

# Генеративные языковые модели GPT3 few-/zero-shot learning

Алексей Бурлаков

# План

- Ресар: языковые модели
- GPT и другие трансформерные декодеры
- Методы генерации текста
- Few-/zero-shot learning

**Ресар: языковые модели**

# У каждого из нас есть своя языковая модель!

- Мы используем нашу «внутреннюю» языковую модель, чтобы выбрать правильный вариант:

Я хочу назвать моего кота наполеон.

Я хочу назвать моего кота на поле он.



Я хочу назвать моего кота Наполеон.

*Но как быть машине?*

Компьютеру на помощь приходит вероятность!

# Вероятность предложения

**Probability of a sentence** = вероятность предложения на естественном языке

Примеры:

$P(\text{Кот лежит на диване}) > P(\text{На диване кот лежит})$

$P(\text{Красивая девочка играла в мяч}) > P(\text{В мяч играла девочка красивая})$

# Языковые модели (Language models)

- Мы не знаем истинные вероятности предложения  
(= последовательности токенов).
- Но мы можем обучить **языковую модель**  
**оценивать** эти вероятности.
- Обучаем модель на корпусе текстов, чтобы она могла  
оценить вероятность предложения.

# Языковое моделирование

Задача языкового моделирования: **предсказать вероятность текстового фрагмента ИЛИ предсказать вероятность следующего слова в последовательности**

## Зачем?

- “Более вероятный” часто означает “лучше”
  - Can be used for spelling correction, translation, speech recognition etc.
- Сэмплируя из распределения можем генерировать тексты

# Языковое моделирование

Задача языкового моделирования: **предсказать вероятность текстового фрагмента ИЛИ предсказать вероятность следующего слова в последовательности**

**Как?**

**Решение:** цепное правило

$$\begin{aligned} &P(\text{Я хочу назвать кота Наполеон}) = \\ &= P(\text{Наполеон} | \text{Я хочу назвать кота}) \times P(\text{Я хочу назвать кота}) = \\ &= P(\text{Наполеон} | \text{Я хочу назвать кота}) \times P(\text{кота} | \text{Я хочу назвать}) \times \\ &\quad P(\text{назвать} | \text{Я хочу}) \times P(\text{хочу} | \text{Я}) \times P(\text{Я}) \end{aligned}$$



# Языковые модели (Language models)

## • Типы языковых моделей:

- **Count-based Models** (Statistical) – основаны на статистических методах.
- **Neural Language Models** – нейросетевые модели!

Сегодня рулят большие нейросети



# Оценка языковых моделей

Как оценить качество языковой модели?

# Перплексия и правдоподобие

Как оценить качество языковой модели?

Когда мы тренируем LM, мы (эквивалентно)

- *maximize log-likelihood:*
- *minimize cross-entropy:*
- ***minimize perplexity***

(неуверенность модели  
в тексте):

обратная вероятность  
тестового набора,  
нормализованная по  
количеству слов

$$LL = \log p(\text{text}) = \sum_{i=1}^n \log p(y_i | y_{<i})$$

$$CE = -\frac{1}{n} \sum_{i=1}^n \log p(y_i | y_{<i}) = -\frac{LL}{n}$$

$$PP(W) = \sqrt[n]{\frac{1}{P(w_1, w_2, \dots, w_N)}}$$

# Перплексия и правдоподобие

Как оценить качество языковой модели?

Когда мы тренируем LM, мы (эквивалентно)

- *maximize log-likelihood:*
- *minimize cross-entropy:*
- ***minimize perplexity***

(*неуверенность модели  
в тексте*):

обратная вероятность  
тестового набора,  
нормализованная по  
количеству слов

**Выражение через Энтропию:**

$$LL = \log p(\text{text}) = \sum_{i=1}^n \log p(y_i | y_{<i})$$

$$CE = -\frac{1}{n} \sum_{i=1}^n \log p(y_i | y_{<i}) = -\frac{LL}{n}$$

$$PP(W) = \sqrt[n]{\frac{1}{P(w_1, w_2, \dots, w_N)}}$$

$$PP(W) = 2^{H(W)} = 2^{-\frac{1}{N} \log_2 P(w_1, w_2, \dots, w_N)}$$

# Перплексия и правдоподобие

Как оценить качество языковой модели?

