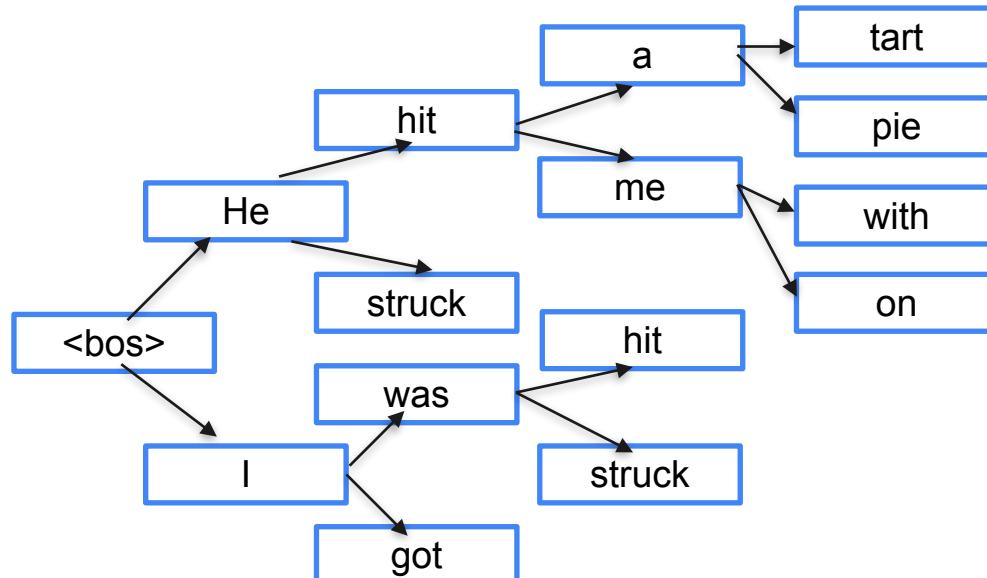
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

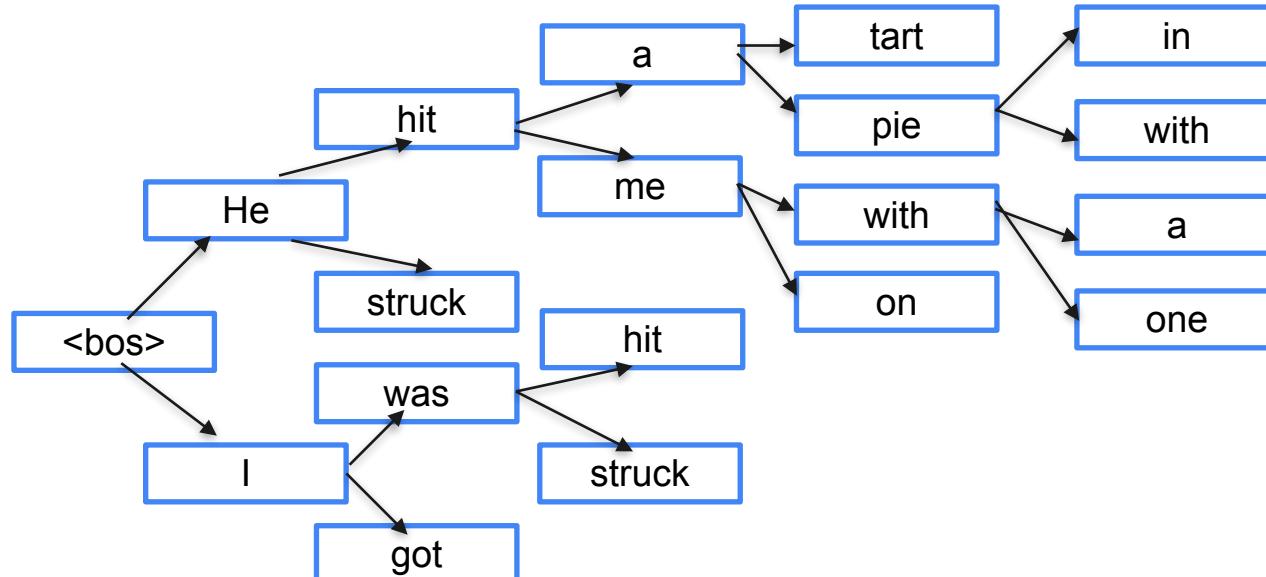
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

