Natural Language Processing course

(stream 6, spring 2024)

Отчет

Сравнительный анализ результатов работы алгоритмов открытых библиотек Python для распознавания адресов на примере сообщений о ДТП и ЧС в TG-каналах.

Подготовил: Павлов Д.Н.

2024 г.

ГИБДД. Использования данного канала может дополнить картину ситуации на улицах города, используя относительно дешевые технические решения.

Разработка сервиса для поиска сообщений о ДТП в социальных сетях подразумевает решение ряда задач: определение перечня групп и каналов, разработка парсера для считывания и разбора сообщений, выделение ключевых слов или тематик, которые необходимо отслеживать, классификация инцидентов и определение географических координат инцидента, интеграция в существующие решения для формирования тревожных сообщений операторам и диспетчерам системы ИТС. В нашей работе мы сосредоточимся на узкой задаче – определение адреса инцидента. Целью работы является проверка возможности стандартными средствами библиотек решить задачу извлечения адреса из текстов с сообщениями об инцидентах в телеграмм-каналах.

Подготовил Павлов Д.Н.

Введение.

С декабря 2018 г. в Российской Федерации принят и активно развивается национальный проект «Безопасные и качественные автомобильные дороги», целью которого является приведение автомобильных дорог регионального значения и дорожной сети городских агломераций в нормативное состояние. В рамках национального проекта «Безопасные и качественные автомобильные дороги» предусмотрены мероприятия по внедрению интеллектуальных транспортных систем (ИТС) в городских агломерациях с населением от 300 000 человек. Одним из элементов региональной ИТС является подсистема детектирования ДТП и ЧС.

Эта одна из ключевых подсистем, которая позволяет не только оперативно управлять транспортными потоками, но и служит дублирующим источником данных о ДТП и ЧС в городе для оперативных служб.

Определение и фиксация техническими средствами факта ДТП чаще всего решается методами машинного зрения с использованием тысяч видеокамер, установленных на улично-дорожной сети. Наиболее известными разработчиками коммерческих решений являются: ООО «ТраффикДэйта» (г. Пермь), ООО «РоадАР» (Татарстан), ООО ВойсЛинк (г. Москва), ООО "НПО "ИТС. СОФТ" (г. Москва), ООО "УРБАНТЕХ-ИТ" (г. Москва) и ряд других. Вместе с тем, у похода детектирования ДТП с использованием видеоаналитики есть ряд существенных недостатков: высокие вычислительные требования к edge-устройствам или к серверным мощностям, ограниченные зоны видимости камер, значительная стоимость внедрения и обслуживания подобных систем. Поиск альтернативных источников данных является крайне актуальной задачей.

Развитие социальных сетей расширило не только возможности по быстрой передачи сообщений при коммуникациях между людьми, но и предоставляет инструмент быстрого сбора критичной информации по разным направлениям. Практически в каждом городе есть телеграмм-канал или группа в сети «VK», посвященная теме ДТП, инцидентам на дороге, в которых пользователи оперативно выкладывают сообщений об авариях, инцидентах, заторах и мобильных постах

1. Обзор работ.

Задача извлечения именованных сущностей (Named Entity Recognition или NER) – одна из ключевых задач в сфере обработки естественного языка, заключается в автоматическом распознавании именованных сущностей в тексте, таких как имена людей, названия организаций, адреса, географические названия и имена других значимых объектов. Решение этой задачи включает в себя два основных этапа: обнаружение именованных сущностей в тексте и определение их класса (например, имя человека, название города и т.д.).

В настоящее время существует немало обзорных работ по поиску и извлечению адреса из текста. Основными задачами таких работ авторы обозначают сравнение результатов работы библиотек с открытым исходным кодом, иногда на примере конкретного корпуса текстов. Например, сравнение работы open-source Питон библиотек для решения задачи распознавания именованных сущностей, в которой рассматриваются возможности и даны примеры для работы со следующими библиотеками: NLTK, StandfordCoreNLP, Spacy, Flair, DeepPavlov, deepmipt/ner, Polyglot, AdaptNLP, Stanza, AllenNLP, HanLP, PullEnti, Natasha, ner-d, [1]. Еще один анализ работы open-source библиотек на языке Python для решения задачи распознавания именованных сущностей [2]. Даны результаты использования для библиотек Stanza, Natasha, Pullenti. Встречаются и обзоры более редких библиотек или сервисов Apache NlpCraft, OpenNlp [3].

