Детоксификация текстов

В. Ганеева,
Я. Лабенская

Чего мы хотели:

На первом этапе работы:

1. Анализ данных
2. Препроцессинг
3. T5 и вариации
4. LSTM и вариации

Что мы делали:

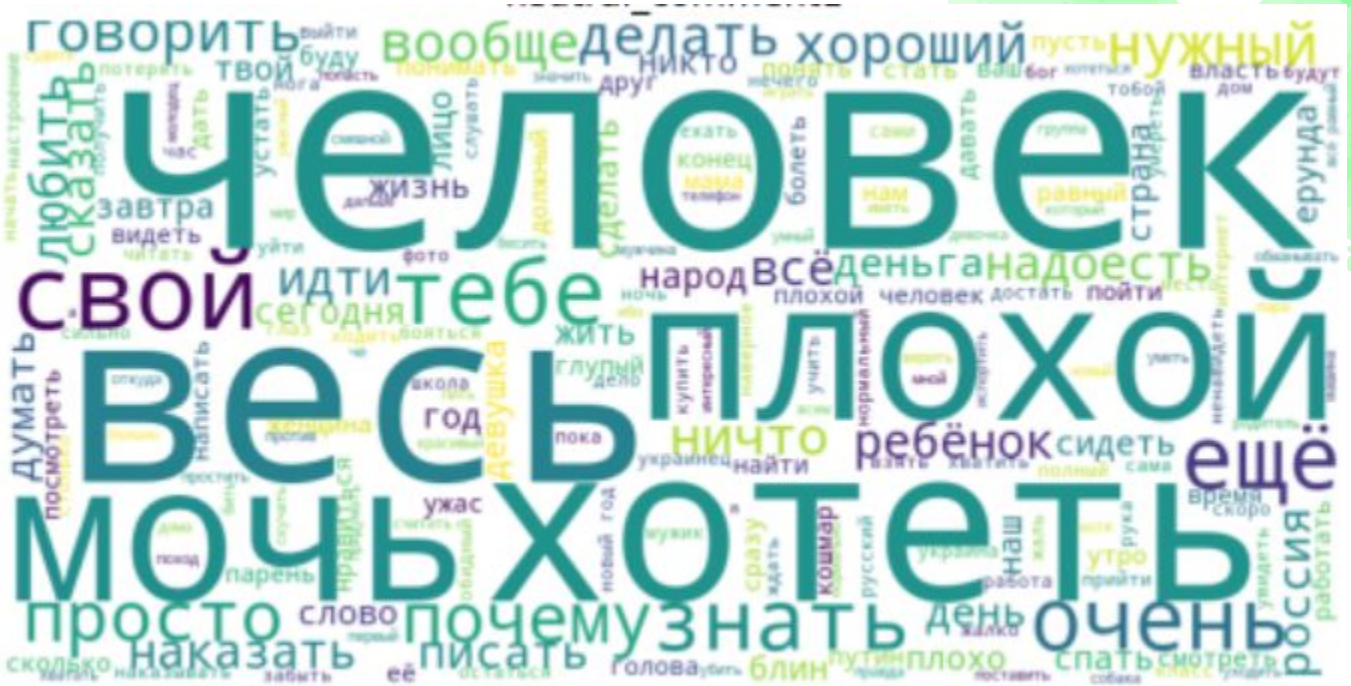
Процесс работы

1. **Сделали препроцессинг**. Столкнулись с проблемами: спеллчекер не знает мат. Токенизация и лемматизация очень плохо работают с особенностями комментариев вроде скобочек в конце слова с цифрами, пропущенных пробелов и текстовых смайликов.

