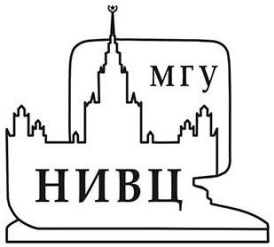


ИНТЕЛЛЕКТ

Токенизация, оценка качества LLM

к.ф.-м.н. Тихомиров М.М.

НИВЦ МГУ имени М. В. Ломоносова

Токенизация

Токенизация до subword tokenization

Как можно представить текст для нейронной сети перед векторизацией?

- Символы: ['Б', 'о', 'л', 'ь', 'ш', 'и', 'е', ' ', 'я', 'з', 'ы', 'к', 'о', 'в', 'ы', …]
  - Семантика единицы минимальна.
  - Длина последовательности = количеству символов.

- Слова: ['Большие', 'языковые', 'модели', 'в', 'вопросно-ответных', …]
  - Разных слов только на одном языке миллионы.
  - Богатая морфология “ухудшает” ситуацию.

- Леммы: ['большой', 'языковой', 'модель', 'в', 'вопросно-ответный', …]
  - Основной рабочий вариант раньше,
  - Размеры словаря ~200-500 тыс. слов, остальное UNK.
  - Теряется морфология.

BPE

- **Компромисс** между **символами** и **словами**.
- **Старт с** словаря, состоящего из **символов** (каждый символ - токен).
- Последовательное **объединение** наиболее частотных пар токенов.
- Больше **нет OOV**, а размер словаря может быть небольшим (например, 32 тыс.)!
- Наиболее популярный на данный момент.

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)
```


Sennrich R., Haddow B., Birch A. Neural machine translation of rare words with subword units //arXiv preprint arXiv:1508.07909. – 2015.

Unigram

- Подход **сверху-вниз** в отличие от BPE.
- Стартует с **большого словаря:** наиболее частотные слова корпуса + подстроки + все символы.
- Используя **униграмм** языковую модель и текущий словарь токенов, рассчитываются вероятности всех текстовых последовательностей (предложений) корпуса.
- После чего рассчитывается, удаление каких токенов из словаря меньше всего повышает значение лосс функции:

$$\mathcal{L} = \sum_{s=1}^{|D|} \log(P(X^{(s)})) = \sum_{s=1}^{|D|} \log\Big(\sum_{\mathbf{x} \in \mathcal{S}(X^{(s)})} P(\mathbf{x}) \Big)$$

- 10-20% токенов удаляется, после чего повторение прошлых шагов.

Algorithm 2 Unigram LM (Kudo, 2018)

Input: set of strings D, target vocab size k
procedure UNIGRAMLM(D, k)
  $V \leftarrow$ all substrings occurring more than once in D (not crossing words)
  while $|V| > k$ **do** ▷ Prune tokens
    Fit unigram LM θ to D
    for $t \in V$ **do** ▷ Estimate token 'loss'
      $L_t \leftarrow p_\theta(D) - p_{\theta'}(D)$
      where θ' is the LM without token t
    end for
    Remove $\min(|V| - k, \lfloor \alpha |V| \rfloor)$ of the tokens t with highest L_t from V,
    where $\alpha \in [0, 1]$ is a hyperparameter
  end while
  Fit final unigram LM θ to D
  return V, θ
end procedure

Kudo T. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates //Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). – 2018. – С. 66-75.

Unigram vs BPE

- **BPE** проще и **быстрее** строить, меньше нагрузка на RAM.
  - Обучение **Unigram** на корпусе из 40GB текстов занимает **1T RAM.**
- **BPE** интуитивно **понятнее**.
- **Unigram** разбивает текст **ближе к морфологии языка**, чем **BPE**, что особенно заметно на языках с богатой морфологией (например русский).

Method	English (w.r.t. CELEX2)			Japanese (w.r.t. MeCab)		
	Precision	Recall	F1	Precision	Recall	F1
BPE	38.6%	12.9%	19.3%	78.6%	69.5%	73.8%
Uni. LM	62.2%	20.1%	30.3%	82.2%	72.8%	77.2%

Table 3: Correspondence of subword boundaries between unsupervised tokenization methods and morphological reference segmentations.

Bostrom K., Durrett G. Byte Pair Encoding is Suboptimal for Language Model Pretraining //Findings of the Association for Computational Linguistics: EMNLP 2020. – 2020. – С. 4617-4624.

sentencepiece

- Пакет от google для обучения токенизации (BPE/Unigram)
- Результат достаточно просто потом “конвертируется” в huggingface токенизацию
- Особое представление пробелов

```
>>> import sentencepiece as spm
>>> s = spm.SentencePieceProcessor(model_file='spm.model')
>>> for n in range(5):
...     s.encode('New York', out_type=str, enable_sampling=True, alpha=0.1, nbest_size=-1)
...
['▁', 'N', 'e', 'w', '▁York']
['▁', 'New', '▁York']
['▁', 'New', '▁Y', 'o', 'r', 'k']
['▁', 'New', '▁York']
['▁', 'New', '▁York']
```

```
[32] list(tokenizer_json['model']['vocab'].items())[250:270]

[('<0xF7>', 250),
 ('<0xF8>', 251),
 ('<0xF9>', 252),
 ('<0xFA>', 253),
 ('<0xFB>', 254),
 ('<0xFC>', 255),
 ('<0xFD>', 256),
 ('<0xFE>', 257),
 ('<0xFF>', 258),
 ('___', 259),
 ('_____', 260),
 ('▁t', 261),
 ('in', 262),
 ('er', 263),
 ('▁a', 264),
 ('he', 265),
 ('on', 266),
 ('re', 267),
 ('▁s', 268),
 ('en', 269)]
```

```
tokenizer_json['model']['merges'][:20]

[['▁', 't'],
 ['i', 'n'],
 ['e', 'r'],
 ['▁', 'a'],
 ['h', 'e'],
 ['o', 'n'],
 ['r', 'e'],
 ['▁', 's'],
 ['e', 'n'],
 ['a', 't'],
 ['o', 'r'],
 ['▁t', 'he'],
 ['▁th', 'e'],
 ['▁', 'the'],
 ['e', 's'],
 ['▁', 'w'],
 ['a', 'n'],
 ['▁', 'c'],
 ['i', 's'],
 ['i', 't']]
```

https://github.com/google/sentencepiece

sentencepiece

Из интересных параметров:

- split_digits
- byte_fallback
- character_coverage

```
Usage: ../build/src/spm_train [options] files

  --input (comma separated list of input sentences)  type: std::string default: ""
  --input_format (Input format. Supported format is `text` or `tsv`.)  type: std::string default: ""
  --model_prefix (output model prefix)  type: std::string default: ""
  --model_type (model algorithm: unigram, bpe, word or char)  type: std::string default: "unigram"
  --vocab_size (vocabulary size)  type: int32 default: 8000
  --accept_language (comma-separated list of languages this model can accept)  type: std::string default: ""
  --self_test_sample_size (the size of self test samples)  type: int32 default: 0
  --character_coverage (character coverage to determine the minimum symbols)  type: double default: 0.9995
  --input_sentence_size (maximum size of sentences the trainer loads)  type: std::uint64_t default: 0
  --shuffle_input_sentence (Randomly sample input sentences in advance. Valid when --input_sentence_size > 0)  type: bool defa
  --seed_sentencepiece_size (the size of seed sentencepieces)  type: int32 default: 1000000
  --shrinking_factor (Keeps top shrinking_factor pieces with respect to the loss)  type: double default: 0.75
  --num_threads (number of threads for training)  type: int32 default: 16
  --num_sub_iterations (number of EM sub-iterations)  type: int32 default: 2
  --max_sentencepiece_length (maximum length of sentence piece)  type: int32 default: 16
  --max_sentence_length (maximum length of sentence in byte)  type: int32 default: 4192
  --split_by_unicode_script (use Unicode script to split sentence pieces)  type: bool default: true
  --split_by_number (split tokens by numbers (0-9))  type: bool default: true
  --split_by_whitespace (use a white space to split sentence pieces)  type: bool default: true
  --split_digits (split all digits (0-9) into separate pieces)  type: bool default: false
  --treat_whitespace_as_suffix (treat whitespace marker as suffix instead of prefix.)  type: bool default: false
  --allow_whitespace_only_pieces (allow pieces that only contain (consecutive) whitespace tokens)  type: bool default: false
  --control_symbols (comma separated list of control symbols)  type: std::string default: ""
  --control_symbols_file (load control_symbols from file.)  type: std::string default: ""
  --user_defined_symbols (comma separated list of user defined symbols)  type: std::string default: ""
  --user_defined_symbols_file (load user_defined_symbols from file.)  type: std::string default: ""
  --required_chars (UTF8 characters in this flag are always used in the character set regardless of --character_coverage)  typ
  --required_chars_file (load required_chars from file.)  type: std::string default: ""
  --byte_fallback (decompose unknown pieces into UTF-8 byte pieces)  type: bool default: false
  --vocabulary_output_piece_score (Define score in vocab file)  type: bool default: true
  --normalization_rule_name (Normalization rule name. Choose from nfkc or identity)  type: std::string default: "nmt_nfkc"
  --normalization_rule_tsv (Normalization rule TSV file. )  type: std::string default: ""
  --denormalization_rule_tsv (Denormalization rule TSV file.)  type: std::string default: ""
  --add_dummy_prefix (Add dummy whitespace at the beginning of text)  type: bool default: true
  --remove_extra_whitespaces (Removes leading, trailing, and duplicate internal whitespace)  type: bool default: true
  --hard_vocab_limit (If set to false, --vocab_size is considered as a soft limit.)  type: bool default: true
  --use_all_vocab (If set to true, use all tokens as vocab. Valid for word/char models.)  type: bool default: false
  --unk_id (Override UNK (<unk>) id.)  type: int32 default: 0
  --bos_id (Override BOS (<s>) id. Set -1 to disable BOS.)  type: int32 default: 1
  --eos_id (Override EOS (</s>) id. Set -1 to disable EOS.)  type: int32 default: 2
  --pad_id (Override PAD (<pad>) id. Set -1 to disable PAD.)  type: int32 default: -1
  --unk_piece (Override UNK (<unk>) piece.)  type: std::string default: "<unk>"
  --bos_piece (Override BOS (<s>) piece.)  type: std::string default: "<s>"
  --eos_piece (Override EOS (</s>) piece.)  type: std::string default: "</s>"
  --pad_piece (Override PAD (<pad>) piece.)  type: std::string default: "<pad>"
  --unk_surface (Dummy surface string for <unk>. In decoding <unk> is decoded to `unk_surface`.)  type: std::string default: "
  --train_extremely_large_corpus (Increase bit depth for unigram tokenization.)  type: bool default: false
  --random_seed (Seed value for random generator.)  type: uint32 default: 4294967295
  --enable_differential_privacy (Whether to add DP while training. Currently supported only by UNIGRAM model.)  type: bool defa
  --differential_privacy_noise_level (Amount of noise to add for DP)  type: float default: 0
  --differential_privacy_clipping_threshold (Threshold for clipping the counts for DP)  type: std::uint64_t default: 0
  --help (show help)  type: bool default: false
  --version (show version)  type: bool default: false
  --minloglevel (Messages logged at a lower level than this don't actually get logged anywhere)  type: int default: 0
```

https://github.com/google/sentencepiece

tiktoken

- Библиотека от OpenAI
- Специальное представление токенов в модели
  - Словарь хранится в специальном представлении
  - Мержи происходят тоже в нем