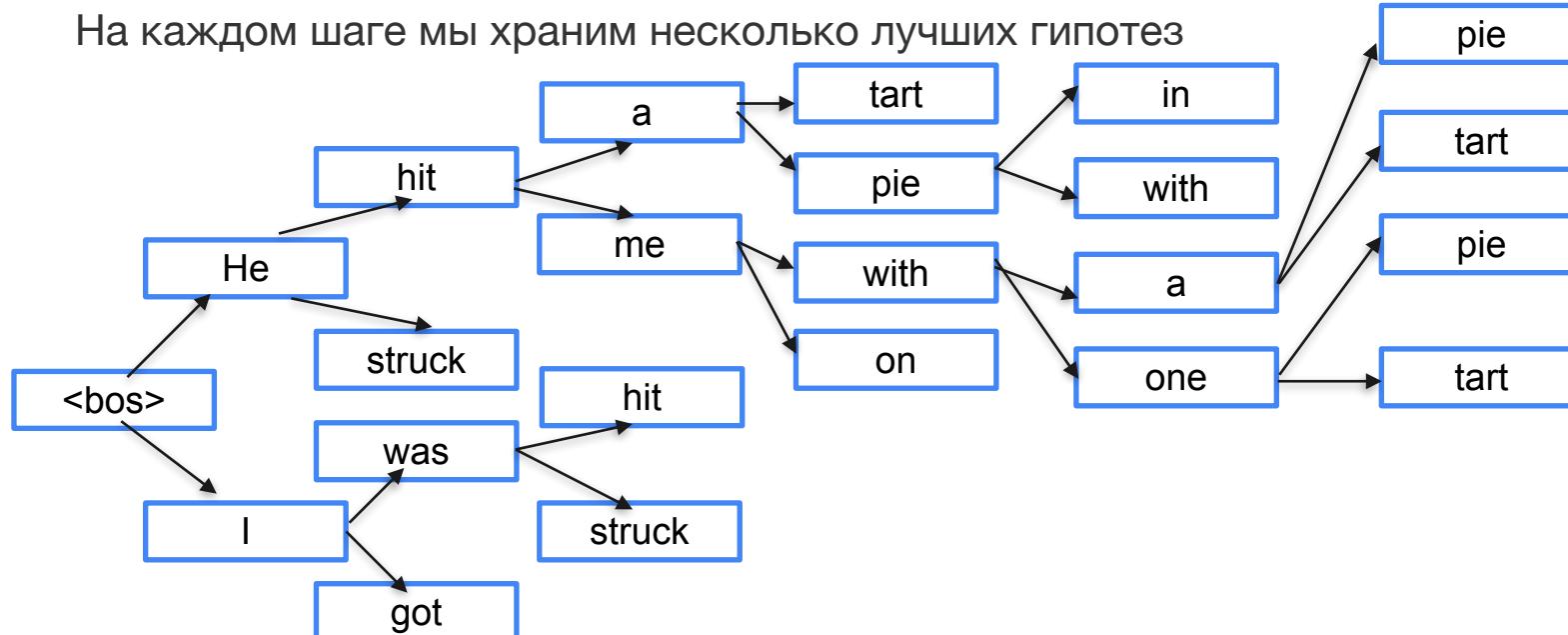
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