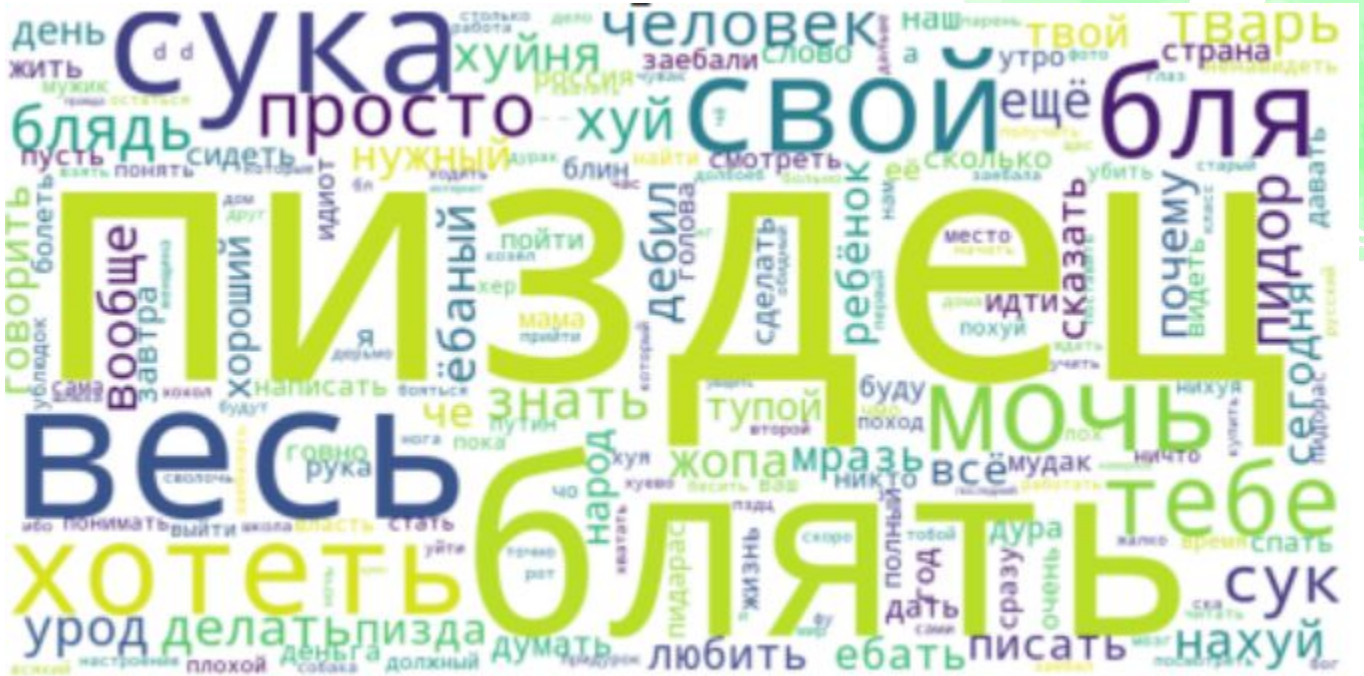
2. **Обучили rut5-base с разными параметрами**. Попробовали mt5, rut5-small и ещё несколько моделей. Проблема: модели слишком большие, чтобы обучать их локально, порой всё ещё большеваты для колаба без про-аккаунта, а также все эксперименты с ними прекращаются в тот момент, когда у нас заканчивается лимит времени.

3. **Для LSTM сложнее подготовить данные** в силу особенностей токенизации – можно пометить как “то, что нужно исключать” те слова, которые в детоксифицированном виде просто написаны по-другому.

Нейтральные комментарии



Токсичные комментарии



Какие бывают комментарии

1. **Полностью токсичные.**
   Те, в которых нам нужно заменять все слова, и не факт, что нужно просто заменить их нейтральными.

   “на х,,й твоя мамка хороша а ты сука рот свой поганый закрой”

2. **Частично токсичные**.
   Мат или оскорбление можно удалить без потери смысла.

   “О, а есть деанон этого петуха?” “О, а есть деанон?”

3. **Частично токсичные, но не оскорбительные**.
   Матерные слова можно заменить на нормативные синонимы, и оскорбительно не будет

   “упаси боже такую мать, которая ребёнка готова пиздить”

   “упаси боже такую мать, которая ребёнка готова ударить”

Проблемы базовых решений

Токенизация

Мы попробовали три разных варианта – stanza, udpipe и spacy. Результат не очень

```
'блятьпиздецлишьбыбалынабралисьсука',
```

```
preprocess('надо((сука')
```

```
(['надо((сука'], ['надо((сука'])
```

Словарный подход

Нам нужно выделять те слова,, которые в контекстах могут и быть оскорблениями, и нет

```
'голубые',
'пидорком',
'тарелку',
'уф',
'флагом',
```

Препроцессинг

Идея 1: нормализация

Стоит произвести обычную нормализацию текста - избавиться от знаков препинания, хэштегов и эмодзи

Идея 2: автокоррекция

В данных очевидно много опечаток, которые исправлены в нейтральных вариантах тех же комментариев - то есть, в нашем таргете. Можем ли мы подключить к этой системе автокорректор для исправления опечаток еще в рамках окончательного препроцессинга?

