курс «Глубокое обучение»

Языковые модели

Александр Дьяконов

7 ноября 2022 года

План

Моделирование языка (Language Modeling)

Параметрическое оценивание

RNN-моделирование языка

Подходы к генерированию

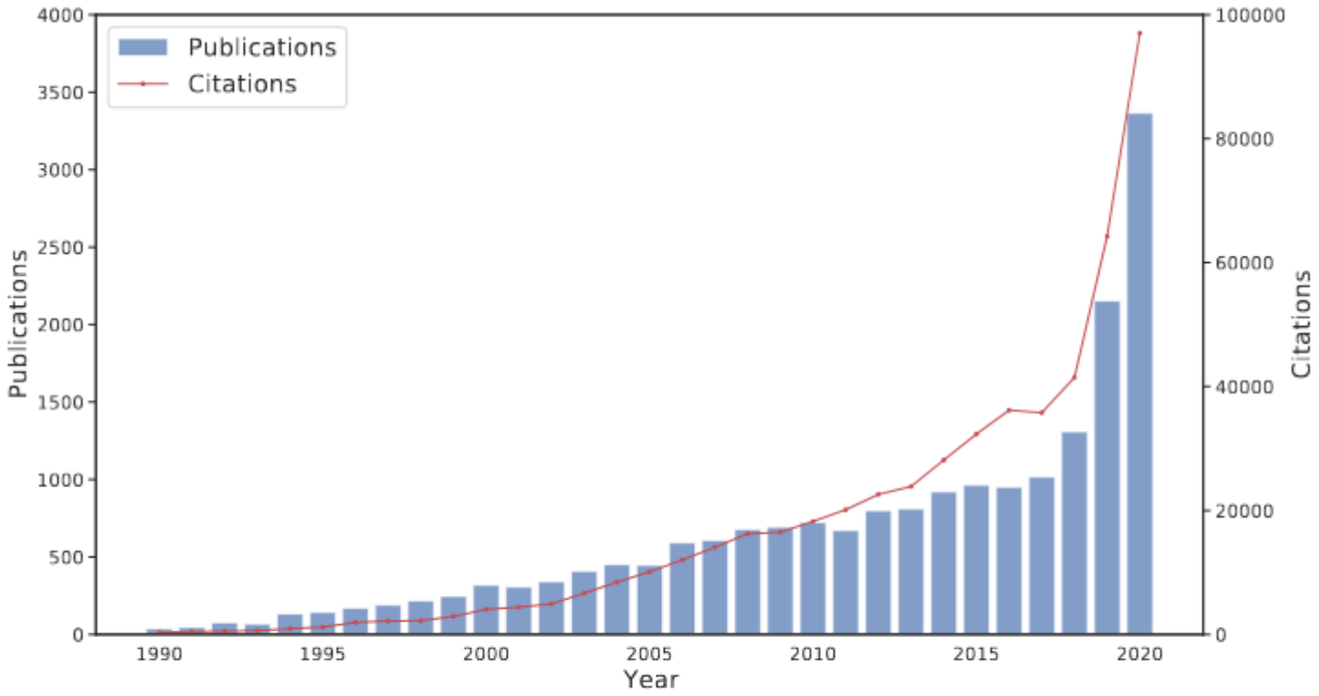
Beam Search (метод луча)

GPT / GPT-2 / GPT-3

BPE и другие способы представления текстов

Извлечение обучающих данных (на примере GPT-2)

Популярность моделирования языка



(a) The number of publications on "language models" and their citations in recent years.

Xu Han et al. «Pre-Trained Models: Past, Present and Future» // https://arxiv.org/pdf/2106.07139.pdf

Моделирование языка (Language Modeling)

Вероятность текста

$$p(x_1,\ldots,x_n)$$

Предсказание следующего слова

$$p(x_n \mid x_1,\ldots,x_{n-1})$$

свойство Маркова

$$p(x_n \mid x_{n-k},\ldots,x_{n-1})$$

```
в лесу родилась ёлочка 0.4
                белочка 0.2
                лисичка 0.1
                берёзка 0.05
                функция 0.0002
                ...
```

Языковые модели в жизни (Language Models)

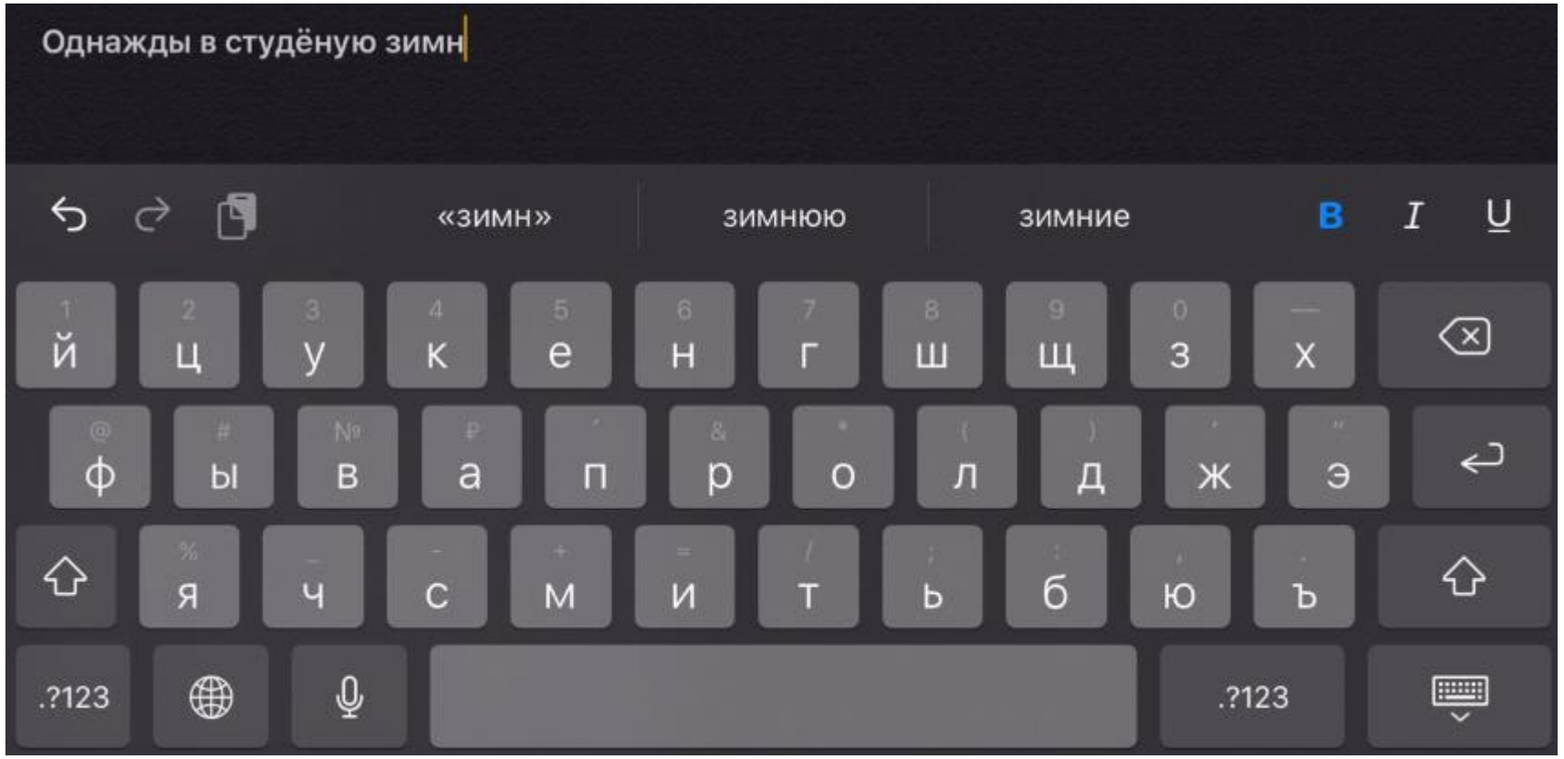
Google

анализ малых

- анализ малых **данных**
- анализ малых **литературных форм**
- анализ малых **данных гос auc**
- анализ малых **выборок**
- анализ малых **предприятий**
- анализ **бесконечно** малых
- **дьяконов** анализ малых **данных**
- **структурный** анализ малых **групп**
- **статистический** анализ малых **выборок**
- анализ **бесконечно** малых **на английском**

Поиск в Google | Мне повезёт!

Пожаловаться на неприемлемые подсказки



Моделирование языка: n-gram Language Models

учимся генерировать текст – как было до DL... n-gram Language Models

Насколько вероятно предложение
«кот поймал в мешок дровосека»

Unigram Modelling

$$p(\text{кот}) \cdot p(\text{поймал}) \cdot p(\text{в}) \cdot p(\text{мешок}) \cdot p(\text{дровосека})$$

Bigram Modelling

$$p(\text{кот}) \cdot p(\text{поймал} \mid \text{кот}) \cdot p(\text{в} \mid \text{поймал}) \cdot p(\text{мешок} \mid \text{в}) \cdot p(\text{дровосека} \mid \text{мешок})$$

Trigram Modelling

$$p(\text{кот}) \cdot p(\text{поймал} \mid \text{кот}) \cdot p(\text{в} \mid \text{кот}, \text{поймал}) \cdot p(\text{мешок} \mid \text{поймал}, \text{в}) \ldots$$

~~в лесу родилась~~ ёлочка, в лесу она MASK

Проблема

в корпусе может не быть некоторых сочетаний

Сглаживание (по Лапласу)

$$p(x_t \mid x_{t-n},\ldots,x_{t-1}) = \frac{\#(x_{t-n},\ldots,x_{t-1},x_t)+\alpha}{\#(x_{t-n},\ldots,x_{t-1})+\alpha\,|V|}$$

Backoff (примерно так...)

при $\#(x_{t-n},\ldots,x_{t-1}) = 0$

$$p(x_t \mid x_{t-n},\ldots,x_{t-1}) = \alpha(x_{t-n},\ldots,x_{t-1})\frac{\#(x_{t-n+1},\ldots,x_{t-1},x_t)}{\#(x_{t-n+1},\ldots,x_{t-1})}$$

умножаем на некоторый «понижающий множитель»

или через частоты меньших порядков (лк с ними)

Марковская парадигма

Проблема

Маленькое обобщение (Lack of Generalization)

если в выборке только

(идти, в, сад), (идти, в, огород)

тогда проблемы при

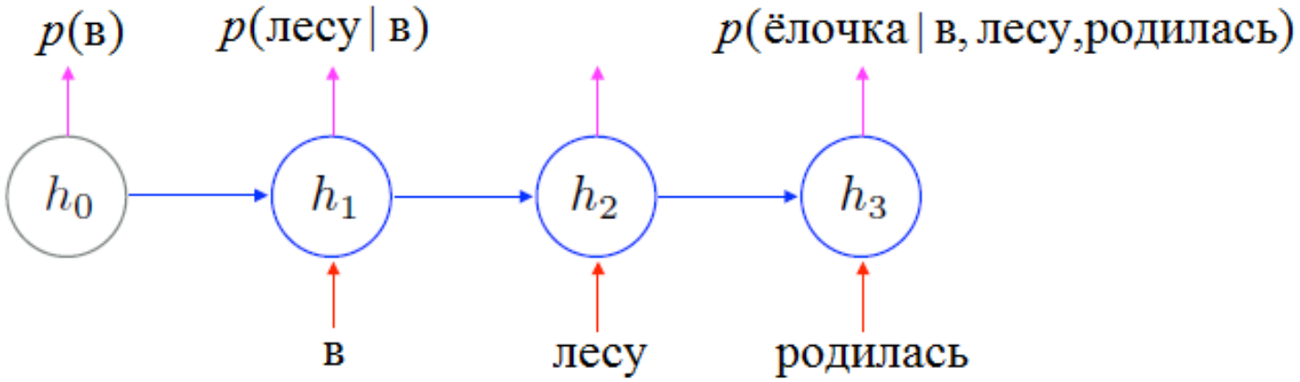
p(идти, в, парк) = ?

