Поиск контактной информации в объявлениях

Андрей Егоров

Общий план решения

- Решив вторую, решаем первую почти автоматом => начал со второй
- Вторую задачу свел к NER на тексте с частичным adversarial obfuscation
  - Трансформер как текущий стандарт для задач NER
- Объекты упрощаем до поля description
  - Удобно и дает достаточное качество

NER

Препроцессинг

- Оставили только символы стандартной пунктуации и те, которые могут встретиться в контактной информации (@, #, / и тд)
  - Более осмысленное разделение на предложения
  - Уменьшение количества токенов внутри предложений
- Разделение на предложения - razdel проекта natasha
  - Стандартные BERT модели не могут обработать строки длиной более чем 512 токенов (300-400 слов)

Количество предложений в объекте

- Какие-то объекты состоят из одного предложения (назовем простыми), какие-то - более чем из одного (назовем составными). Ряд наблюдений:
  - Контактная информация в простых и составных объектах не отличается (те же приемы)
  - В простых объектах лучше соотношение сигнал / шум: в составных не понятно, в каких именно предложениях находятся контакты
  - Научившись работать с простыми объектами, на инференсе легко обобщить на сложные, если представить их как набор простых
- Начать работу следует с простых объектов - и их оказалось достаточно

Разметка и датасет

- Решил потратить силы на разметку:
  - Создание качественного датасета под задачу всегда окупается; часто все равно приходишь к такой необходимости
  - Альтернативой могло бы быть решить классификацию и посмотреть активации через captum - решил не рисковать
  - Разметка простых объектов достаточно ненапряжна, много готовых разметчиков (prodigy, doccano, ner-annotator и тд)
- Датасет clad_ner (CLassified ADs):
  - 8000 объектов, содержащих named entities в кодировке BIO, и 24000 “пустых” (для защиты от ложных срабатываний, чтобы модель хорошо научилась работать с классом 0)
  - Чтобы в дальнейшем потенциально можно было провести более глубокий анализ, выделил 3 класса: phone, phone-adversarial, other-contact-info
  - Подкласс transformers.dataset - доступна вся функциональность

Обучение модели NER

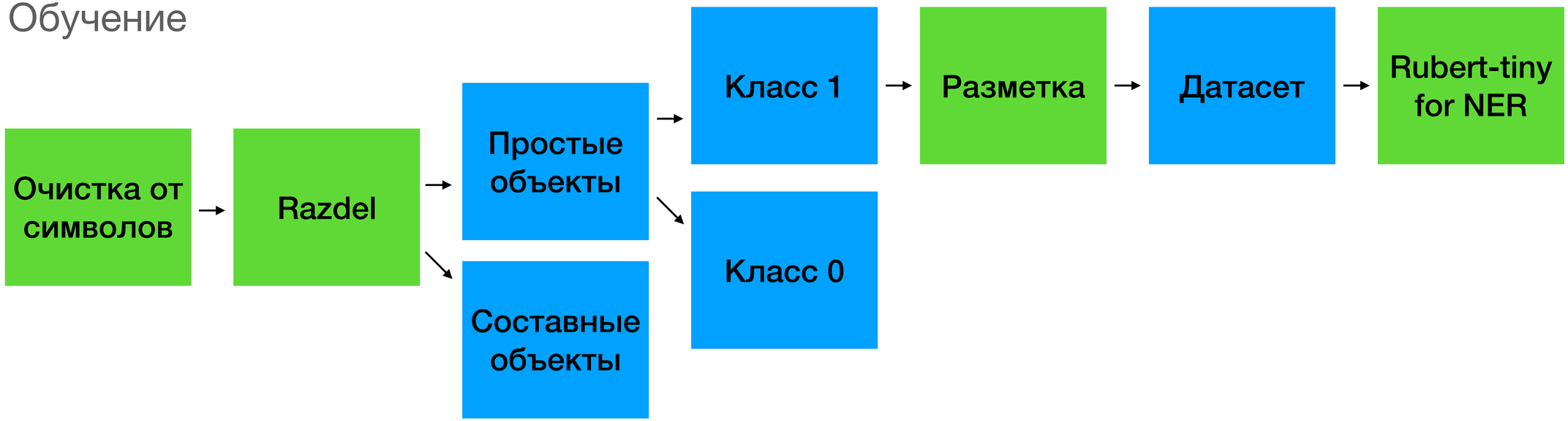
- Стандартные процедуры библиотеки Transformers
- RuBERT - отличное качество, но не влезал в ограничения времени на inference
- Остановился на rubert-tiny (num_epochs=10, AdamW, lr=2e-5, weight_decay=0.01)
  - Лучшее качество без переобучения в районе 4 эпохи

Заключительный штрих

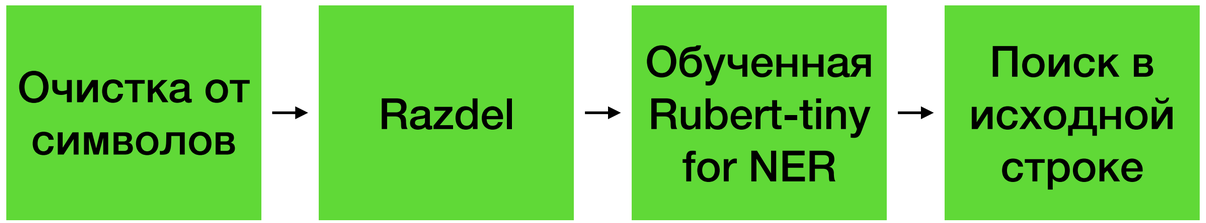
- Наш очищенный текст - своего рода “спрямляющее пространство”
- Решив задачу в нем, надо перейти обратно в исходное, чтобы найти точные координаты
  - Небольшой дополнительный фильтр - если в найденном меньше 5 символов, считаем, что ничего не нашли
- Найдем, где в исходной строке символы в том же порядке. При этом между ними могут быть удаленные символы (для простоты - от 0 до 9 любых символов)
- 5 и 9 выступают в качестве гиперпараметров, подобраны на валидации

