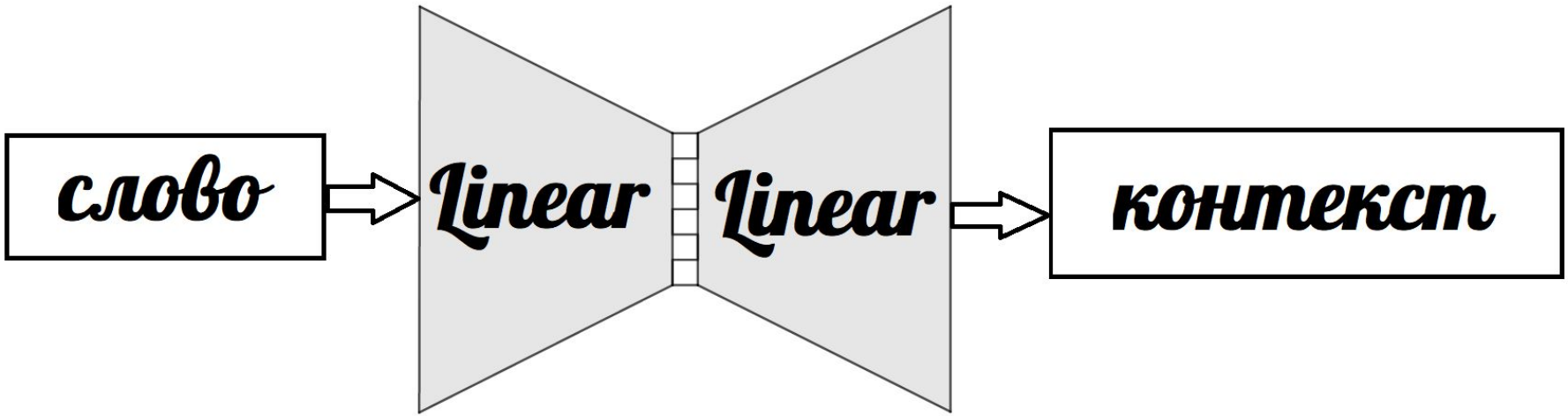
# BERT и другое

Привет! Меня зовут Коля Карташев и сегодня я сделаю доклад про берт.

# Word2Vec.




слово = значение = контекст

# На самом деле:

слово

\+

контекст

= значение

# Несколько (вполне банальных) примеров:

—"Скрипичный ключ помещает ноту «соль» первой октавы на вторую линейку нотоносца."

—"Первые ключи, как и замки, появились одновременно с первыми цивилизациями."

—"Ключ - естественный выход подземных вод на земную поверхность на суше или под водой"

# Word2Vec:

$\sum$ (замки, скрипичный, вод, выход, ...) = ключ

Word2Vec просто скидывает в кучу все слова, встречающиеся рядом с нашим, и выдает некий “усредненный” вектор-представление для всей кучи.

# Чего бы нам хотелось:

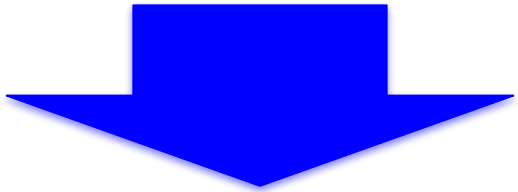
скрипичный ключ → $w_i$ ~ басовый ключ

первые ключи, как и замки → $w_j$ ~ дверной ключ

ключ - <...> подземных вод → $w_k$ ~ речной ключ

# Как придумать эмбеддинг с учетом контекста?

- Идея Word2Vec - получать векторные представления слова и контекста, пытаясь предсказывать из одного другое
- Чтобы получить эмбеддинг содержащий максимальное количество контекстной информации, нужно будет работать с предложениями целиком.