```
list(tokenizer_json['model']['vocab'].items())[130000:130020]
```

```
[('ìÍ˜', 130000),
 ('à¸кà¸Ïà¸²à¸¢', 130001),
 ('ĠkullanÄ±l', 130002),
 ('ĠtÃ´', 130003),
 ('ãģ«ãĤĪãĤĬ', 130004),
 ('ĠëÍIJíķĪ', 130005),
 ('Ġ×¢×ij×ķ×j×ķ', 130006),
 ('ĠriÃª', 130007),
 ('ĠriÃªng', 130008),
 ('ĠyakÄ±n', 130009),
 ('Ø²Ø§', 130010),
 ('Å»', 130011),
 ('×IJ×ķ×Ł×I', 130012),
 ('Ø´Ø§Ø±Ùĥ', 130013),
 ('ĠÐ±ÐµÑģ', 130014),
 ('×´', 130015),
 ('ĠØ§Ø¨ÙĬ', 130016),
 ('ĠTá»ķng', 130017),
 ('ÙĬØ¸', 130018),
 ('ÅĽwiad', 130019)]
```

```
[43] tokenizer_json['model']['merges'][:20]
```

```
[['Ġ', 'Ġ'],
 ['ĠĠ', 'ĠĠ'],
 ['i', 'n'],
 ['Ġ', 't'],
 ['ĠĠĠĠ', 'ĠĠĠĠ'],
 ['e', 'r'],
 ['ĠĠ', 'Ġ'],
 ['o', 'n'],
 ['Ġ', 'a'],
 ['r', 'e'],
 ['a', 't'],
 ['s', 't'],
 ['e', 'n'],
 ['o', 'r'],
 ['Ġt', 'h'],
 ['Ġ', 'Ġ'],
 ['Ġ', 'c'],
 ['l', 'e'],
 ['Ġ', 's'],
 ['i', 't']]
```

```
def bytes_to_unicode():
    """
    Returns list of utf-8 byte and a mapping to unicode strings. We specifically avoids mapping to whitespace/control
    characters the bpe code barfs on.