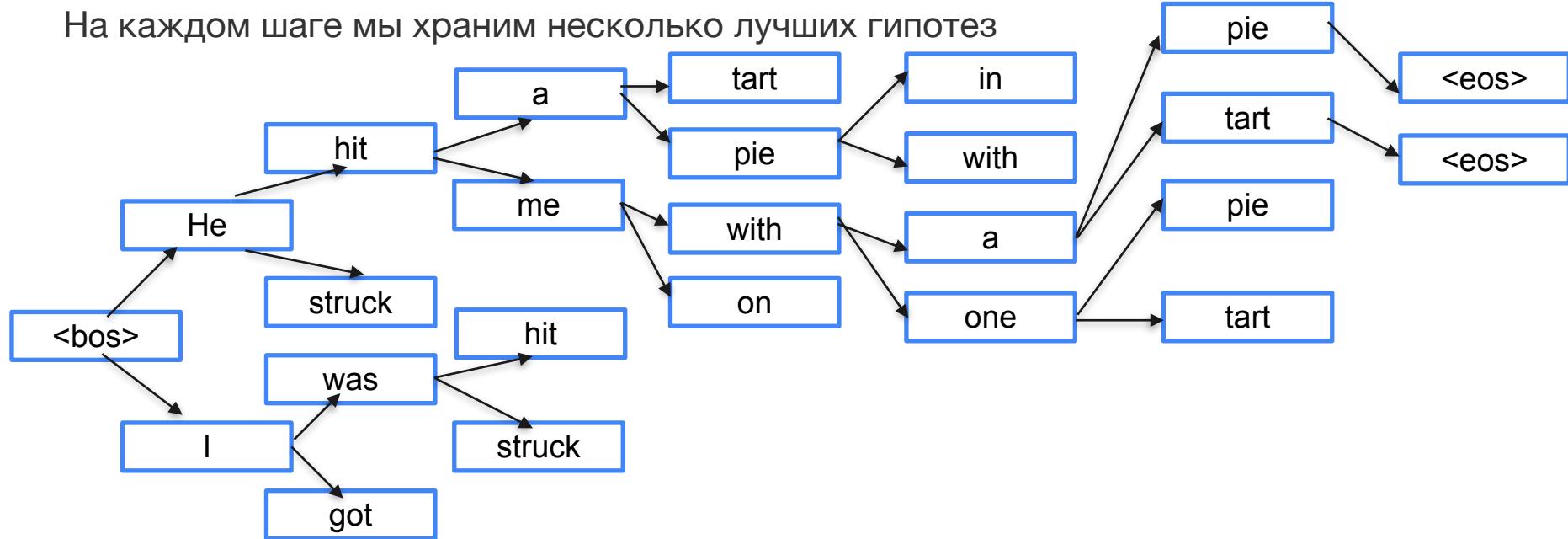
На каждом шаге мы храним несколько лучших гипотез



Сгенерируем beam_size вероятных токенов

Beam-search, пример

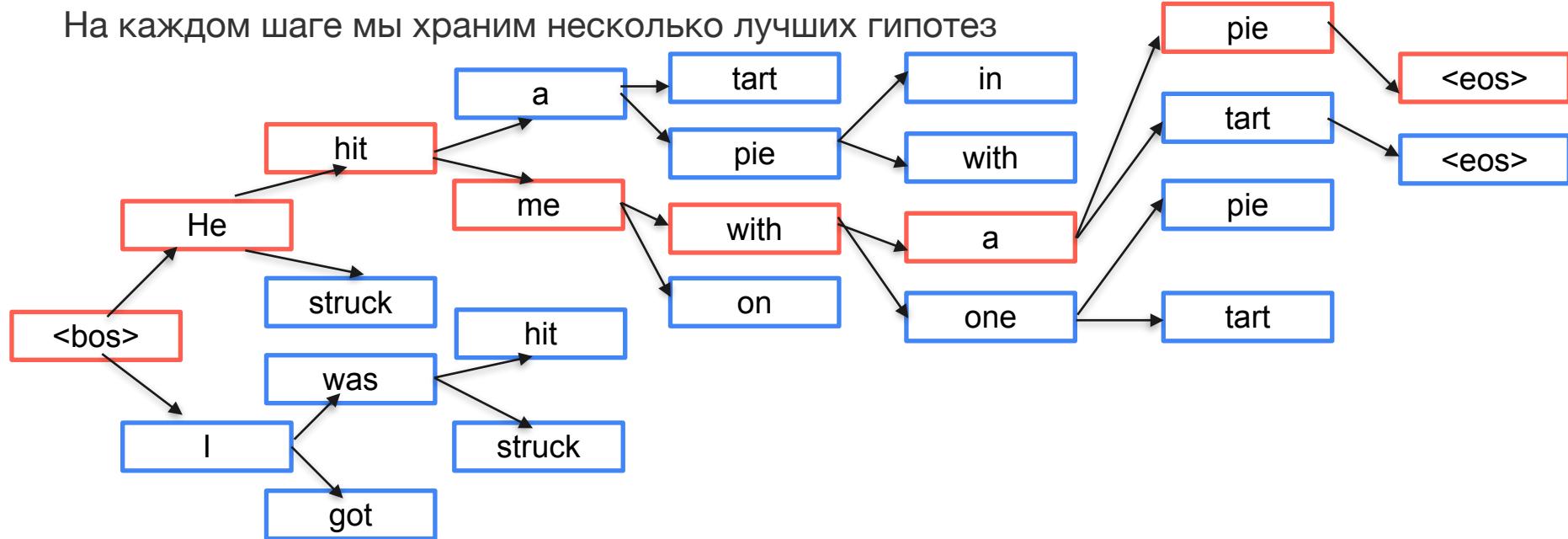
На каждом шаге мы храним несколько лучших гипотез



