

# Введение в NLP

# План

- Краткая история развития NLP
- О курсе: план, формы контроля, оценка
- Введение в NLP
- Практика

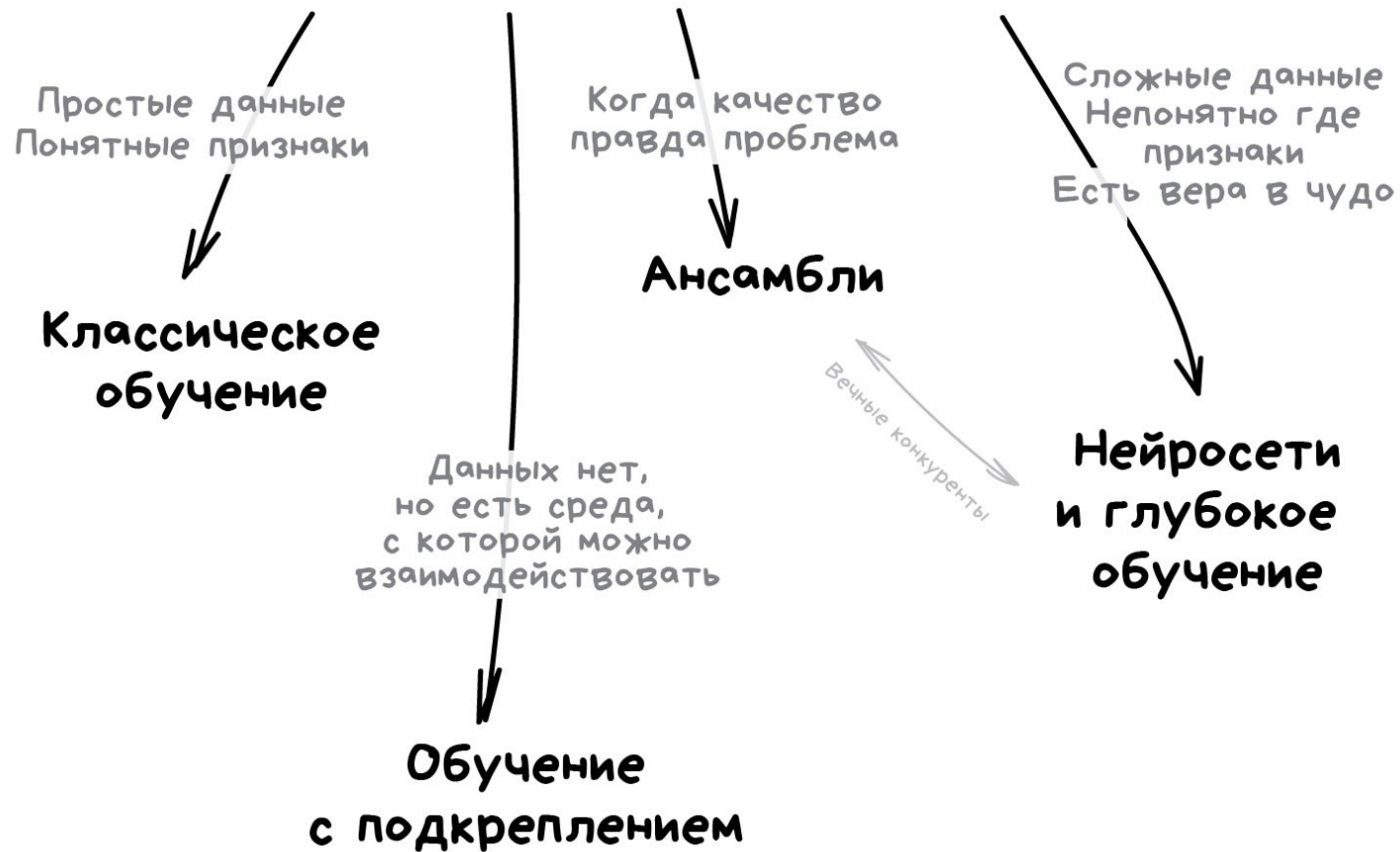
# Краткая история развития NLP

# Машинное обучение

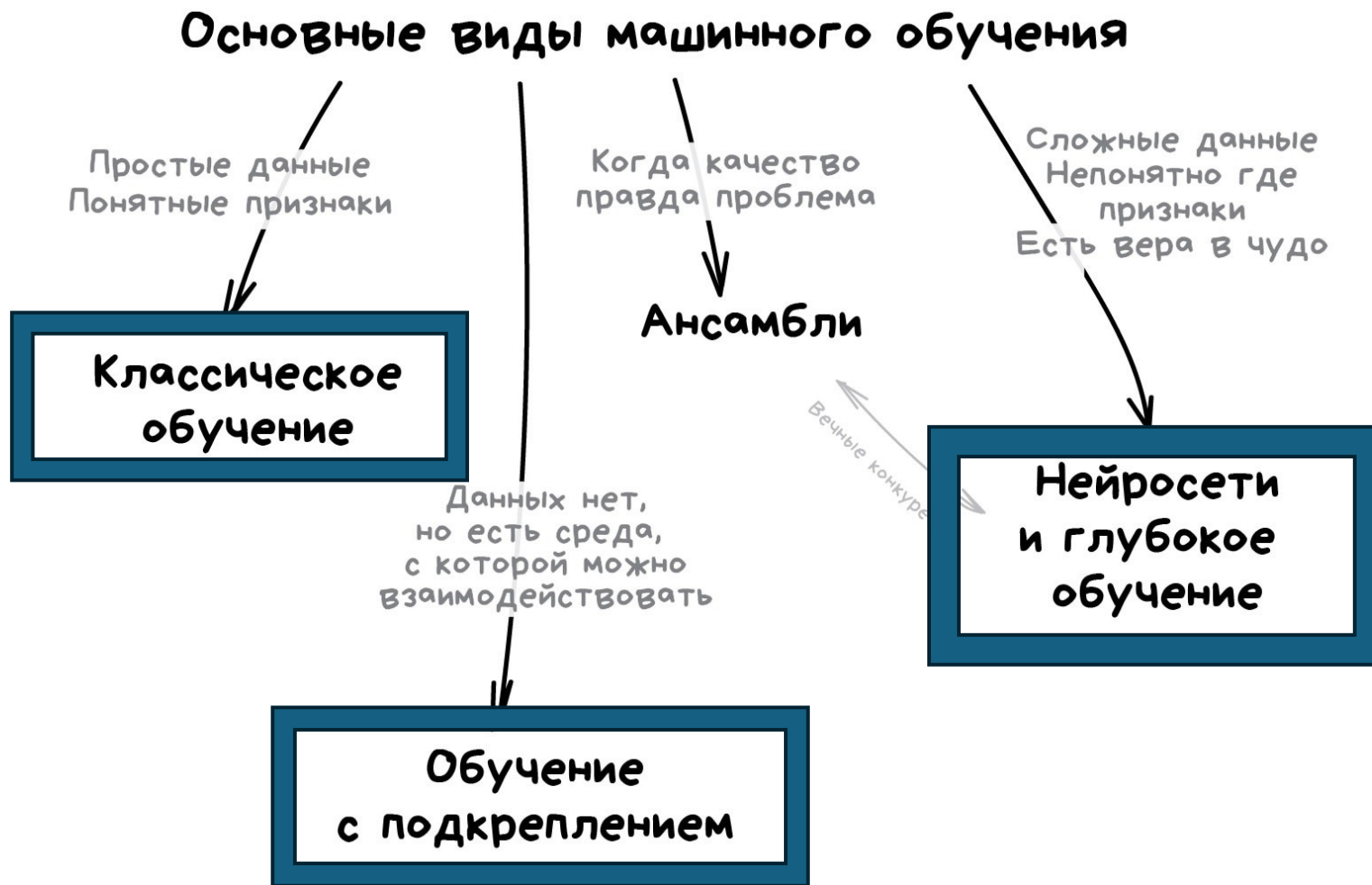


# Машинное обучение

## Основные виды машинного обучения



# Машинное обучение: где NLP?



# Обучение на текстах

Для обучения используется простой прием: часть [ ] в маскируется. Задача нейросети предположит [ ] ие слова были пропущены.