    The reversible bpe codes work on unicode strings. This means you need a large # of unicode characters in your vocab
    if you want to avoid UNKs. When you're at something like a 10B token dataset you end up needing around 5K for
    decent coverage. This is a significant percentage of your normal, say, 32K bpe vocab. To avoid that, we want lookup
    tables between utf-8 bytes and unicode strings.
    """
    bs = (
        list(range(ord("!"), ord("~") + 1)) + list(range(ord("¡"), ord("¬") + 1)) + list(range(ord("®"), ord("ÿ") + 1))
    )
    cs = bs[:]
    n = 0
    for b in range(2**8):
        if b not in bs:
            bs.append(b)
            cs.append(2**8 + n)
            n += 1
    cs = [chr(n) for n in cs]
    return dict(zip(bs, cs))
```

https://github.com/openai/tiktoken

Ruadapt: адаптация больших языковых моделей на русский язык

Идея - исследовать влияние **токенизатора** на **адаптацию** LLM на **русский язык.**

- Токенизатор преобразует текст в последовательность токенов:
  - интеллект -> ['инте', 'лле', 'кт'] -> [26133, 9997, 3931] -> эмбеддинги.
  - Вычислительная эффективность модели существенно зависит от токенизатора.

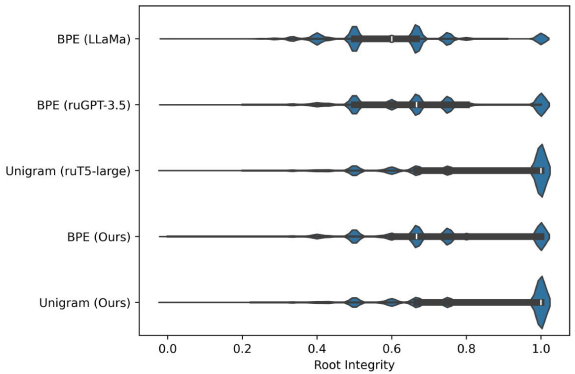
Ставились следующие вопросы:

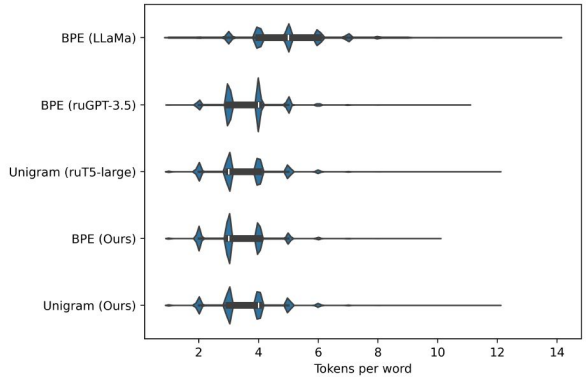
- Какой алгоритм токенизации лучше подходит для русского языка: **BPE** или **Unigram**?
- Можно ли адаптировать модель, обучая только новые слои?
  - Инициализация усреднением.
- Насколько можно повысить вычислительную эффективность модели на русских текстах и не упадет ли качество?

Сравнение токенизации с точки зрения русского языка

Датасет (~30GB), используемый для обучения токенизации и модели, состоял из: новости, литература, русская wikipedia, соцсеть пикабу, stackoverflow, некоторый процент англ. wikipedia.

Root Integrity - максимальное пересечение токена с корнем слова. Оценивался на **RuMorphs-Words**.





САПИН А. С. Построение нейросетевых моделей морфологического и морфемного анализа текста //Труды Института системного программирования РАН. – 2021. – Т. 33. – №. 4. – С. 117-130.

Сравнение качества модели в зависимости от токенизации

- Бенчмарк: Russian Super Glue
- Решалась задача адаптации LLaMa-7B на русский язык путем замены токенизации

	LiDiRus	RCB	PARus	MuSeRC	TERRa	RUSSE	RWSD	DaNetQA	RuCoS	mean
llama7b	0,361	0,462	0,672	0,799	0,860	0,624	0,682	0,866	**0,802**	0,681
llama7b_rulm_raw	0,392	0,494	0,688	0,805	0,859	0,631	0,669	**0,871**	0,791	0,689
llama7b_rulm_bpe	0,365	0,509	0,684	0,782	0,844	0,626	**0,747**	0,824	0,737	0,680
llama7b_rulm_unigram	**0,412**	**0,561**	0,732	0,800	**0,875**	**0,660**	0,675	0,865	0,756	**0,704**
llama7b_rulm_unigram_hm	0,387	0,546	**0,750**	**0,815**	0,866	**0,660**	0,740	0,812	0,758	**0,704**

Сравнение качества моделей на RSG с дообучением

	LiDiRus	RCB	PARus	MuSeRC	TERRa	RUSSE	RWSD	DaNetQA	RuCoS	mean
saiga7b	0,084	0,412	0,528	0,311	0,514	0,484	**0,675**	0,676	0,319	0,445
saiga7b_rulm_raw	0,025	0,373	**0,610**	0,310	0,523	**0,587**	0,584	0,783	0,474	0,474
saiga7b_rulm_bpe	0,149	0,429	0,596	0,344	**0,647**	0,478	0,636	0,757	0,397	0,493
saiga7b_rulm_unigram	0,194	**0,432**	0,568	0,313	0,591	**0,587**	0,630	**0,789**	**0,477**	**0,509**
saiga7b_rulm_unigram_hm	**0,198**	0,413	0,584	**0,349**	0,533	**0,587**	0,578	**0,789**	0,475	0,501

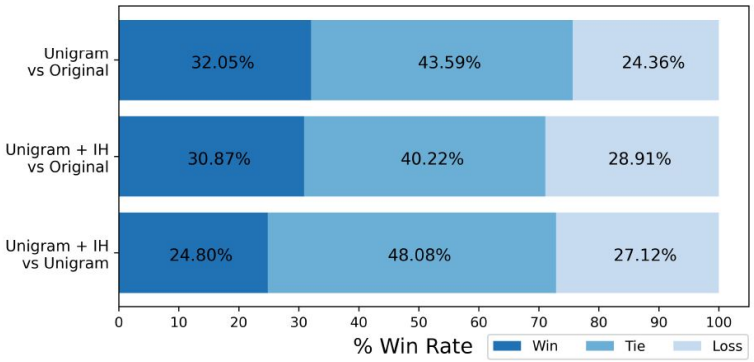
Сравнение качества инструктивных версий моделей на RSG в zero-shot

Tikhomirov M., Chernyshev D. Impact of Tokenization on LLaMa Russian Adaptation //arXiv preprint arXiv:2312.02598. – 2023.

Оценка качества и вычислительной эффективности

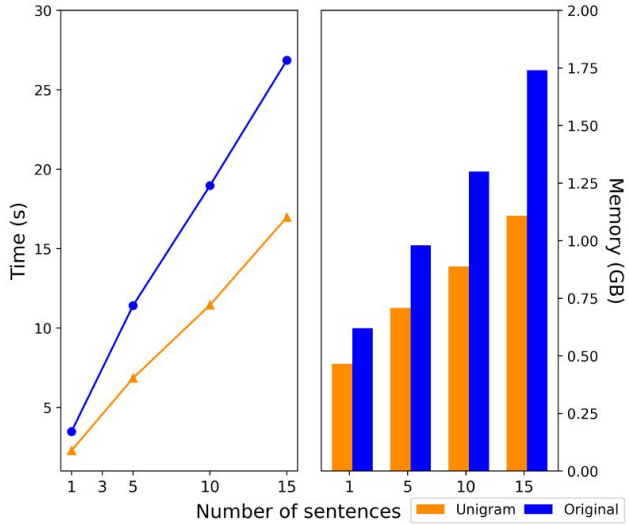
Сравнение путем выбора лучшей генерации из двух (side-by-side).

Было подготовлено 78 вопросов для моделей,15 аннотаторов.



Сравнение качества инструктивных версий моделей людьми

До **60%** прироста в скорости **при генерации** и до **35%** прироста в скорости **при обучении**.



Сравнение вычислительной эффективности при генерации

Исследование адаптации более современных моделей

- Больше ресурсов -> подбор гиперпараметров
- Более мультиязычные модели, чем раньше
- Сравнили расширение vs замена токенизации
- llama3 на tiktoken, mistral на sentencepiece

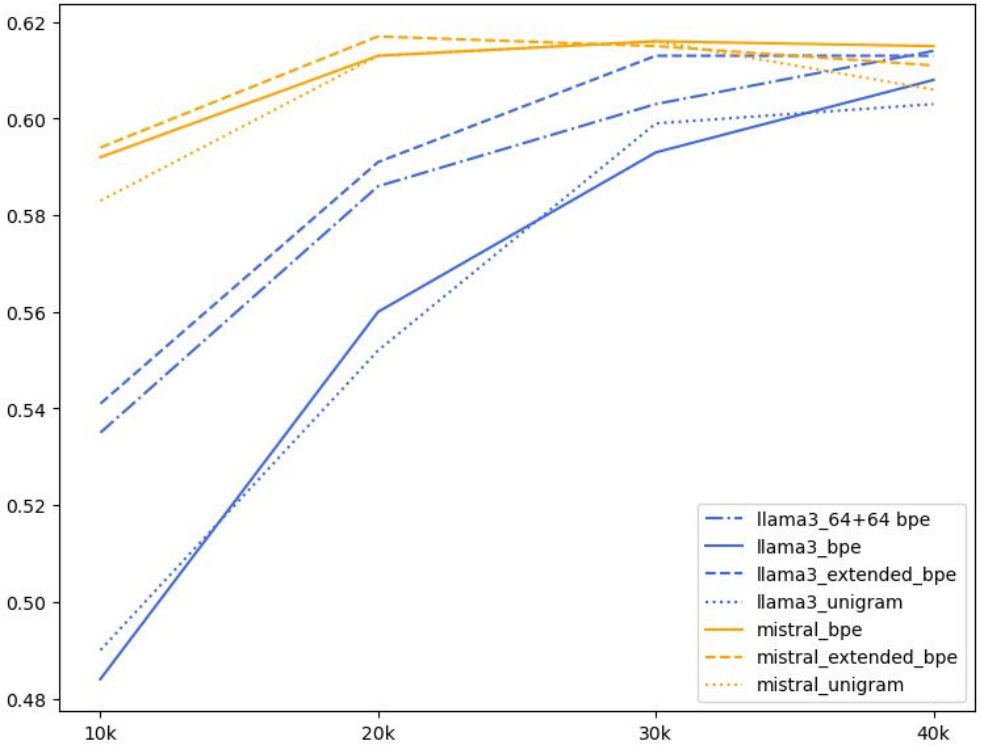
Краткие выводы:

- batch size -> количество шагов оптимизации важно
- learning rate - оказывает существенное влияние на более “тяжелые” для адаптации модели

model_name	tokenization	step	mean
mistral	bpe	10k	**0,616**
mistral	bpe	20k	0,606
mistral	bpe	30k	**0,616**
mistral	bpe	**40k**	**0,616**
mistral	bpe	full	0,613
mistral	unigram	10k	0,596
mistral	unigram	20k	0,609
mistral	unigram	30k	0,614
mistral	unigram	**40k**	**0,616**
mistral	unigram	full	0,615
mistral	extended_bpe	10k	0,6
mistral	extended_bpe	20k	0,609
mistral	extended_bpe	30k	**0,617**
mistral	extended_bpe	**40k**	**0,617**
mistral	extended_bpe	full	0,615
llama3	bpe	10k	0,604
llama3	bpe	**20k**	**0,618**
llama3	bpe	30k	**0,617**
llama3	bpe	full (38k)	**0,617**
llama3	unigram	10k	0,602
llama3	unigram	20k	**0,609**
llama3	unigram	**30k**	**0,609**
llama3	unigram	full (38k)	0,607
llama3	extended_bpe	10k	0,609
llama3	extended_bpe	**20k**	**0,627**
llama3	extended_bpe	30k	0,621
llama3	extended_bpe	40k	0,622
llama3	extended_bpe	full	0,624
llama3	64+64 bpe	10k	0,607
llama3	64+64 bpe	20k	0,605
llama3	64+64 bpe	30k	0,616
llama3	64+64 bpe	**40k**	**0,62**
llama3	64+64 bpe	full	**0,619**
mistral	raw	-	0,604
llama3	raw	-	0,629

Зависимость процедуры адаптации от LR

- Mistral адаптируется быстро
- Llama требует более существенного изменения эмбедингов
  - tiktoken на 128т. токенов плохо ложится на новую токенизацию?



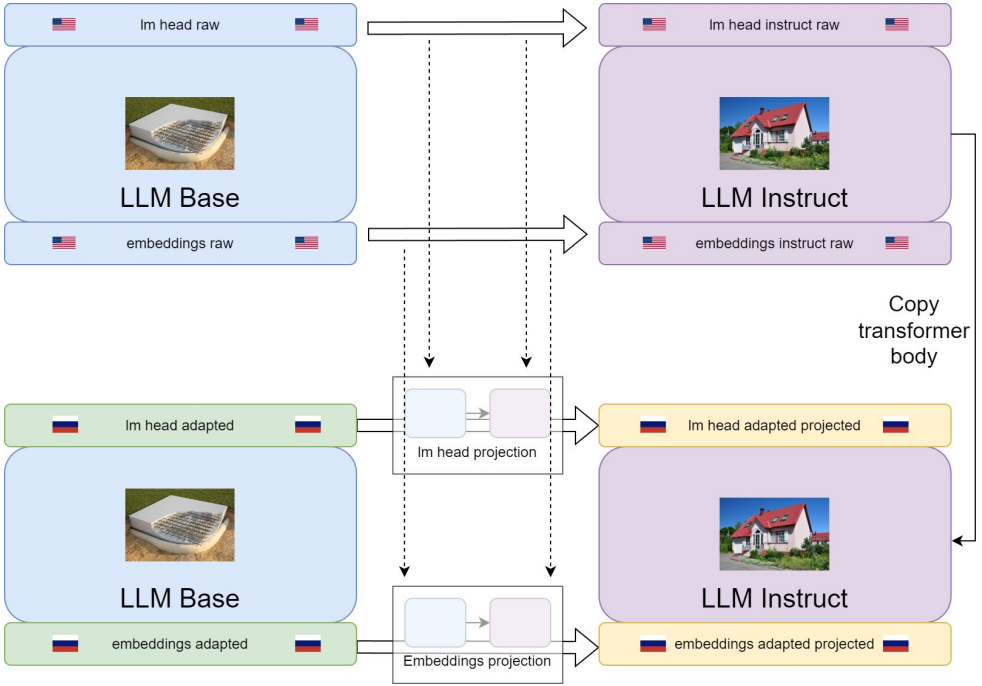
Проблемы адаптации модели на язык

- Адаптация происходит для **базовых моделей**, не инструктивных
- Соответственно требуется воспроизводить процедуру инстракт тюнинга с базы
  - LLaMa-3 обучалась на 10 миллионах инструкций (датасет не опубликован!)
