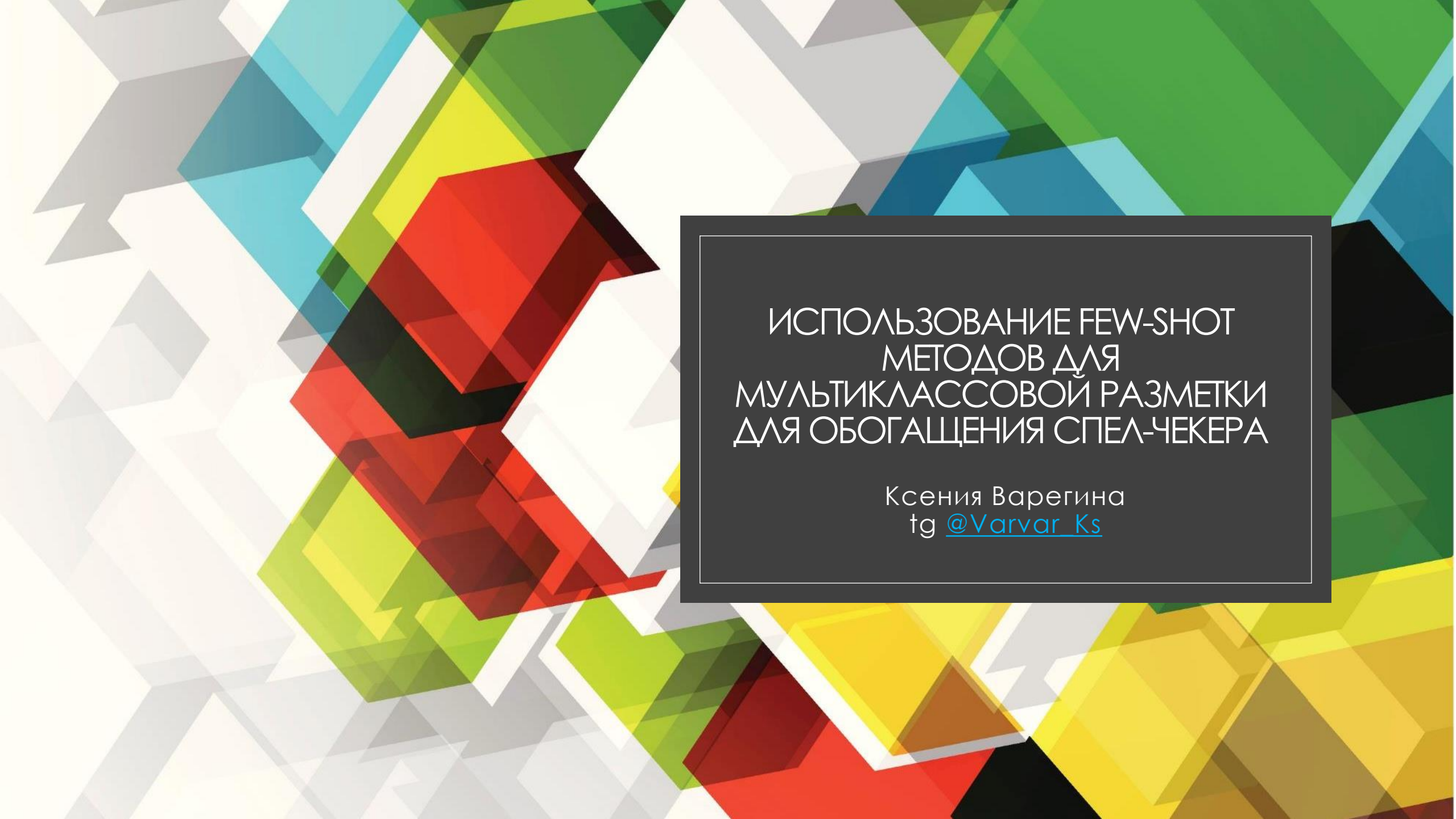


ИСПОЛЬЗОВАНИЕ FEW-SHOT МЕТОДОВ ДЛЯ МУЛЬТИКЛАССОВОЙ РАЗМЕТКИ ДЛЯ ОБОГАЩЕНИЯ СПЕЛ-ЧЕКЕРА

Ксения Варегина
tg @Varvar_Ks

Задача: отличать правильные незнакомые слова от ошибок

Стандартный подход к поиску ошибок в тексте: **выделить в нём слова, которых не знает модель**, основанная на большом словаре и умеющая склонять (спрягать) слова из своей базы.