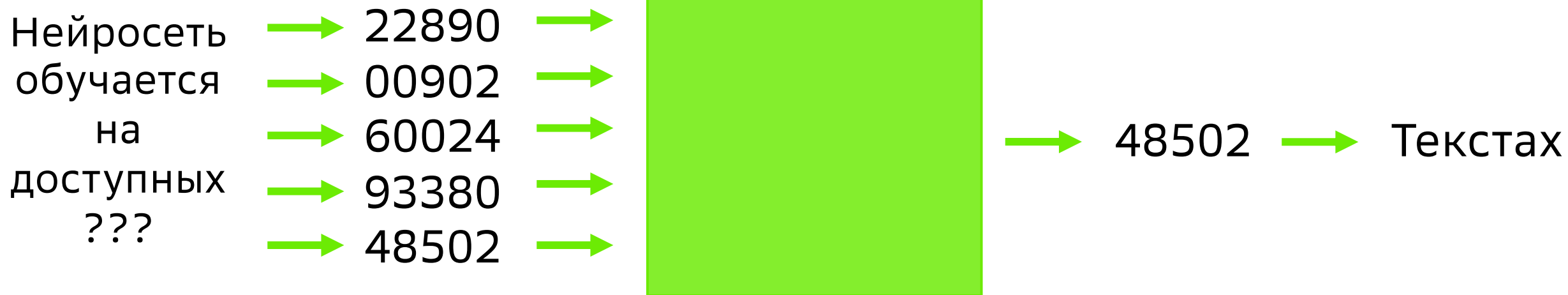
При этом нейросеть обучается [ ] всех доступных текстах. Это огромный массив данных.

# Обучение на текстах

Для обучения используется простой прием: часть слов маскируется. Задача нейросети предположить, какие слова были пропущены.

При этом нейросеть обучается на всех доступных текстах. Это огромный массив данных.

# Обучение на текстах

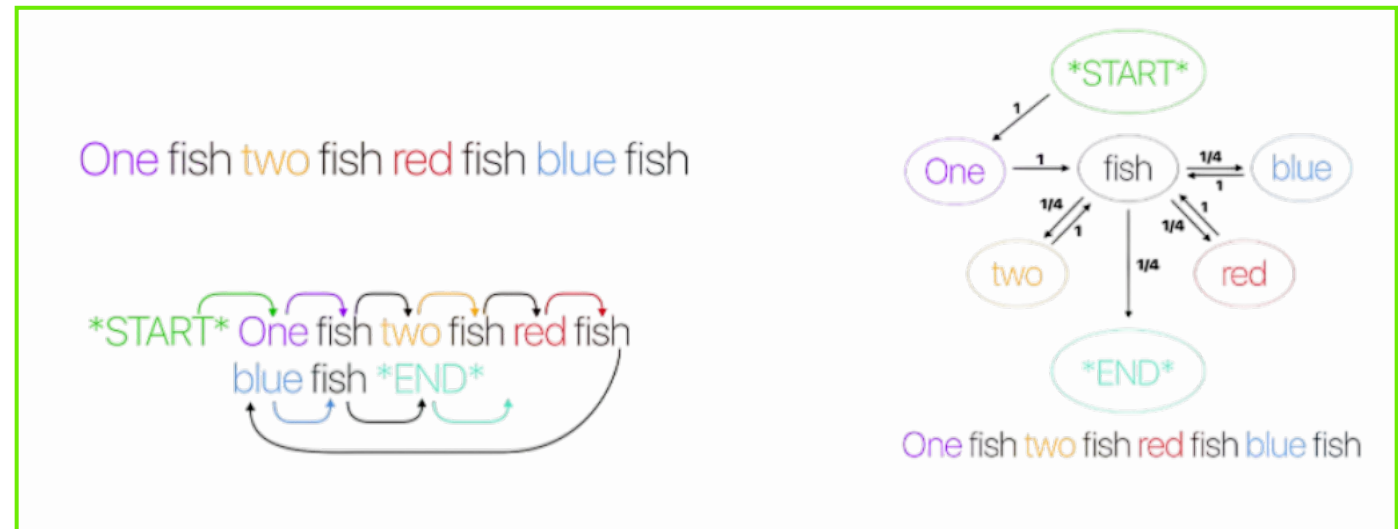


# Цепь Маркова

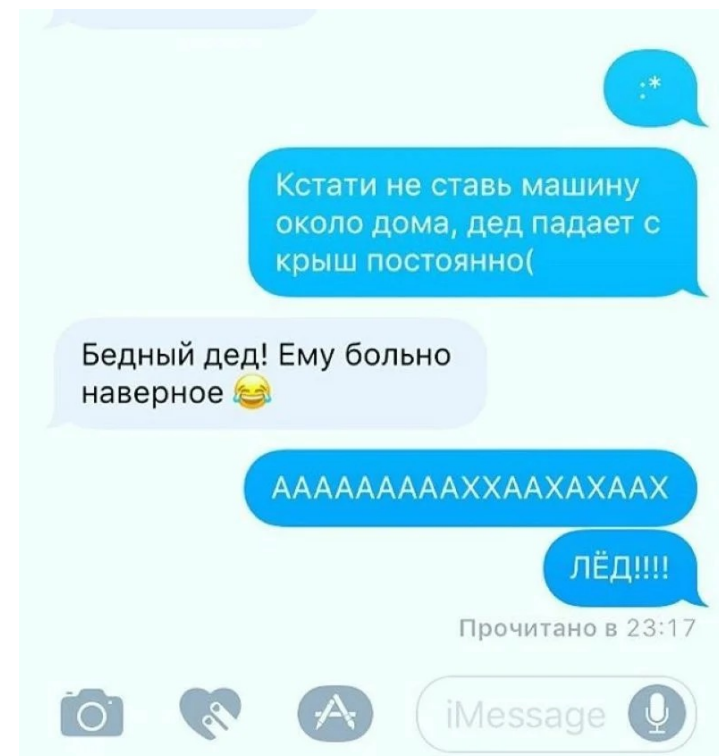
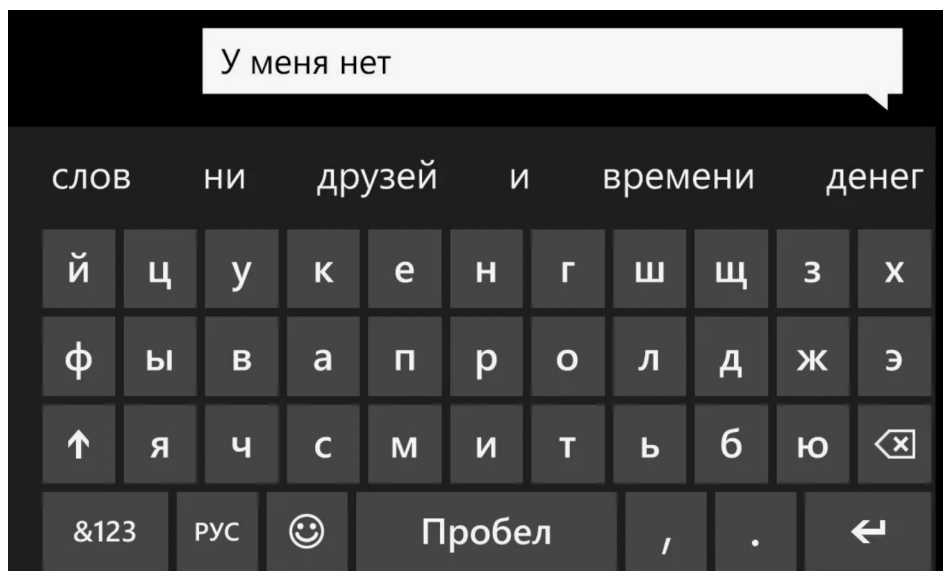
**Слово** – отдельный объект