  - Лучшая версия модели mistral openchat-3.5 обучалась на датасете, который также закрыт

- Нужно быть аккуратным, чтобы не потерять исходные знания модели

Learned Embeddings Propagation

- Еще не опубликованный метод
- Модели после LEP являются инструктивными и адаптированными моделями
- Присутствует некоторая потеря качества, так что нужен небольшое дообучение



Learned Embeddings Propagation

- Можно по-разному расчитывать проекцию, но общий тренд один:
  - Лучше база -> лучше результат после lep
  - Лучше инстракт -> лучше результат после lep
  - Extended лучше BPE / Unigram

Learned Embedding Propagation

Table 3. Darumeru zero-shot evaluation results for Learned Embedding Propagation methods.

Vocab	LEP method	Micro-Avg	DaruMMLU	DaruMERA	DaruSum	DaruCopy (Ru)	DaruCopy (En)
OpenChat-3.5							
BPE	Swap	**0,587**	**0,528**	**0,526**	0,277	**0,829**	0,988
	Overlap	0,584	0,525	0,523	0,281	0,818	0,986
	Conversion	0,583	0,526	0,524	**0,284**	0,791	**0,993**
Unigram	Swap	0,556	**0,517**	0,517	0,282	0,614	0,985
	Overlap	**0,572**	0,514	0,534	0,297	**0,680**	0,981
	Conversion	0,565	**0,515**	**0,519**	**0,301**	0,651	**0,999**
Extended	Swap	**0,608**	0,535	0,540	0,298	**0,907**	0,999
	Overlap	0,607	0,535	0,539	**0,307**	0,898	0,999
	Conversion	**0,609**	**0,535**	**0,541**	**0,306**	**0,909**	0,999
LLaMa-3 (instruct)							
BPE	Swap	0,565	0,544	0,486	0,317	0,729	0,999
	Overlap	0,569	0,546	0,489	0,314	0,753	0,999
	Conversion	**0,570**	0,546	0,490	0,318	0,754	0,999
Unigram	Swap	0,582	0,545	0,488	0,313	0,865	0,999
	Overlap	0,580	0,545	0,482	0,314	0,876	0,999
	Conversion	**0,584**	**0,545**	**0,488**	**0,315**	0,889	0,994
Extended	Swap	0,592	0,557	0,498	0,319	0,921	0,969
	Overlap	**0,597**	0,556	0,504	**0,321**	**0,936**	0,964
	Conversion	**0,597**	0,556	0,501	0,318	0,921	**0,994**
Optimized	Swap	0,594	0,554	0,499	**0,327**	0,928	0,970
	Overlap	0,586	0,553	0,495	0,323	0,925	0,925
	Conversion	**0,598**	**0,555**	**0,500**	**0,324**	**0,928**	**0,995**

Learned Embeddings Propagation

- После дополнительного SFT шага качество уверенно растет.
- Дополнительные примеры с инструкциями на копирование абзацев текста помогают

Table 5. Benchmark results for continued instruction-tuning of LEP-Conversion models

Model	dataset	Micro-Avg	DaruMMLU	DaruMERA	DaruSum	DaruCopy (EN)	DaruCopy (RU)
OpenChat 3.5 (original)	-	0,607	**0,543**	0,526	0,322	0,999	0,917
	saiga d7	0,611	0,540	**0,528**	**0,325**	0,999	0,945
	+copy task	**0,615**	0,541	0,524	0,324	**1,000**	**0,995**
OpenChat 3.5 (Unigram)	-	0,564	0,515	0,519	0,301	0,998	0,646
	saiga d7	0,599	**0,532**	0,556	0,316	0,999	0,754
	+copy task	**0,630**	0,530	**0,559**	**0,321**	**1,000**	**0,999**
OpenChat 3.5 (Extended)	-	0,610	0,535	0,541	0,307	0,999	0,914
	saiga d7	0,616	**0,543**	**0,566**	0,319	0,999	0,845
	+copy task	**0,632**	0,541	0,563	**0,321**	**1,000**	**0,989**
LLaMa-3-8B instruct (original)	-	0,61	0,571	0,510	0,322	1,000	0,972
	saiga d7	0,615	**0,576**	0,512	0,329	1,000	0,983
	+copy task	**0,616**	0,575	**0,513**	**0,332**	**1,000**	**0,995**
LLaMa-3-8B instruct (Extended)	-	0,603	0,557	0,503	0,328	0,990	0,956
	saiga d7	0,614	**0,568**	0,519	0,338	0,995	0,961
	+copy task	**0,618**	0,565	**0,521**	**0,339**	**1,000**	**0,984**
LLaMa-3-8B instruct (Optimized)	-	0,603	0,553	0,506	0,326	0,995	0,949
	saiga d7	0,611	**0,555**	0,515	0,336	**1,000**	0,971
	+copy task	**0,617**	**0,555**	**0,522**	**0,339**	**1,000**	**0,989**

Model	Micro-Avg	DaruMMLU	DaruMERA	DaruSum	DaruCopy (EN)	DaruCopy (RU)
Openchat 3.5 (Mistral-7B)	0,607	0,543	0,526	**0,322**	0,999	0,917
LLaMa-3-8B (Instruct)	**0,610**	0,571	0,510	**0,322**	**1,000**	**0,972**
Saiga (LLaMa-3-8B)	0,608	0,574	0,514	0,320	0,995	0,939
Vikhr-5.2 (Mistral-7B)	0,587	0,494	**0,573**	0,308	0,959	0,693
Qwen-2 7B	0,613	**0,624**	0,548	0,300	0,938	0,842
Mistral Nemo (12B)	**0,639**	0,592	**0,576**	0,320	0,998	0,924

Что дает нам адаптация моделей на язык

Токенизация

RefalMachine/ruadapt_qwen2.5_3B_ext_u48_instruct_v4

А льп ак а — домашнее моз олен ого е животное , предполож ительно произошедше е от вик ун ьи (виг они). Раз водят в высок огор ном по я се Южной Амер ики (Ан ды). На сегодняшний день там обитает около трёх миллионов аль пак , большая часть из которых нас еляет Перу . Вы ращ ивают аль пак для стр иж ки шерсти , из которой делают тёпл ые и мягкие од ея ла , пл ед ы и одеж ду , а из мех а делают предметы для дома .

(94 tokens / 375 characters)

Qwen/Qwen2.5-3B-Instruct

А ль п ак а — дом аш нее м оз ол ен ого е живот ное , пред пол ож итель но произ о шедш ее от в ик ун ь и (в иг он и). Раз вод ят в выс ок о г ор ном поя се Ю ж ной А мер ики (А н ды). На сегодня ш ний день та м об ит ает около тр ё х миллион ов а ль п ак , больш ая часть из которых нас еля ет Пер у . Вы ращ ивают а ль п ак для стр иж ки ш ер сти , из которой дел ают т ё пл ые и мяг кие од е я ла , пл ед ы и од ежду , а из мех а дел ают предмет ы для дома .

(140 tokens / 375 characters)

Ru Arena General

LLM Benchmark | About | Submit here!

Search

Separate multiple queries with ';'.

Select Columns to Display:

score | 95% CI | lower | upper | avg_tokens | std_tokens | lc_score

model	score	95% CI	avg_tokens
DeepSeek - DeepSeek-V3-Chat	96.3	+0.7 / -0.8	665.97
-chatgpt-4o-latest	94.75	+0.8 / -0.7	693.15
yi-lightning	93.46	+0.9 / -0.9	636.68
o1-mini	93.45	+0.7 / -0.8	791.18
RefalMachine-RuadaptQwen-32B-Pro_v1	92.18	+1.2 / -1.2	563.43
claude-3-opus-20240229	91.3	+1.0 / -1.1	468.69
gpt-4-1106-preview	90.89	+1.1 / -1.1	541.66
T-Tech-T-pro-it-1.0	90.87	+1.1 / -1.2	502
o1-preview	90.8	+0.9 / -1.0	664.89
RefalMachine-RuadaptQwen2.5-32B-instruct-v1	90.47	+1.1 / -1.0	527.86
SberDevices-GigaChatMaxWithoutFilter	89.96	+1.2 / -1.4	523.95

“Особенности” токенизации в современных пакетах [1]

- В **huggingface** два реализации токенизации: быстрая и обычная.
- Их поведение для одинаковых моделей **должно быть эквивалентно, но!**
- **Пробелы** для токенизации **важны**