Выход: моделирование языка с помощью НС

Немарковские модели: RNN-подход

$$p(x_t,\ldots,x_T)=\prod_{t=1}^{T} p(x_t \mid x_1,\ldots,x_{t-1})$$

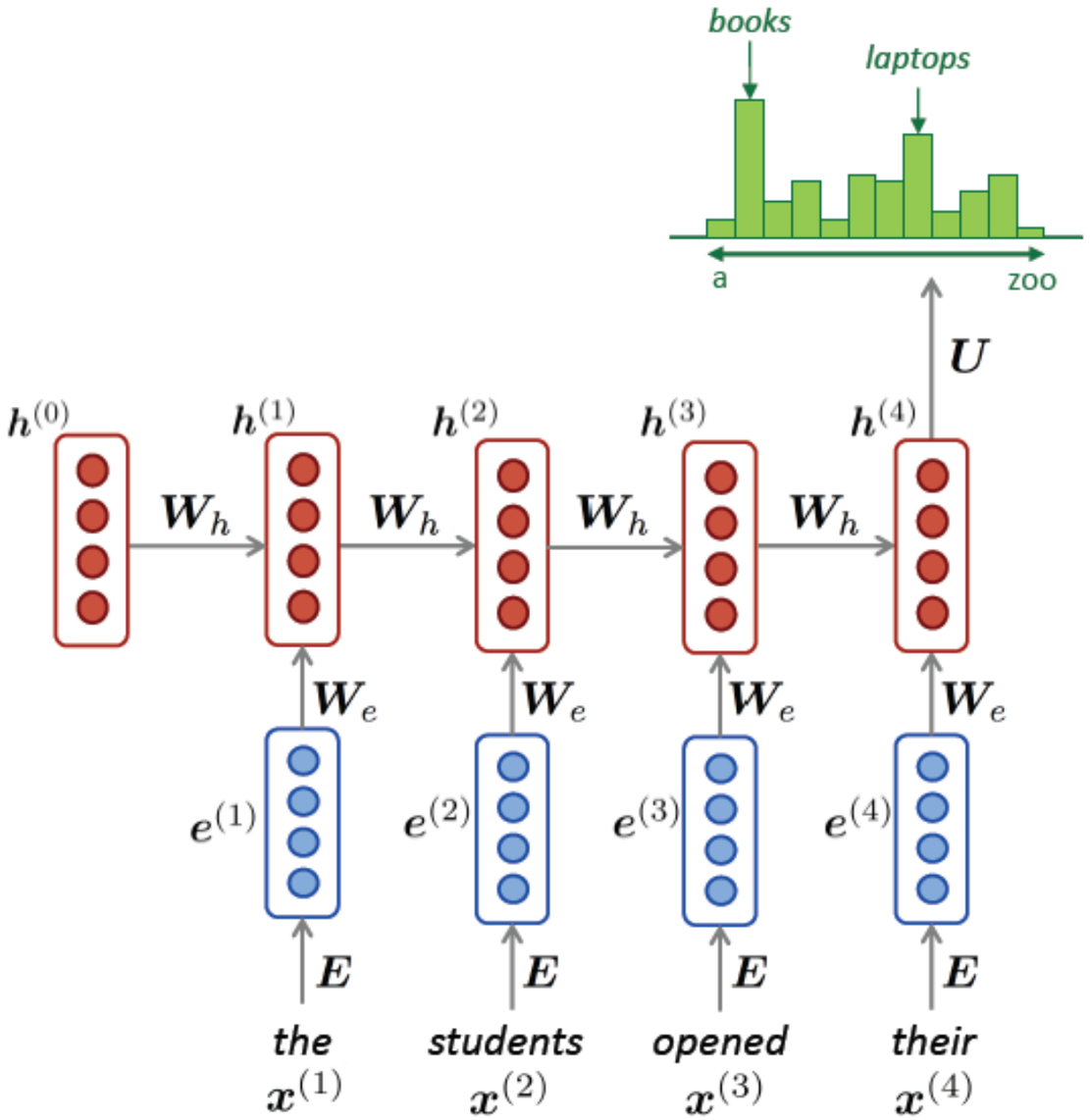
т.е. зависимость от всех слов предложения!



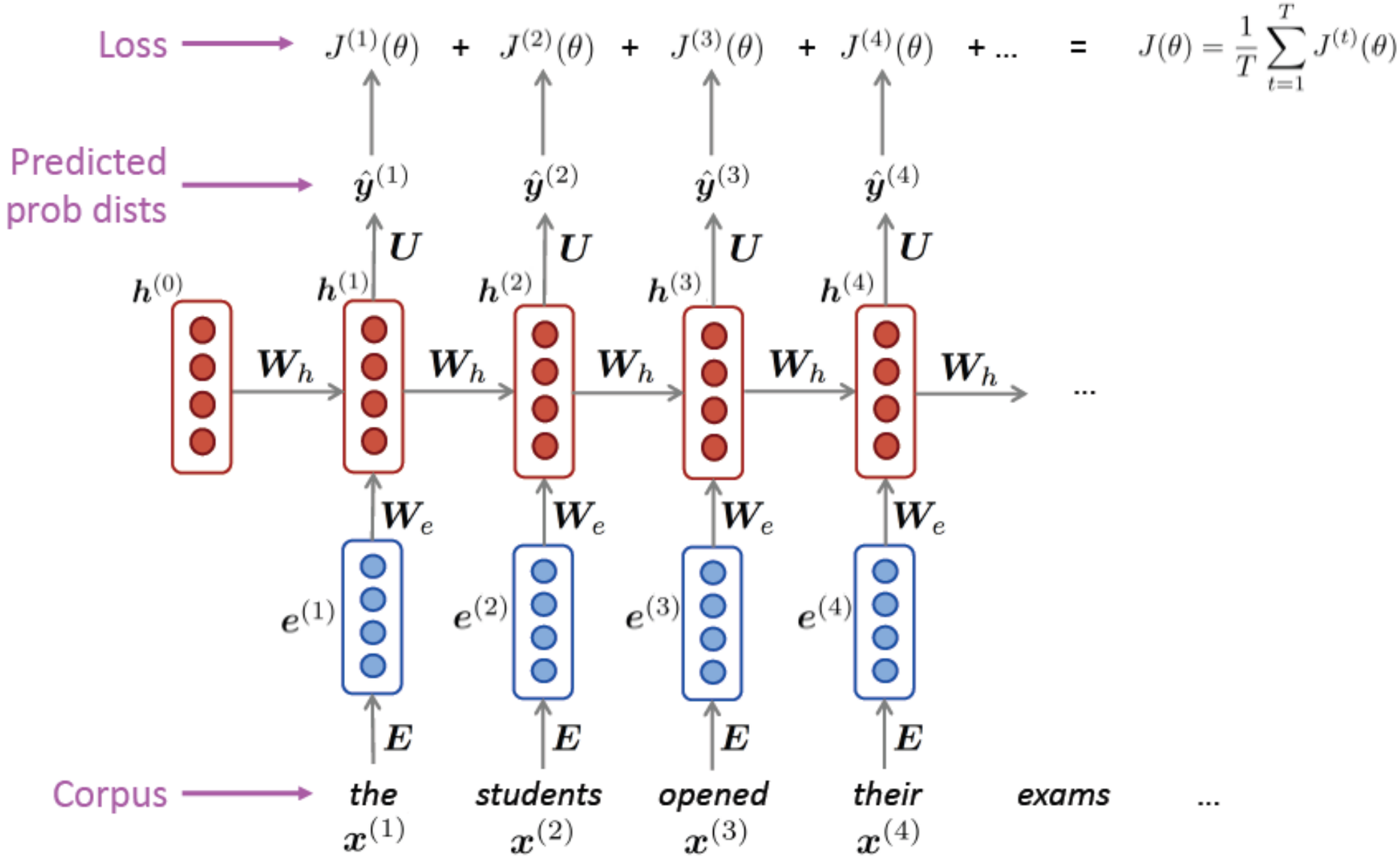
рекуррентная сеть – можно обрабатывать последовательности любой длины!

RNN-моделирование языка

$$\hat{\boldsymbol{y}}^{(4)} = P(\boldsymbol{x}^{(5)}|\text{the students opened their})$$



RNN-моделирование языка: обучение



http://web.stanford.edu/class/cs224n/

Генерирование текста с помощью RNN

итераций	вывод
100	tyntd-iafhatawiaoihrdemot lytdws e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e plia tklrgd t o idoe ns,smtt h ne etie h,hregtrs nigtike,aoaenns lng
300	"Tmont thithey" fomesscerliund Keushey. Thom here sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
700	Aftair fall unsuch that the hall for Prince Velzonski's that me of her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort how, and Gogition is so overelical and ofter.
2000	"Why do what that day," replied Natasha, and wishing to himself the fact the princess, Princess Mary was easier, fed in had oftened him. Pierre aking his soul came to the packs and drove up his father-in-law women.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Подходы к генерации

$$p(x_1,\ldots,x_T)=\prod_{t=1}^{T} p(x_t \mid x_1,\ldots,x_{t-1}) \to \max$$

$$\frac{1}{T}\sum_{t=1}^{T}\log p(x_t \mid \ldots) \to \max$$

лучше среднее арифметическое, чтобы не было коротких предложений

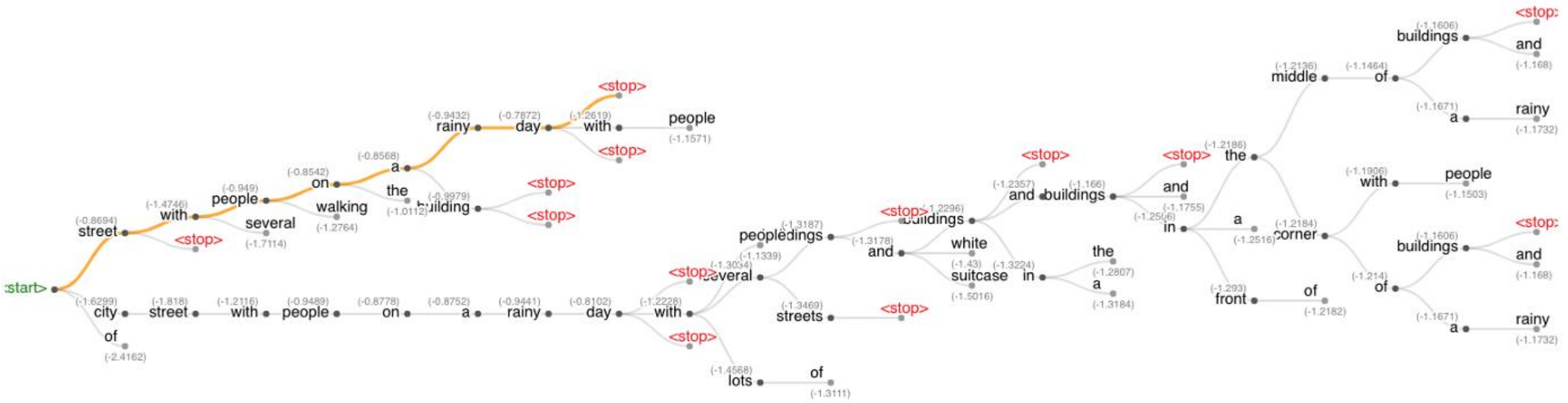
Генерация текста по картинке

Greedy decoder **Large building in the snow in the**
Beam search **Large building in a barn**
Pure sampling decoder **Photo of green boxes in the snow**
Top-k sampling decoder **Large building in the snow away from below**

+ более умные методы (см. дальше)

https://www.katnoria.com/nlg-decoders/

Beam Search (метод луча)