**Обучение** – подсчет вероятностей

**Генерация** – блуждание по графу



# Цепь Маркова: применение

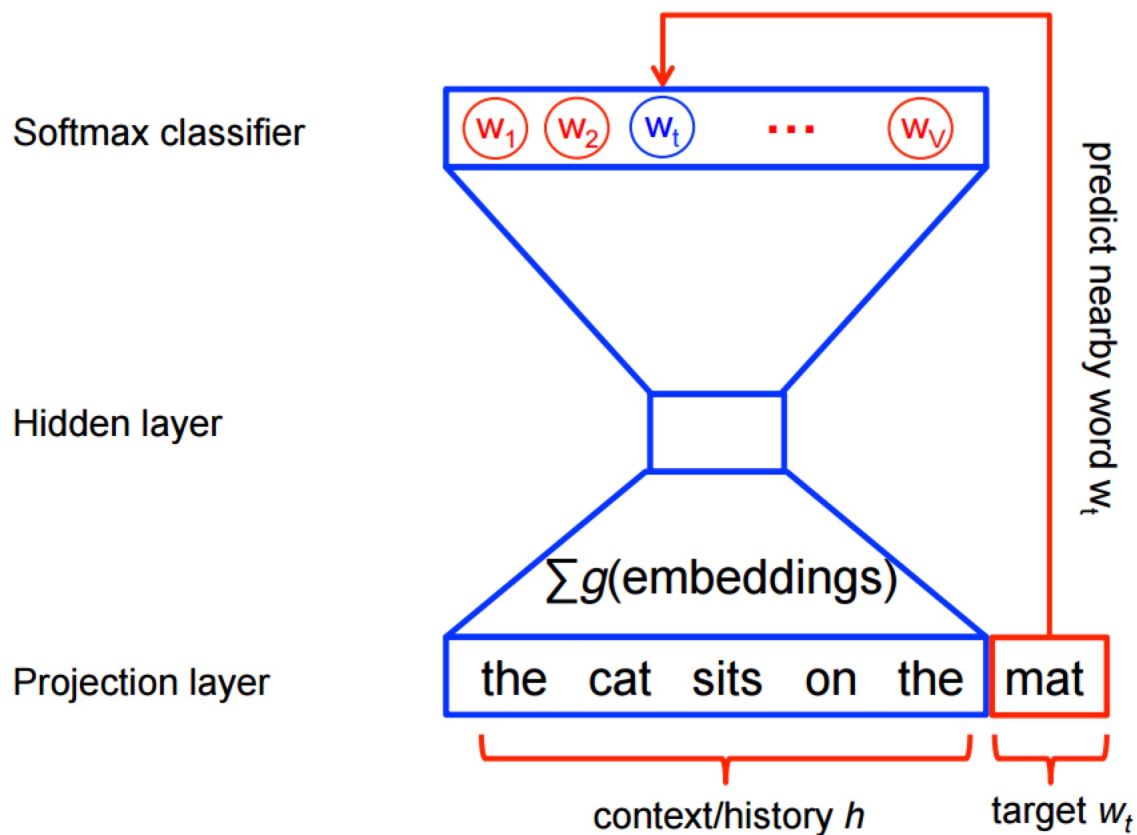


# Word2Vec

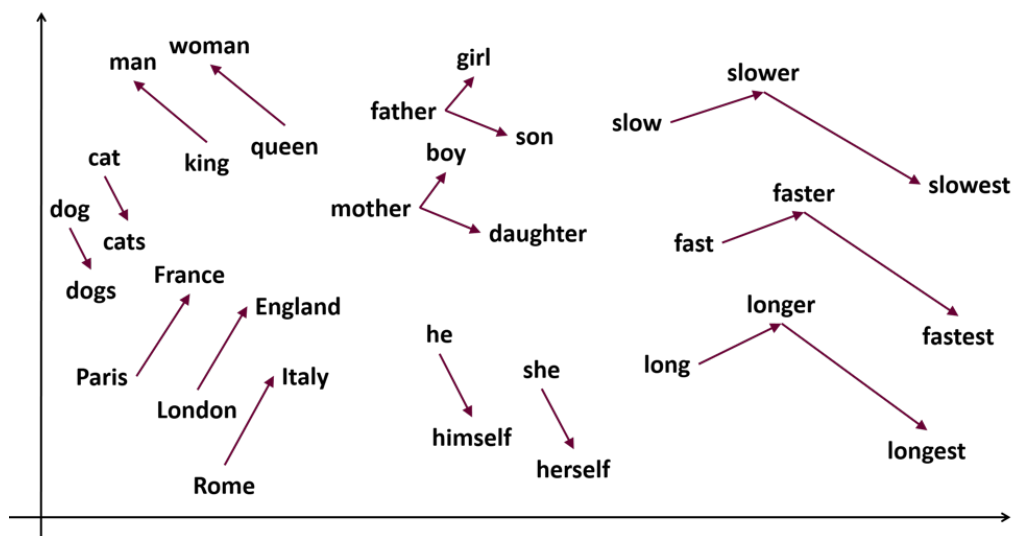
**Слово** – вектор

**Обучение** – подбор весов нейросети

**Генерация** – подсчет вероятности на основе нескольких слов



# Word2Vec: применение



Google

Google Search

I'm Feeling Lucky

Яндекс

Карты Маркет Новости Словари ещё

Найти

Например, воздушный кондиционер

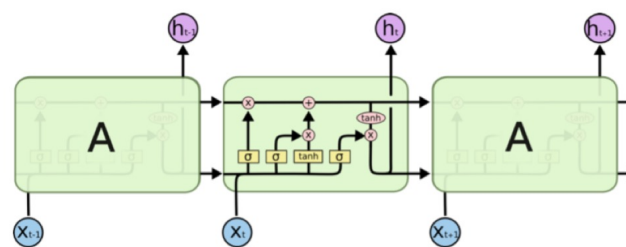
# LSTM (Long short-term memory)