```
text1 = "<s>system\nТы — Сайга</s><s>user\nКакой результат уровненеия 2+2*2? Объясни свой результат</s><s>bot\n"
text2 = "<s> system\nТы — Сайга </s> <s> user\nКакой результат уровненеия 2+2*2? Объясни свой результат </s> <s> bot\n"
```

```
[19] from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("IlyaGusev/saiga_mistral_7b_merged", use_fast=False)
tokens1 = tokenizer(text1)["input_ids"]
print(tokens1)
```

```
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
[1, 1587, 13, 28875, 28829, 1040, 9177, 28819, 2618, 2, 1, 2188, 13, 22117, 13108, 20959, 28786, 1351, 5658, 28778, 8977, 28772, 28811, 28
```

```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("IlyaGusev/saiga_mistral_7b_merged", use_fast=True)
tokens2 = tokenizer(text1)["input_ids"]
print(tokens2)
```

```
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
[1, 6574, 13, 28875, 28829, 1040, 9177, 28819, 2618, 700, 28713, 3409, 28713, 28767, 1838, 13, 22117, 13108, 20959, 28786, 1351, 5658, 287
```

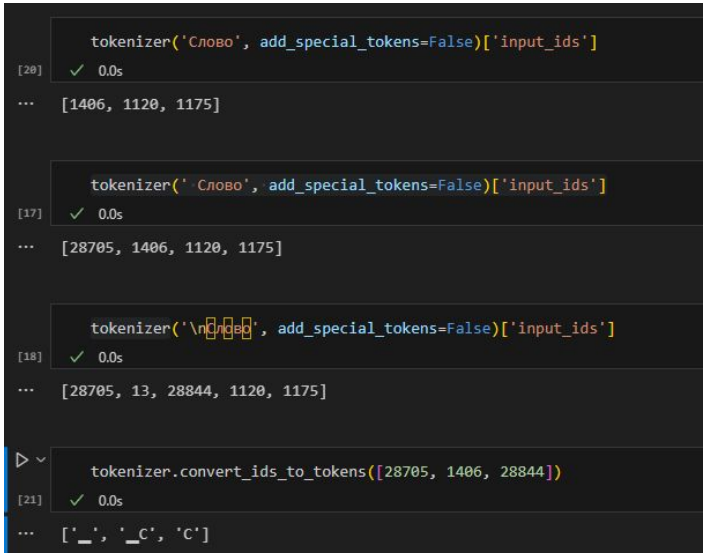
```
[17] from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("IlyaGusev/saiga_mistral_7b_merged", use_fast=True)
tokens3 = tokenizer(text2)["input_ids"]
print(tokens3)
```

```
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
[1, 1587, 13, 28875, 28829, 1040, 9177, 28819, 2618, 2, 1, 2188, 13, 22117, 13108, 20959, 28786, 1351, 5658, 28778, 8977, 28772, 28811, 28
```

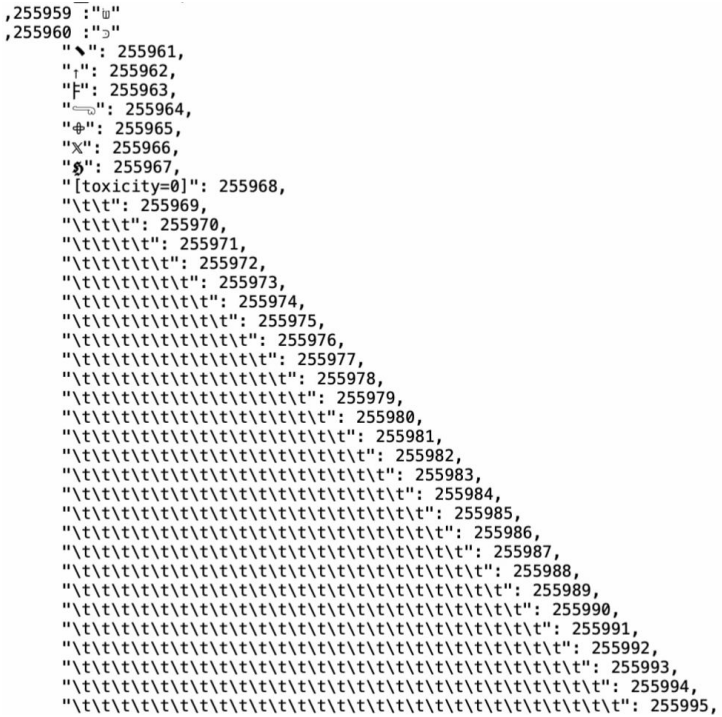
https://huggingface.co/

“Особенности” токенизации в современных пакетах [2]

- Токенизатор всегда считает первое слово последовательности “пробельным”.
- Представление для “ Слово” и “\nСлово” отличны.
- Разные id токенов -> разные вектора на вход.



Пример словаря токенов в модели Gemma