Beam search decoder with k=3 and max steps as 51

На каждом шаге декодера храним *k* наиболее вероятных варианта

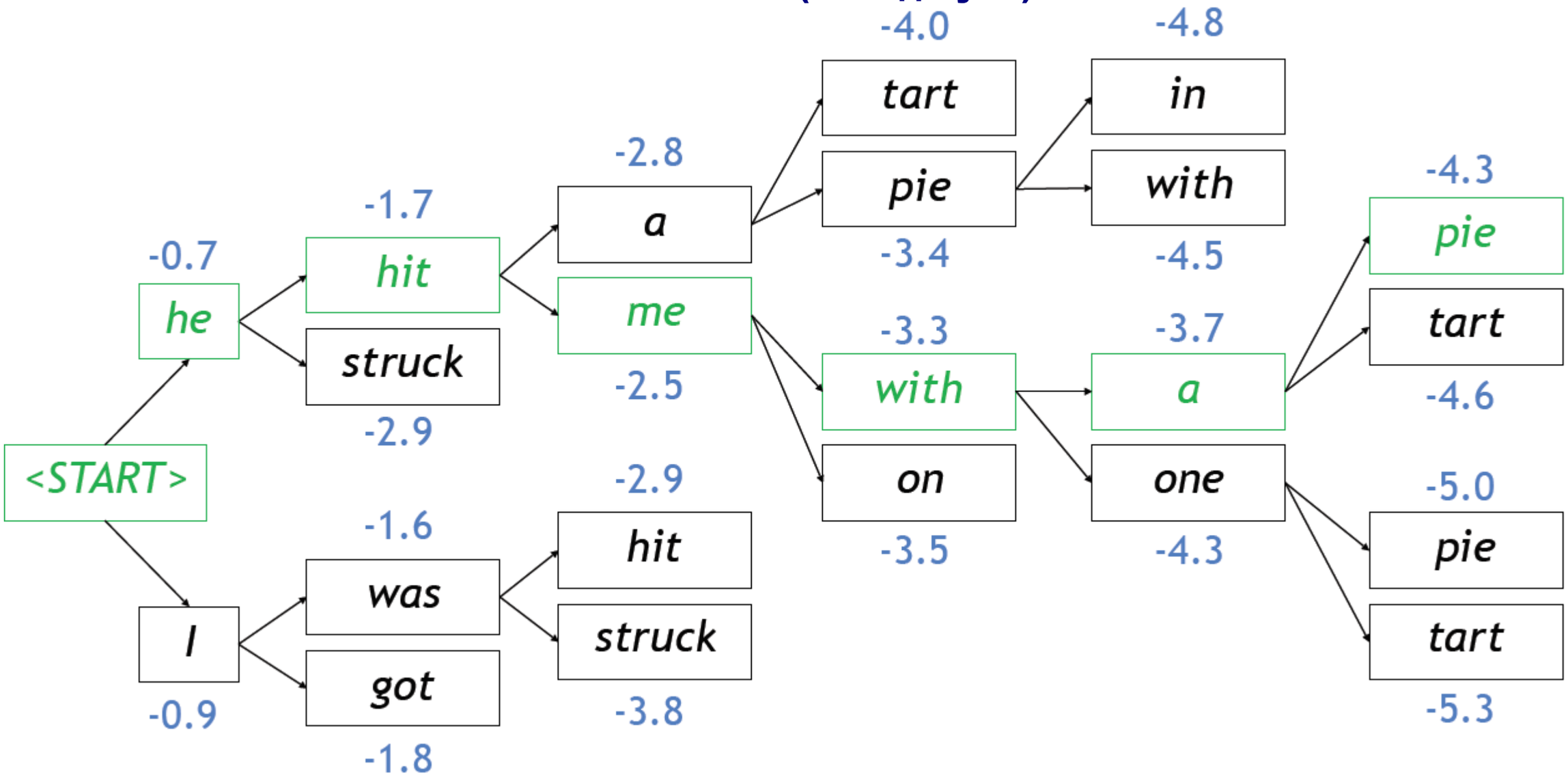
часто продолжают до какой-то максимальной длины *T*

или пока не будет *n* законченных вариантов

https://www.katnoria.com/nlg-decoders/

Sam Wiseman, Alexander M. Rush «Sequence-to-Sequence Learning as Beam-Search Optimization» https://arxiv.org/abs/1606.02960

Beam Search (метод луча)



Пример для k=2 http://web.stanford.edu/class/cs224n/

Выбор параметра k в методе луча

Beam size	Model response
1	*I love to eat healthy and eat healthy*
2	*That is a good thing to have*
3	*I am a nurse so I do not eat raw food*
4	*I am a nurse so I am a nurse*
5	*Do you have any hobbies?*
6	*What do you do for a living?*
7	*What do you do for a living?*
8	*What do you do for a living?*

Маленькие значения – релевантно, но часто неязыковая фраза,
большие – грамматически верная фраза, но слишком общая

дальше будут стратегии сэмплирования

https://cs224n.stanford.edu/

Ещё стратегии сэмплирования – Stochastic Decoding

1. Сэмплирование с температурой

$$p(x_t = x \mid x_1, \ldots, x_{t-1}) = \text{softmax}(u_1 / \tau, \ldots, u_l / \tau)$$

2. Топ-k (Top-k Sampling)

$$p'(x_t = x \mid x_1, \ldots, x_{t-1}) = \begin{cases} p(x_t = x \mid x_1, \ldots, x_{t-1}) / \ p', & x \in \text{top}(k), \\ 0, & \text{иначе.} \end{cases}$$

3. Nucleus (Top-p) Sampling

вместо используем $\text{top}(k)$

$$\sum_{x \in \text{sort}} p(x_t = x \mid x_1, \ldots, x_{t-1}) \geq p$$

Оценка языковых моделей

Перплексия (perplexity)

должна быть как можно меньше

$$p(x_1,\ldots,x_T)^{-1/T} = \prod_{t=1}^{T}\left(\frac{1}{p(x_t \mid x_1,\ldots,x_{t-1})}\right)^{1/T}$$

степень для нормировки

в методе луча используют такую же нормировку

Применение LM

Кроме «чистой» генерации текстов…

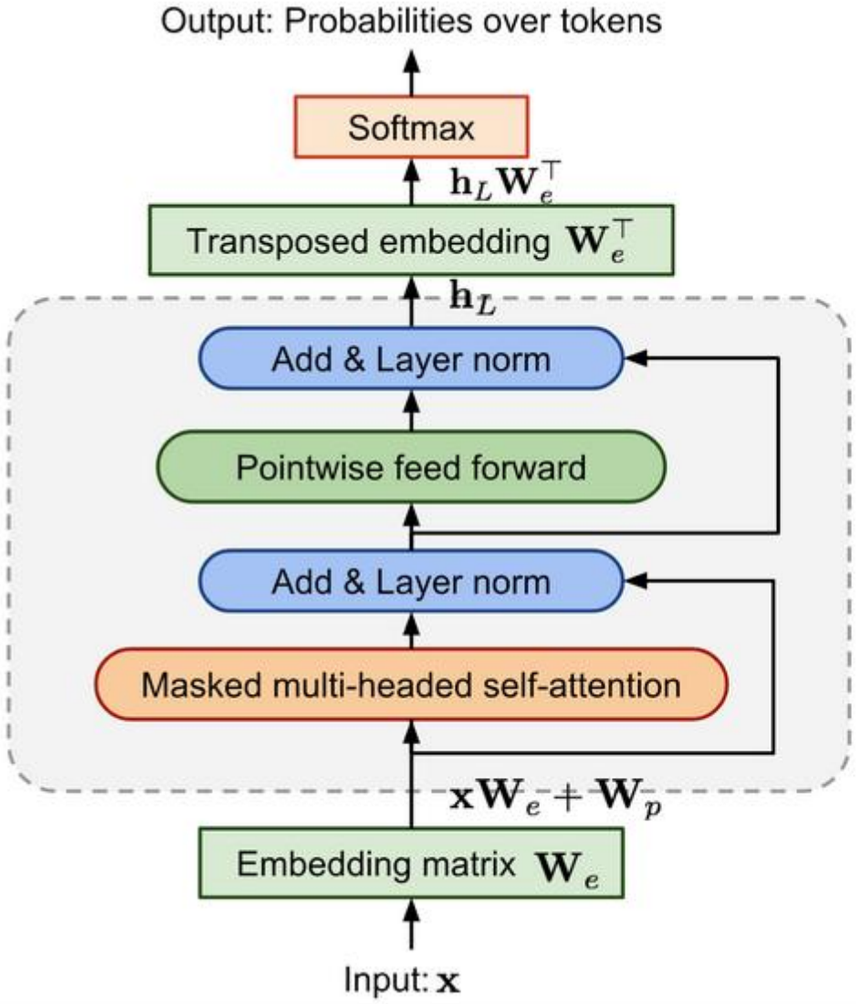
Машинный перевод: выбор подходящего варианта

Распознавание речи: выбор подходящего варианта

Проверка текста: нахождение ошибок

Набор текста: подсказка вариантов

GPT (OpenAI) – архитектура



Transformer Block
Repeat x L=12

$$\mathbf{h}_\ell = \text{transformer\_block}(\mathbf{h}_{\ell-1})$$
$$\ell = 1, \ldots, L$$

https://lilianweng.github.io/lil-log/2019/01/31/generalized-language-models.html

GPT (OpenAI) – архитектура

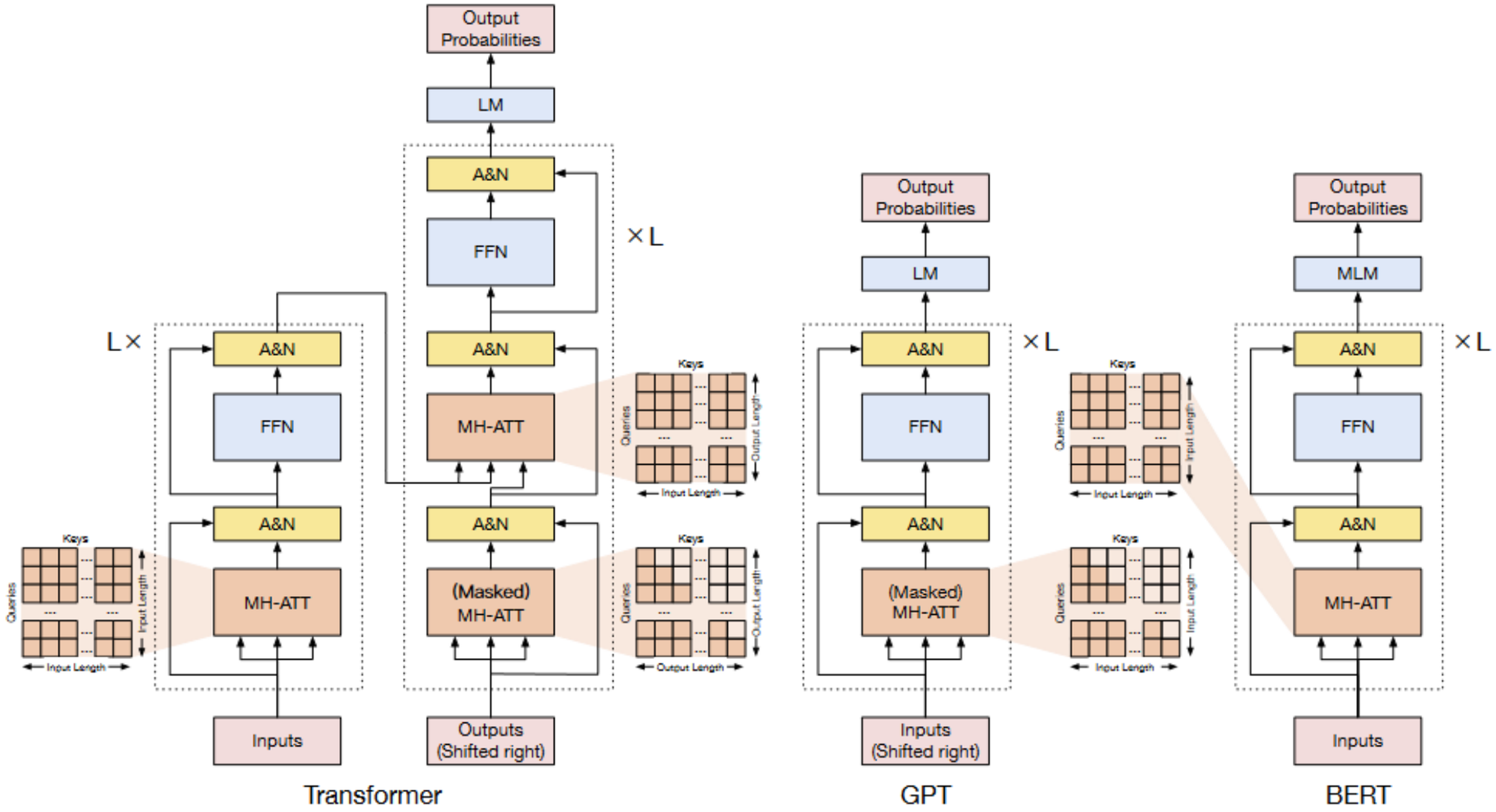


Figure 5: The architecture of Transformer, GPT, and BERT.

https://arxiv.org/pdf/2106.07139.pdf

GPT (OpenAI) – моделирование языка

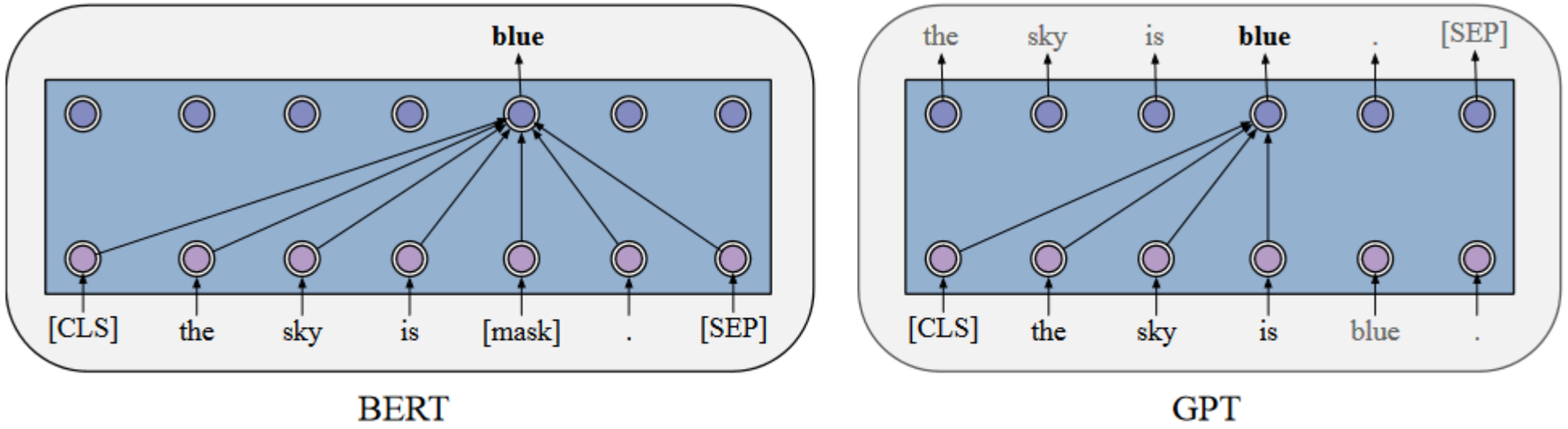


Figure 7: The difference between GPT and BERT in their self-attention mechanisms and pre-training objectives.