Декомпозиция NER (recap)

Обучение



Инференс



Классификация

Препроцессинг и объекты

- Модель для NER явно уже неплохо осведомлена о задаче - хотелось бы переиспользовать ее знания, взяв веса без token classification головы
- Одинаковый препроцессинг
- Простые/составные объекты:
  - В составном не понятно, за счет каких предложений весь объект получил метку 1
  - Если разбить сложные на простые и назначить всем метку исходного сложного, модель получит на вход много противоречивого шума
- Снова попытаемся начать с простых объектов и найти способ свести результат воедино для сложных, представив их как набор простых

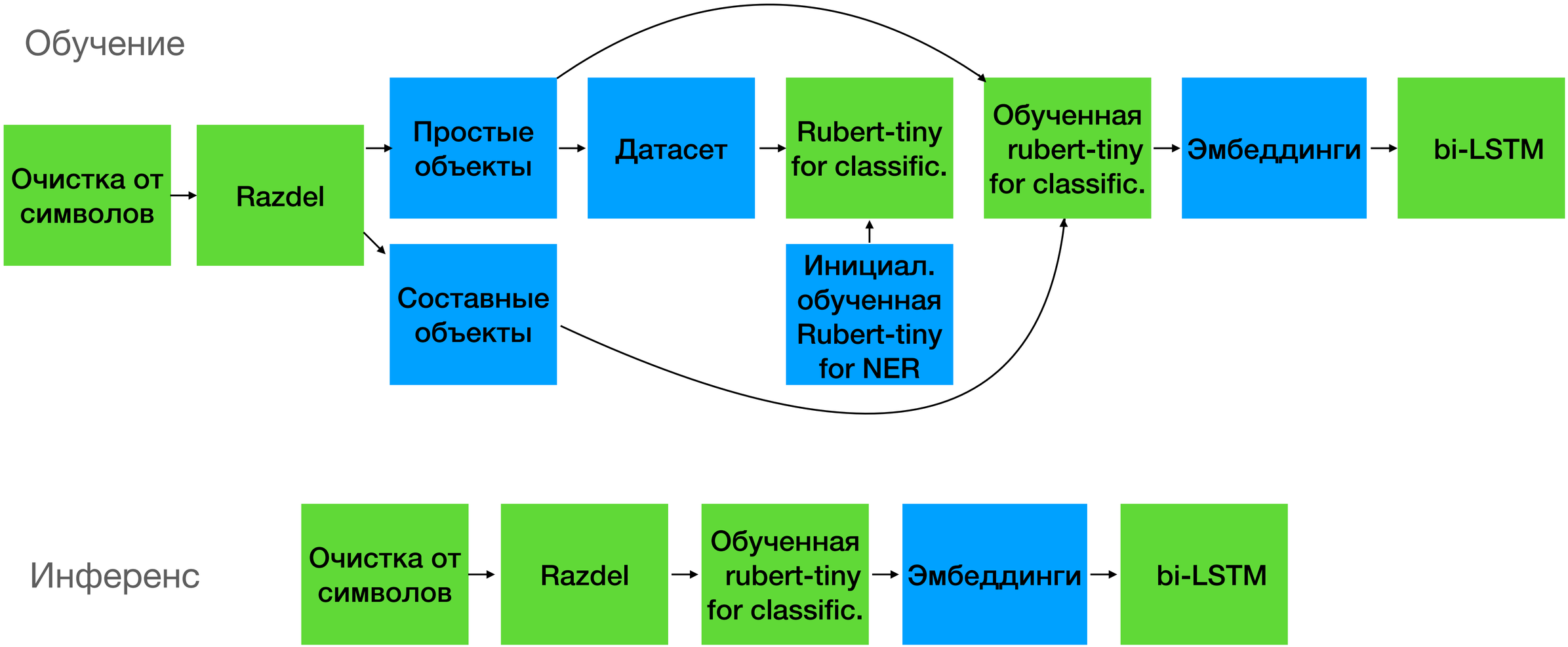
Датасет и обучение

- Датасет clad_class: 100K объектов
  - 50K простых объектов каждого класса - ограничение по числу простых объектов класса 1
- Rubert-tiny (num_epochs=3, AdamW, lr=2e-5, weight_decay=0.01)
  - Лучший результат достигается быстро - уже после 2 эпохи возникает переобучение

Работа с составными объектами

- Свести воедино уже задача сложнее - нет очевидного способа собрать вместе классификации отдельных предложений в составном объекте
  - Простые вещи, вроде максимума, не тянут по метрике
- Выходы rubert-tiny до классификационной головы использованы как эмбеддинги, таким образом новые объекты - последовательности эмбеддингов переменной длины
  - Эмбеддинги получены на сбалансированной выборке по категориям объявлений (всего около 120K объектов), с бОльшим акцентом на сложные категории (oversampled)
- Поверх них обучена bi-LSTM (15 эпох, AdamW, lr=1e-3, lr scheduler)

Декомпозиция классификации (recap)



Анализ потенциальных недостатков

- Потенциальные недостатки:
  - Классификационная модель не end-to-end
  - Разметка - накладные расходы
  - Задействована не вся исходная обучающая выборка
- Что можно было бы попробовать:
  - Не очищать от символов - может какие-то были важными признаками того, что подстрока не является контактной информацией
  - Подход при помощи captum
  - Модель без ограничения на длину последовательности

Спасибо за внимание