```
,255959 :"ш"
,255960 :"ↄ"
     "↘": 255961,
     "↑": 255962,
     "╞": 255963,
     "⇦": 255964,
     "⊕": 255965,
     "𝕏": 255966,
     "𝔇": 255967,
     "[toxicity=0]": 255968,
     "\t\t": 255969,
     "\t\t\t": 255970,
     "\t\t\t\t": 255971,
     "\t\t\t\t\t": 255972,
     "\t\t\t\t\t\t": 255973,
     "\t\t\t\t\t\t\t": 255974,
     "\t\t\t\t\t\t\t\t": 255975,
     "\t\t\t\t\t\t\t\t\t": 255976,
     "\t\t\t\t\t\t\t\t\t\t": 255977,
     "\t\t\t\t\t\t\t\t\t\t\t": 255978,
     "\t\t\t\t\t\t\t\t\t\t\t\t": 255979,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t": 255980,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255981,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255982,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255983,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255984,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255985,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255986,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255987,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255988,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255989,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255990,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255991,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255992,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255993,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255994,
     "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t": 255995,
```

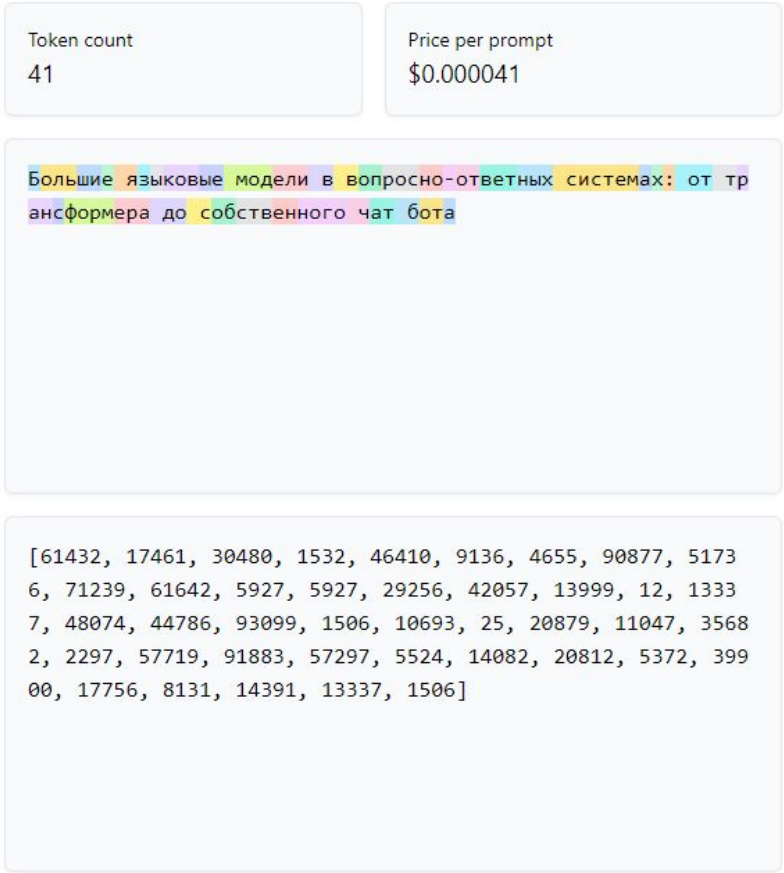
Разница в токенизации для языков

- Токенизация полностью **зависит от корпуса**, на котором обучалась.
- Если в корпусе было мало какого-нибудь языка, то и слово будет разбиваться на много токенов.
- Чем **больше токенов**, тем **дольше** ее **обработка**!
- В **LLaMa** и других открытых моделях в среднем **2 символа** на **токен** для **русского языка**.

Пример токенизации текста в ChatGPT: английский

https://tiktokenizer.vercel.app/

Пример токенизации текста в ChatGPT: русский



https://tiktokenizer.vercel.app/

Оценка качества LLM

Оценка качества LLM

Обычная оценка качества в ML: 1) Есть четкая задача 2) Есть четкий вариант правильного и не правильного ответа 3) различные формулы оценки.

Как оценивать LLM, особенно инструктивные?

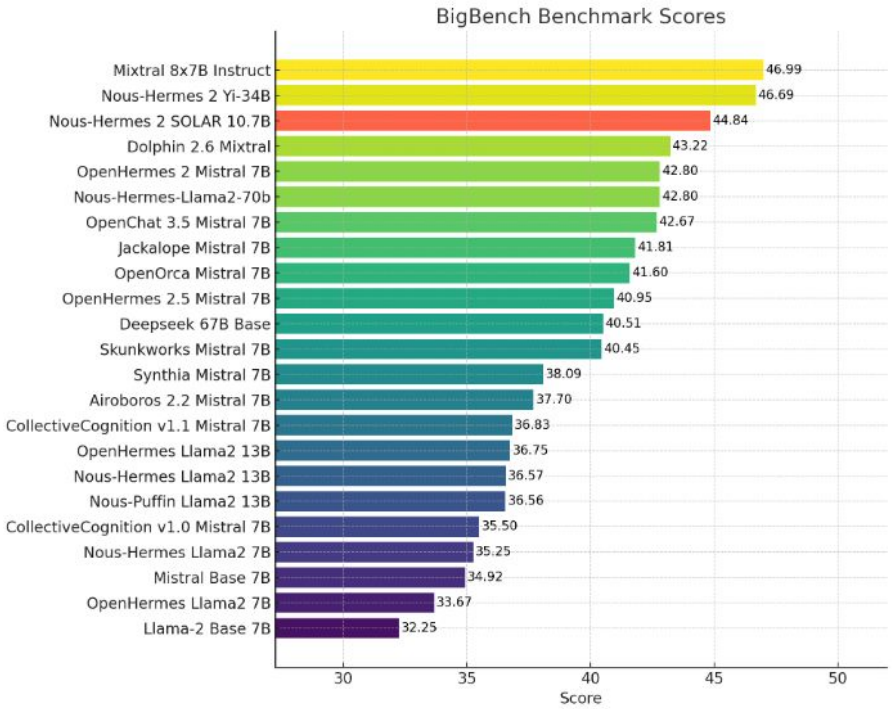
- zero-shot / few-shot при решении конкретных задач.
- Следование инструкциям.
- Полезность, вредность, токсичность.
- Сравнение side-by-side людьми
- Сравнение side-by-side намного более мощной LLM.

MMLU

- Классический бенчмарк для оценки **знаний** модели.
- Вопросы с множественным выбором, 4 варианта ответа, оценка вероятности генерации A, B, C, D.
- Покрытие различных областей от физики до философии, разный уровень (колледж, старшая школа, профессиональный).
- Считают в zero-shot и few-shot с k=5.
  - Качество с few-shot у базовых и инстракт моделей обычно схожие.
- Для русского языка есть переводной вариант: https://github.com/NLP-Core-Team/mmlu_ru

BigBench

- 200+ NLP задач.
- Есть Lite и Hard версии.
- Пример из карточки модели NousResearch/Nous-Hermes-2-SOLAR-10.7B



Srivastava A. et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models //arXiv preprint arXiv:2206.04615. – 2022.

Russian Super Glue

- Относительно старый бенчмарк на данный момент.
- Содержит 9 различных датасетов.
- Имеет тренировочные данные для каждого из них.
- Уровень “человека” с учетом обучения по сути уже давно превзойден.

https://russiansuperglue.com/leaderboard/2

Russian Super Glue

Leaderboard

We have improved the datasets. Please, change the leaderboard for the version (1.0/1.1) you are looking for, clicked on the button below.
You can switch between the scores and inference speed leaderboard as well. Click on the button Performance.

Version 1.0 | Performance*

* More information about speed scores and RAM are available here.

Rank	Name	Team	Link	Score	LiDiRus	RCB	PARus	MuSeRC	TERRa	RUSSE	RWSD	DaNetQA	RuCoS
1	HUMAN BENCHMARK	AGI NLP	i	**0.811**	0.626	0.68 / 0.702	0.982	0.806 / 0.42	0.92	0.805	0.84	0.915	0.93 / 0.89
2	ruadapt Solar 10.7 twostage	RCC MSU	i	**0.805**	0.591	0.597 / 0.594	0.916	0.946 / 0.837	0.927	0.739	0.844	0.933	0.82 / 0.797
3	Mistral 7B LoRA	Saiga team	i	**0.763**	0.46	0.529 / 0.573	0.824	0.927 / 0.787	0.888	0.758	0.786	0.919	0.83 / 0.816
4	FRED-T5 1.7B finetune	SberDevices	i	**0.762**	0.497	0.497 / 0.541	0.842	0.916 / 0.773	0.871	0.823	0.669	0.889	0.9 / 0.902
5	Golden Transformer v2.0	Avengers Ensemble	i	**0.755**	0.515	0.384 / 0.534	0.906	0.936 / 0.804	0.877	0.687	0.643	0.911	0.92 / 0.924
6	LLaMA-2 13B LoRA	Saiga team	i	**0.718**	0.398	0.489 / 0.543	0.784	0.919 / 0.761	0.793	0.74	0.714	0.907	0.78 / 0.76
7	Saiga 13B LoRA	Saiga team	i	**0.712**	0.436	0.439 / 0.5	0.694	0.898 / 0.704	0.865	0.728	0.714	0.862	0.85 / 0.83
8	YaLM p-tune (3.3B frozen + 40k trainable params)	Yandex	i	**0.711**	0.364	0.357 / 0.479	0.834	0.892 / 0.707	0.841	0.71	0.669	0.85	0.92 / 0.916
9	ruadapt LLaMA-2 7B LoRA	RCC MSU	i	**0.71**	0.417	0.545 / 0.555	0.756	0.894 / 0.695	0.876	0.668	0.708	0.878	0.76 / 0.733
10	FRED-T5 large finetune	SberDevices	i	**0.706**	0.389	0.456 / 0.546	0.776	0.887 / 0.678	0.801	0.775	0.669	0.799	0.87 / 0.863
11	RuLeanALBERT	Yandex Research	i	**0.698**	0.403	0.361 / 0.413	0.796	0.874 / 0.654	0.812	0.789	0.669	0.76	0.9 / 0.902
12	FRED-T5 1.7B (only encoder 760M) finetune	SberDevices	i	**0.694**	0.421	0.311 / 0.441	0.806	0.882 / 0.666	0.831	0.723	0.669	0.735	0.91 / 0.911
13	ruT5-large finetune	SberDevices	i	**0.686**	0.32	0.45 / 0.532	0.764	0.855 / 0.608	0.775	0.773	0.669	0.79	0.86 / 0.859
14	ruRoberta-large finetune	SberDevices	i	**0.684**	0.343	0.357 / 0.518	0.722	0.861 / 0.63	0.801	0.748	0.669	0.82	0.87 / 0.867
15	gpt-3.5-turbo zero-shot	Saiga team	i	**0.682**	0.422	0.484 / 0.505	0.888	0.817 / 0.532	0.795	0.596	0.714	0.878	0.68 / 0.667
16	Golden Transformer v1.0	Avengers Ensemble	i	**0.679**	0.0	0.406 / 0.546	0.908	0.941 / 0.819	0.871	0.587	0.545	0.917	0.92 / 0.924