Когда мы тренируем LM, мы (эквивалентно)

- *maximize log-likelihood:*
- *minimize cross-entropy:*
- ***minimize perplexity***

(неуверенность модели  
в тексте):

обратная вероятность  
тестового набора,  
нормализованная по  
количеству слов

**Выражение через Энтропию:**

$$LL = \log p(\text{text}) = \sum_{i=1}^n \log p(y_i | y_{<i})$$

$$CE = -\frac{1}{n} \sum_{i=1}^n \log p(y_i | y_{<i}) = -\frac{LL}{n}$$

$$PP(W) = \sqrt[n]{\frac{1}{P(w_1, w_2, \dots, w_N)}}$$

$$PP(W) = 2^{H(W)} = 2^{-\frac{1}{N} \log_2 P(w_1, w_2, \dots, w_N)}$$

Для оценки, мы вычисляем эти же величины на тестовых текстах

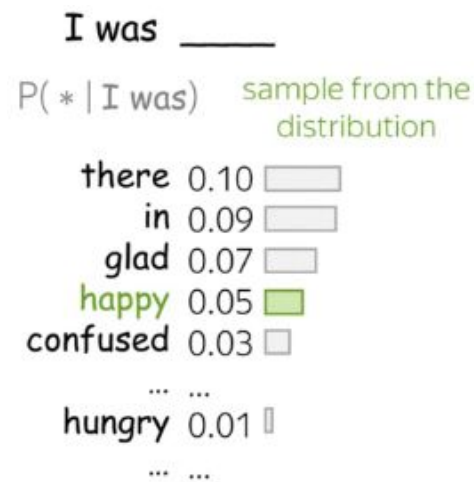
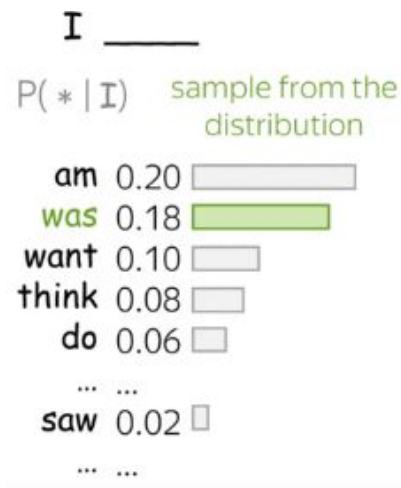
Подробнее тут: <https://habr.com/ru/companies/wunderfund/articles/580230/>

# Задача языкового моделирования для предобучения

- Это **сложная** задача даже (для людей)
- Вынуждает языковые модели **учить разные аспекты языка**
- Не нужно размеченных данных, **только тексты**

# Основные стратегии генерации текста

- Полный перебор
  - **НИрИальНо** -  $O(V^T)$ , но гарантирует лучший текст
- Greedy decoding (*самое вероятно на текущем шаге*)
  - Не гарантирует лучшего текста
- Сэмплирование
- Beam search (*на каждом этапе следим за несколькими лучами*)



# Языковое моделирование наше все

Большинство проблем можно сформулировать как задачу языкового моделирования

## Few Shot Prompt and Predicted Answer

The following are multiple choice questions about high school mathematics.

How many numbers are in the list 25, 26, ..., 100?

(A) 75 (B) 76 (C) 22 (D) 23

Answer: B

Compute  $i + i^2 + i^3 + \dots + i^{258} + i^{259}$ .

(A) -1 (B) 1 (C)  $i$  (D)  $-i$

Answer: A

If 4 daps = 7 yaps, and 5 yaps = 3 baps, how many daps equal 42 baps?

(A) 28 (B) 21 (C) 40 (D) 30

Answer: C

Translate each sentence into a string of emojis.

English: That cat ate the fish.

Emojis: 🐱😋🐟

English: What is this, a house for ants?

Emojis: 🏠🐜?

English: The quick brown fox jumps over the lazy dog.

Emojis: 🦊👉🦊🐶🐕

English: One small step for man, one giant leap for mankind.

Emojis: 🚀🌍👤



# Языковое моделирование наше все

Большинство проблем можно сформулировать как задачу языкового моделирования



Языковая модель способна решить их все разом!

## Few Shot Prompt and Predicted Answer

The following are multiple choice questions about high school mathematics.

How many numbers are in the list 25, 26, ..., 100?

(A) 75 (B) 76 (C) 22 (D) 23

Answer: B

Compute  $i + i^2 + i^3 + \dots + i^{258} + i^{259}$ .

(A) -1 (B) 1 (C)  $i$  (D)  $-i$

Answer: A

If 4 daps = 7 yaps, and 5 yaps = 3 baps, how many daps equal 42 baps?

(A) 28 (B) 21 (C) 40 (D) 30

Answer: C

Translate each sentence into a string of emojis.

English: That cat ate the fish.

Emojis: 🐱😋🐟

English: What is this, a house for ants?

Emojis: 🏠🐜?

English: The quick brown fox jumps over the lazy dog.

Emojis: 🦊👉🦊👈🐕

English: One small step for man, one giant leap for mankind.