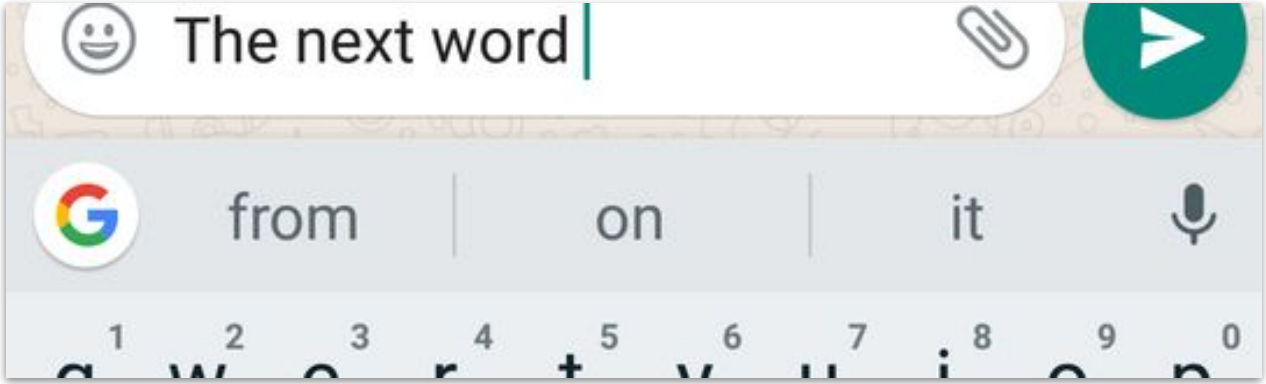
**Рекуррентные модели**

# Так ли хорошо делить предложения по словам?

- Много редких слов, для которых недостаточно информации для нормального обучения
- Порой приходится использовать токены <UNK>, что, иногда, выкидывает важную информацию из данных.
- Различные формы слова воспринимаются как разные слова
- Размеры словарей замедляют скорость работы и увеличивают размеры модели.

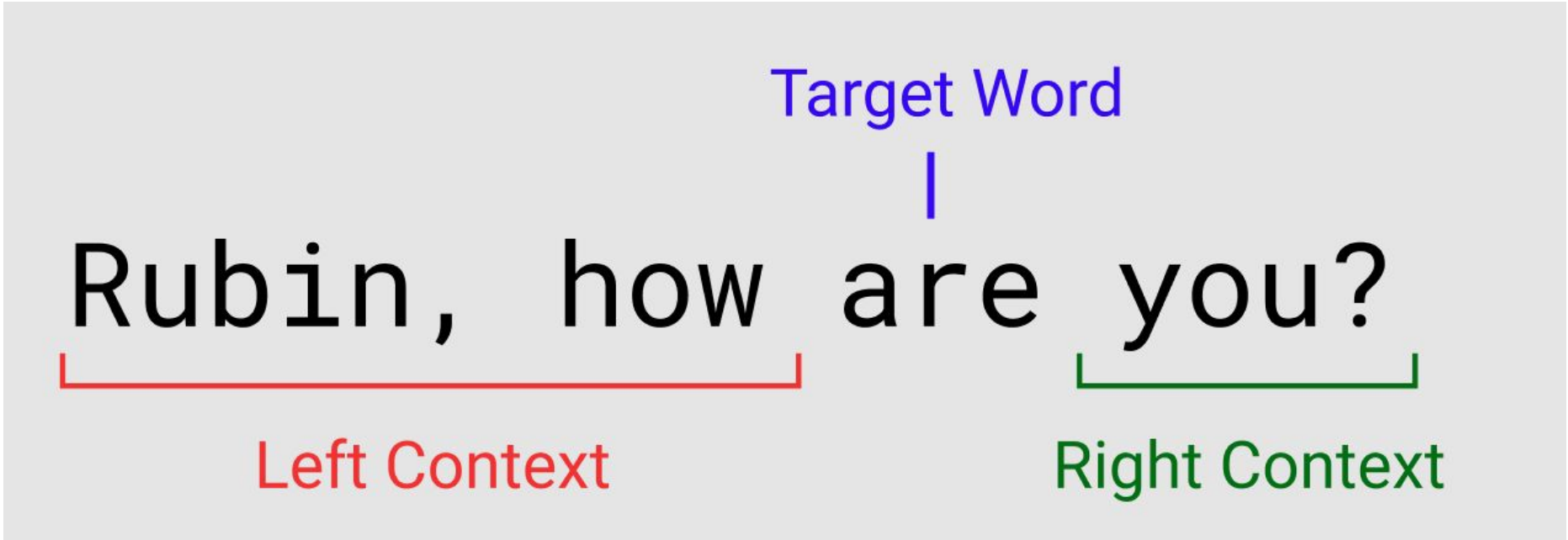
BERT, GPT, и другие продвинутые модели NLP часто используют свои **токенайзеры**: в них слова могут представляются в виде нескольких токенов.