Нормализация

Textacy: надстройка над Spacy, позволяющая автоматически удалять лишние пробелы, пунктуацию и эмодзи

Токенизатор: NLTK (word_tokenize)

Автокоррекция

Вариант 1: модуль autocorrect, предобученный для русского языка

Вариант 2: модель rut5-small-normalizer: трансформер t5, обученный на русском языке и дообученный для задачи автокоррекции

t5

Ввод [244]:
```
number = 210
print('original: '+str(df['toxic_comment'][number]))
print('autocorrect module: '+str(spellcheck(df['toxic_comment'][number])))
print('t5: '+str(t5_autocorrect(df['toxic_comment'][number])))
```

```
original: Жириновский очень точно сформулировала лозунг отечественного ресентимента.
autocorrect module: Жириновский очень точно сформулировала лозунг отечественного ресентимента.
t5: Жириновский очень точно сформулировал лозунг Отечественного Ресертмента.
```

t5

Ввод [424]:

```
number = 1680
work_str = str(preprocess(df['toxic_comment'][number]))
print('original: '+str(df['toxic_comment'][number]))
print('original-preproc: '+work_str)
print('autocorrect module: '+str(spellcheck(work_str)))
print('t5: '+str(t5_autocorrect(df['toxic_comment'][number])))
```

```
original: какая блядь сейчас бруснику собирает руки отрывать за это надо
original-preproc: какая блядь сейчас бруснику собирает руки отрывать за это надо
autocorrect module: какая блядь сейчас бруснику собирает руки отрывать за это надо
t5: Какая блядь сейчас собирает руки, отрывать за это надо?
```

t5

Ввод [362]:

```
number = 1080
work_str = str(preprocess(df['toxic_comment'][number]))
print('original: '+str(df['toxic_comment'][number]))
print('original-preproc: '+work_str)
print('autocorrect module: '+str(spellcheck(work_str)))
print('t5: '+str(t5_autocorrect(work_str)))
```

```
original: а значит и драный их батька такой-же фашизюка, если не пресёк это
original-preproc: а значит и драный их батька такой же фашизюка если не пресёк это
autocorrect module: а значит и драный их батька такой же фашизма если не пресёк это
t5: А значит, такой же батька сидит на этой страшнейшей массизе.
```

Словарь бейзлайна

вымандившаяся
упиздяшивающую
помандяхивавшее
обмандохивался
отмандимтесь
припиздюривавшее
выхуякивавши
напиздошивающая
припиздиться
отпиздякающий
распиздяшимся
пропизживающеюся

Дообученный autocorrect

Ввод [621]:

```
number = 3240
work_str = str(preprocess(df['toxic_comment'][number]))
print('original: '+str(df['toxic_comment'][number]))
print('original-preproc: '+work_str)
print('autocorrect module: '+str(spellcheck(work_str)))
print('autocorrect updated: '+str(spellcheck_updated(work_str)))
print('t5: '+str(t5_autocorrect(df['toxic_comment'][number])))
```

```
original: козел. это же и дети увидят. и его дети в том числе. вырастут такими же ублюдами.
original-preproc: козел это же и дети увидят и его дети в том числе вырастут такими же ублюдами
autocorrect module: козел это же и дети увидят и его дети в том числе вырастут такими же блюдами
autocorrect updated: козел это же и дети увидят и его дети в том числе вырастут такими же ублюдками
t5: Козел. Это же какие-то люди увидят, чтобы детей увидят. И его детям в том числе вырастут такими же Ублюдами.
```

Дообученный autocorrect

Ввод [617]:

```
number = 3200
work_str = str(preprocess(df['toxic_comment'][number]))
print('original: '+str(df['toxic_comment'][number]))
print('original-preproc: '+work_str)
print('autocorrect module: '+str(spellcheck(work_str)))
print('autocorrect updated: '+str(spellcheck_updated(work_str)))
print('t5: '+str(t5_autocorrect(df['toxic_comment'][number])))
```

```
original: не кукла а мумия молью трахнутая
original-preproc: не кукла а мумия молью трахнутая
autocorrect module: не кукла а мумия молью тронутая
autocorrect updated: не кукла а мумия молью трахнутая
t5: А кукла не молью трахнутая.
```

Предварительное сравнение

	STA	SIM	FL	J
Baseline (t5)	0.753606	0.805467	0.816638	0.504346
Preproc (+punct)	**0.830605**	0.737583	0.759188	0.478918
Preproc (-punct)	0.753747	**0.805508**	**0.816859**	**0.504625**

Проблемы: почему падает SIM?