Emojis: 🚀🌍👤

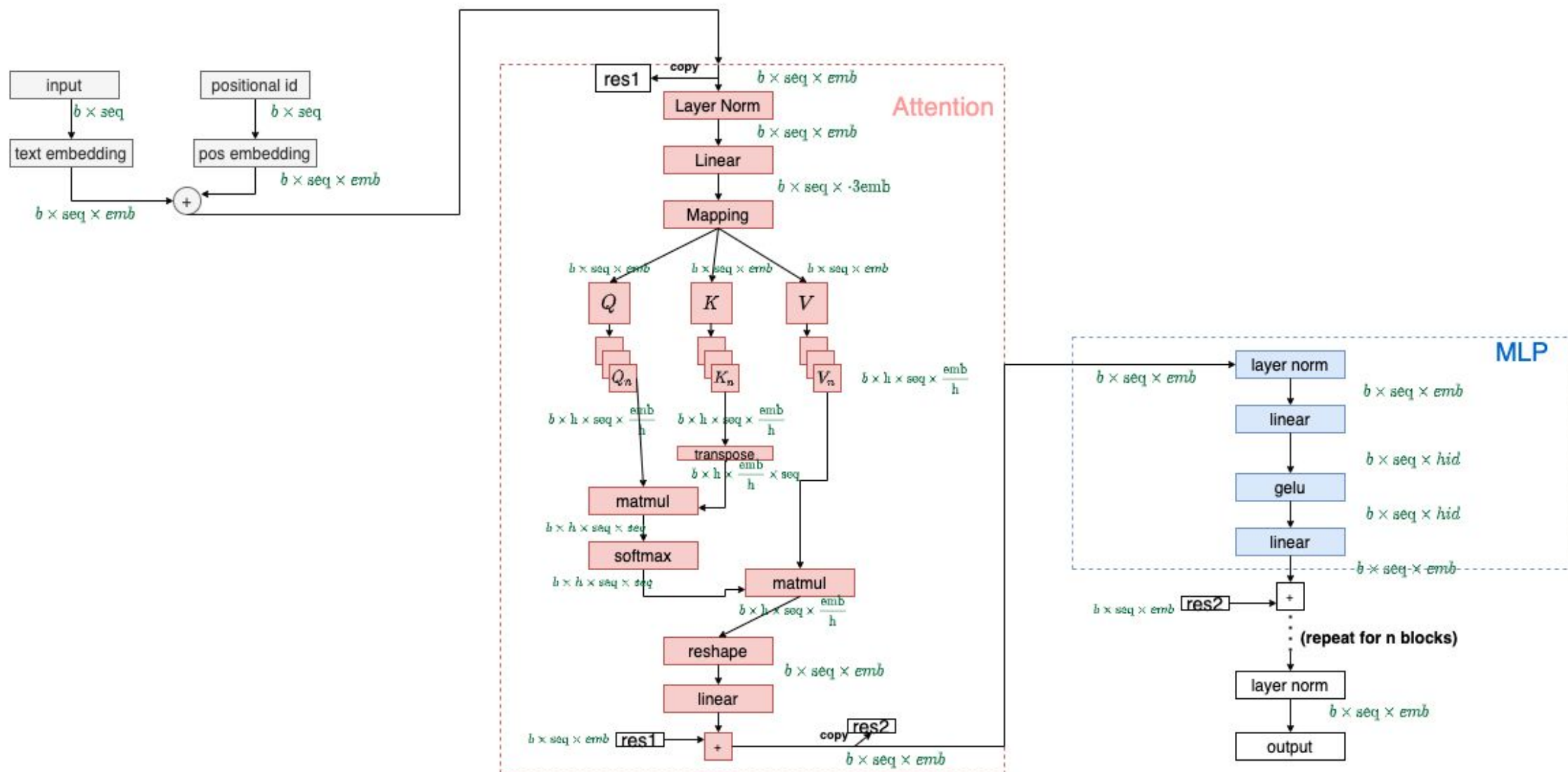
GPT family

# Generative Pretrained Transformers

## Основная идея:

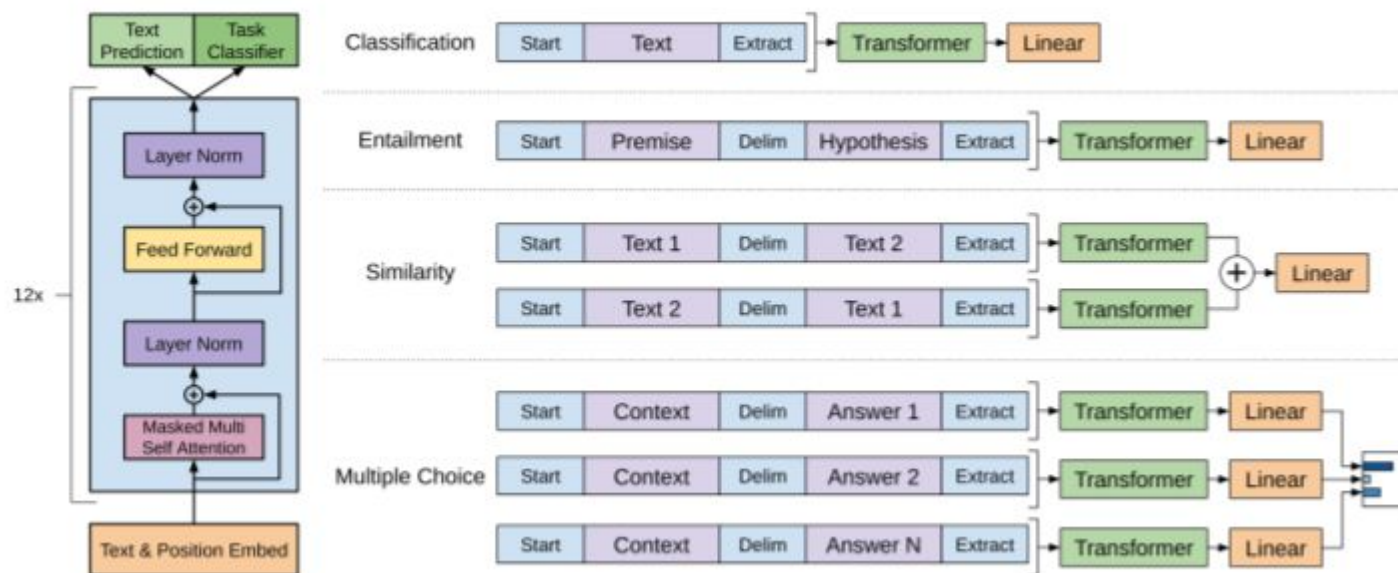
- *Предобучить большой трансформерный декодер на задачу языкового моделирования*
- Сформулировать задачу NLP как задачу *продолжения текста*
- Использовать GPT модель без дополнительного дообучения (или с небольшим дообучением)

# Архитектура GPT



# GPT-1

- *Improving Language Understanding by Generative Pre-Training, Radford et al, 2018*
  - Статья опубликована за несколько месяцев до BERT
- Идея: предобучить трансформерный декодер на задачу языкового моделирования, затем дообучать под NLP задачи



# GPT-2

- *Language Models are Unsupervised Multitask Learners, Radford et al, 2019*
- Модель растёт:
  - Обучение на 40GB текстов, размер 1.5B параметров (vs 117M в GPT-1)
- Показывает хорошие zero-shot способности на ряде задач
- Самая большая версия генерирует качественные и естественные тексты

# GPT-3

- *Language Models are Few-Shot Learners, Brown et al, 2020*
- ОООООчень большая: **училась на 70GB текстов,**  
размер - **175B parameters (!!!)**

# GPT-3

- *Language Models are Few-Shot Learners, Brown et al, 2020*
- ОООООчень большая: **училась на 70GB текстов,**  
размер - **175B parameters (!!!)**
- **Few-shot подход**
  - формулируем задание в виде промпта на естественном языке и применяем модель к задачам NLP без дообучения, показывая ей отдельные примеры-образцы

## Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ..... ← prompt