# Language modelling

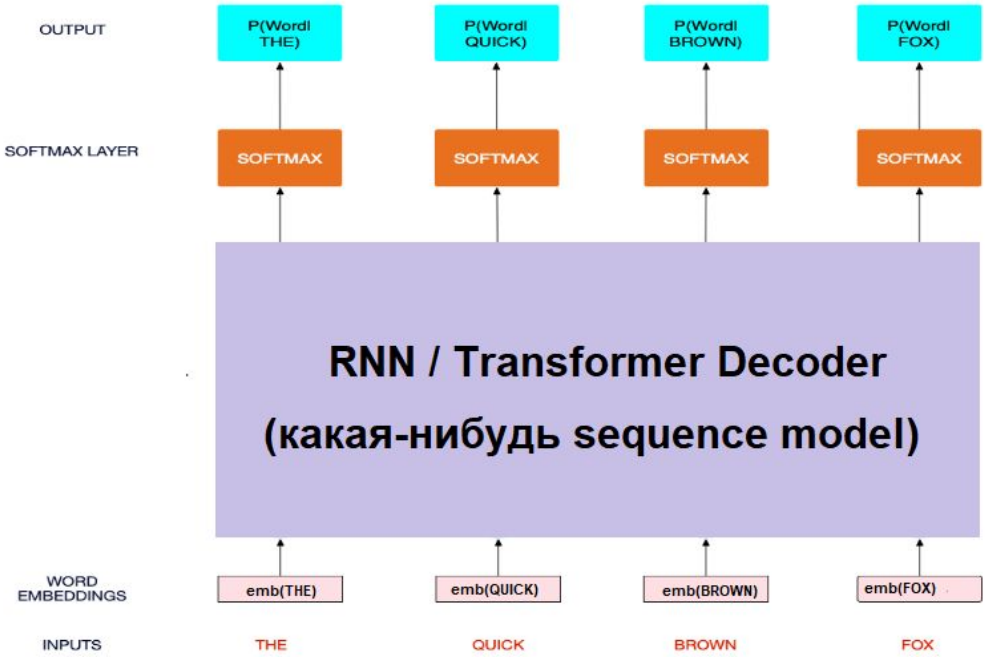


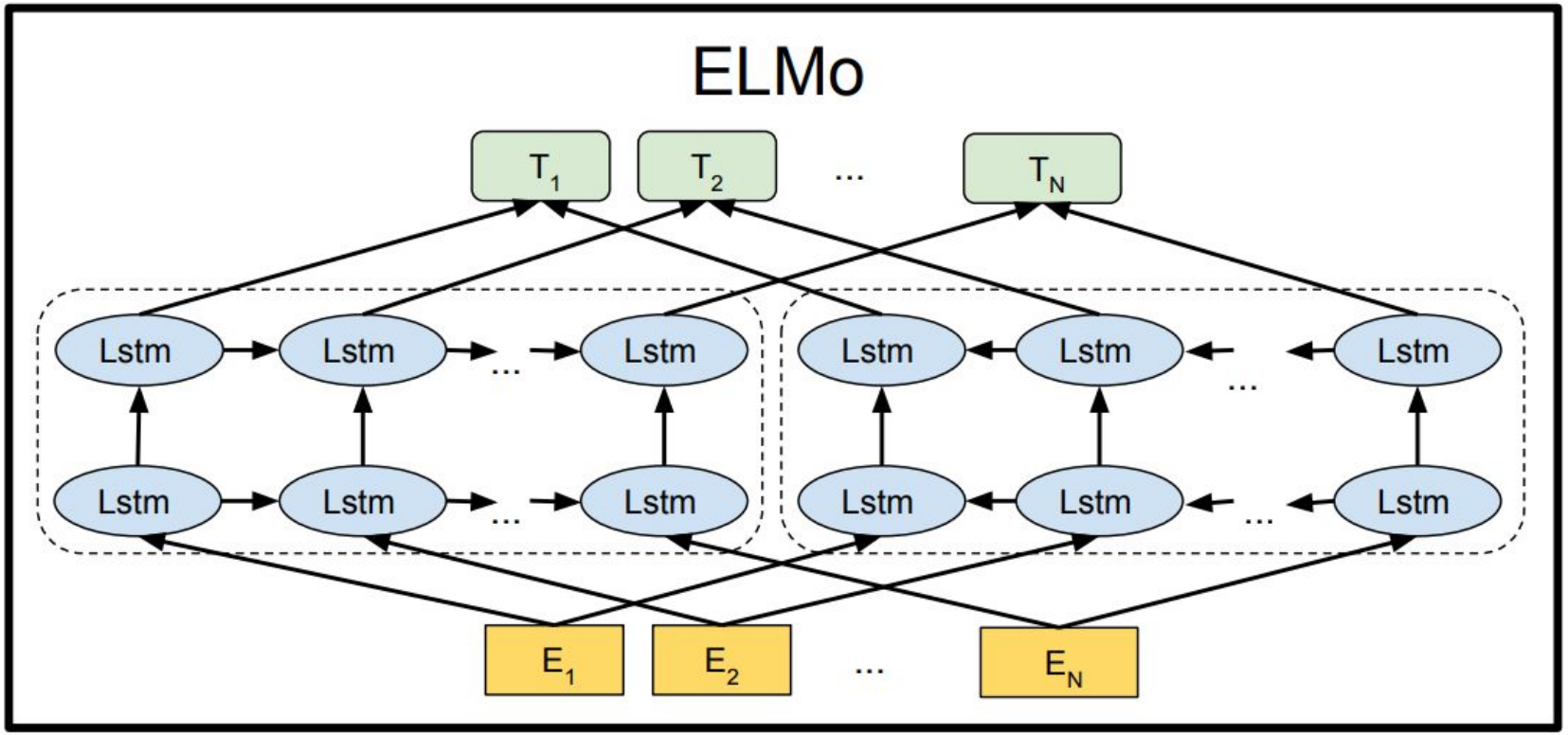

Простой пример language