Слова в тексте, с которыми такая модель незнакома, скорее всего будут неправильными — но не обязательно.

К какой задаче можно свести?

- Морфология (предсказывание морф информации)
- NER
- Классификация

Мы выбрали для себя путь NER – выделение классов слов, в том числе ошибочных.

1. Spacy
2. DeepPavlov
3. pullenti
4. stanza
5. Natasha

Существующие NER модели не показали результаты, которые можно было бы использовать.

Но можно собрать/разметить данные, чтобы дообучить существующий NER или получить данные для обогащения системы спел-чекера.

Какие есть данные?

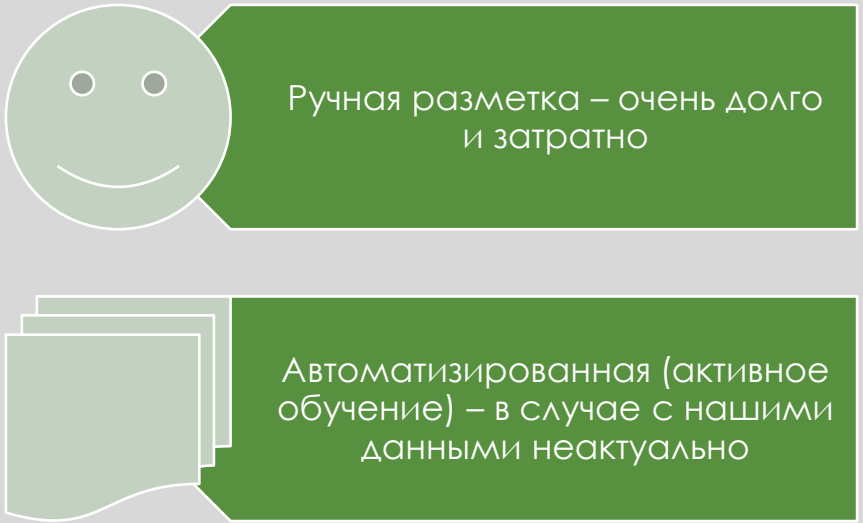
Файл, в котором исчислены все слова, которых ОРФО не знает

	N	Log	W	Dict_UD	dict.opcorpora	wiki_freq
0	367.0	-3.817249	хз	NaN	NaN	24
1	488.0	-3.909896	чо	NOUN an	PRCL Infr,Dist	351
2	538.0	-3.939917	зп	NaN	NaN	18
3	540.0	-3.940406	пикабу	NaN	NaN	NaN
4	579.0	-3.963836	как-будто	NaN	NaN	19

Кажется, все слова русского интернета (но точное происхождение неизвестно) +

Похожий файл со словами, который ОРФО знает

Что можно сделать?



Ручная разметка – очень долго и затратно

Автоматизированная (активное обучение) – в случае с нашими данными неактуально

Пайплайн

~~пошел не по плану~~ изменялся по ходу

Содержание

Здесь и далее демонстрация тетрадки

Этапы:

- Изучить промпт-инжиниринг
- Разобраться в подключении/использования API ChatGPT
- Подготовить тестовые данные
- Экспериментировать с промптами
- Прогнать тестовые данные с учетом ограниченности памяти контекста
- Оценить последовательность/точность разметки

Основные части промпта для few-shot

Подготовка примеров

```
examples = shuffle(examples)
errors = list(examples.loc[examples['tag'] == 'ERR']['word'])[:7]
neologisms = list(examples.loc[examples['tag'] == 'NEW']['word'])[:7]
slang = list(examples.loc[examples['tag'] == 'SLN']['word'])[:7]
person_names = list(examples.loc[examples['tag'] == 'PER']['word'])[:7]
companies = list(examples.loc[examples['tag'] == 'ORG']['word'])[:7]
normal_words = list(examples.loc[examples['tag'] == 'NORM']['word'])[:7]
geo = list(examples.loc[examples['tag'] == 'GPE']['word'])[:7]
print(*neologisms)
```

```
кремлеботская загуглить суицидницы обочинщика стендапы репостнул роадмап
```