$$\mathcal{L}(\mathcal{X}) = \sum_{i=1}^{n+1} \log P(x_i | x_{i-k}, ..., x_{i-1}; \Theta)$$

GPT (OpenAI) – настройка на конкретную задачу

Пример – классификация

Пропускаем через трансформер (декодировщик)
используем скрытое состояние только последнего токена

$$P(y \mid x_1, \ldots, x_n) = \text{softmax}(\mathbf{h}_L^{(n)} \mathbf{W}_y)$$

ошибка = сумма ошибки LM и классификации:

$$\mathcal{L}_{\text{cls}} = \sum_{(\mathbf{x},y)\in\mathcal{D}} \log P(y \mid x_1, \ldots, x_n) = \sum_{(\mathbf{x},y)\in\mathcal{D}} \log \text{softmax}(\mathbf{h}_L^{(n)}(\mathbf{x}) \mathbf{W}_y)$$

$$\mathcal{L}_{\text{LM}} = -\sum_i \log p(x_i \mid x_{i-k}, \ldots, x_{i-1})$$

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{LM}}$$

GPT (OpenAI) – любая задача не требует изменения архитектуры

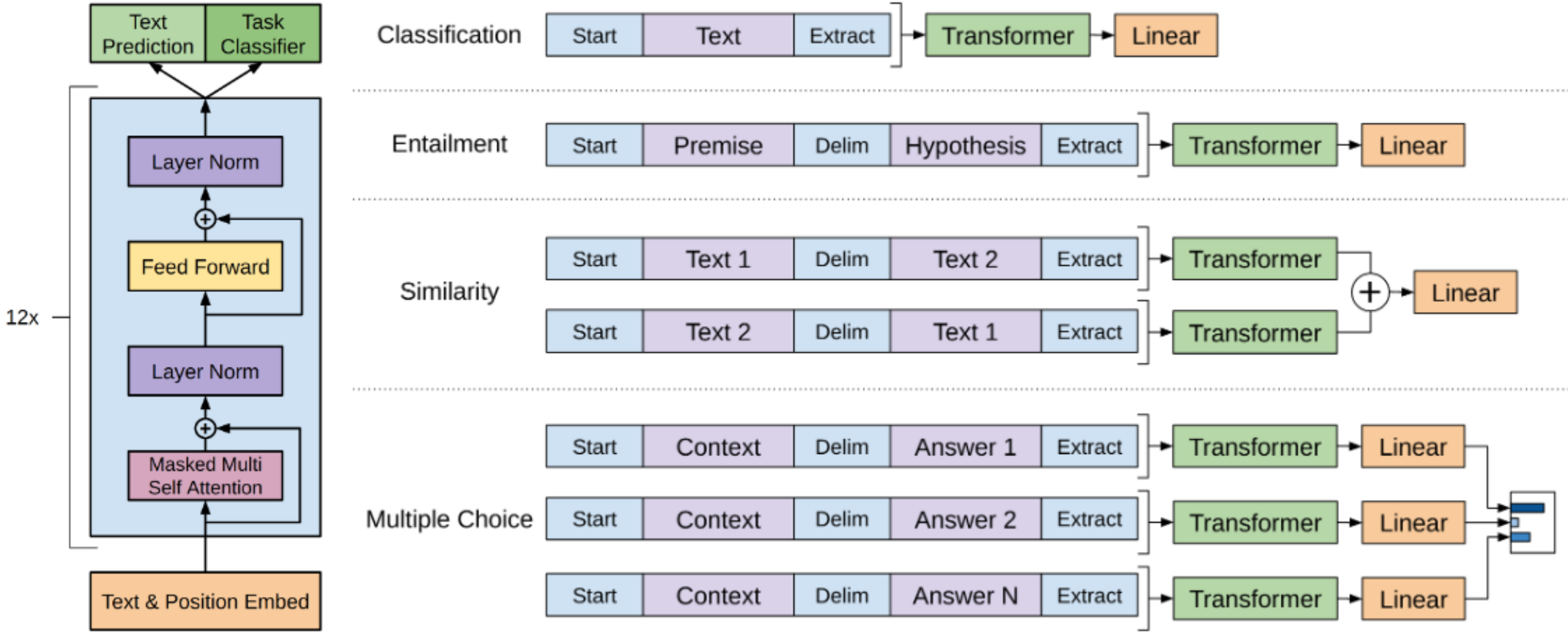
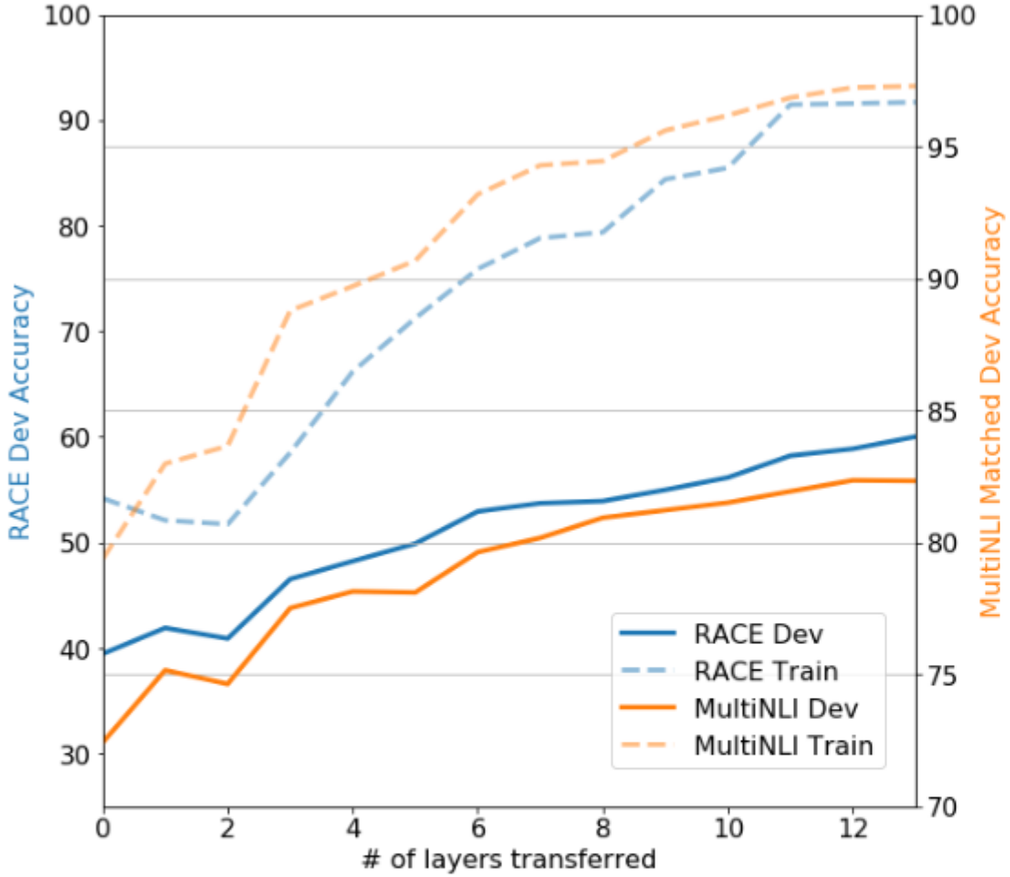


Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

если в задаче несколько входных предложений – они разделяются спецтокеном

GPT (OpenAI)



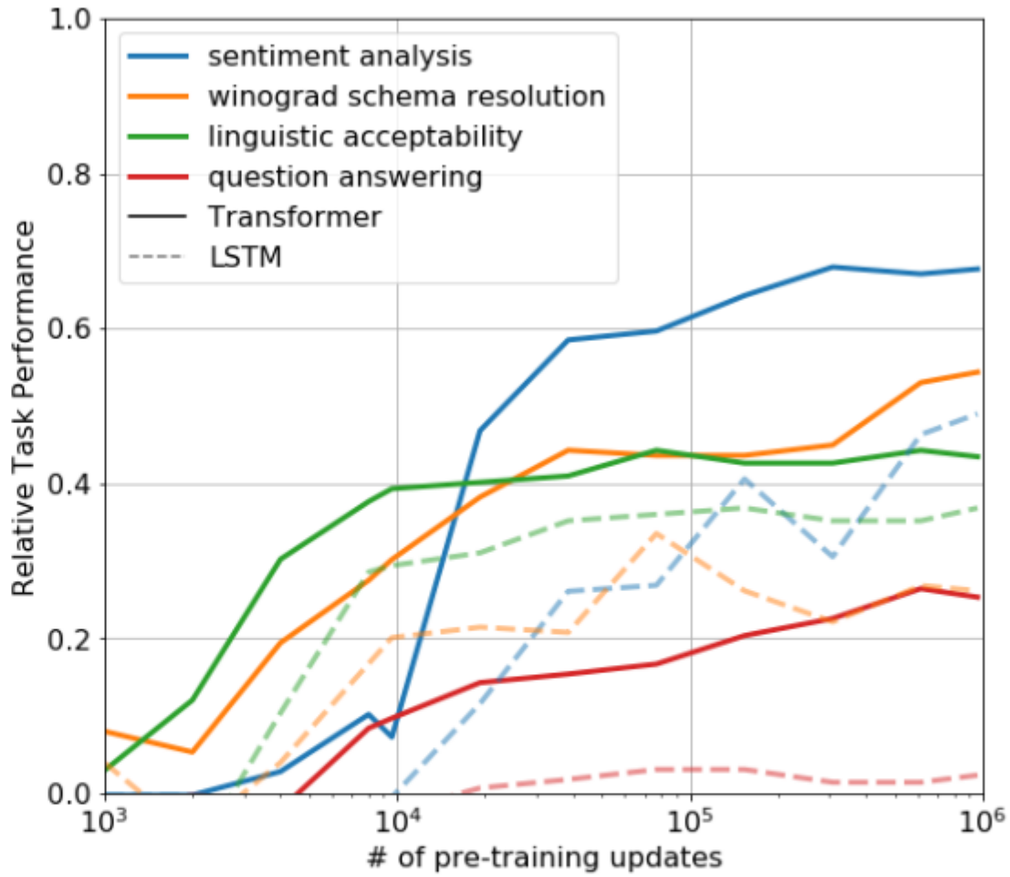


Figure 2: (**left**) Effect of transferring increasing number of layers from the pre-trained language model on RACE and MultiNLI. (**right**) Plot showing the evolution of zero-shot performance on different tasks as a function of LM pre-training updates. Performance per task is normalized between a random guess baseline and the current state-of-the-art with a single model.