modelling в реальной жизни

# Language modelling (LM)



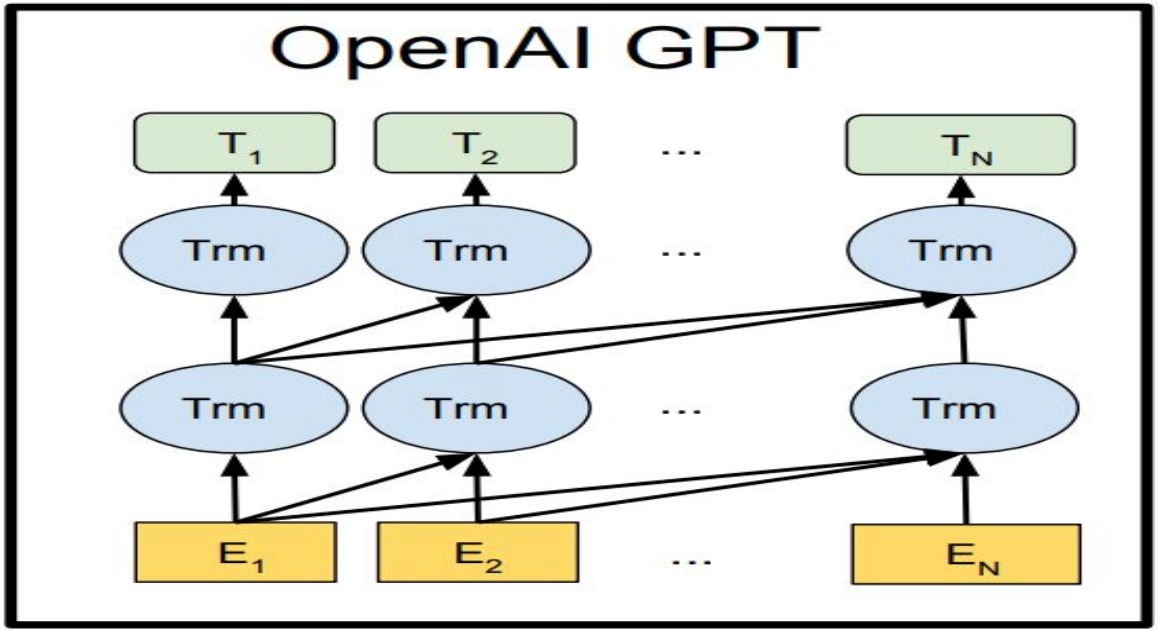
# Language modelling (LM)