```

## Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← examples
3 peppermint => menthe poivrée ←
4 plush girafe => girafe peluche ←
5 cheese => ..... ← prompt
```



# Применение GPT-3: NLU

## Prompt

The following is a list of companies and the categories they fall into:

Apple, Facebook, Fedex

Apple

Category:

## Sample response

Technology

Facebook

Category: Social Media

Fedex

Category: Delivery

A simple prompt for extracting airport codes from text.

## Prompt

Extract the airport codes from this text:

Text: "I want to fly from Los Angeles to Miami."

Airport codes: LAX, MIA

Text: "I want to fly from Orlando to Boston"

Airport codes:

## Sample response

MCO, BOS

# Применение GPT-3 : NLG

Explain a piece of Python code in human understandable language.

## Prompt

```
# Python 3
def remove_common_prefix(x, prefix, ws_prefix):
    x["completion"] = x["completion"].str[len(prefix) :]
    if ws_prefix:
        # keep the single whitespace as prefix
        x["completion"] = " " + x["completion"]
    return x

# Explanation of what the code does

#
```

## Sample response

The code above is a function that takes a dataframe and a prefix as input and returns a dataframe with the prefix removed from the completion column.

Turn a product description into ad copy.

## Prompt

Write a creative ad for the following product to run on Facebook aimed at parents:

Product: Learning Room is a virtual environment to help students from kindergarten to high school excel in school.