Ориг: своих увидела и голосок прорезался ? мыышь серогорбая))))
База: своих увидела и голосок прорезался? мыышь серогорбая))))
Препроц: своих увидела и голосок прорезался

Ориг: и в 3 подъезде бывает такое. поймать бы чухана!
и в 3 подъезде бывает такое. поймать бы его!
Препроц: и в 3 подъезде бывает такое поймать бы чулана

Ориг: убить суку, это не отец
База: Это не отец
Препроц: Наказать этого человека это не отец

Ориг: вискаря въебу и расскажу тебе что почём
База: вискаря ударю и расскажу тебе что почём
Препроц: викария въебу и расскажу тебе что почём

T5 и вариации

mT5.
Почему нет – cuda error даже в колабе с оптимизацией

ru-T5 base
Как и в бейзлайне организаторов, но на других настройках. Спойлер: стало хуже

ru-T5 small
Мы надеялись, что она даст нам похожий результат с меньшими затратами памяти. Нет, лучше не стало, мы получили не очень хороший процесс обучения на наших данных

T5-paraphrase
Мы с параметрами организаторов и сейчас возлагаем на её тюнинг наибольшие надежды

ru-t5 small

Модель номер 3: ru-t5 small, мы надеялись, что она даст нам похожий результат с меньшими затратами памяти. Нет, лучше не стало, мы получили не очень хороший процесс обучения на наших данных

```
epoch 0, step 100/100: train loss: 2.7762  val loss: 15.4076
epoch 0, step 200/200: train loss: 2.6242  val loss: 15.2856
epoch 0, step 300/300: train loss: 2.4576  val loss: 15.2085
epoch 0, step 400/400: train loss: 2.3768  val loss: 15.4989
epoch 0, step 500/500: train loss: 2.2971  val loss: 15.2845
epoch 0, step 589/589: train loss: 2.2559  val loss: 15.1246
```

t5-paraphrase

По обучению всё вроде выглядит неплохо, но результат получается не тот, который мы могли бы ожидать

```
epoch 1, step 1600/11580: train loss: 1.0332  val loss: 1.0315
epoch 1, step 1700/11680: train loss: 0.9346  val loss: 1.0483
epoch 1, step 1800/11780: train loss: 1.0792  val loss: 1.0275
epoch 1, step 1900/11880: train loss: 1.1330  val loss: 1.0284
epoch 1, step 2000/11980: train loss: 1.0519  val loss: 1.0404
12000 12000
```

```
paraphrase('вы чо курите блять ?', model, temperature=0.0)
```

```
'Ну бля'
```

LSTM и вариации

Данные для обучения

1 – слова, которые не встретились в обработанных комментариях

```
(['О', ' ', 'а', 'есть', 'деанон', 'этого', 'петуха'],
 ['0', '0', '0', '0', '0', '1', '1'])
```

Результат на четырёх эпохах

```
lstm_crf_get_answer('ты сука рот свой поганый закрой пидарас нашёлся')
```

```
[1, 1, 1, 1, 1, 1, 1, 1]
```

```
lstm_crf_get_answer('кот лежит на столе')
```

```
[0, 0, 0, 0]
```

Предварительное сравнение

	STA	SIM	FL	J
Baseline (t5)	0.753606	0.805467	0.816638	0.504346
Baseline-preproc (+punct)	**0.830605**	0.737583	0.759188	0.478918
Baseline-preproc (-punct)	0.753747	**0.805508**	**0.816859**	**0.504625**
ruT5paraphraser(-preproc, 3000 steps)	0.554153	0.390466	0.610752	0.061559
ruT5paraphraser(+preproc, 12000 steps)	0.52703	0.391150	0.616564	0.061009

Дальнейшие планы:

1. **Мы запустили разные модели** и осознали, что мы можем их обучить. При этом нас подводит настройка параметров обучения: наши модели выглядят нормально по лоссу, но при этом плохо справляются с задачей. Что это значит: вероятно, нам необходимо изучить тему замораживания слоёв в t5 и также попробовать добавить эпох обучению и ещё изменять параметры. Нам необходимо лучшее решение для дальнейшей экстракции

2. **Мы получили данные для LSTM** и прочих генераций последовательностей. Нужно дообучить CRF-LSTM на пять-семь эпох и попробовать другие модели в этом месте. Здесь мы соревнуемся со словарным бейзлайном

3. **Нам нужно запустить FELIX.** Это иная архитектура, и параллельно с первой задачей мы можем обнаружить, что эта модель подходит нам лучше: она изменяет последовательность, а не генерирует новую

Спасибо за внимание!

```
print(paraphrase(['пиздец блять'], model, temperature=50.0, beams=20))
```

```
['У меня пиздец какой пиздец, блять.']
```