Задача: BiLSTM(context) = word

# Следующий неизбежный шаг - Трансформеры



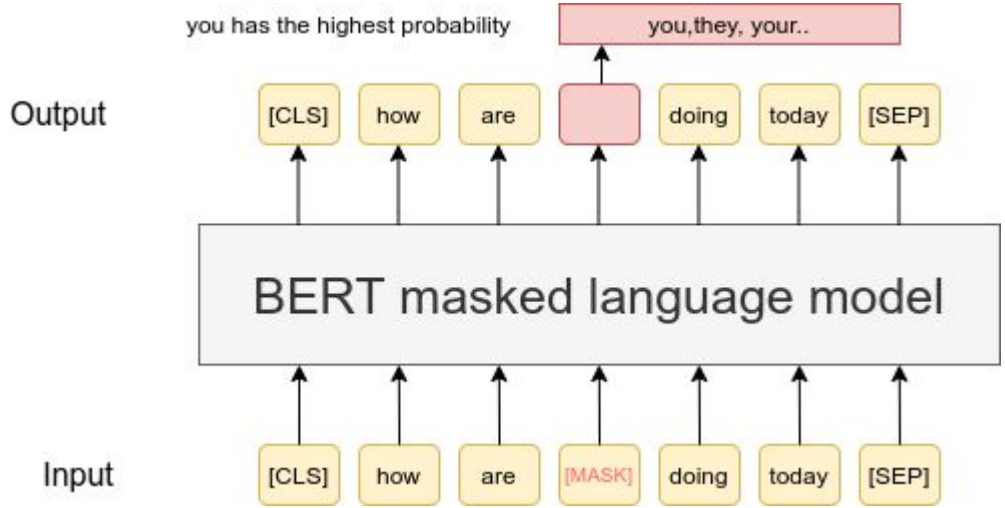

Все очень просто - обычный Transformer Decoder, обычное Language Modelling, обычные тонны гпу и денег.

# GPT: плюсы и минусы

OpenAI recently published GPT-3, the largest language model ever trained. GPT-3 has 175 billion parameters and would require 355 years and $4,600,000 to train - even with the lowest priced GPU cloud on the market.[1]

+ Можно генерировать тексты! Иногда даже осмысленные!!
+ Можно потратить миллионы долларов!
- Мы не можем использовать и правый и левый контекст - декодер разрешает нам идти только в одну сторону для генерации.
- Отчасти в силу этого, обучить модель очень сложно - и чем лучше модель, тем сложнее это сделать.
- Нам нужно генерировать естественный язык, при этом не запуская модель для каждого слова, используя все предложение за раз.

# Решение - заменить LM на Masked LM.




Учимся предсказывать истинные значения слов, замененные в оригинальном предложении.

# Masked Language Modelling

1. Случайно выбираем 15% токенов, по которым будет считаться Loss для модели.
2. Заменим 80% из них на токен [MASK]
3. Заменим 10% из них на случайный токен из словаря
4. Оставим 10% токенов неизменными
5. Передадим все предложение (в том числе с невыбранными в 1 пункте токенами) в BERT.
6. Предскажем оригинальные токены (BERT не знает какие токены мы заменили и какие выбрали в 1 пункте)

# Question 1:

Q: Почему мы выбираем токены, по которым будем брать функцию потерь? Почему не брать ее по всем токенам?