17	xlm-roberta-large (Facebook) finetune	SberDevices	i	**0.654**	0.369	0.328 / 0.457	0.59	0.809 / 0.501	0.798	0.765	0.669	0.757	0.89 / 0.886
18	mdeberta-v3-base (Microsoft) finetune	SberDevices	i	**0.651**	0.332	0.27 / 0.489	0.716	0.825 / 0.531	0.783	0.727	0.669	0.708	0.87 / 0.868
19	Saiga2 70B zero-shot	Saiga team	i	**0.643**	0.365	0.385 / 0.461	0.82	0.669 / 0.098	0.811	0.59	0.831	0.878	0.69 / 0.678
20	Saiga Mistral 7B zero-shot	Saiga team	i	**0.635**	0.322	0.436 / 0.5	0.698	0.84 / 0.553	0.807	0.587	0.727	0.839	0.58 / 0.571
21	ruT5-base finetune	Sberdevices	i	**0.635**	0.267	0.423 / 0.461	0.636	0.808 / 0.475	0.736	0.707	0.669	0.769	0.85 / 0.847
22	ruBert-large finetune	SberDevices	i	**0.62**	0.235	0.356 / 0.5	0.656	0.778 / 0.436	0.704	0.707	0.669	0.773	0.81 / 0.805

https://russiansuperglue.com/leaderboard/2

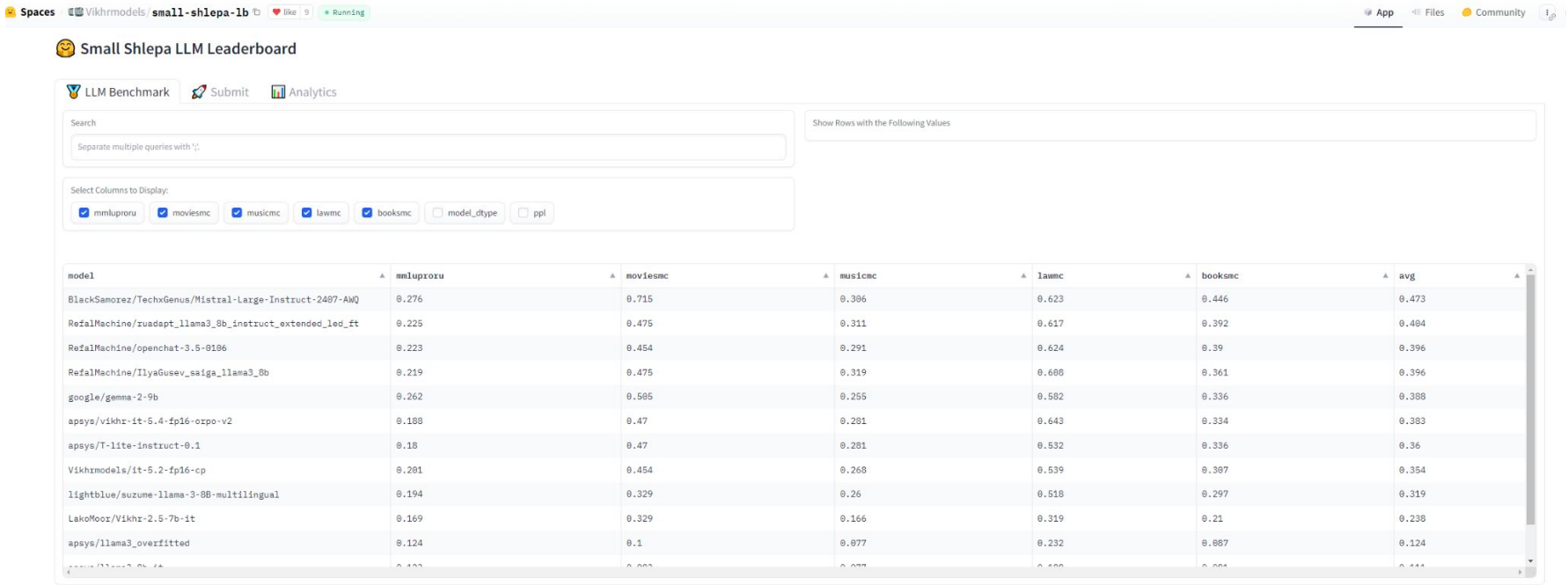
MERA

- Основной разработчик бенча: Сбер (заявлено как Альянс ИИ)
- Содержит 21 задачу
- Недавно получил апдейт, обновился лидерборд

	Модель, команда	Результат	RWSD	PARus	RCB	MultiQ	ruWorldTree	ruOpenBookQA	CheGeKa
1	GPT4o MERA	0.642	0.496	0.944	0.557 / 0.521	0.572 / 0.431	0.985 / 0.985	0.935 / 0.935	0.553 / 0.464
2	Meta-Llama-3.1-405B-In... 405.0B MERA	0.59	0.677	0.902	0.598 / 0.548	0.623 / 0.453	0.981 / 0.981	0.955 / 0.765	0.506 / 0.413
3	GigaChat Max GIGACHAT	0.588	0.665	0.928	0.58 / 0.423	0.486 / 0.322	0.975 / 0.975	0.918 / 0.737	0.469 / 0.397
4	Mistral-Large-Instruct-2... 123.0B MERA	0.574	0.635	0.932	0.55 / 0.531	0.63 / 0.471	0.975 / 0.975	0.915 / 0.915	0.458 / 0.356
5	GPT4o-mini MERA	0.57	0.577	0.918	0.571 / 0.507	0.509 / 0.379	0.956 / 0.956	0.875 / 0.874	0.293 / 0.233
6	Qwen2-72B-Instruct 72.7B MERA	0.57	0.658	0.944	0.511 / 0.484	0.58 / 0.447	0.985 / 0.789	0.945 / 0.945	0.324 / 0.267
7	Qwen2.5-32B-Instruct-A... 32.0B mizinovmv	0.567	0.619	0.93	0.573 / 0.539	0.564 / 0.426	0.987 / 0.987	0.935 / 0.935	0.172 / 0.12
8	Meta-Llama-3.1-70B-Ins... 70.6B MERA	0.554	0.554	0.932	0.596 / 0.526	0.607 / 0.443	0.977 / 0.977	0.913 / 0.731	0.353 / 0.291
9	Meta-Llama-3-70B-Instr... 70.6B MERA	0.528	0.519	0.892	0.587 / 0.493	0.595 / 0.421	0.958 / 0.958	0.905 / 0.727	0.327 / 0.269
10	GigaChat Pro 30.0B GIGACHAT	0.516	0.362	0.884	0.575 / 0.258	0.302 / 0.212	0.931 / 0.931	0.873 / 0.7	0.397 / 0.329
11	Phi-3.5-MoE-instruct 41.9B MERA	0.487	0.465	0.864	0.546 / 0.486	0.446 / 0.349	0.971 / 0.971	0.88 / 0.88	0.178 / 0.139
12	Mixtral-8×22B-Instruct-... 140.6B MERA	0.486	0.473	0.87	0.578 / 0.372	0.521 / 0.366	0.916 / 0.733	0.835 / 0.669	0.338 / 0.267
13	Qwen1.5-32B-Chat 32.5B MERA	0.482	0.377	0.904	0.521 / 0.456	0.452 / 0.318	0.931 / 0.932	0.86 / 0.86	0.119 / 0.082
14	GigaChat Lite+ 7.0B GIGACHAT	0.477	0.435	0.85	0.53 / 0.278	0.287 / 0.208	0.886 / 0.886	0.785 / 0.63	0.268 / 0.214
15	GigaChat Lite 7.0B GIGACHAT	0.477	0.427	0.842	0.523 / 0.275	0.285 / 0.207	0.89 / 0.89	0.78 / 0.626	0.274 / 0.219
16	Qwen2-57B-A14B-Instruct 57.4B MERA	0.471	0.342	0.894	0.541 / 0.449	0.48 / 0.348	0.941 / 0.941	0.888 / 0.888	0.19 / 0.144
17	Phi-3-medium-4k-instruct 14.0B MERA	0.465	0.488	0.896	0.495 / 0.435	0.361 / 0.174	0.962 / 0.962	0.873 / 0.873	0.135 / 0.091
18	ruadapt llama3-8B-instr... 8.4B RCC MSU	0.447	0.542	0.828	0.534 / 0.446	0.483 / 0.334	0.838 / 0.837	0.773 / 0.772	0.146 / 0.101
19	Qwen2-7B-Instruct 7.6B MERA	0.445	0.462	0.82	0.541 / 0.353	0.442 / 0.333	0.926 / 0.742	0.783 / 0.782	0.069 / 0.043
20	Phi-3-medium-128k-inst... 14.0B MERA	0.441	0.485	0.9	0.546 / 0.459	0.361 / 0.162	0.954 / 0.764	0.863 / 0.691	0.134 / 0.087
21	Phi-3-small-8k-instruct 7.4B MERA	0.438	0.523	0.836	0.55 / 0.398	0.361 / 0.231	0.935 / 0.749	0.825 / 0.661	0.08 / 0.053

https://mera.a-ai.ru/ru

Shlepa



model	mmluproru	moviesmc	musicmc	lawmc	booksmc	avg
BlackSamorez/TechxGenus/Mistral-Large-Instruct-2407-AWQ	0.276	0.715	0.306	0.623	0.446	0.473
RefalMachine/ruadapt_llama3_8b_instruct_extended_led_ft	0.225	0.475	0.311	0.617	0.392	0.404
RefalMachine/openchat-3.5-0106	0.223	0.454	0.291	0.624	0.39	0.396
RefalMachine/IlyaGusev_saiga_llama3_8b	0.219	0.475	0.319	0.608	0.361	0.396
google/gemma-2-9b	0.262	0.505	0.255	0.582	0.336	0.388
apsys/vikhr-it-5.4-fp16-orpo-v2	0.188	0.47	0.281	0.643	0.334	0.383
apsys/T-lite-instruct-0.1	0.18	0.47	0.281	0.532	0.336	0.36
Vikhrmodels/it-5.2-fp16-cp	0.201	0.454	0.268	0.539	0.307	0.354
lightblue/suzume-llama-3-8B-multilingual	0.194	0.329	0.26	0.518	0.297	0.319
LakoMoor/Vikhr-2.5-7b-it	0.169	0.329	0.166	0.319	0.21	0.238
apsys/llama3_overfitted	0.124	0.1	0.077	0.232	0.087	0.124

Бенчмарк от Vikhr: “mmlu” с выбором из 12 вариантов

https://huggingface.co/spaces/Vikhrmodels/small-shlepa-lb

Проблемы современных бенчмарков

- Качество на конкретных задачах не отражает пользовательский опыт.
- Сложно оценивать следование инструкциям.
- Модели переобучают на то, чтобы иметь более высокие значения на бенчмарках

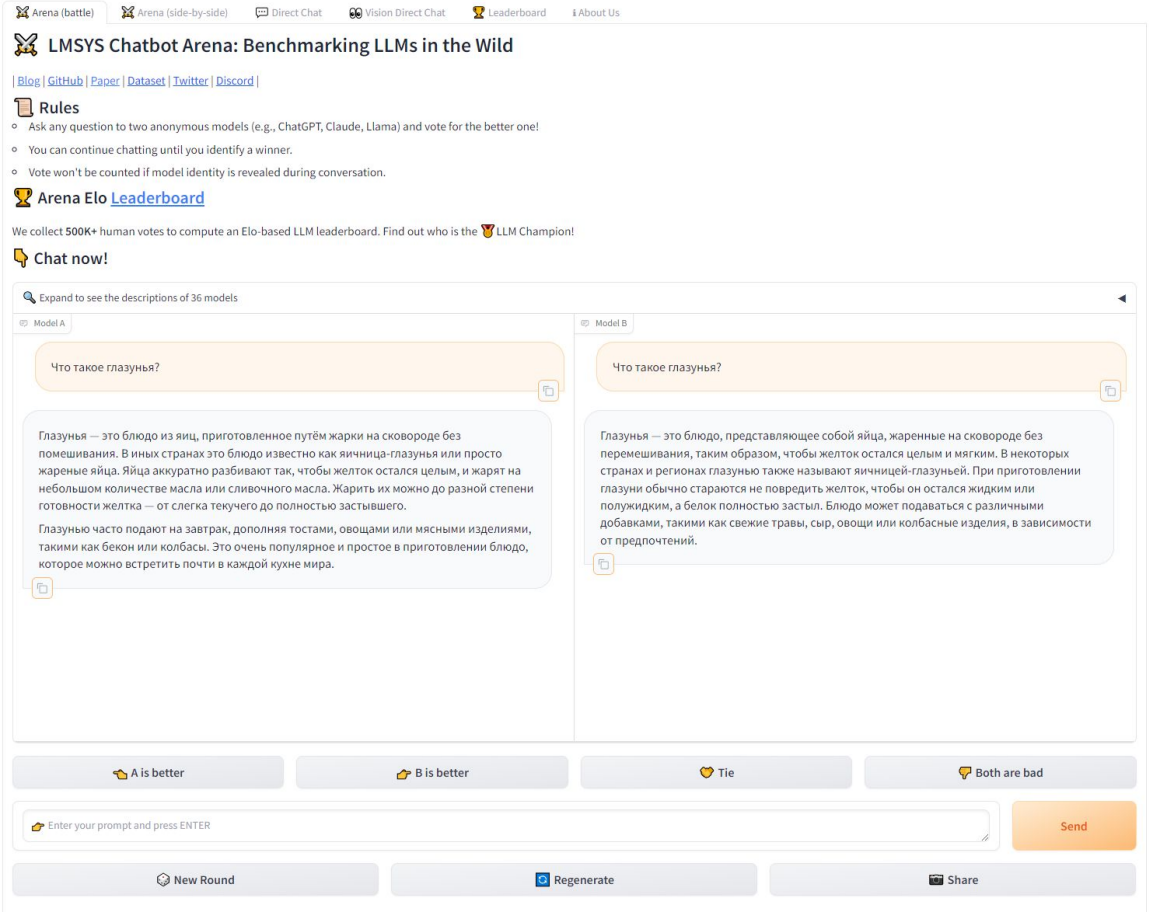
Арены

Вместо сравнения на бенчмарках: сравнение side-by-side.

- Сравнение за счет других больших моделей.
  - MT-bench
  - alpaca-eval

- Сравнение за счет оценки людьми.
  - lmsys

lmsys


Arena (battle)
Arena (side-by-side)
Direct Chat
Vision Direct Chat
Leaderboard
About Us
LMSYS Chatbot Arena: Benchmarking LLMs in the Wild
| Blog | GitHub | Paper | Dataset | Twitter | Discord |
Rules
Ask any question to two anonymous models (e.g., ChatGPT, Claude, Llama) and vote for the better one!
You can continue chatting until you identify a winner.
Vote won't be counted if model identity is revealed during conversation.
Arena Elo Leaderboard
We collect 500K+ human votes to compute an Elo-based LLM leaderboard. Find out who is the LLM Champion!
Chat now!
Expand to see the descriptions of 36 models
Model A
Что такое глазунья?
Глазунья — это блюдо из яиц, приготовленное путём жарки на сковороде без помешивания. В иных странах это блюдо известно как яичница-глазунья или просто жареные яйца. Яйца аккуратно разбивают так, чтобы желток остался целым, и жарят на небольшом количестве масла или сливочного масла. Жарить их можно до разной степени готовности желтка — от слегка текучего до полностью застывшего.