GPT2 (2019, OpenAI)

1.5 млрд параметров (10×GPT)

обучение – новый датасет «WebText»

BPE

MQAN

(new←GPT-1) Layer normalization → вход каждого под-блока / после self-attention-блока

(new←GPT-1) другая инициализация

vocabulary = 50 257 (стал больше)

context size = 1024 (больше)

batchsize = 512 (больше)

при инициализации меньше вес Residual layers

SOTA 7 из 8 задач (zero-shot setting – без подстраивания под задачи)

https://blog.openai.com/better-language-models/

https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

GPT2 – размеры

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

Table 2. Architecture hyperparameters for the 4 model sizes.

d – размерность пространства представления токенов

48, а не 12 слоёв

GPT2 – датасет «WebText»

~ 1 млн web-страниц / 45 млн ссылок / 8 млн. документов 40Гб

ссылки с Reddit ≥3 кармы (т.е. отбором человека)

удалили Wiki ! (чтобы тестировать на других датасетах)

экстракторы текстов:

Dragnet (Peters & Lecocq, 2013) and Newspaper (https://github.com/codelucas/newspaper)

Есть гипотеза, что Wiki плоха для обучения…

GPT2 – Предобработка

lower-casing

tokenization

out-of-vocabulary tokens

Unicode → UTF-8

BPE (Byte Pair Encoding)

потом кодируем частые слова и буквы (из которых состоят редкие слова)

GPT2 – задачи

- question answering
- machine translation
- reading comprehension
- summarization

Решение всех задач на основе – Language modeling

$$p(x) = \prod_{i=1}^{n} p(s_n | s_1, \ldots, s_{n-1})$$

p(output |input, task) – с помощью трансформера

предсказывает следующее слово в предложении
тут нет маскирования как в BERT

GPT2 – MQAN (Multitask Learning as Question Answering)

«переведи …»

«ответь на вопрос …»

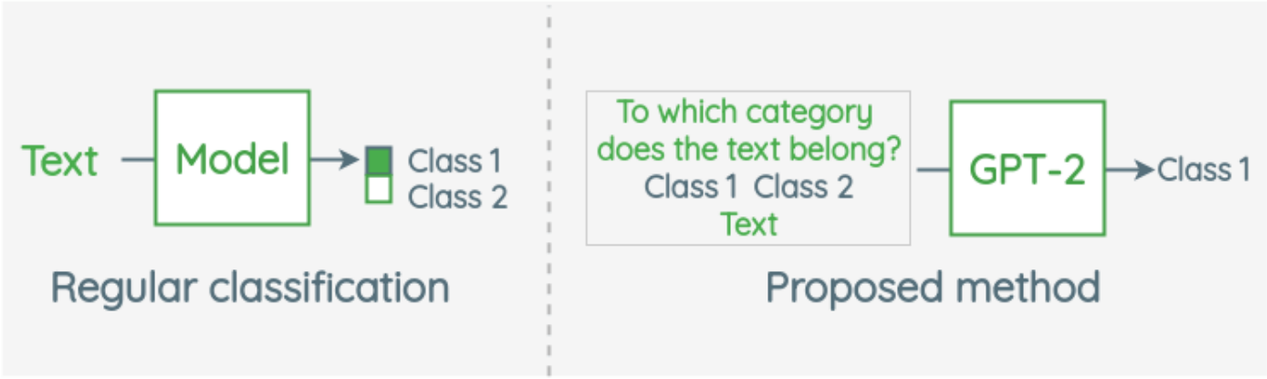
«TL;DR:»

– надо задавать правильные вопросы;)

без дообучения с учителем на специализированных данных!

zero-shot task transfer – особая концепция в NLP

GPT2 – MQAN



Question	Text	Answer
To which category does the text belong?:" Positive Sentiment ", " Negative Sentiment "	the film is one of the year's best	Positive Sentiment

GPT2 – MQAN

Facts	Q: What is human life expectancy in the United States? A: Human life expectancy in the United States is 78 years. Q: Which is heavier, a mouse or an elephant? A: An elephant is heavier than a mouse. Q: Who was president of the United States in 1801? A: Thomas Jefferson was president of the United States in 1801. Q: Who was president of the United States in 1600? A: Queen Elizabeth I was president of the United States in 1600.
Common Sense	Q: What is your favorite animal? A: My favorite animal is a dog.
Logic	Q: When counting, what number comes before 100? A: Ninety-nine comes before 100. Q: Reverse the following array: [1, 3, 5, 6, 10, 4, 2, 77] A: [10, 6, 4, 2, 77, 3, 5, 1]

https://lacker.io/ai/2020/07/06/giving-gpt-3-a-turing-test.html

Результат 2019 – GPT2

Language Models are Unsupervised Multitask Learners

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gong et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

Alec Radford et. al. «Language Models are Unsupervised Multitask Learners»
https://www.ceid.upatras.gr/webpages/faculty/zaro/teaching/alg-ds/PRESENTATIONS/PAPERS/2019-Radford-et-al_Language-Models-Are-Unsupervised-Multitask-%20Learners.pdf

Question	Generated Answer	Correct	Probability
Who wrote the book the origin of species?	Charles Darwin	✓	83.4%
Who is the founder of the ubuntu project?	Mark Shuttleworth	✓	82.0%
Who is the quarterback for the green bay packers?	Aaron Rodgers	✓	81.1%
Panda is a national animal of which country?	China	✓	76.8%
Who came up with the theory of relativity?	Albert Einstein	✓	76.4%
When was the first star wars film released?	1977	✓	71.4%
What is the most common blood type in sweden?	A	✗	70.6%
Who is regarded as the founder of psychoanalysis?	Sigmund Freud	✓	69.3%
Who took the first steps on the moon in 1969?	Neil Armstrong	✓	66.8%
Who is the largest supermarket chain in the uk?	Tesco	✓	65.3%
What is the meaning of shalom in english?	peace	✓	64.0%
Who was the author of the art of war?	Sun Tzu	✓	59.6%
Largest state in the us by land mass?	California	✗	59.2%
Green algae is an example of which type of reproduction?	parthenogenesis	✗	56.5%
Vikram samvat calender is official in which country?	India	✓	55.6%
Who is mostly responsible for writing the declaration of independence?	Thomas Jefferson	✓	53.3%
What us state forms the western boundary of montana?	Montana	✗	52.3%
Who plays ser davos in game of thrones?	Peter Dinklage	✗	52.1%
Who appoints the chair of the federal reserve system?	Janet Yellen	✗	51.5%
State the process that divides one nucleus into two genetically identical nuclei?	mitosis	✓	50.7%
Who won the most mvp awards in the nba?	Michael Jordan	✗	50.2%
What river is associated with the city of rome?	the Tiber	✓	48.6%
Who is the first president to be impeached?	Andrew Johnson	✓	48.3%
Who is the head of the department of homeland security 2017?	John Kelly	✓	47.0%
What is the name given to the common currency to the european union?	Euro	✓	46.8%
What was the emperor name in star wars?	Palpatine	✓	46.5%
Do you have to have a gun permit to shoot at a range?	No	✓	46.4%
Who proposed evolution in 1859 as the basis of biological development?	Charles Darwin	✓	45.7%
Nuclear power plant that blew up in russia?	Chernobyl	✓	45.7%
Who played john connor in the original terminator?	Arnold Schwarzenegger	✗	45.2%

Table 5. The 30 most confident answers generated by GPT-2 on the development set of Natural Questions sorted by their probability according to GPT-2. None of these questions appear in WebText according to the procedure described in Section 4.

GPT3 (2020, OpenAI)

Архитектура как в GPT2, но 96 слоёв, 96 головок

dim(word embeddings) = 12 888 (вместо 1600)

окно контекста = 2048 (вместо 1024)

175 млрд параметров (100×GPT2, 10×Turing NLG)

«alternating dense and locally banded sparse attention patterns»

как в Sparse-трансформере

Обучающий датасет – больше данных

Тестирование в режиме few-shot без fine-tuning

Не файнтьюнили GPT-3 под задачу!

Генерация совсем правдоподобных историй

on-the-fly tasks – на которых не обучалась (сложить два числа, запрос SQL и т.п.)

https://arxiv.org/abs/2005.14165

GPT3 – Обучающий датасет

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Table 2.2: Datasets used to train GPT-3. "Weight in training mix" refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.

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

GPT3 – zero/one/few-shot learning

The three settings we explore for in-context learning

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
```

One-shot

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

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  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
```

Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

```
sea otter => loutre de mer             ← example #1
        ↓
   gradient update
        ↓
peppermint => menthe poivrée           ← example #2
        ↓
   gradient update
        ↓
        ...
        ↓
plush giraffe => girafe peluche        ← example #N
        ↓
   gradient update
        ↓
cheese =>                              ← prompt
```

Figure 2.1: Zero-shot, one-shot and few-shot, contrasted with traditional fine-tuning. The panels above show four methods for performing a task with a language model – fine-tuning is the traditional method, whereas zero-, one-, and few-shot, which we study in this work, require the model to perform the task with only forward passes at test time. We typically present the model with a few dozen examples in the few shot setting. Exact phrasings for all task descriptions, examples and prompts can be found in Appendix G.

GPT3 (2020, OpenAI)

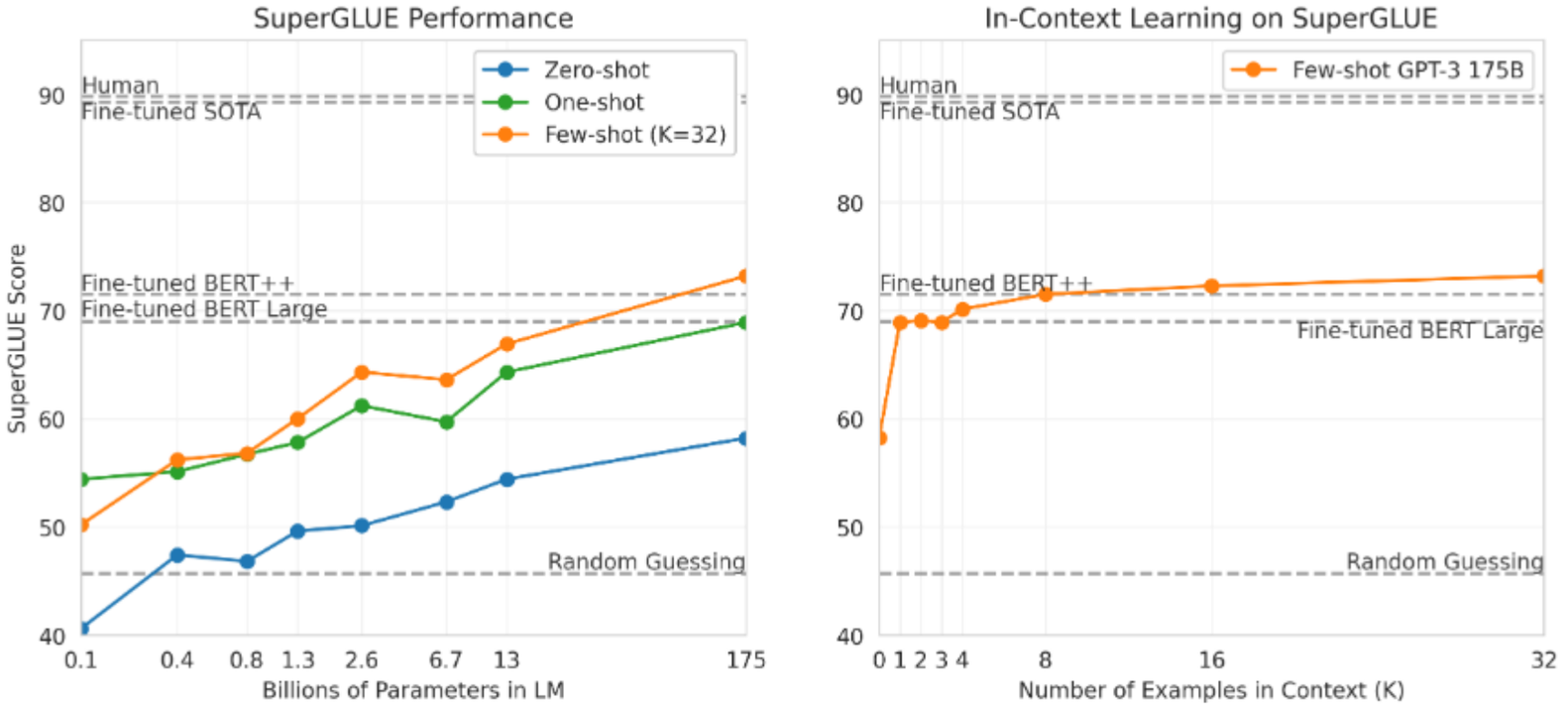


Figure 3.8: Performance on SuperGLUE increases with model size and number of examples in context. A value of $K = 32$ means that our model was shown 32 examples per task, for 256 examples total divided across the 8 tasks in SuperGLUE. We report GPT-3 values on the dev set, so our numbers are not directly comparable to the dotted reference lines (our test set results are in Table 3.8). The BERT-Large reference model was fine-tuned on the SuperGLUE training set (125K examples), whereas BERT++ was first fine-tuned on MultiNLI (392K examples) and SWAG (113K examples) before further fine-tuning on the SuperGLUE training set (for a total of 630K fine-tuning examples). We find the difference in performance between the BERT-Large and BERT++ to be roughly equivalent to the difference between GPT-3 with one example per context versus eight examples per context.