## Sample response

Are you looking for a way to help your child excel in school? Look no further than Learning Room! Our virtual environment is designed to help students from kindergarten to high school learn and grow. With our help, your child will be able to get ahead in school and achieve their academic goals.

# Применение GPT-3: генерация кода

Translate text into programmatic commands.

## Prompt

Convert this text to a programmatic command:

Example: Ask Constance if we need some bread

Output: send-msg `find constance` Do we need some bread?

Contact the ski store and figure out if I can get my skis fixed before I leave on Thursday

## Sample response

send-msg `find ski-store` Can I get my skis fixed before I leave on Thursday?

Translate natural language to SQL queries.

## Prompt

```
### Postgres SQL tables, with their properties:
#
# Employee(id, name, department_id)
# Department(id, name, address)
# Salary_Payments(id, employee_id, amount, date)
#
### A query to list the names of the departments which employed
more than 10 employees in the last 3 months
SELECT
```

## Sample response

```
DISTINCT department.name
FROM department
JOIN employee ON department.id = employee.department_id
JOIN salary_payments ON employee.id =
salary_payments.employee_id
WHERE salary_payments.date > (CURRENT_DATE - INTERVAL '3
months')
GROUP BY department.name
HAVING COUNT(employee.id) > 10;
```

# Детали имплементации GPT

- Все модели GPT – ванильные трансформерные декодеры
  - GPT-3 дополнительно использует sparse attention
- Контекст 2048 токенов

Model Name	$n_{\text{params}}$	$n_{\text{layers}}$	$d_{\text{model}}$	$n_{\text{heads}}$	$d_{\text{head}}$	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	$6.0 \times 10^{-4}$
GPT-3 Medium	350M	24	1024	16	64	0.5M	$3.0 \times 10^{-4}$
GPT-3 Large	760M	24	1536	16	96	0.5M	$2.5 \times 10^{-4}$
GPT-3 XL	1.3B	24	2048	24	128	1M	$2.0 \times 10^{-4}$
GPT-3 2.7B	2.7B	32	2560	32	80	1M	$1.6 \times 10^{-4}$
GPT-3 6.7B	6.7B	32	4096	32	128	2M	$1.2 \times 10^{-4}$
GPT-3 13B	13.0B	40	5140	40	128	2M	$1.0 \times 10^{-4}$
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	$0.6 \times 10^{-4}$

**Table 2.1:** Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

# GPT-4

- Основана на GPT
- Принимает на вход изображения и тексты, генерирует текст
- Контекстное окно - 8,192 токенов
- Есть версия с 32,768 токенами (50 страниц текста !!!)
- **НЕТ СТАТЬИ**
- **НЕТ ИНФОРМАЦИИ про архитектуру**

Архитектура:

Любопытно, что GPT-4 закрыла собой долгие споры о том, куда все-таки нужно ставить LayerNorm в Трансформере!

# Родственники GPT

- **GPT-Neo от EleutherAI (English, публичная)**
- LaMDA от Google (English, диалоговая, закрытая)
- Megatron-Turing NLG от Microsoft от Nvidia (English, закрытая)
- PanGu- $\alpha$  от Huawei (Chinese, публичная)
- Wu Dao by BAAI (Chinese, закрытая)
- HyperCLOVA by Naver (Korean, закрытая)
- **RuGPT by Sber (Russian, публичная)**
- YaLM by Yandex (Russian, публичная)
- **XGLM by Meta (multilingual, публичная)**
- **mGPT by Sber (multilingual, публичная)**
- ChatGPT (multilingual, закрытая)
- GPT-4 (multilingual, закрытая)
- **Llama, Mistral, OPT, GLM, PaLM, Falcon, MPT, Claude, Phi, ...**

# Возможное будущее GPT

- Аналогичные модели для других языков
- GPT подобные модели для других модальностей (картинки)
- Больше разных стратегий генерации текста
- Ускорение генерации
- Квантизация весов
- Большое контекстное окно
- Различные размеры
  - Больше для лучше качества?
  - Меньше для эффективности?

# Prompt tuning



# Prompt-tuning: motivation

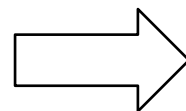
- Fine-tuning:
  - дорого
  - вычислительно затратно
  - нужны размеченные данные
- Zero-shot:
  - эффективен
  - сложно подбирать промпты

# Prompt-tuning: motivation

- Zero-shot:
  - эффективен
  - сложно подбирать промпты

## *Tweet sentiment*

```
positive = 'Веселый твит' + tweet_text  
negative = 'Грустный твит' + tweet_text
```



0.7 accuracy

```
positive = 'Веселый твит' + tweet_text + ')))'  
negative = 'Грустный твит' + tweet_text + '(((
```