**Слово** – набор ***N*-грамм**,  
каждая

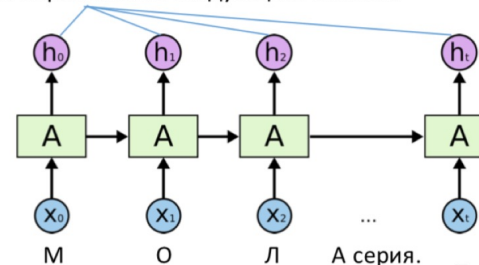
***N*-грамма** кодируется  
вектором

**Обучение** – подбор весов  
рекуррентной нейросети

**Генерация** – подсчет  
вероятности на основе всего  
предыдущего контекста



Вектора вероятностей следующего символа



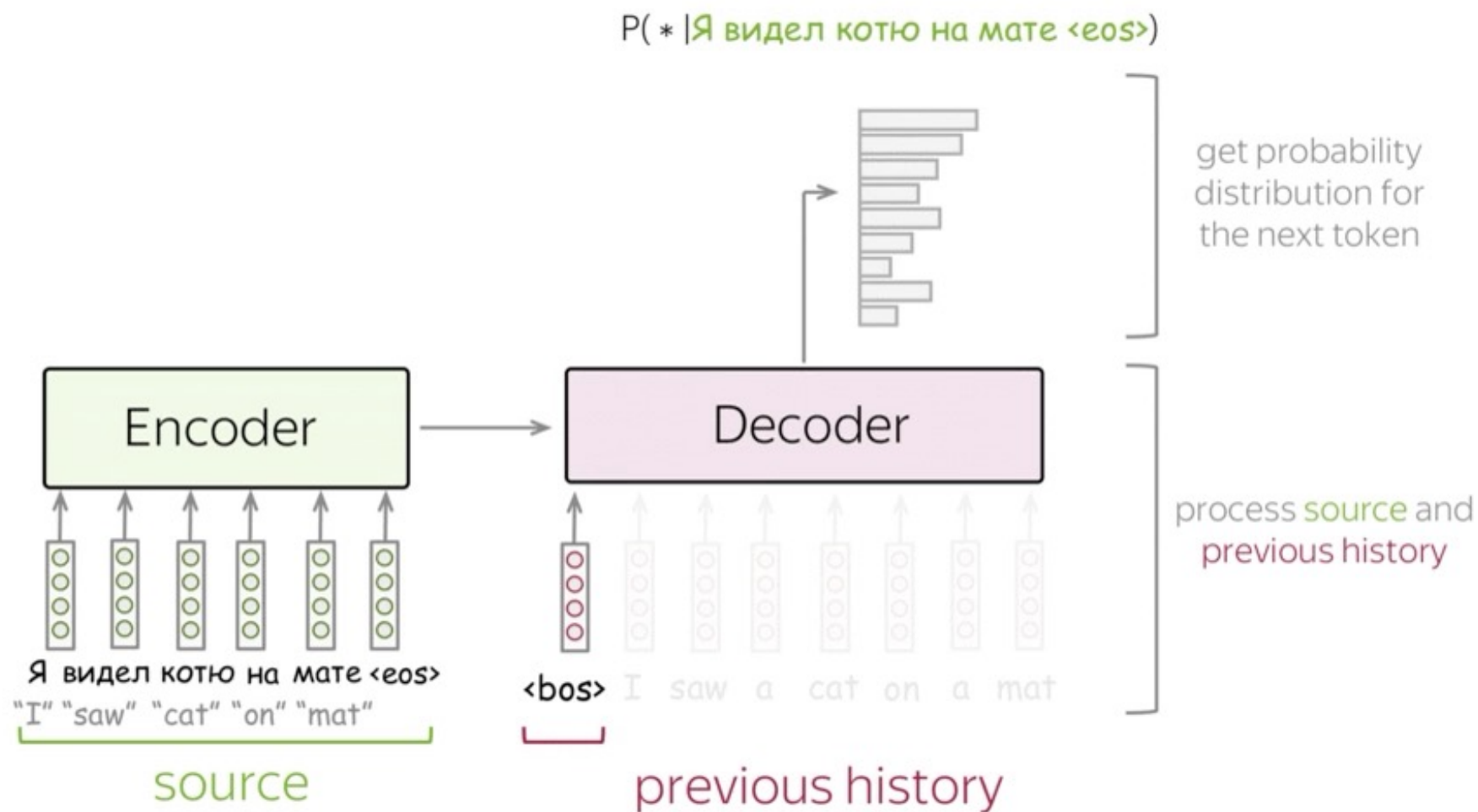
**Выборка для обучения:**

МОЛА серия  
ЭРИКСЛУНД картина  
ДРАГЕТ стеллаж  
ЛЬЮСОГА простыня  
ВОГСБЕРГ / СПОРРЕН рабочий стул

**Результат генерации:**

СЭРДЕРГ серияльная  
БУДОРЕ фидушка  
ЛЕЗОССХОЛЬЕНЕ подушель и течодвик  
СТРАПЛА подушка шкаф  
ИКОТЕРУМ стенник-чвечик  
ЛАПСО подашники  
ВЭБЛИГ вросвал  
ЛИДАЛЕН гирмы,двечное 9 телкая для дков