но SuperGLUE не самое хорошее задание для GPT-3

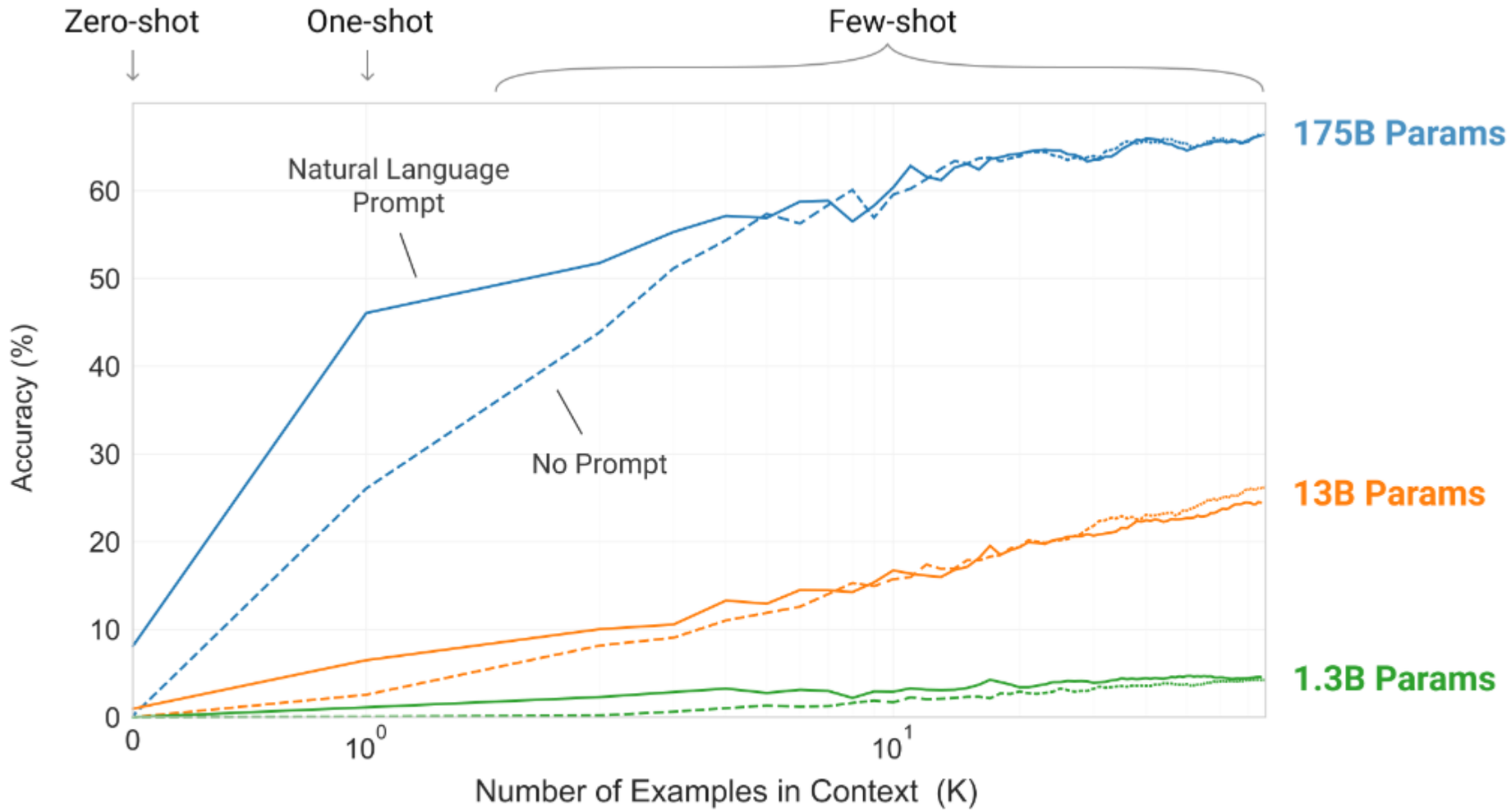


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

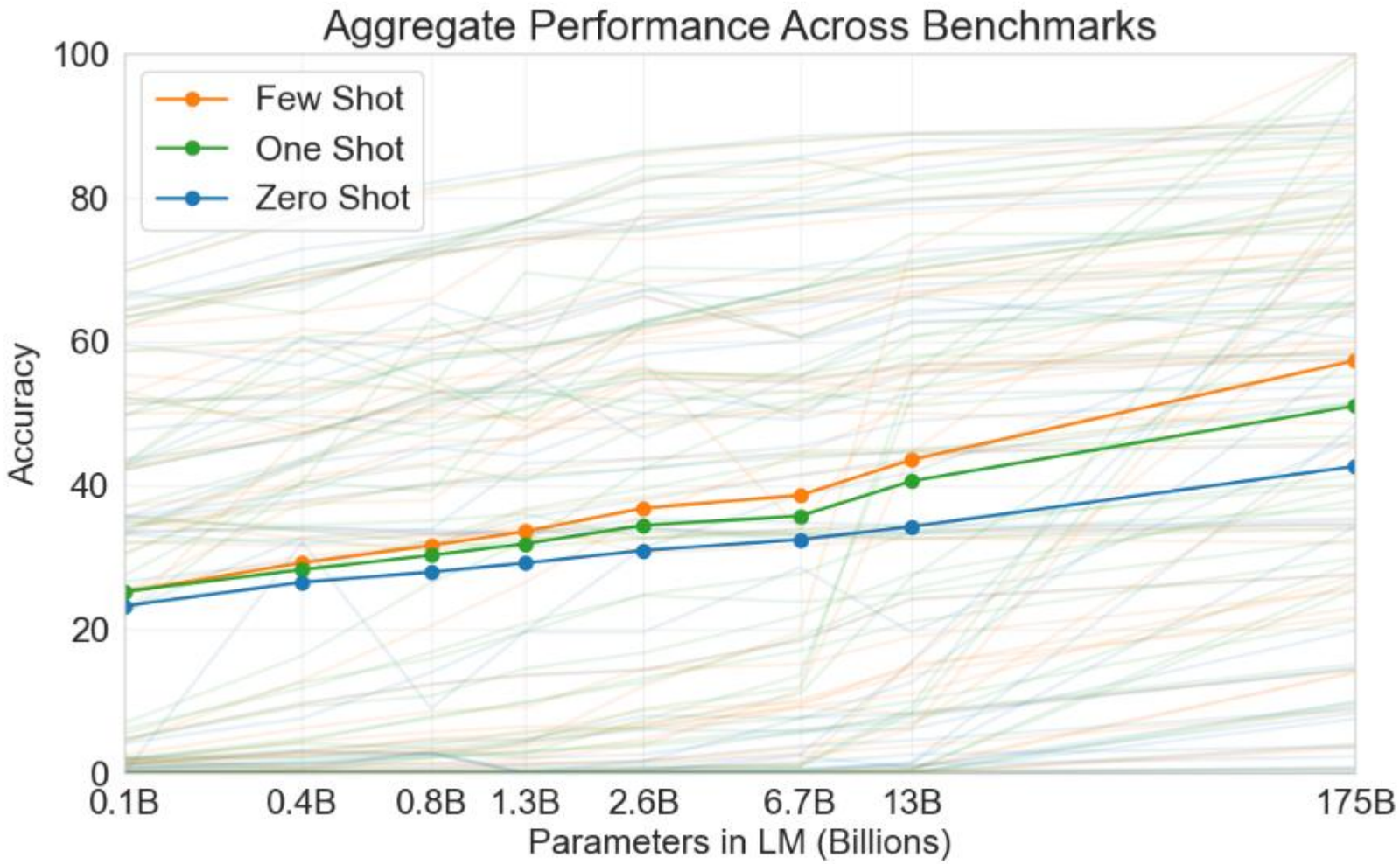
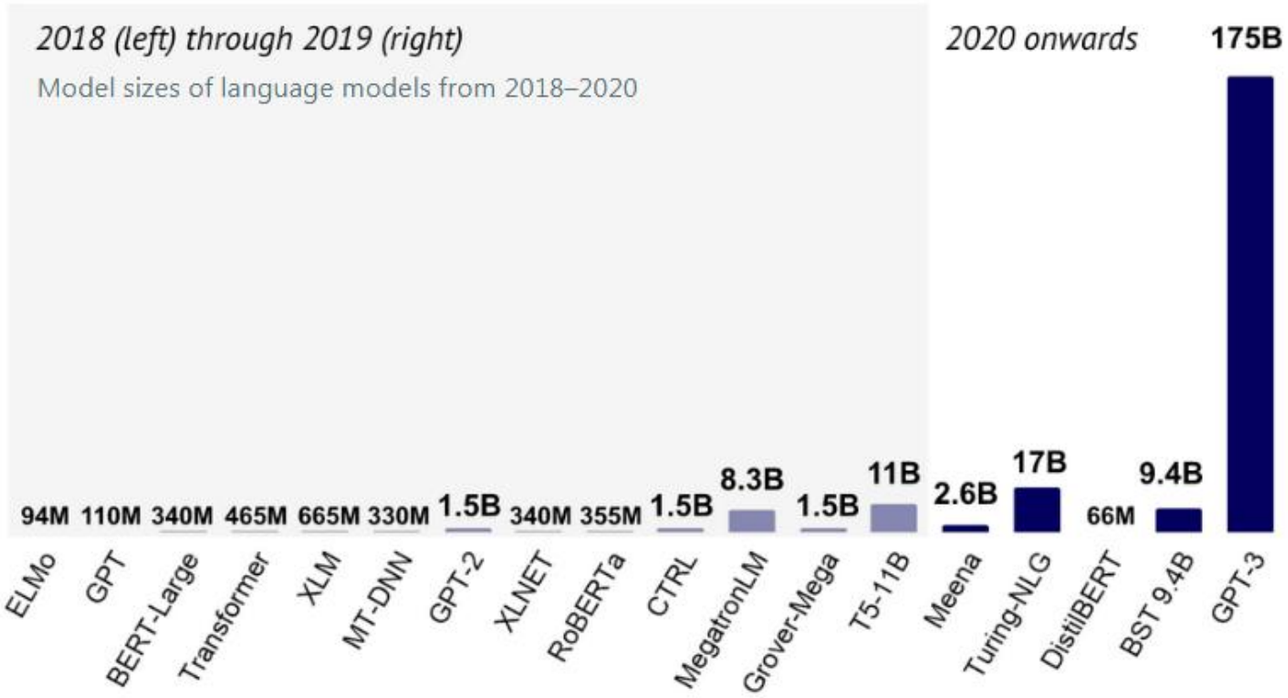
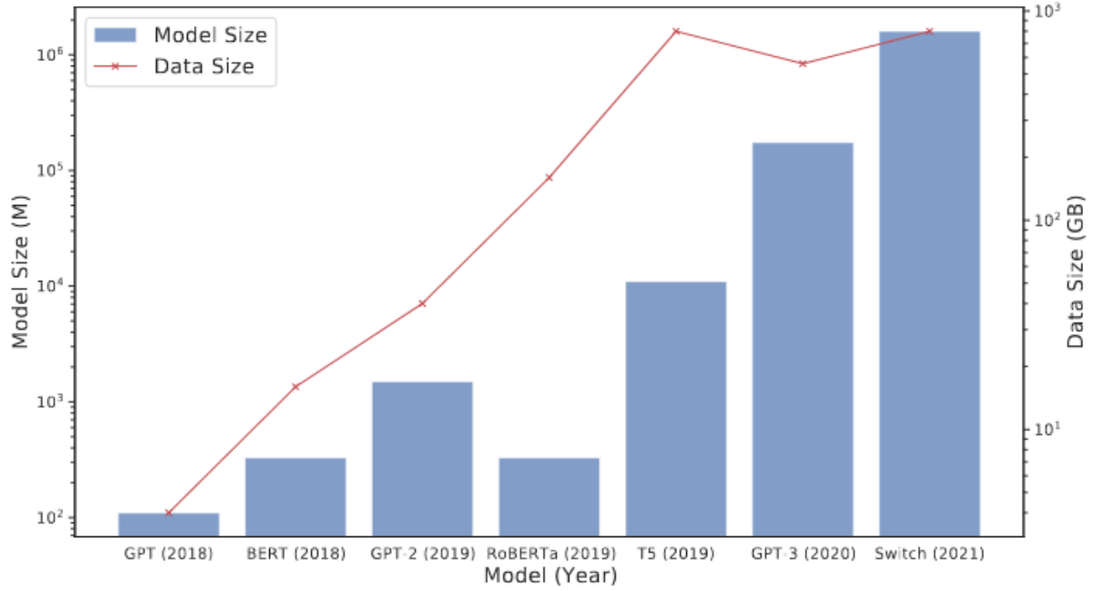


Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

GPT3 – размер модели





(b) The model size and data size applied by recent NLP PTMs. A base-10 log scale is used for the figure.

https://www.stateof.ai/

https://arxiv.org/pdf/2106.07139.pdf

Представление слов

- **как подавать текст в НС**

что такое токен и как его представлять

- **решение проблемы OOV слов**

(можно ещё помечать слова не из словаря <UNKNOWN>)

- **составление более адекватного словаря**

редкие слова разбивать на подслова

частые пары объединять

какой размер словаря должен быть?