0.9 accuracy

# Давайте обучать промпты градиентным спуском

- Prefix-Tuning: Optimizing Continuous Prompts for Generation<sup>1</sup>
- GPT Understands, Too (P-Tuning)<sup>2</sup>
- The Power of Scale for Parameter-Efficient Prompt Tuning<sup>3</sup>
- Continuous prompt tuning for Kussian: how to learn prompts efficiently with rugpt3?<sup>4</sup>

---

<sup>1</sup>Li et al. (Jan 2021)

<sup>2</sup>Liu et al. (Mar 2021)

<sup>3</sup>Lester et al. (Apr 2021)

<sup>4</sup>Konodvuk et Tikhonova (Dec 2021)

# Идея

Translate English to French

sea otter => loutre de mer

plush giraffe => girafe peluche

cheese => **⟨MASK⟩**

# Идея

Translate English to French

sea otter => loutre de mer

plush giraffe => girafe peluche

cheese => **<MASK>**

## Формат затравки:

Translate English to French

{word\_in\_english} => {word\_in\_french}

{word\_in\_english} => {word\_in\_french}

{word\_in\_english} => **<MASK>**

## Элементы:

instance queries

targets

task instructions

# Идея

## Общий формат:

⟨TI⟩ ⟨task context⟩ ⟨TI⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨MASK⟩

## Элементы:

instance queries   targets   task instructions   task context   instance context

# Идея

## Общий формат:

⟨TI⟩ ⟨task context⟩ ⟨TI⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨MASK⟩

## Zero-shot:

⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨MASK⟩

## Элементы:

instance queries   targets   task instructions   task context   instance context



# Идея

## Общий формат:

⟨TI⟩ ⟨task context⟩ ⟨TI⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨instance target⟩  
⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨MASK⟩

## Zero-shot:

⟨TI⟩ ⟨instance context⟩ ⟨TI⟩ ⟨instance query⟩ ⟨TI⟩ ⟨MASK⟩

## Пример:

⟨learned instructions⟩ Москва была основана в 1147 году на Москве-реке.  
⟨learned instructions⟩ Была ли Москва основана в 12 веке?  
⟨learned instructions⟩ ⟨MASK⟩



# Идея

⟨learned instructions⟩ Москва была основана в 1147 году на Москве-реке.

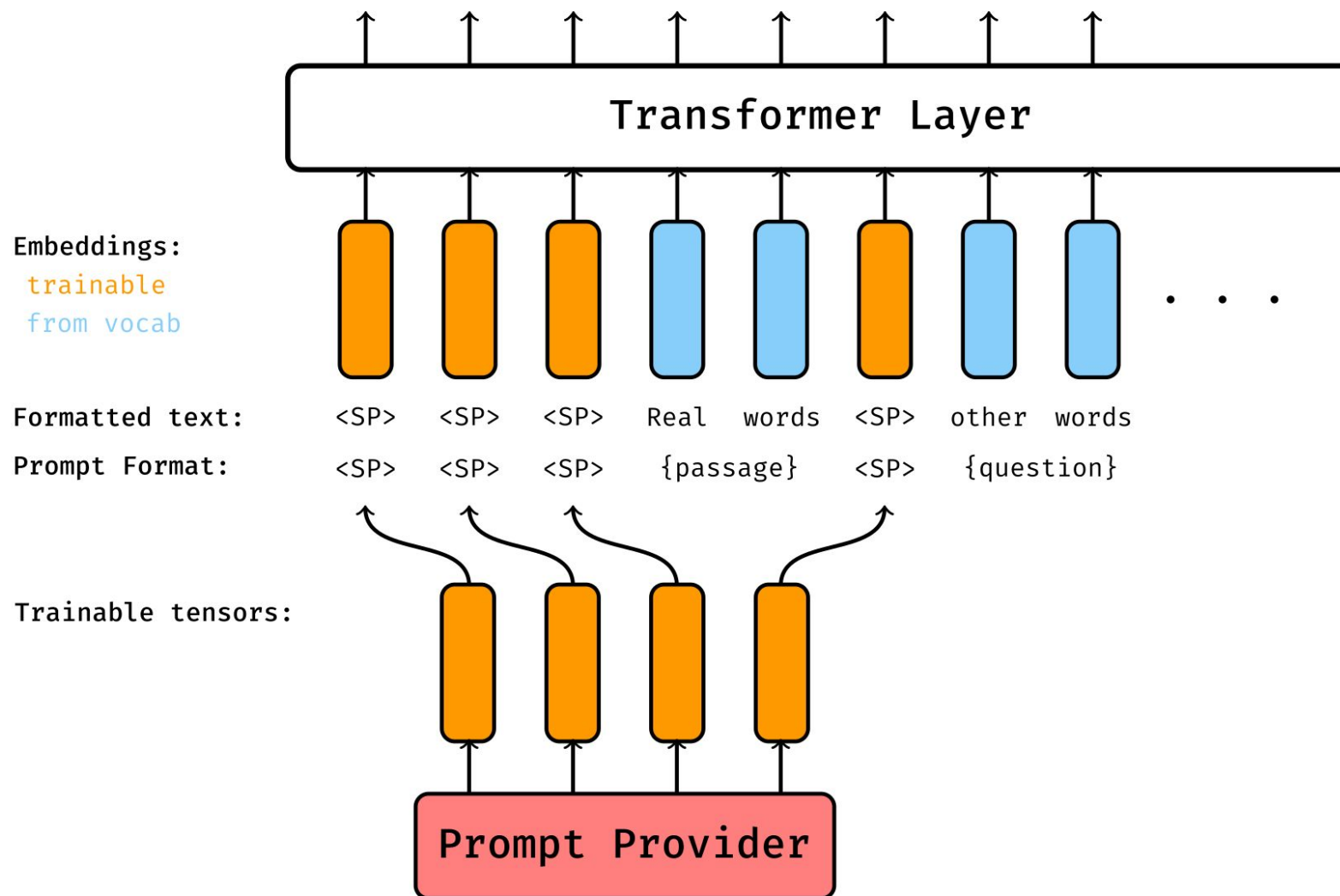
⟨learned instructions⟩ Была ли Москва основана в 12 веке?