Описание групп сущностей

```
groups_specifications = f"""Groups description, examples and tag:
1. Neologisms, occasional words, ```{neologisms}```, NEW
2. Organisations, companies, apps, ```{companies}```, ORG
3. Slang and swear words, ```{slang}```, SLN
4. Normal correct words, ```{normal_words}```, NORM
5. Misspelled, incorrect words, ```{errors}```, ERR
6. Surnames, ```{person_names}```, PER
7. Cities and countries, ```{geo}```, GEO"""
```

Как изменялся промпт

```
prompt = f"""

Your task is to determine which NER group Russian word should be classify every Russian word from the list and label the word with the group tag.

Do the classification based on ```{groups_specifications}``` , only the tags from specifications can be used.

The list of Russian words to label: ```{test}```

Pay close attention to errors (ERR), they include:

  - misspelled like 'сайах', 'могди'

  - imposible Russian words like 'мемяца', 'никогла'

  - words with unnecessary hyphen.

 Present results in csv format with '\t' separator with no heading

"""
```

Опасности:

- Непоследовательность разметки. Решение – прогон одного и того же фрагмента датасета несколько раз, сравнение результатов.
- Неточность разметки. Решение – ручная разметка датасета, оценка по стандартным метрикам.

Недочеты получения данных посредством ChatGPTd:

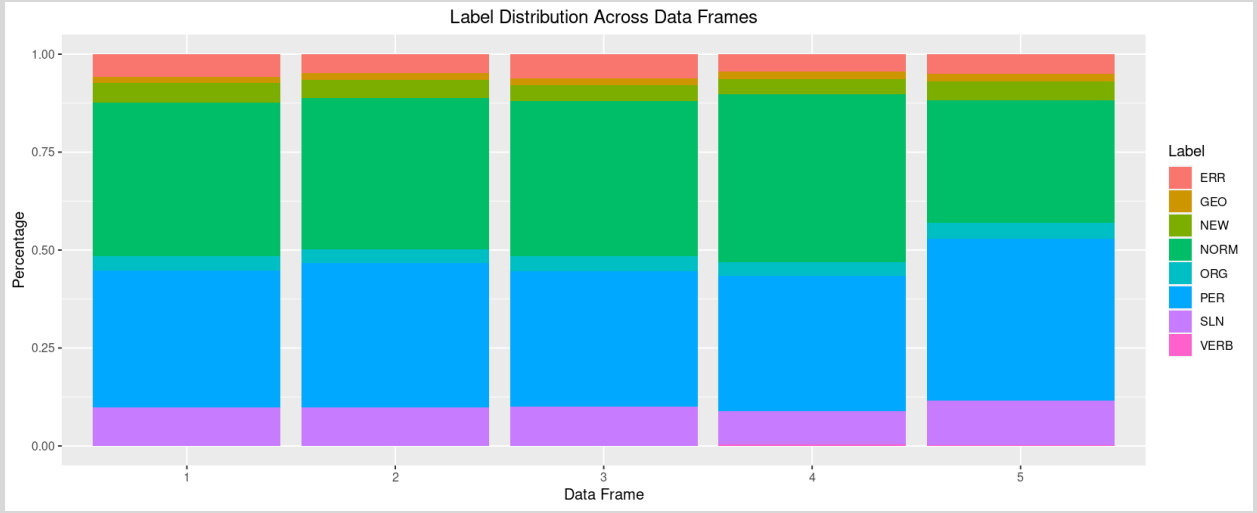
- Вывод результатов только в формате строки, нарушение форматирования итогового файла
- Потеря части разметки, нарушение последовательности в файлах

```
страданиями     Sотогреется     NORM
```

- Галюцинации

```
хапанули     VERB
```

Последовательна ли разметка?