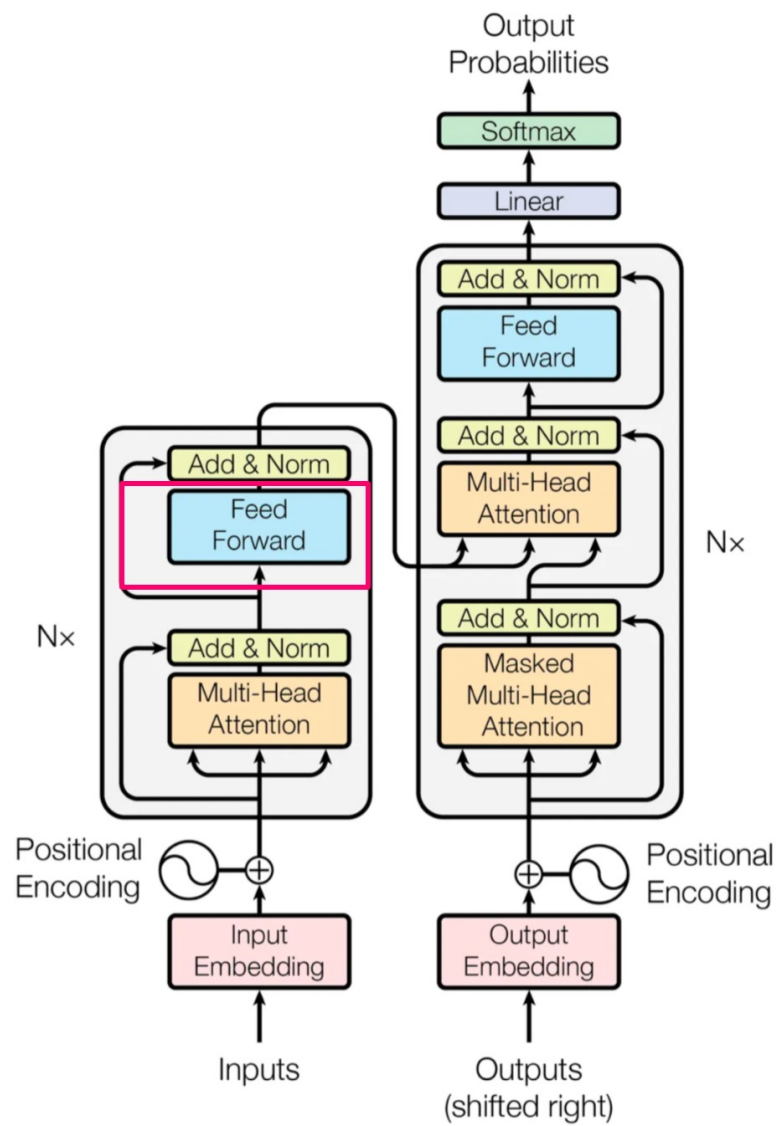
# LSTM : применение



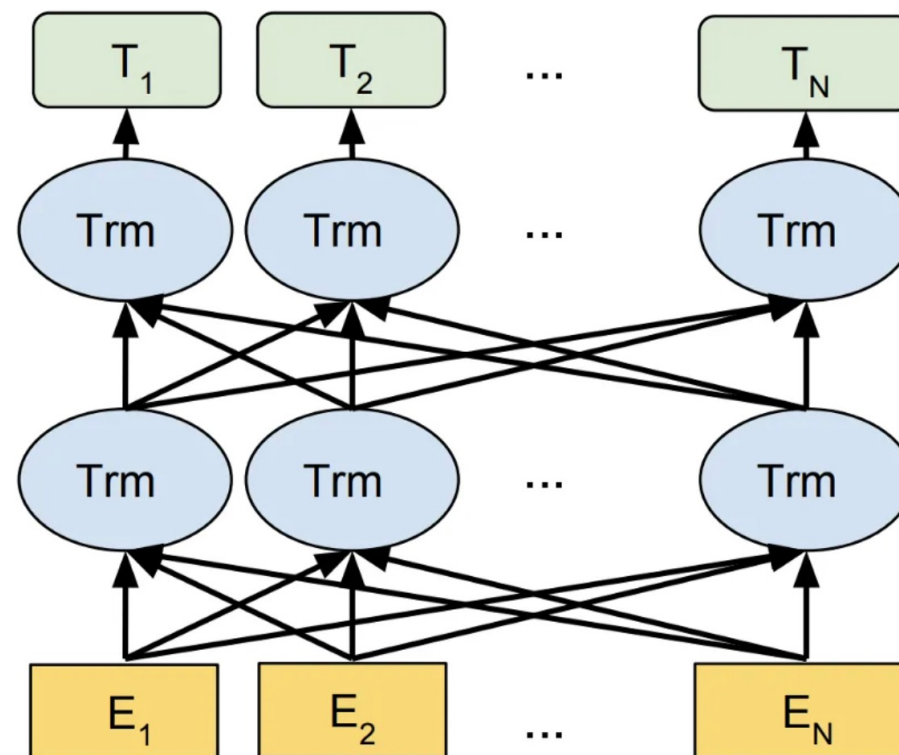
# LSTM : применение



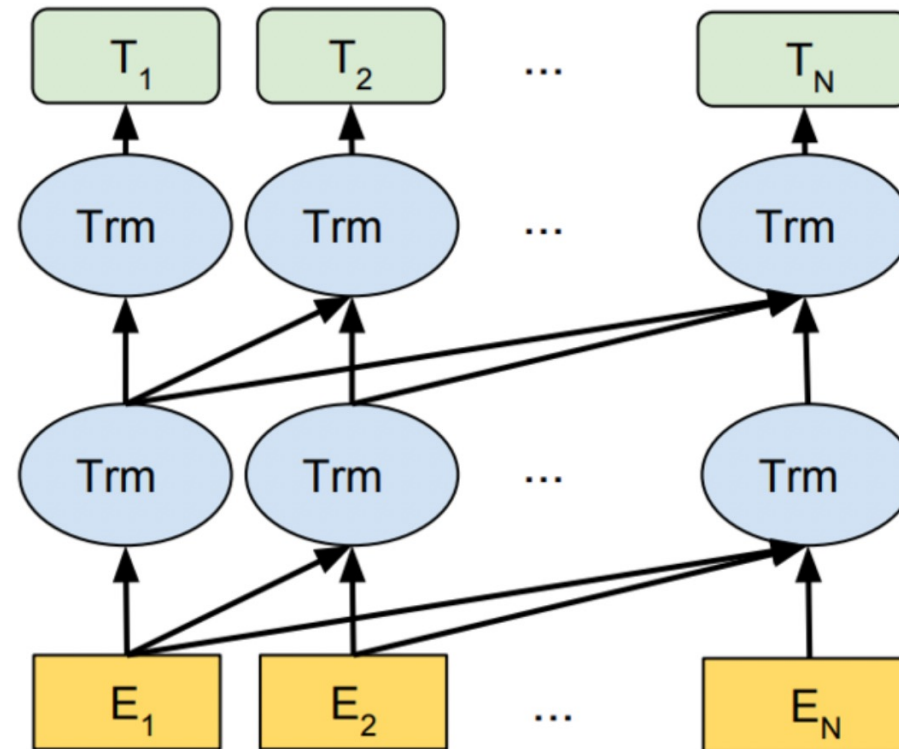
# Transformers



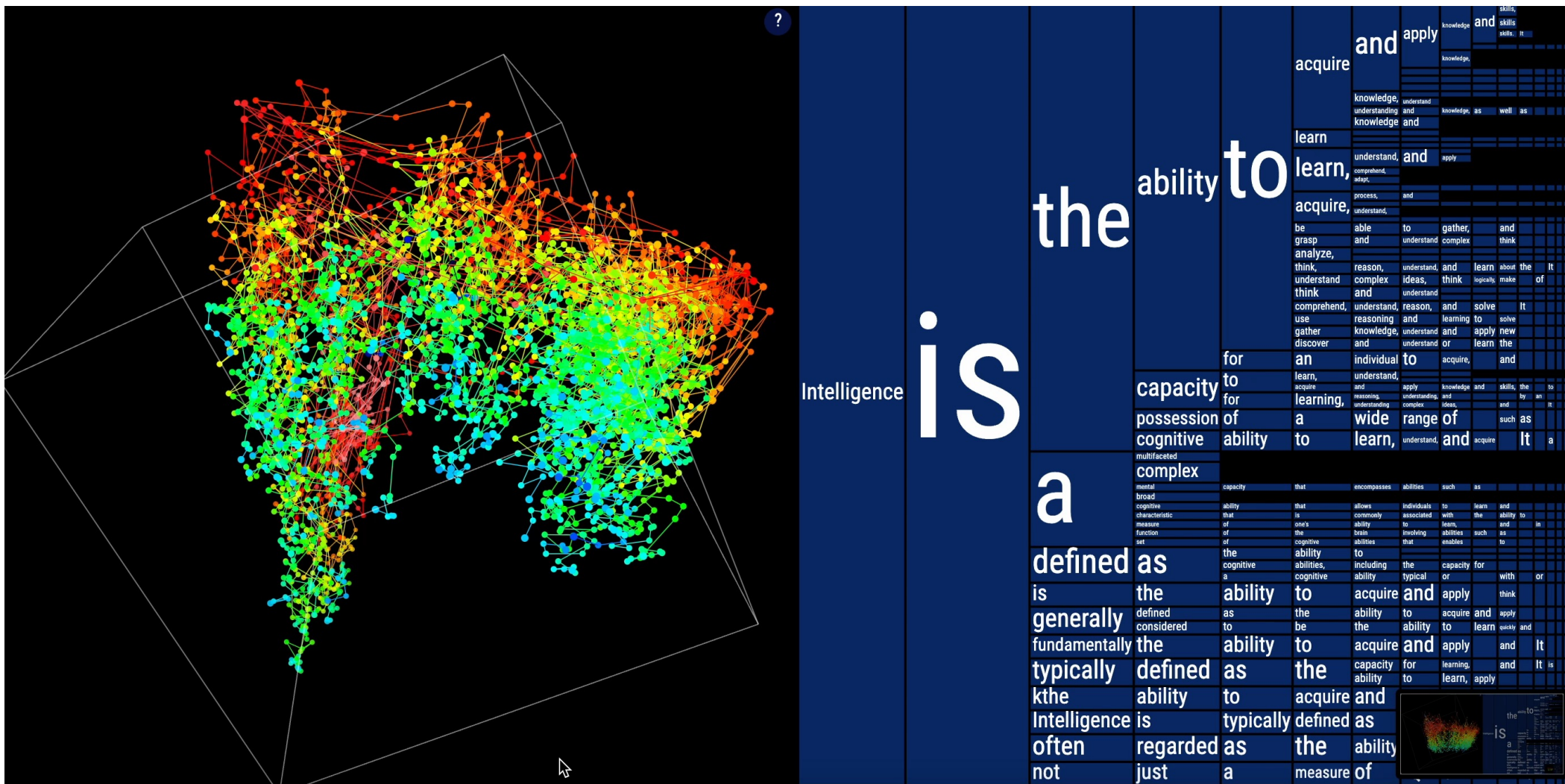
# BERT (Bidirectional Encoder Representations from Transformers)



# OpenAI GPT (Generative Pre-trained Transformer)



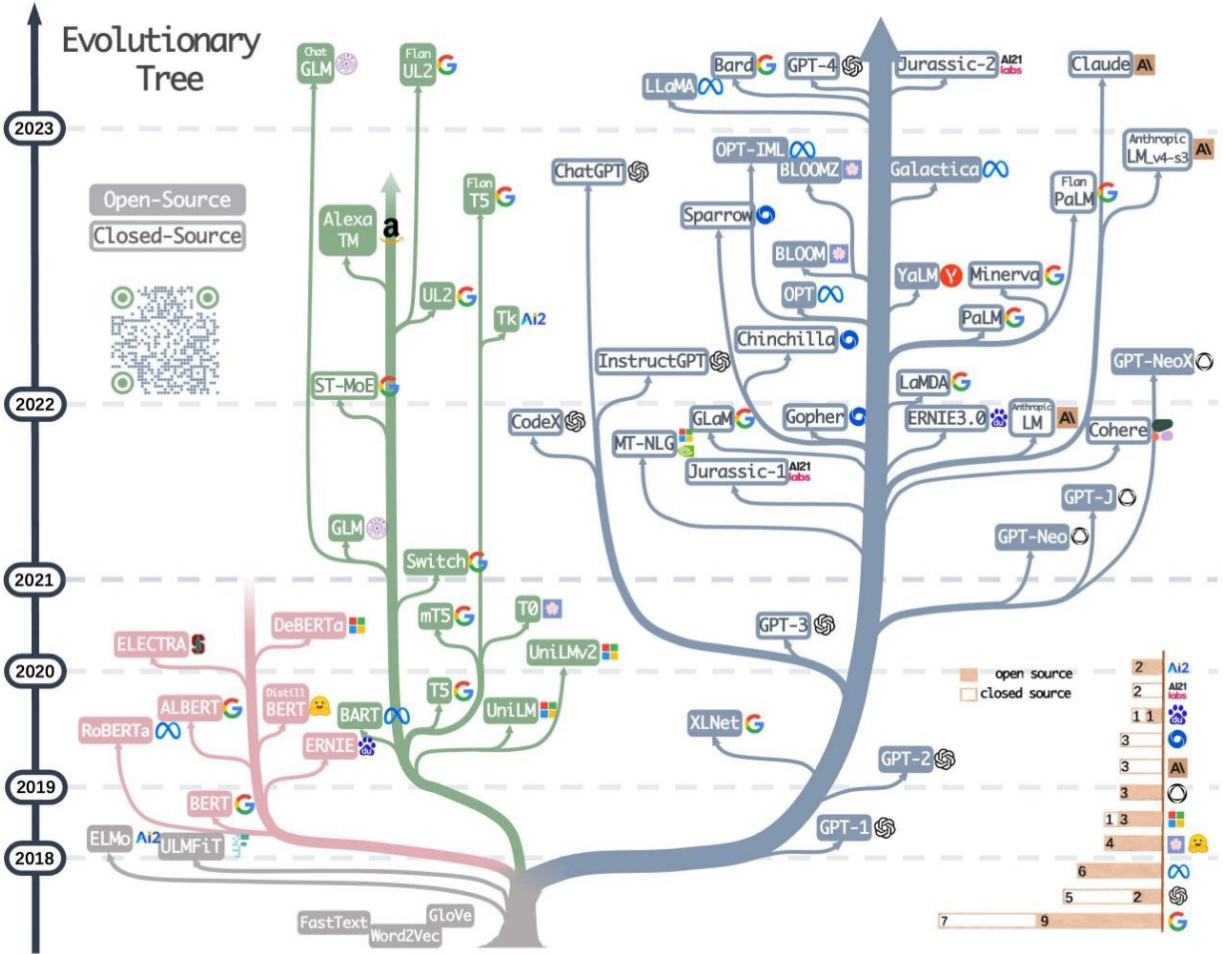
# Визуализация работы



# Рост мощностей

Название	Дата релиза <sup>[a]</sup>	Разработчик	Число параметров <sup>[b]</sup>	Размер корпуса текстов
BERT	2018	Google	340 миллионов <sup>[9]</sup>	3,3 миллиарда слов <sup>[9]</sup>
GPT-2	2019	OpenAI	1,5 миллиарда <sup>[11]</sup>	40GB <sup>[12]</sup> (~10 миллиардов токенов) <sup>[13]</sup>
GPT-3	2020	OpenAI	175 миллиардов <sup>[5]</sup>	499 миллиардов токенов <sup>[13]</sup>