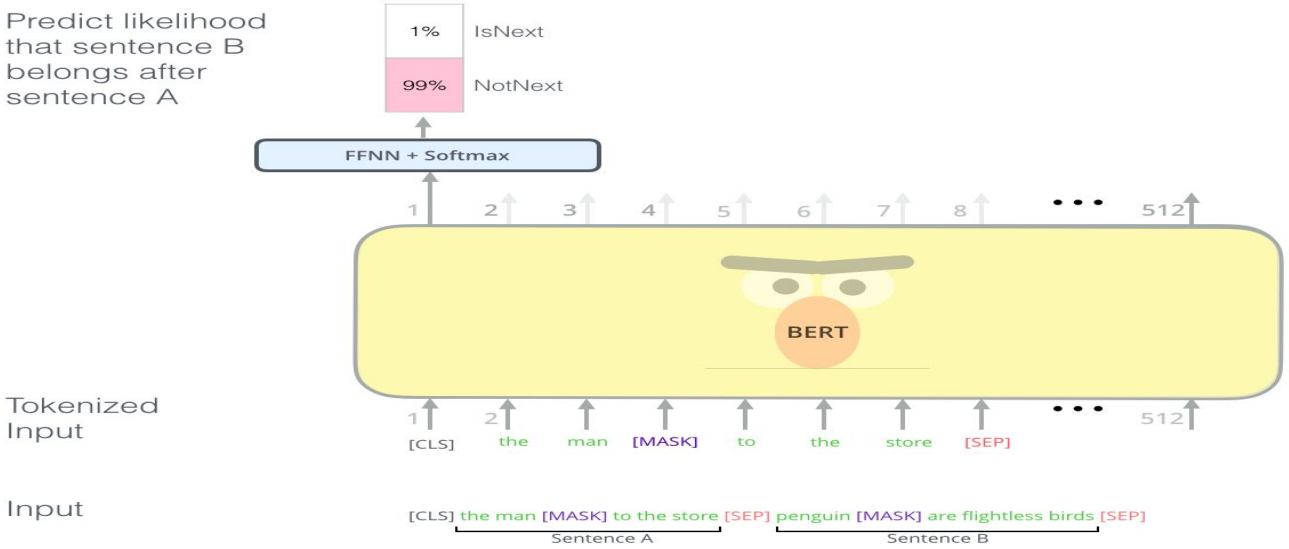
A: Мы не хотим чтобы BERT переучивался на выдачу оригинальных слов как ответа, так как мы хотим обучить его извлекать максимум информации из контекста. При этом, если заменить больше 15% токенов, нам не будет хватать контекста для предсказания замененных символов.

# Question 2:

Q: Что будет если заменить не 80%, а 100% токенов на [MASK]? Почему мы заменяем часть на случайные, а часть на оригинальные?

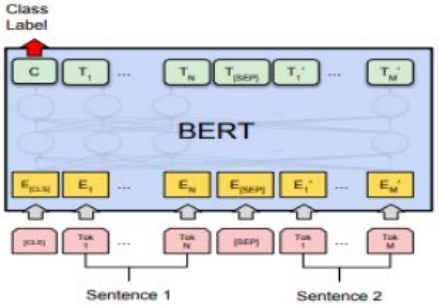
A: Мы хотим, чтобы в основном берт извлекал информацию из контекста. Одновременно, мы хотим чтобы в эмбеддинге слова содержалась информация и о нем самом. Значит, чтобы BERT все же учитывал оригинальные слова, мы хотим дать ему часть слов неизменными. Однако, мы также не хотим, чтобы BERT верил нам наслово, про слова которые мы ему передали, как бы абсурдны они ни были. Поэтому, иногда мы будем подавать ему на вход ложные слова, таким образом подготавливая берт не “ломаться” когда в тексте встречаются ошибки / опечатки.