Сгенерировали все возможные гипотезы. Теперь выберем гипотезу с наивысшим скором

Beam-search, пример

На каждом шаге мы храним несколько лучших гипотез



Beam-search

Обычно генерируем до тех пор пока:

- не превысили время генерации
- У нас нет N гипотез, из которых выбирать

Обычно добавляют нормировку оценок по длине, чтобы избежать предпочтений к более коротким генерациям

$$\frac{1}{t} \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

Снова языковые модели

Формальная задача языкового моделирования может быть записана так:

$$P(y_1, y_2, \dots, y_n) = P(y_1) \cdot P(y_2|y_1) \cdot P(y_3|y_1, y_2) \cdot \dots \cdot P(y_n|y_1, \dots, y_{n-1}) = \prod_{t=1}^n P(y_t|y_{<t})$$

Условная языковая модели

Формальная задача языкового моделирования может быть записана так:

$$P(y_1, y_2, \dots, y_n) = \prod_{t=1}^n p(y_t | y_{<t})$$

А задача условного языкового моделирования может быть записана так:

$$P(y_1, y_2, \dots, y_n, | \textcolor{green}{x}) = \prod_{t=1}^n p(y_t | y_{<t}, \textcolor{green}{x})$$

condition on source x

Языковые модели для систем машинного перевода

При заданном тексте с исходного языка, необходимо решать следующую математическую задачу

$$P(y|x) = P(y_1|x) P(y_2|y_1, x) P(y_3|y_1, y_2, x) \dots P(y_T|y_1, \dots, y_{T-1}, x)$$

Probability of next target word, given target words so far and source sentence x

Перплексия языковых моделей

Перплексия - обратная вероятность тестового набора, нормализованная по количеству слов

$$\text{Perplexity}(W) = \sqrt[N]{\frac{1}{\prod_{i=1}^N P(w_i | w_{i-1}, \dots, w_1)}}$$

Перплексию иногда называют взвешенным коэффициентом ветвления, потому что она показывает из какого количества слов приходится выбирать модели следующее слово для продолжения генерации

Перплексия языковых моделей

Перплексия - обратная вероятность тестового набора, нормализованная по количеству слов

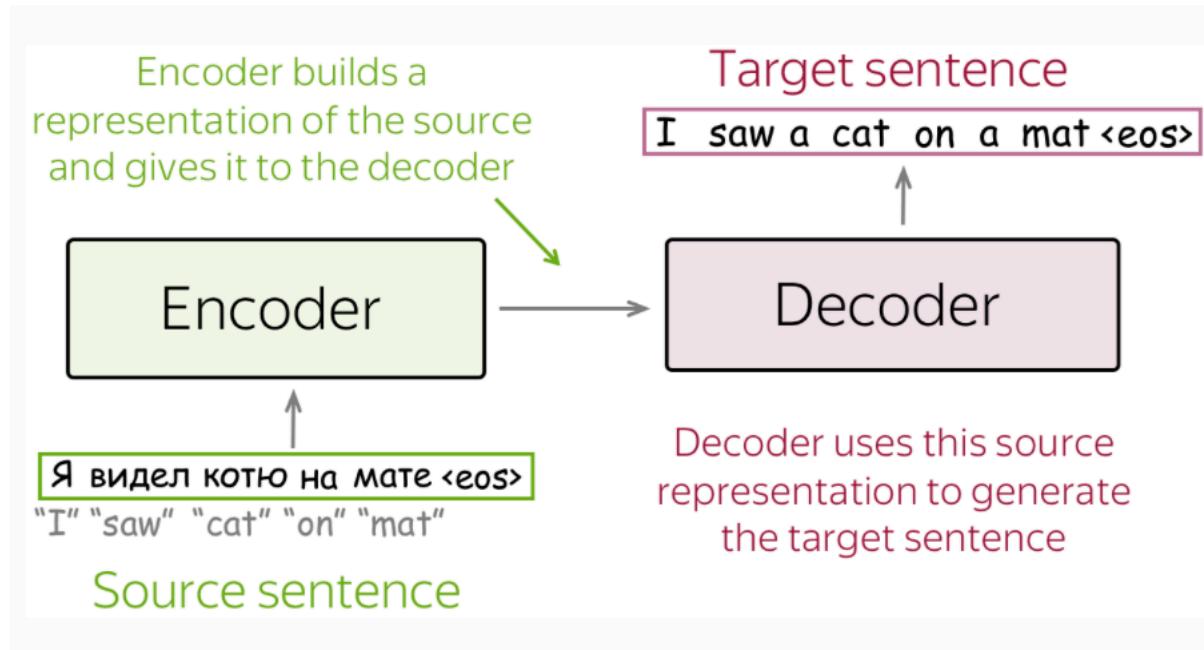
$$\text{Perplexity}(W) = \sqrt[N]{\frac{1}{\prod_{i=1}^N P(w_i | w_{i-1}, \dots, w_1)}}$$