«subword» → «sub» + «word»

«machine learning» → ML_word

https://medium.com/@makcedward/how-subword-helps-on-your-nlp-model-83dd1b836f46

https://mlexplained.com/2019/11/06/a-deep-dive-into-the-wonderful-world-of-preprocessing-in-nlp/

Представление слов

токенизация на подслова ← **сейчас это**

- byte-pair encoding (BPE)
- wordpiece
- unigram language model
- sentencepiece

посимвольный подход (представления слов из анализа символов)

- Посимвольная модель для представления слов: Compositional Character Model
- Посимвольные модели: Character-Aware NLM

гибридный подход (действуем на уровне слов, если надо – на уровне символов)

- Compositional Character Model
- Character-Aware NLM

Токенизация на подслова (Subword Tokenization)

Причины:

Неформальные слова

«Yeeees! Goooood!»

Транслитерация

«файнтьюнинг»

Динамический словарь

«айфонизация…»

Сложности с разделеним слов

安理会认可利比亚问题柏林峰会成果

ف ل ق ن ا ه

Lebensversicherungsgesellschaftsangestellter

«Abwasserbehandlungsanlange» (нем.) – станция очистки сточных вод

Byte Pair Encoding (BPE)

идея ~ Huffman encoding

возник в работе сжатия данных,
потом в машинном переводе
чтобы обучать на данных с одним словарём,
а работать на более широком разнообразии данных

На этапе препроцессинга данных.

Byte Pair Encoding (BPE)

Обучение BPE

- **Слово = последовательность токенов (пока символов) + специальный символ конца** (изначально использовались unicode-символы)
- **Словарь = все токены** (на нулевой итерации – символы)
- **Повторять пока не достигли ограничения на размер словаря**
  - **Назначаем новым токеном объединение двух существующих токенов, которое встречается чаще других пар в корпусе**

Применение BPE (возможны варианты)

идём по всем токенам по убыванию частоты – находим соответствующую последовательность символов в корпусе, заменяем на токен

Rico Sennrich et al. «Neural Machine Translation of Rare Words with Subword Units» https://arxiv.org/abs/1508.07909
ещё есть работа [Philip Gage, 1994] откуда, собственно, термин BPE

Byte Pair Encoding (BPE)

AABABCABBAABAC

AA – 2
AB – 4 AB = D
BA – 3
BC – 1
CA – 1
BB – 1
AC – 1

ADDCDBADAC

AD – 2 AD = E
DD – 1
DC – 1
CD – 1
DB – 1
DA – 1
AC – 1

EDCDBEAC

На практике

«l$_{</w>}$ like$_{</w>}$ ke$_{</w>}$» → «l$_{</w>}$», «li», «##ke$_{</w>}$», «ke$_{</w>}$»

- различают токен изолированный и токен внутри слова
- есть специальный символ конца слова

Algorithm 1 Learn BPE operations

```
import re, collections

def get_stats(vocab):
  pairs = collections.defaultdict(int)
  for word, freq in vocab.items():
    symbols = word.split()
    for i in range(len(symbols)-1):
      pairs[symbols[i],symbols[i+1]] += freq
  return pairs

def merge_vocab(pair, v_in):
  v_out = {}
  bigram = re.escape(' '.join(pair))
  p = re.compile(r'(?<!\S)' + bigram + r'(?!\S)')
  for word in v_in:
    w_out = p.sub(''.join(pair), word)
    v_out[w_out] = v_in[word]
  return v_out

vocab = {'l o w </w>' : 5, 'l o w e r </w>' : 2,
         'n e w e s t </w>':6, 'w i d e s t </w>':3}
num_merges = 10
for i in range(num_merges):
  pairs = get_stats(vocab)
  best = max(pairs, key=pairs.get)
  vocab = merge_vocab(best, vocab)
  print(best)
```

r · → r·
l o → lo
lo w → low
e r· → er·

Figure 1: BPE merge operations learned from dictionary {‘low’, ‘lowest’, ‘newer’, ‘wider’}.

BPE в GPT2

- вход – последовательность байтов (а не юникод-символов)

- не сливают символы разного типа (e.g. буквы и знаки пунктуации)

«dog», «dog!», «dog?»

Byte Pair Encoding (BPE)

name	segmentation	shortlist	vocabulary source	vocabulary target	BLEU single	BLEU ens-8	CHRF3 single	CHRF3 ens-8	unigram F_1 (%) all	unigram F_1 (%) rare	unigram F_1 (%) OOV
syntax-based (Sennrich and Haddow, 2015)					24.4	-	55.3	-	59.1	46.0	37.7
WUnk	-	-	300 000	500 000	20.6	22.8	47.2	48.9	56.7	20.4	0.0
WDict	-	-	300 000	500 000	22.0	24.2	50.5	52.4	58.1	36.8	**36.8**
C2-50k	char-bigram	50 000	60 000	60 000	**22.8**	**25.3**	51.9	53.5	58.4	40.5	30.9
BPE-60k	BPE	-	60 000	60 000	21.5	24.5	**52.0**	53.9	58.4	40.9	29.3
BPE-J90k	BPE (joint)	-	90 000	90 000	**22.8**	24.7	51.7	**54.1**	**58.5**	**41.8**	33.6

Table 2: English→German translation performance (BLEU, CHRF3 and unigram F_1) on newstest2015. Ens-8: ensemble of 8 models. Best NMT system in bold. Unigram F_1 (with ensembles) is computed for all words ($n = 44085$), rare words (not among top 50 000 in training set; $n = 2900$), and OOVs (not in training set; $n = 1168$).

name	segmentation	shortlist	vocabulary source	vocabulary target	BLEU single	BLEU ens-8	CHRF3 single	CHRF3 ens-8	unigram F_1 (%) all	unigram F_1 (%) rare	unigram F_1 (%) OOV
phrase-based (Haddow et al., 2015)					24.3	-	53.8	-	56.0	31.3	16.5
WUnk	-	-	300 000	500 000	18.8	22.4	46.5	49.9	54.2	25.2	0.0
WDict	-	-	300 000	500 000	19.1	22.8	47.5	51.0	54.8	26.5	6.6
C2-50k	char-bigram	50 000	60 000	60 000	**20.9**	**24.1**	49.0	51.6	55.2	27.8	17.4
BPE-60k	BPE	-	60 000	60 000	20.5	23.6	**49.8**	52.7	55.3	29.7	15.6
BPE-J90k	BPE (joint)	-	90 000	100 000	20.4	**24.1**	49.7	**53.0**	**55.8**	**29.7**	**18.3**

Table 3: English→Russian translation performance (BLEU, CHRF3 and unigram F_1) on newstest2015. Ens-8: ensemble of 8 models. Best NMT system in bold. Unigram F_1 (with ensembles) is computed for all words ($n = 55654$), rare words (not among top 50 000 in training set; $n = 5442$), and OOVs (not in training set; $n = 851$).

из оригинальной статьи (CHRF3 – a character n-gram F_3-score – хорош для оценки перевода [Popović, 2015])

WordPiece

WordPiece – это BPE, но при объединении максимизируем правдоподобие, а не частоту

В BERT был реализован WordPiece, но в RoBERTa показали, что использование BPE особо ничего не меняет

Schuster, Nakajima «Japanese and Korea voice search», 2012
https://static.googleusercontent.com/media/research.google.com/ja//pubs/archive/37842.pdf

WordPiece (in BERT)

1. Подготавливаем большой корпус
2. Определяем желаемы размер словаря подслов (sudwords)
3. Представляем слово = последовательность букв
4. Строим языковую модель LM
5. Новое слово получаем объединяя 2 существующих, максимизируя правдоподобие
6. Если не достигли ограничения на размер словаря или порога для правдоподобия, повторяем п. 5

WordPiece (in BERT)

If my understanding is correct, this means that aside from just the bigram frequency, the frequency of the original symbols that constitute the bigram are also taken into account. The log likelihood of a sentence in a unigram language model (assuming independence between the words in a sentence) is simply the sum of the log frequencies of its constituent symbols. This means merging two symbols will increase the total log likelihood by the log likelihood of the *merged* symbol and decrease it by the log likelihood of the two *original* symbols. Assuming we merge symbols x and y, the increase in the log likelihood is

$$\log p(x,y) - \log p(x) - \log p(y) = \log \frac{\log(p(x))}{\log(p(x))\log(p(y))}$$

Извлечение обучающих данных (на примере GPT-2)

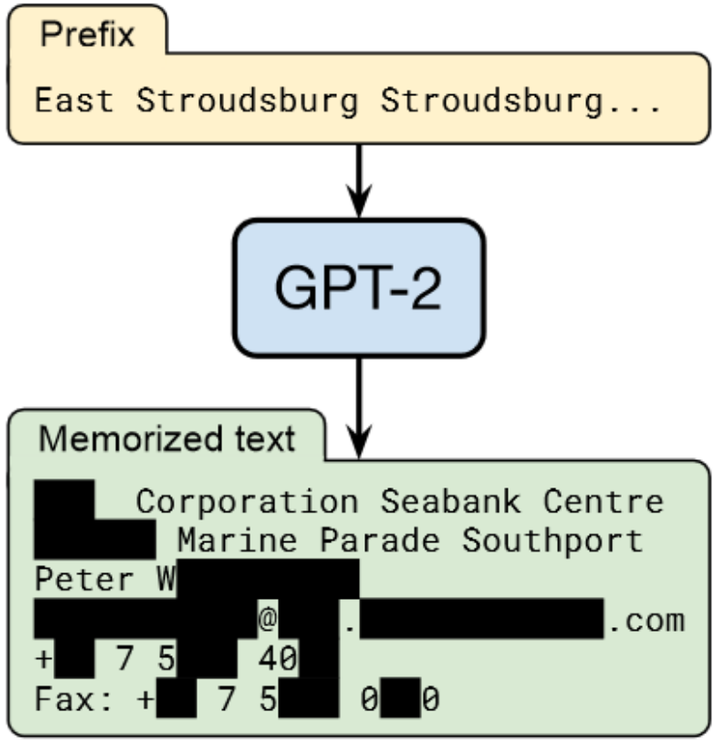


Figure 1: **Our extraction attack.** Given query access to a neural network language model, we extract an individual person's name, email address, phone number, fax number, and physical address. The example in this figure shows information that is all accurate so we redact it to protect privacy.

Nicholas Carlini et al. «Extracting Training Data from Large Language Models» // https://arxiv.org/pdf/2012.07805.pdf

«Атаки на сеть» – по извлечению персональной информации

Сети обучают на больших корпусах, в которых есть и персональные данные:
names, phone numbers, and email addresses
Например, GMail's auto-complete model – обучалось на частной переписке!

Можно ли их выудить из обученной модели?

Ошибочно считалость, что SOTA-модели не переобучены $\Rightarrow$ нет опасности
у GPT-2 ошибка на обучении на 10% меньше, чем ошибка на тесте

«membership inference attack» – есть ли такой пример в обучении [Reza Shokri, 2017]
«model inversion attacks» – извлечение конкретных примеров

«differentially-private training» – хороший способ защиты, но сейчас не о нём…

«Атаки на сеть» – по извлечению персональной информации

Definition 1 (Model Knowledge Extraction) *A string s is* extractable[4] *from an LM f_θ if there exists a prefix c such that:*

$$s \leftarrow \underset{s' :\, |s'|=N}{\arg\max}\, f_\theta(s' \mid c)$$

Definition 2 (k-Eidetic Memorization) *A string s is k-eidetic memorized (for $k \geq 1$) by an LM f_θ if s is extractable from f_θ and s appears in at most k examples in the training data X: $|\{x \in X : s \subseteq x\}| \leq k$.*

«Атаки на сеть» – по извлечению персональной информации

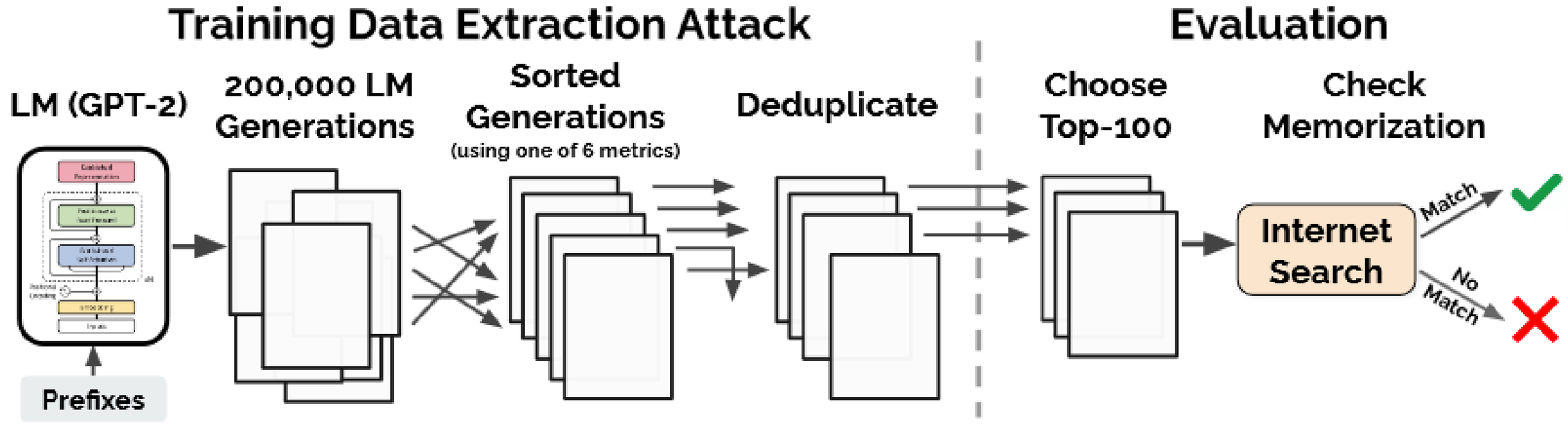


Figure 2: **Workflow of our extraction attack and evaluation. Attack.** We begin by generating many samples from GPT-2 when the model is conditioned on (potentially empty) prefixes. We then sort each generation according to one of six metrics and remove the duplicates. This gives us a set of potentially memorized training examples. **Evaluation.** We manually inspect 100 of the top-1000 generations for each metric. We mark each generation as either memorized or not-memorized by manually searching online, and we confirm these findings by working with OpenAI to query the original training data.