# Next Sentence Prediction

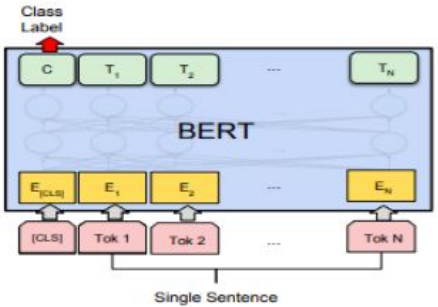



Нам бы хотелось, чтобы берт умел обменивать контекст между предложениями. Так мы можем удачнее строить эмбеддинг целых текстов. Решение - NSP.

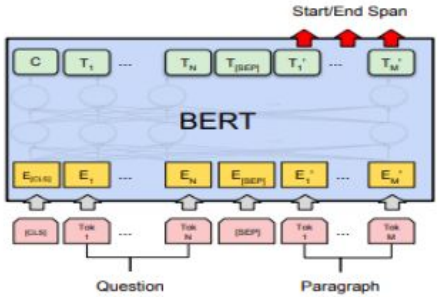
# Задачи:



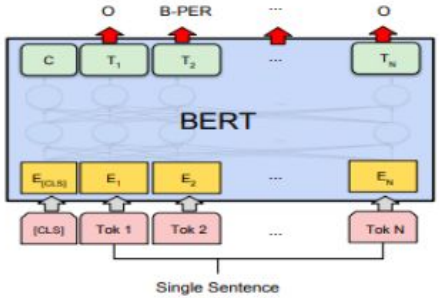

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG




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



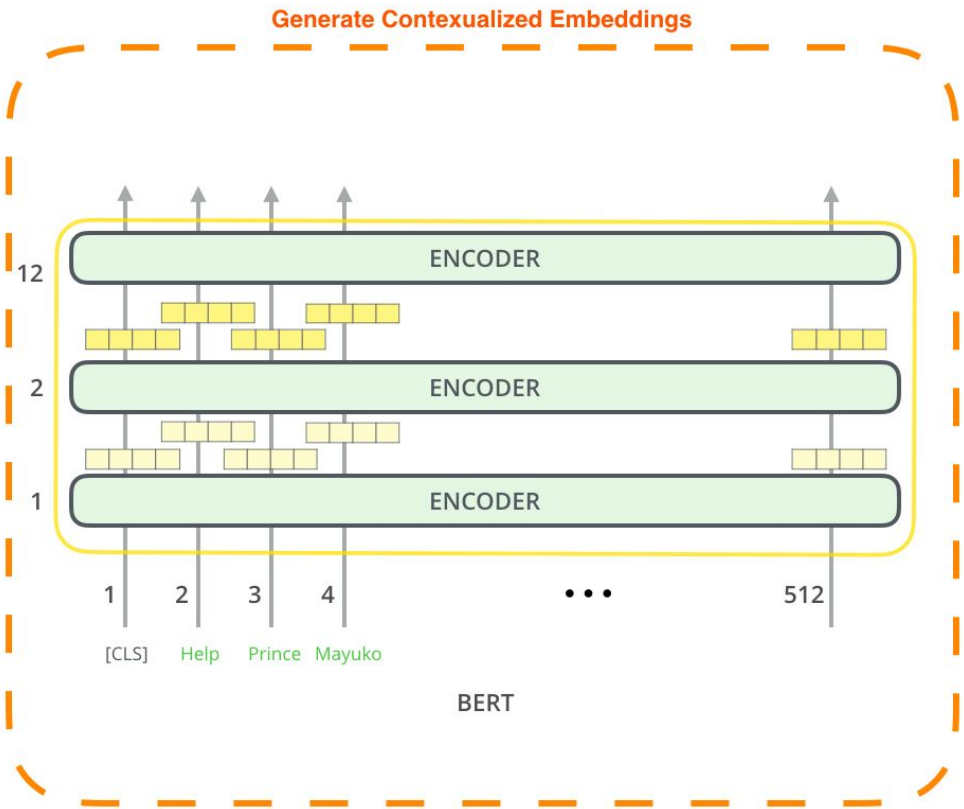

(c) Question Answering Tasks: SQuAD v1.1



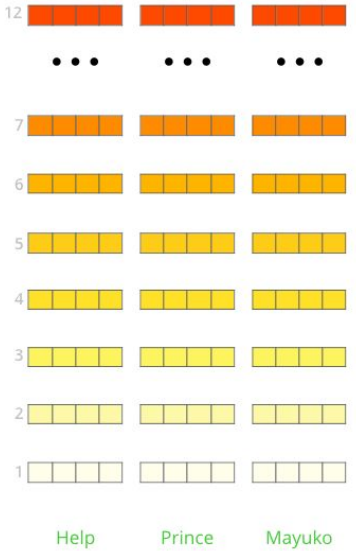

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Предобученный на общих языковых задачах берт можно файн-тюнить на широкий ассортимент разных языковых задач - слева приведены примеры из оригинальной статьи

# BERT - Bidirectional Encoder Representations from Transformers.
# Можно ли работать с самими векторными представлениями?



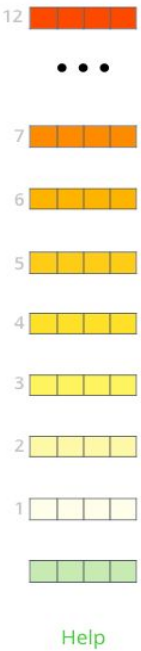

The output of each encoder layer along each token's path can be used as a feature representing that token.




**But which one should we use?**

# Сравнение различных подходов.

**What is the best contextualized embedding for "Help" in that context?**
For named-entity recognition task CoNLL-2003 NER