$$\text{Perplexity}(W) = \sqrt[N]{\frac{1}{\prod_{i=1}^N \hat{y}(w_i)}} = \exp\left(\frac{1}{N} \sum_{i=1}^N -\log(\hat{y}(w_i))\right) = \exp(J(\theta))$$

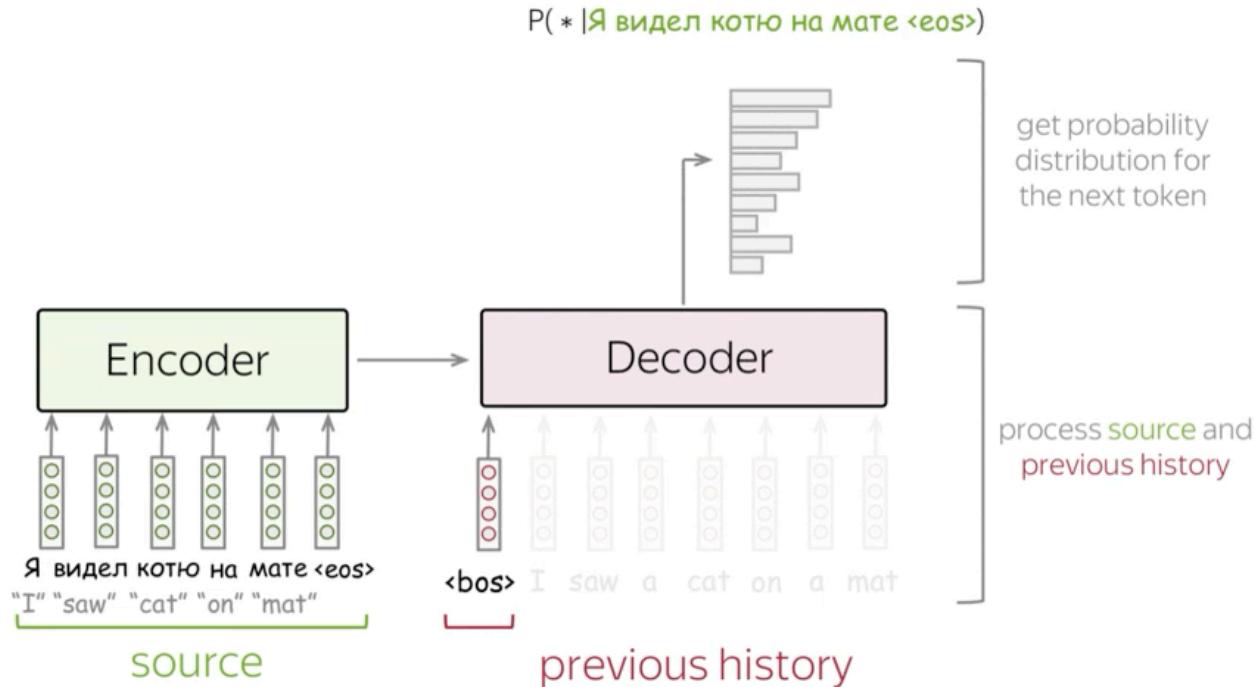
Получаем связь перплексии и обычной кросс-энтропии, с помощью которой мы обучаем языковые модели