«Атаки на сеть» – по извлечению персональной информации

подаём префикс (генерируем 256 токенов)

модель генерирует 200 000 примеров

используем разные стратегии сэмплирования

1) top-n (изначальная)

2) Decaying Temperature – температура уменьшается при генерации

$$\mathrm{softmax}(z/t)$$

3) Conditioning on Internet Text (+ top-n) – начать с естественного контекста (Common Crawl)

предсказываем, где могло быть запоминание

1) смотрим на перплексию

2-3) смотрим на перплексию GPT-2 small / GPT-2 medium

(если у более примитивной модели перплексия меньше, то у нас запоминание)

4) как сильно сжимает текст zlib (+ тут отлавливание повторений)

5) Comparing to Lowercased Text (сравнение с перплексией этого же текста в нижнем регистре)

6) Perplexity on a Sliding Window (50 токенов)

«Атаки на сеть» – по извлечению персональной информации

число конфигураций эксперимента

3 (способов сэмплирования) × 6 (оценок дословности)

выбираем по 100 примеров из 1000

всего 1800 примеров

из них нашли 604 запоминания

параллельно устраняем fuzzy-дубликаты

интернет поиск + ассесоры для нахождения, есть где-то такая фраза дословно

ищем дословные повторы в обучении

(сравнение по 3-граммам)

Category	Count
US and international news	109
Log files and error reports	79
License, terms of use, copyright notices	54
Lists of named items (games, countries, etc.)	54
Forum or Wiki entry	53
Valid URLs	50
Named individuals (non-news samples only)	46
Promotional content (products, subscriptions, etc.)	45
High entropy (UUIDs, base64 data)	35
Contact info (address, email, phone, twitter, etc.)	32
Code	31
Configuration files	30
Religious texts	25
Pseudonyms	15
Donald Trump tweets and quotes	12
Web forms (menu items, instructions, etc.)	11
Tech news	11
Lists of numbers (dates, sequences, etc.)	10

Table 1: Manual categorization of the 604 memorized training examples that we extract from GPT-2, along with a description of each category. Some samples correspond to multiple categories (e.g., a URL may contain base-64 data). Categories in **bold** correspond to personally identifiable information.

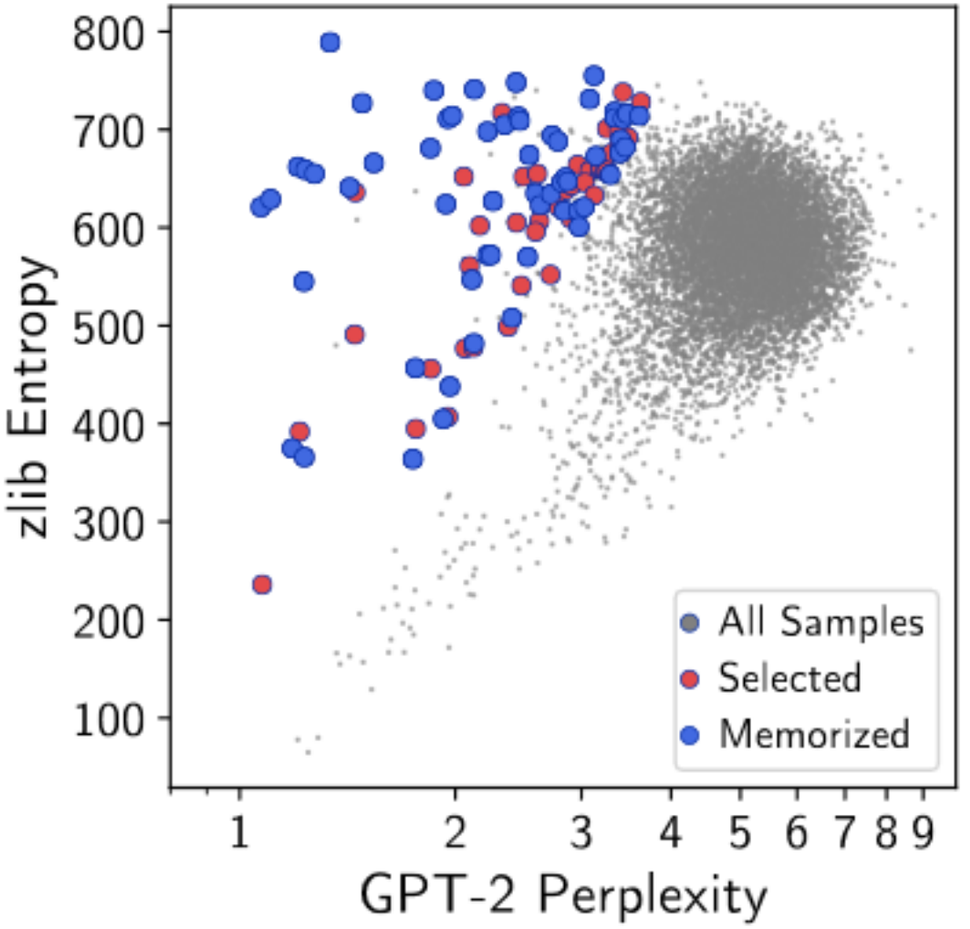


Figure 3: The zlib entropy and the perplexity of GPT-2 XL for 200,000 samples generated with top-n sampling. In red, we show the 100 samples that were selected for manual inspection. In blue, we show the 59 samples that were confirmed as memorized text. Additional plots for other text generation and detection strategies are in Figure 4.

Inference Strategy	Text Generation Strategy		
	Top-n	Temperature	Internet
Perplexity	9	3	39
Small	41	42	58
Medium	38	33	45
zlib	59	46	67
Window	33	28	58
Lowercase	53	22	60
Total Unique	191	140	273

Table 2: The number of memorized examples (out of 100 candidates) that we identify using each of the three text generation strategies and six membership inference techniques. Some samples are found by multiple strategies; we identify 604 unique memorized examples in total.

Memorized String	Sequence Length	Occurrences in Data	
		Docs	Total
`Y2...`[redacted]`...y5`	87	1	10
`7C...`[redacted]`...18`	40	1	22
`XM...`[redacted]`...WA`	54	1	36
`ab...`[redacted]`...2c`	64	1	49
`ff...`[redacted]`...af`	32	1	64
`C7...`[redacted]`...ow`	43	1	83
`0x...`[redacted]`...C0`	10	1	96
`76...`[redacted]`...84`	17	1	122
`a7...`[redacted]`...4b`	40	1	311

Table 3: **Examples of $k = 1$ eidetic memorized, high-entropy content that we extract** from the training data. Each is contained in *just one* document. In the best case, we extract a 87-characters-long sequence that is contained in the training dataset just 10 times in total, all in the same document.

total – длина последовательности, в которой он нашёлся

UUID: 1e4bd2a8-e8c8-4a62-adcd-40a936480059

Google search – 3 documents containing this UUID

GPT-2 training – 1 document

Некорректные запоминания

Пример, когда два разных запоминания склеиваются:

**GPT-2 generates a news article about the (real) murder of a woman in 2013,
but then attributes the murder to one of the victims of a nightclub shooting in Orlando in 2016.**

Instagram biography of a pornography producer + describe an American fashion model as a pornography actress

Есть примеры данных, которые уже удалили из Интернета

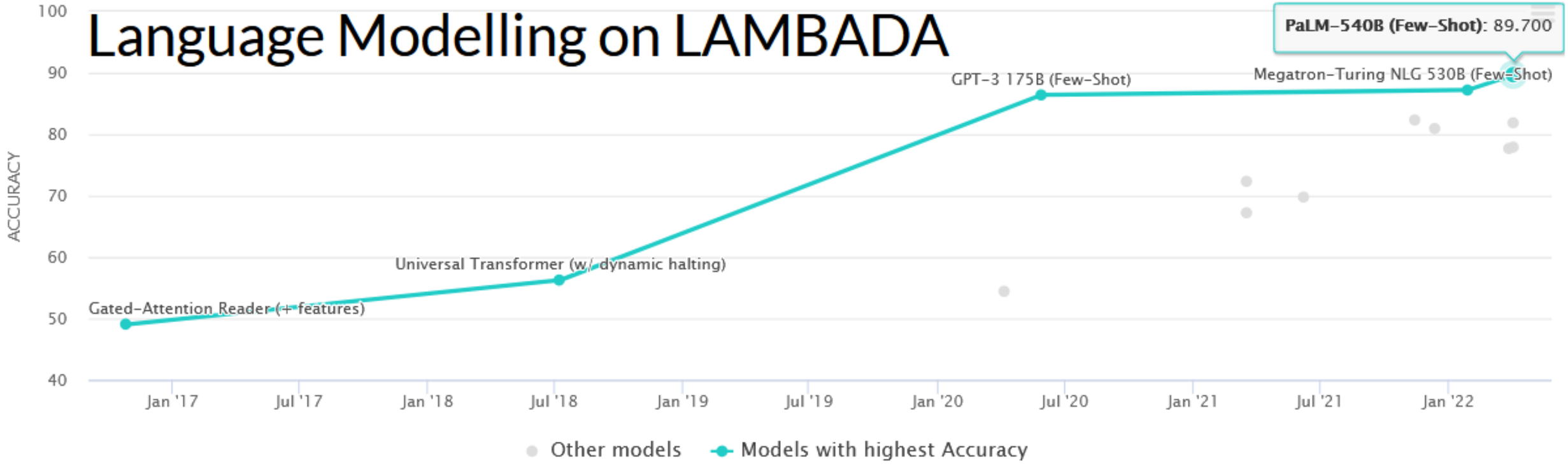
**Большие последовательности: 1450 строк кода,
the entirety of the MIT, Creative Commons, and Project Gutenberg licenses**

Число π

GPT-2 will complete the prompt «3.14159» with the first 25 digits
beam-search – 500 digits
«pi is 3.14159» – 799 digits
«pi begins 3.14159» – 824 digits

контекст важен!
(а это не учитывалось в эксперименте)

SotA


Language Modelling on LAMBADA
PaLM-540B (Few-Shot): 89.700
Megatron-Turing NLG 530B (Few-Shot)
GPT-3 175B (Few-Shot)
Universal Transformer (w/ dynamic halting)
Gated-Attention Reader (+ features)
ACCURACY
100
90
80
70
60
50
40
Jan '17
Jul '17
Jan '18
Jul '18
Jan '19
Jul '19
Jan '20
Jul '20
Jan '21
Jul '21
Jan '22
Other models
Models with highest Accuracy

Итог

ULMfit	01.2018	fast.ai	1 GPU-дней
GPT	06.2018	OpenAI	240 GPU-дней
BERT	10.2018	Google AI	265 TPU-дней
GPT-2	02.2019	OpenAI	>2048 TPU-дней

Языковые модели – предсказывают следующее слово

есть простые n-граммные, если рекуррентные / трансформерные – позволяют учесть весь контекст

Ссылки

хороший курс «Natural Language Processing with Deep Learning»

http://web.stanford.edu/class/cs224n/

Обзор стратегий декодирования

https://lilianweng.github.io/lil-log/2021/01/02/controllable-neural-text-generation.html