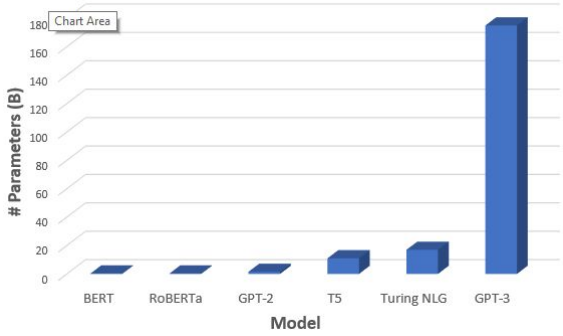


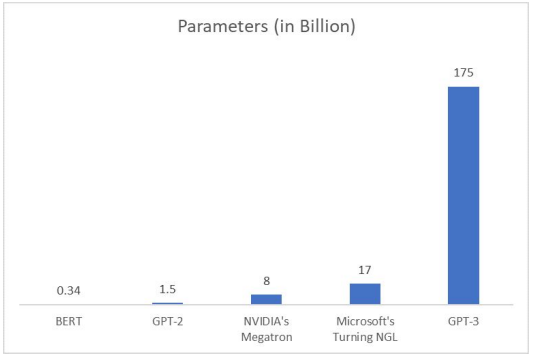

| | | Dev F1 Score |
|---|---|---|
| First Layer | Embedding | 91.0 |
| Last Hidden Layer | 12 | 94.9 |
| Sum All 12 Layers | 12 + ... + 2 + 1 = | 95.5 |
| Second-to-Last Hidden Layer | 11 | 95.6 |
| Sum Last Four Hidden | 12 + 11 + 10 + 9 = | 95.9 |
| Concat Last Four Hidden | 9 10 11 12 | 96.1 |

# Что-нибудь еще?

Помимо берта придумано множество альтернативных моделей - BERT LARGE, XLNet, Megatron-LM, GPT-X и многие, многие другие. Поговорим про GPT.

Помимо попыток придумать новое решение проблемы, есть другой способ улучшить результат - увеличить размер и количество параметров модели.




# Пример работы GPT-2

Как уже говорилось ранее, GPT удобно тем, что имеет встроенную возможность генерировать тексты. Как успехи?

| | |
|---|---|
| SYSTEM PROMPT (HUMAN-WRITTEN) | *In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.* |
| MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES) | The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.<br><br>Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.<br><br>Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow. |

Спасибо!