<https://habr.com/ru/companies/wunderfund/articles/580230/>

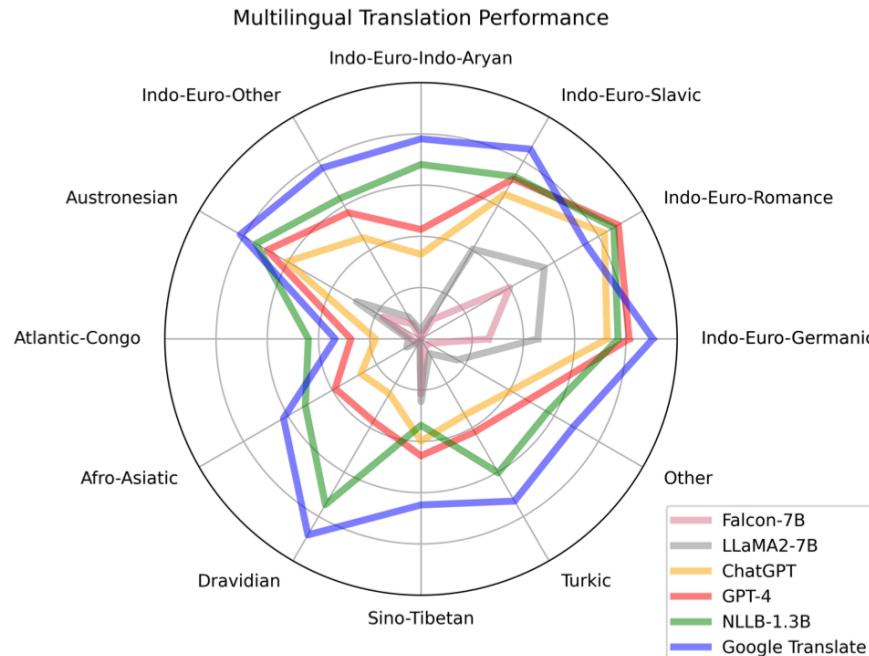
Задача перевода последовательностей



Задача перевода последовательностей



Современный подход к машинному переводу



Современный подход к машинному переводу

- Для популярных языков открытые LLM показывают сопоставимое или даже лучшее качество, чем специализированные модели
- Для задач перевода на малоресурсные языки специализированные модели для перевода(NLLB, LASER, seamless-m4t) справляются заметно лучше
- Текущая SOTA модель для мультиязычного перевода это модель NLLB(200 языков)

Метрики машинного перевода

Сложность оценки перевода в том, что можно перефразировать исходную фразу, но суть перевода останется прежней

Поэтому зачастую автоматизированные метрики используют для оценок по первым бенчмаркам, а более сложную человеческую оценку для финальных оценок

К автоматическим метрикам можно отнести:

- BLEU
- ROUGE
- METEOR
- WER
- итд

BLEU

BLEU (BiLingual Evaluation Understudy) - метрика основанная на подсчете слов (unigrams) и словосочетаний (n-grams) из машинного перевода, также встречающихся в эталоне.

$$BLEU = \text{brevity penalty} \cdot \left(\prod_{i=1}^n \text{precision}_i \right)^{1/n} \cdot 100\%$$

$$\text{brevity penalty} = \min \left(1, \frac{\text{output length}}{\text{reference length}} \right)$$

BLEU

Исходное предложение: Israeli officials are responsible for airport security

Кандидат А: Israeli officials responsibility for airport safety

Кандидат В: Airport security Israeli officials are responsible

Metric	System A	System B
precision (1-gram)	3/6	6/6
precision (2-gram)	1/5	4/5
precision (3-gram)	0/4	2/4
precision (4-gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0 %	52 %

Итоги занятия

- Вспомнили особенности архитектуры Transformer
- Изучили разновидности токенизации по подсловам
- Рассмотрели методы сэмплирования
- Вспомнили задачу машинного перевода и ее метрики

Спасибо за внимание