YaLM 100B	Июнь 2022	Яндекс	100 миллиардов <sup>[33]</sup>	300 миллиардов токенов <sup>[34]</sup>
BLOOM	Июль 2022	Коллаборация под управлением Hugging Face	175 миллиардов <sup>[6]</sup>	350 миллиардов токенов (1,6TB) <sup>[35]</sup>
AlexaTM (Teacher Models)	Ноябрь 2022	Amazon	20 миллиардов <sup>[36]</sup>	1,3 триллиона <sup>[37]</sup>
LLaMA (Large Language Model Meta AI)	Февраль 2023	Meta	65 миллиардов <sup>[39]</sup>	1,4 триллиона <sup>[39]</sup>
GPT-4	Март 2023	OpenAI	Нет данных <sup>[f]</sup>	Нет данных



# Масштабы модели класса GPT3:

- ✓ 600 гб текста  
~миллион книг, 1/50000 индекса Яндекса
- ✓ 175 млрд параметров
- ✓ ~1/600 мозга человека

**Стоимость: 35 млн \$**

О курсе: план, формы контроля,  
оценка

# План

- В курсе 12 сдвоенных занятий: по понедельникам с 11 до 14 часов (с перерывом)
- 4 домашних задания
- Промежуточное тестирование (КР)
- Устный экзамен в конце

Формула оценки:  **$O = 0.6 * ДЗ + 0.1 * КР + 0.3 * Экзамен$**

# Введение в NLP

# Термины

- Токен – последовательность символов (слово, слог, словосочетание и т д)
- Текст (документ) – последовательность токенов
- Корпус – набор текстов
- Токенизация – представление текста в виде последовательности

# Термины

## Токенизация

```
graph TD; A[Токенизация] --> B[Посимвольно]; A --> C[По словам]; A --> D[По частям слов];
```

### Посимвольно

- Очень маленький словарь
- Очень длинные последовательности

### По словам

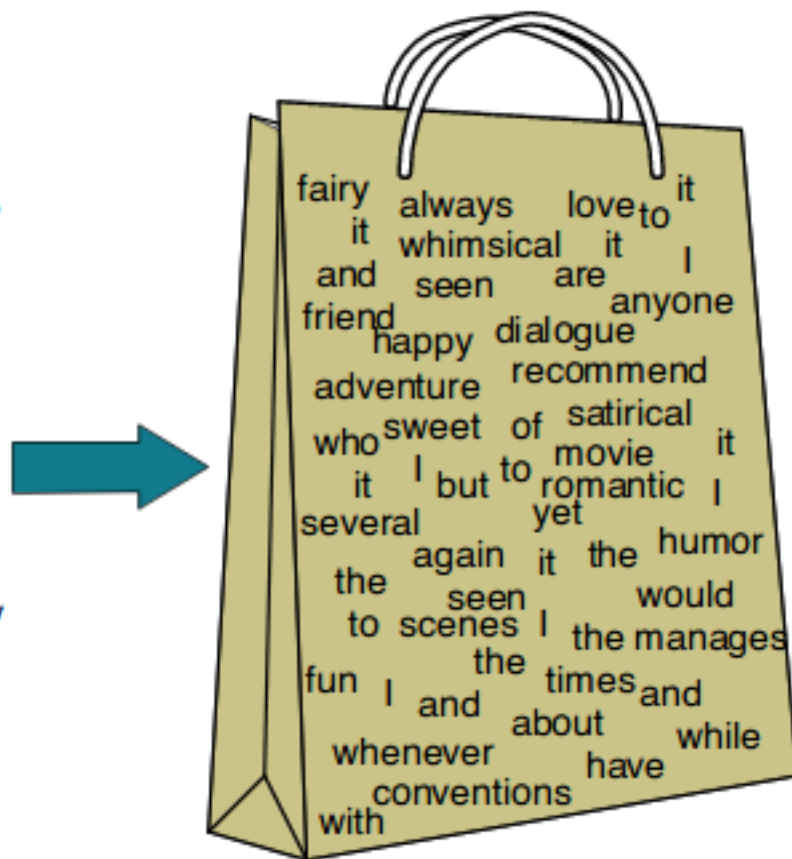
- Очень большой словарь
- Нужен отдельный токен для неизвестных слов (<unk>)

### По частям слов

- Берем лучшее от двух подходов
- Аналогия – mini-batch SGD

# Счетные подходы: кодирование по id ??

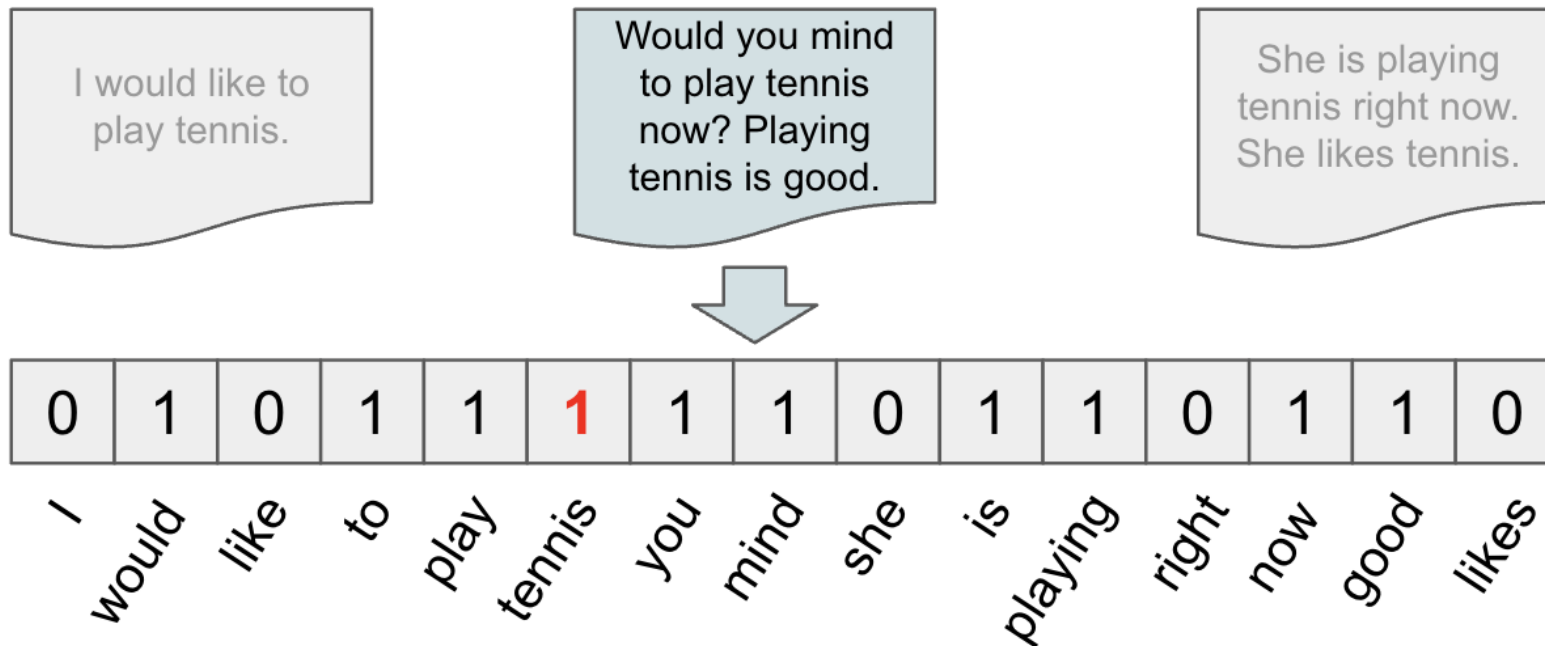
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



word	id
it	1
I	2
the	3
to	4
and	...
seen	
yet	
would	
whimsical	
times	
sweet	
satirical	
adventure	
genre	
fairy	
humor	
have	
great	
...	...