⟨learned instructions⟩ ⟨MASK⟩

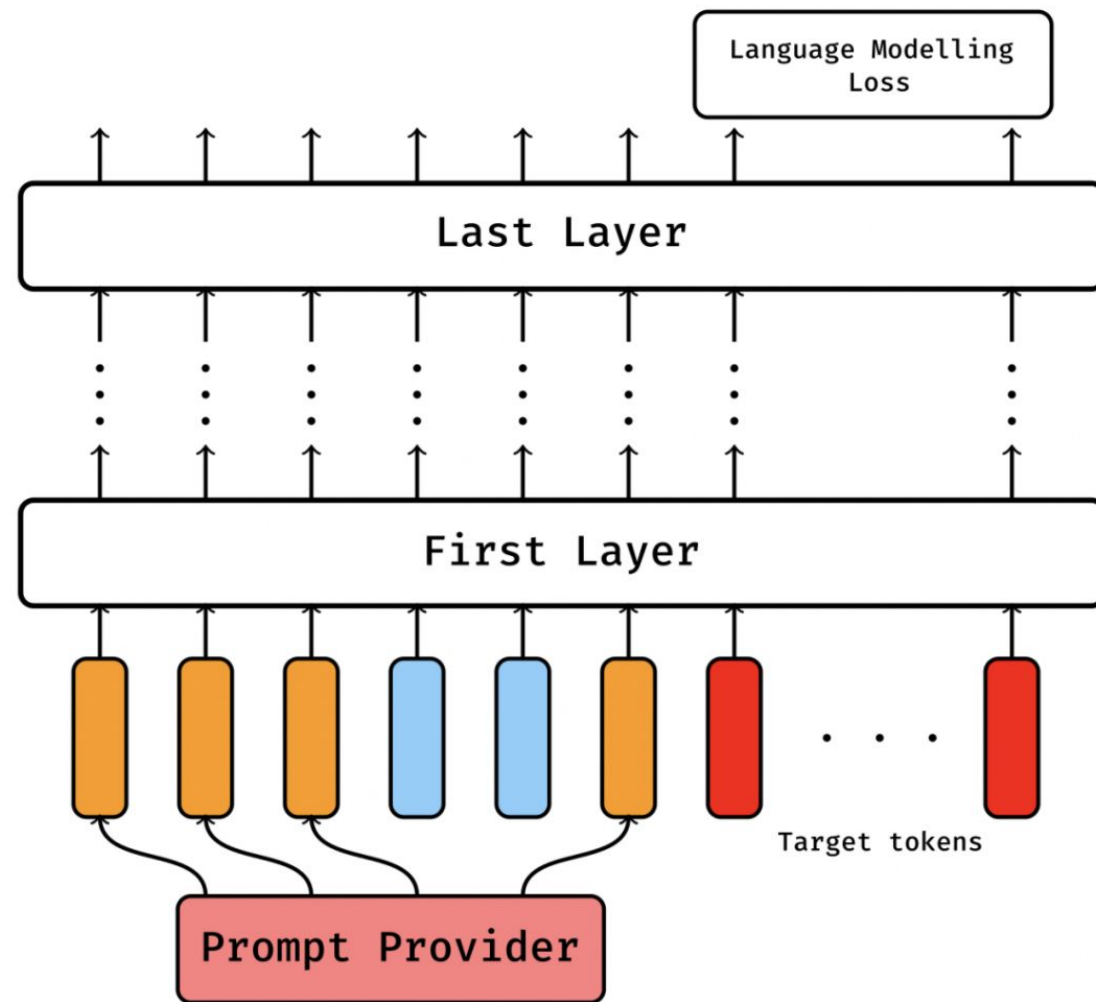
Градиентный спуск для обучения task instructions (TI)