Глазунью часто подают на завтрак, дополняя тостами, овощами или мясными изделиями, такими как бекон или колбасы. Это очень популярное и простое в приготовлении блюдо, которое можно встретить почти в каждой кухне мира.
Model B
Что такое глазунья?
Глазунья — это блюдо, представляющее собой яйца, жаренные на сковороде без перемешивания, таким образом, чтобы желток остался целым и мягким. В некоторых странах и регионах глазунью также называют яичницей-глазуньей. При приготовлении глазуни обычно стараются не повредить желток, чтобы он остался жидким или полужидким, а белок полностью застыл. Блюдо может подаваться с различными добавками, такими как свежие травы, сыр, овощи или колбасные изделия, в зависимости от предпочтений.
A is better
B is better
Tie
Both are bad
Enter your prompt and press ENTER
Send
New Round
Regenerate
Share

lmsys

LMSYS Chatbot Arena Leaderboard

| Vote | Blog | GitHub | Paper | Dataset | Twitter | Discord |

LMSYS Chatbot Arena is a crowdsourced open platform for LLM evals. We've collected over **500,000** human preference votes to rank LLMs with the Elo ranking system.

Arena Elo | Full Leaderboard

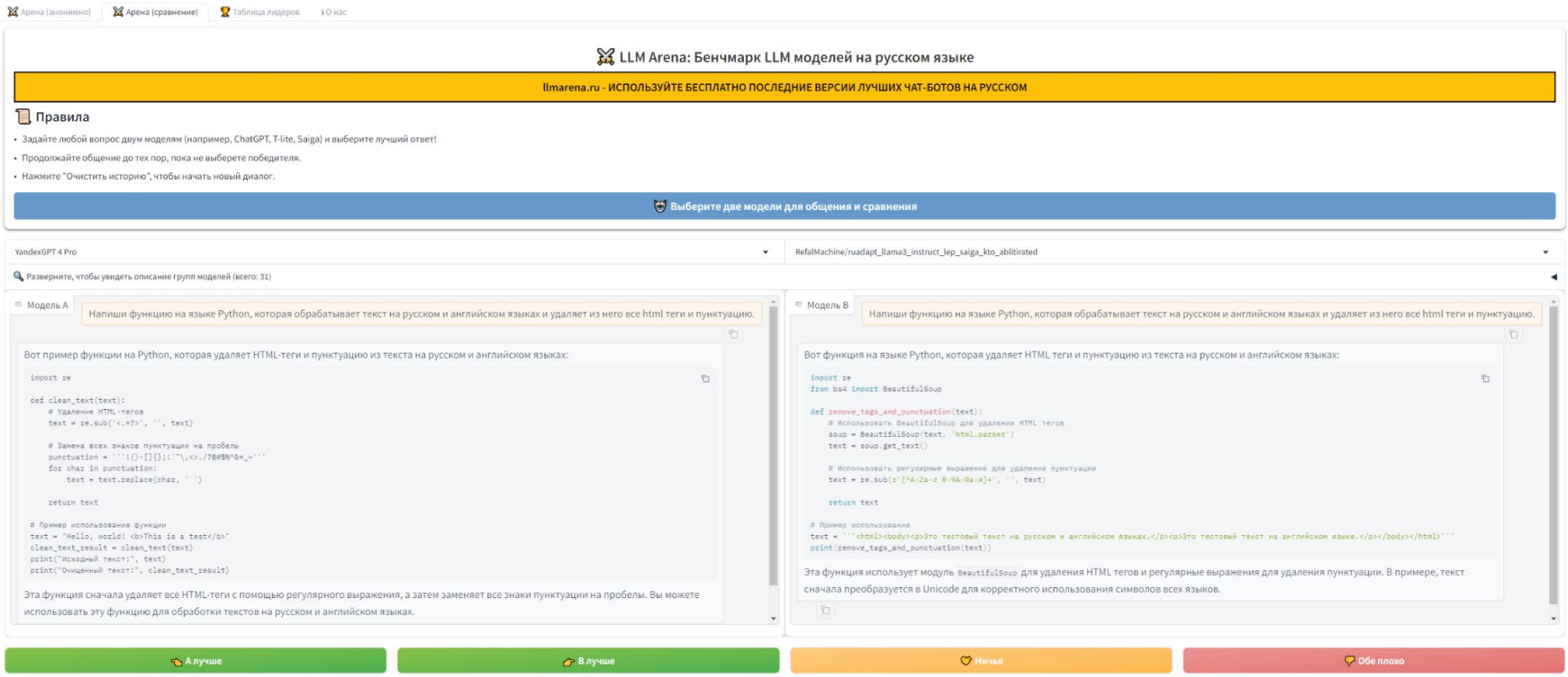
Total #models: **81**. Total #votes: **634676**. Last updated: April 9, 2024.

Contribute your vote at chat.lmsys.org! Find more analysis in the notebook.

Rank	Model	Arena Elo	95% CI	Votes	Organization	License	Knowledge Cutoff
1	Claude 3 Opus	1256	+3/-4	47589	Anthropic	Proprietary	2023/8
1	GPT-4-1106-preview	1254	+3/-4	62657	OpenAI	Proprietary	2023/4
1	GPT-4-0125-preview	1250	+3/-3	47631	OpenAI	Proprietary	2023/12
4	Bard (Gemini Pro)	1208	+5/-5	12468	Google	Proprietary	Online
4	Claude 3 Sonnet	1204	+3/-3	57740	Anthropic	Proprietary	2023/8
6	Command R+	1194	+5/-5	17404	Cohere	CC-BY-NC-4.0	2024/3
6	GPT-4-0314	1189	+4/-3	41292	OpenAI	Proprietary	2021/9
8	Claude 3 Haiku	1182	+3/-4	50689	Anthropic	Proprietary	2023/8
9	GPT-4-0613	1164	+3/-3	60213	OpenAI	Proprietary	2021/9
9	Mistral-Large-2402	1158	+3/-4	35075	Mistral	Proprietary	Unknown
10	Qwen1.5-72B-Chat	1153	+4/-5	27050	Alibaba	Qianwen LICENSE	2024/2
10	Claude-1	1150	+5/-5	21868	Anthropic	Proprietary	Unknown
11	Command R	1149	+3/-3	31871	Cohere	CC-BY-NC-4.0	2024/3
11	Mistral Medium	1148	+4/-5	29807	Mistral	Proprietary	Unknown
15	Qwen1.5-32B-Chat	1138	+4/-6	12130	Alibaba	Qianwen LICENSE	2024/2

Note: we take the 95% confidence interval into account when determining a model's ranking. A model is ranked higher only if its lower bound of model score is higher than the upper bound of the other model's score. See Figure 3 below for visualization of the confidence intervals.

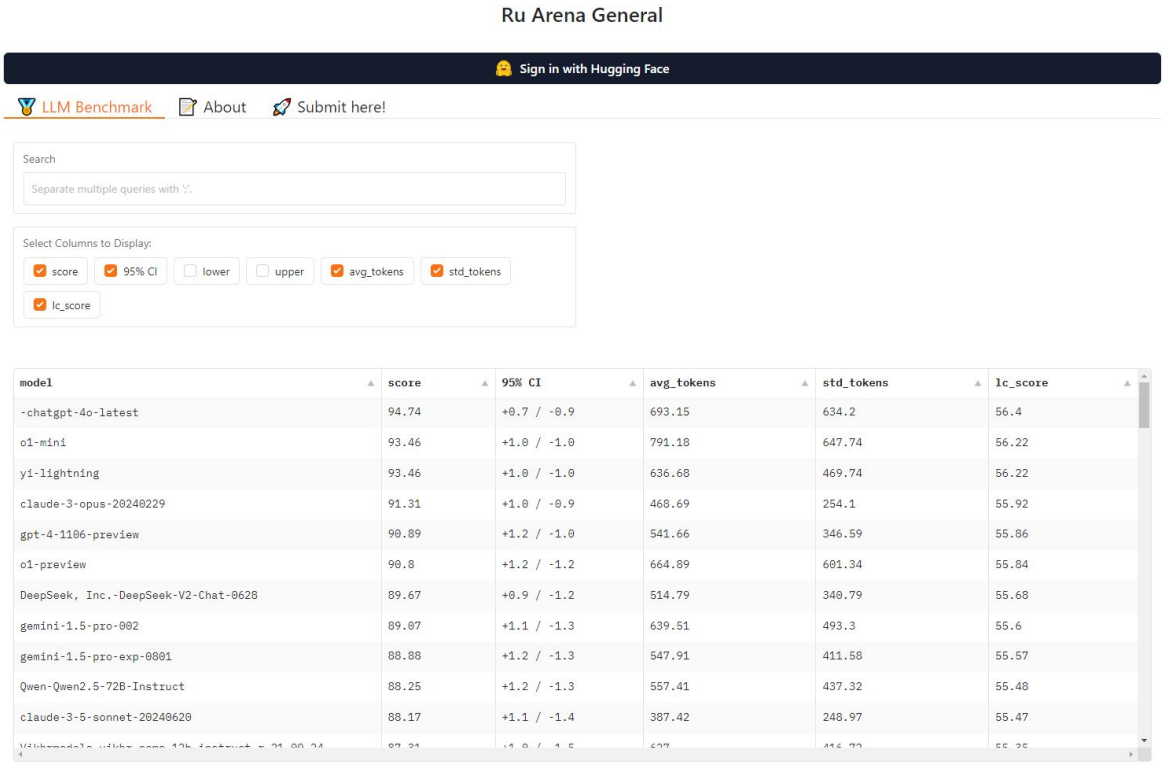
llmarena.ru



https://llmarena.ru/

Ru Arena General

- Тоже от Vikhr
- llm-as-judge
  - gpt4o
- Сравнивается против gpt-3.5



model	score	95% CI	avg_tokens	std_tokens	lc_score
-chatgpt-4o-latest	94.74	+0.7 / -0.9	693.15	634.2	56.4
o1-mini	93.46	+1.0 / -1.0	791.18	647.74	56.22
yi-lightning	93.46	+1.0 / -1.0	636.68	469.74	56.22
claude-3-opus-20240229	91.31	+1.0 / -0.9	468.69	254.1	55.92
gpt-4-1106-preview	90.89	+1.2 / -1.0	541.66	346.59	55.86
o1-preview	90.8	+1.2 / -1.2	664.89	601.34	55.84
DeepSeek, Inc.-DeepSeek-V2-Chat-0628	89.67	+0.9 / -1.2	514.79	340.79	55.68
gemini-1.5-pro-002	89.07	+1.1 / -1.3	639.51	493.3	55.6
gemini-1.5-pro-exp-0801	88.88	+1.2 / -1.3	547.91	411.58	55.57
Qwen-Qwen2.5-72B-Instruct	88.25	+1.2 / -1.3	557.41	437.32	55.48
claude-3-5-sonnet-20240620	88.17	+1.1 / -1.4	387.42	248.97	55.47

https://huggingface.co/spaces/Vikhrmodels/arenahardlb

lm eval

- Известный фреймворк для оценки LLM
- На нем основаны
  - MERA
  - shlepa
  - и др. англ. бенчмарки
- Имеет ряд достоинств и некоторые недостатки (которые со временем исправляют)
  - Нельзя нормально тестировать вариант, когда часть ответа модели уже дана
  - Сложный код с большим количеством legacy

https://github.com/EleutherAI/lm-evaluation-harness

llmtf

- Разрабатываемый мной фреймворк для оценки LLM
- Можно оценивать генерации, вероятность токена, вероятность последовательности
- В основе messages формат работы с моделями.
- Простой способ добавления своих task
- Информативные log файлы (но много весят)

https://github.com/RefalMachine/llmtf_open

Выводы

- Задача сравнения больших языковых моделей до сих пор большая проблема.
- Количество различных англоязычных бенчмарков велико (а какой лучший?), но русскоязычных почти нет.
- А что делать с конкретными предметными областями и задачами?

Выводы [2]

- Арены позволяют сравнивать модели максимально приближенно к усредненному качеству для задач среднего пользователя, но поддерживать их дорого и свои модели на них не потестировать.
  - Имея доступ к GPT-4 можно использовать MT-bench.
- Обычных же бенчмарков недостаточно, так как происходит переобучение на их задачи (иногда намеренное).

Домашнее задание 2

Цель задания:

На основе кода фреймворка llmtf реализовать какую-либо генеративную задачу и оценить с помощью нее 2 модели на выбор.

- Важно выбрать ту задачу, которую можно оценить на основе генерации LLM
- Например, NER, где модель обязана генерировать ответ в структурированном виде, который легко парсить

Домашнее задание 3

- Основные этапы задания:
  - Выбор датасета, разделение его на prompt/test части
  - Конвертация его в message формат или модификация load_dataset функции
  - Реализация основных функций: evaluate, aggregation, create_messages
  - Запуск на тестирование

Задание: оценивание и сроки

- Срок 1 неделя: до 16 марта 23:59.
- Присылать на tikhomirov.mm@gmail.com
  - Название письма: Основы LLM: Задание 2, llmtf task.
  - В письме Ваше полное ФИО, группа, решение и краткий отчет по нему в PDF.
- Оценка по шкале “-/-+/+-/+/++”.
  - ++ за те решения, которые особо мне понравятся чем-либо.