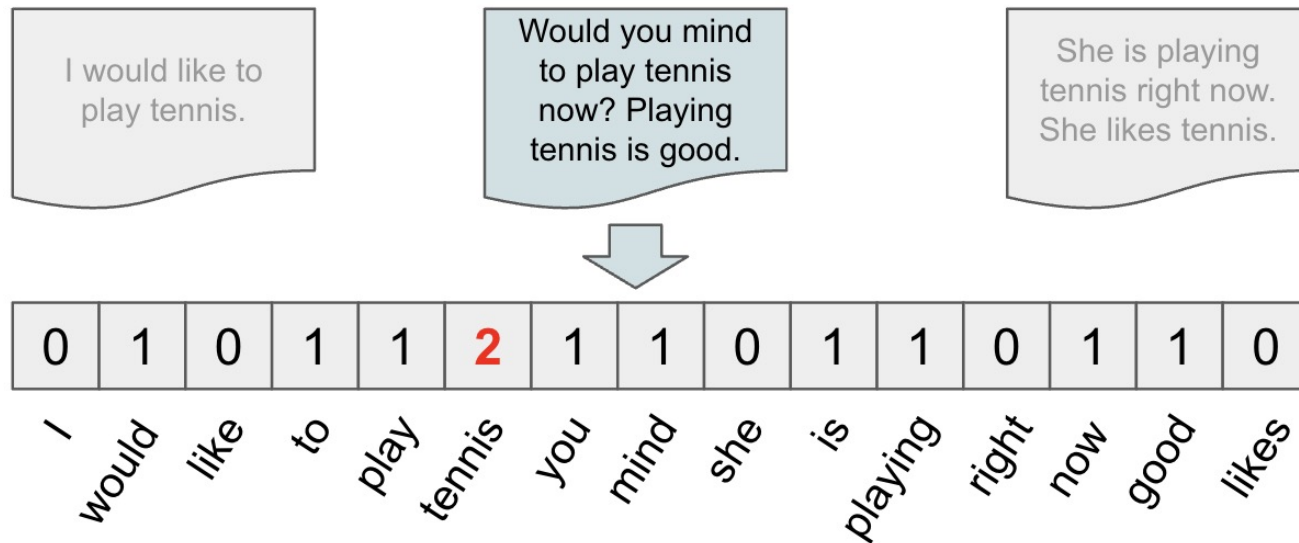
# One-hot encoding

- Составляем словарь из всех слов в корпусе в том виде, в котором они присутствуют в текстах
- Получаем набор из **N** слов (размер словаря)
- Каждый текст = вектор длины N, где каждый элемент отвечает за **факт присутствия** (1 или 0) того или иного слова в тексте



# Мешок слов (Bag of words)

- Составляем словарь из всех слов в корпусе в том виде, в котором они присутствуют в текстах
- Получаем набор из **N** слов (размер словаря)
- Каждый текст = вектор длины N, где каждый элемент отвечает за **количество присутствий** того или иного слова в тексте



# Счетные подходы: мешок слов

	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

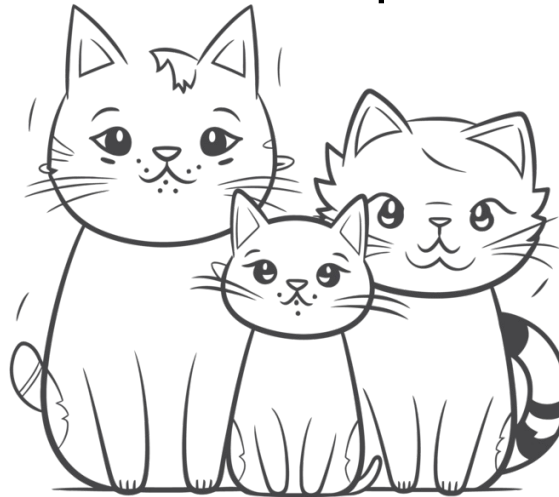
# Счетные подходы: мешок слов

Какие есть недостатки?

# Счетные подходы: мешок слов

- При кодировке теряется информация о порядке слов в тексте

Например, тексты ***I have no cats*** и ***No, I have cats*** будут закодированы одинаково, но имеют противоположный смысл.



- В словаре огромное количество слов (десятки или даже тысячи слов), поэтому каждый текст будет закодирован вектором очень большой длины
- Простой счетчик - не очень хороший вид кодировки, так как в длинных текстах слова встречаются больше раз, чем в коротких

# Tf-Idf

- **TF (Term Frequency)** – как часто встречается слово *в рамках одного конкретного текста*.

$$TF(t, d) = \frac{c_{t,d}}{|d|},$$

# Tf-Idf

- **IDF (Inverse Document Frequency)** – как часто встречается слово *в других текстах*.
- Если слово встречается часто во *всех документах*, то оно не такое важное.

$$IDF(t, D) = \log \frac{|D|}{|d \in D : t \in d|},$$

# Счетные подходы: tf-idf

- TF (Term Frequency) - это число, которое отражает важность слова в конкретном тексте:

$$TF(t, d) = \frac{c_{t,d}}{|d|},$$

- IDF (Inverse Document Frequency) - число, обратное важность слова для всего набора текстов (корпуса)

$$IDF(t, D) = \log \frac{|D|}{|d \in D : t \in d|},$$

Общий вид кодировки:

$$TfIdf(t, d) = TF(t, d) \cdot IDF(t, D).$$

# Счетные подходы

- + Хорошо работают в комбинации с линейными моделями для решения простых задач
- + Быстрые модели, простой результат
- Не работают в сложных задачах NLP
- Не подходят для задач генерации текста

# Классический пайплайн работы с текстами

*Would you mind to play tennis  
now? Playing tennis is good.*

1. Токенизация
2. Очистка от стоп-слов

[~~would~~, ~~you~~, mind, ~~to~~, play, tennis,  
now, playing, ~~is~~, good]

## 3. Нормализация

### Лемматизация

[mind, play, tennis, now,  
good]

### Стемминг

[min, pla, tenn, now, good]

## 4. Очистка от очень редких слов

## 6. Моделирование

## 5. Bag of Words, TF-IDF

# Практика!

<https://colab.research.google.com/drive/1PPdjb9gT63ff125pPrex1KJX0phxK4l1?usp=sharing>