Обучаемые эмбединги получаются через **Prompt provider**

# Детали



# Детали



# ruPrompts Framework

- **Modular structure** for convenient extensibility
- **Integration with HF Transformers**, support for all models with LM head
- **Integration with HF Hub** for sharing and loading pretrained prompts
- **CLI** and configuration system powered by **Hydra**
- **Pretrained prompts** for ruGPT-3

*<https://github.com/ai-forever/ru-prompts>*

# Задачи text2text

- Пословицы
- Суммаризация
- Генерация заголовков
- QA
- Шутки
- Детоксификация

*Предобученные промпты:*

<https://ai-forever.github.io/ru-prompts/pretrained>

# Шутки & Пословицы

## Шутки

**Шел медведь по лесу, видит -- машина горит.**  
Он остановился, достал пистолет и застрелился.

**Умный тостер** не будет жарить хлеб, пока не закончит свои дела.

**После долгих экспериментов свинья-алхимик** синтезировала из навоза метанол.

**Ложась спать, Менделеев** нацарапал на своей диссертации: "Утро вечера мудренее".

- **Доктор**, я больше не могу.
- Что?
- Жить не хочется.
- Что ж, придется немного подождать.

## Пословицы

**Квадрокоптер** – птица не пугливая.

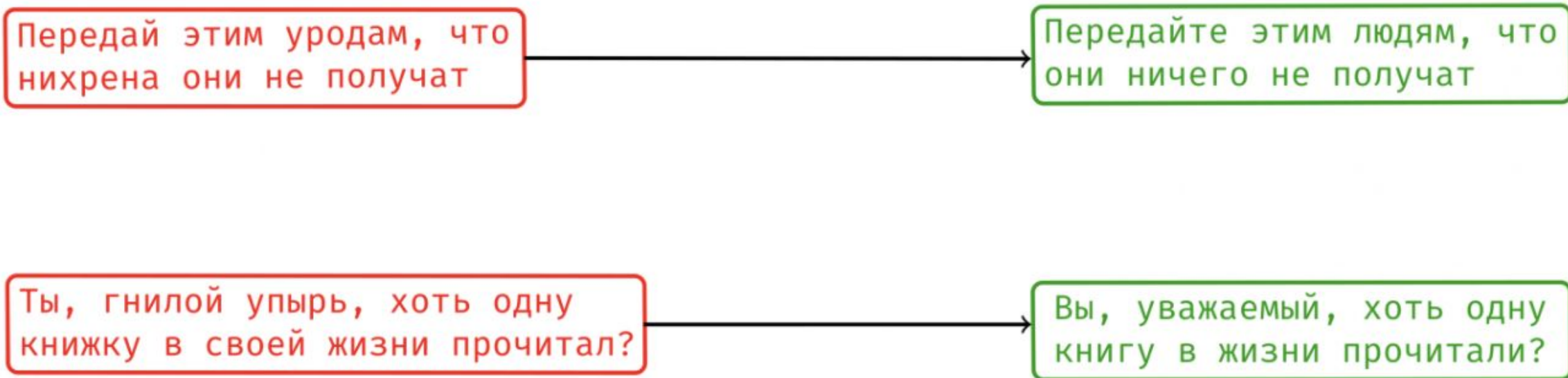
**Нейросеть учить – что** иголку в стоге сена ковырять.

**Программист** – это не тот, кто все знает, а тот, кто все умеет.

**Мудра нейросеть**, да не для всех.

# Детоксификации

Данные: [dataset](#) из соревнования RUSSE 2022 по детоксификации



# Parameter-Efficient Fine-Tuning

<https://github.com/huggingface/peft>

Model	LoRA	Prefix Tuning	P-Tuning	Prompt Tuning	IA3
GPT-2	✓	✓	✓	✓	✓
Bloom	✓	✓	✓	✓	✓
OPT	✓	✓	✓	✓	✓
GPT-Neo	✓	✓	✓	✓	✓
GPT-J	✓	✓	✓	✓	✓
GPT-NeoX-20B	✓	✓	✓	✓	✓
LLaMA	✓	✓	✓	✓	✓
ChatGLM	✓	✓	✓	✓	✓
Mistral	✓				



# LoRA (Low-Rank Adaptation)

Обучаем только матрицы A и B

Все остальные веса модели заморожены