8410 из 10000 слов попали в первые три файла.
7777 слов (92%), среди попавших в первые три выдачи, имеют одинаковые теги во всех случаях

```
Kappa score для result1 and result2: 0.8926768029577331,
  для result1 и result3: 0.9309155201639149,
  для result2 и result3: 0.9220555172532212
```

Точна ли разметка?

```
# по второму результату
get_metrics(annot_df, res_two)
```

```
Precision: 0.776652933919023
Recall: 0.6134699853587116
F1-score: 0.6416417720112945
```

А если допустить, что различить сленг и неологизм сложно (даже человеку), а название компании, например, может стать обычным словом – можно разделить все слова на правильные и нет (то, что и нужно для спел-чекера)

```
# приведем к бинарному виду

annot_df.loc[annot_df['label'] == 'ERR', 'label'] = 1
annot_df.loc[annot_df['label'] != 1, 'label'] = 0

res_one.loc[res_one['label'] == 'ERR', 'label'] = 1
res_one.loc[res_one['label'] == 'NA', 'label'] = 1
res_one.loc[res_one['label'] != 1, 'label'] = 0
```

```
[18] # Метрики ВТОРОГО результата для бинарных классов
     get_bi_metrics(annot_df, res_two)
```

```
Precision: 0.8260960208072362
Recall: 0.8228404099560761
F1-score: 0.7788255949034901
```

```
[23] # Метрики ПЯТОГО результата для бинарных классов
     get_bi_metrics(annot_df, res_five)
```

```
Precision: 0.8297625617891251
Recall: 0.8328358208955224
F1-score: 0.7983166091738105
```

Сложности

Экспериментальность
-
Гибкость разработки

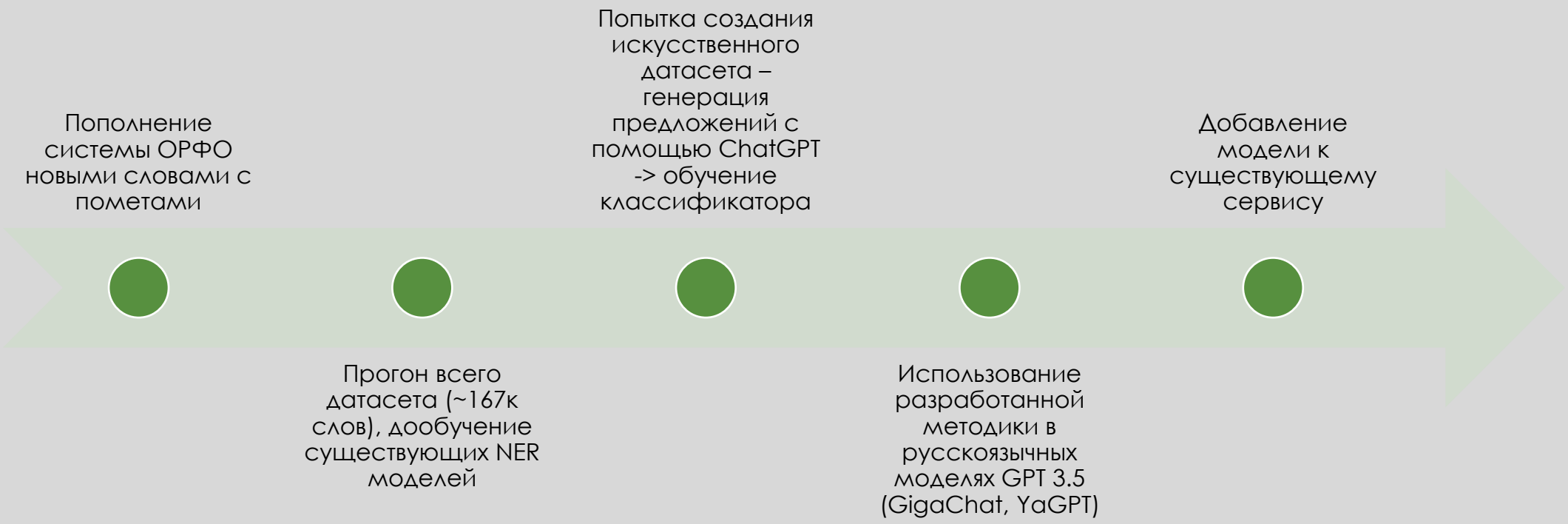
Стоимость
-
10000 слов = $0.70

Отсутствие точных метрик качества
-
“Which is better?” metrics

English LLM
-
Недостаток русских корпусов

Невозможность держать модель на своем сервере
-
Трудно/медленно использовать в проде

Дальнейшее развитие



Спасибо за внимание!

Ксения Варегина
kseniya.varegina@gmail.com
tg @Varvar_Ks

17.06.2023