Для решения задачи поиска и извлечения адресных структур используются различные подходы, включая:

1. Использование предобученных моделей: современные модели глубокого обучения, такие как GPT и его вариации, показывают высокую эффективность в решении задачи NER. Эти модели предварительно обучаются на больших объемах текстовых данных и затем могут быть адаптированы под конкретные задачи.
2. Применение правил и шаблонов: создание правил и шаблонов для идентификации именованных сущностей может быть эффективным подходом, особенно когда речь идет о специфических классах объектов, таких как адреса,

особенно если они извлекаются из относительно формальных документов – справочников, анкет, договоров и т.д.

3. Использование контекстной информации: учет контекста вокруг анализируемых слов помогает улучшить точность распознавания. Например, слово «Пушкин» может быть именем человека, названием города или названием клуба, в зависимости от контекста. Использование предикатов адресных структур («город», «район», «улица») важно для точного распознавания адреса, так как без них появляется необходимость в сравнении адреса со справочными данными, например «Саратовская» может быть улицей, станицей или станцией

4. Лемматизация и стемминг: Преобразование слов к их базовой форме (лемме) или удаление окончаний (стемминг), может упростить процесс распознавания, поскольку уменьшает разнообразие форм слов.

5. Интеграция с другими задачами NLP: задача NER часто решается в рамках более широкой задачи обработки естественного языка, где используются различные методы и подходы для улучшения качества распознавания.

Выбор подхода зависит от специфики задачи, доступных данных и ресурсов. Но чаще всего приходиться совмещать работу сразу нескольких подходов, например в статье [4] описывается тестирование четырёх типов моделей NER, включая DeepPavlov. Перед обработкой данных применялась лемматизация. Использовались регулярные выражения. Результаты сравнивались с данными из готового справочника объектов (в данном случае торговых брендов). Для выбора окончательного ответа применялся принцип голосования. Для решений, ориентированных на рынок РФ по-прежнему актуальны работы по извлечению адреса через сравнение с БД ФАИС (Федеральная информационная адресная система). В работе [5] предлагаются два алгоритма получение адреса путем сравнения с БД ФАИС последовательно или одновременно. Последовательное сравнение – каждый элемент анализируется в очередности его появления. Одновременное сравнение предполагает предварительную очистку от префиксов адресных структур и поиск по всем выявленным объектам сразу. Следует отметить, что предобработка данных, как и для любой задачи аналитики, является значимым этапом на пути разработки успешного продукта. Ряд авторов особым образом обрабатывают документы и сводят их к единому формату. В статье [6] описывается кастомный алгоритм поиска адресов в документах разного типа. На первом этапе происходит сборка адресов из excell, doc, pdf в датафрейм пандас. Далее каждая строка с определенным столбцом, обрабатывается на наличие адресов с помощью регулярных выражений и проводится сравнение использование данного алгоритма с функцией NER NATASHA. Другие авторы активно дорабатывают функционал библиотек. Например, возможны доработки стандартных функций библиотеки Natasha для выполнения поиска адресов путем модификации механизма работы Yargy парсера для обработки специфических правил характерных для пользовательского корпуса текстов [7].

Обращают на себя внимание работы, в которых описывается итерационных подход для обработки и извлечения сущностей, при котором последовательно обрабатываются и устраняются все ошибки и слабые стороны моделей, например опыт разработка собственной NER модели для распознавания брендов [8]. Разметка текстов проводилась в несколько этапов с использованием предобученных моделей и труда разметчиков. Дополнительно были проанализированы особенности наименования брендов, состоящих из нескольких слов или символов. После первичного обучения была проведена дополнительная разметка для тех брендов, которые не распознавались или распознавались с большими ошибками.

Таким образом, извлечение адресной информации и информации о местоположении является актуальной задачей для разных предметных областей. При практическом решении данной задачи активно используются готовые инструменты (библиотеки с предобученными языковыми моделями), разрабатываются специализированные алгоритмы и механизмы обработки текстов, активно используется итерационный подход к разработке.

2. Описание методов и инструментов (библиотек).

Для поиска и выделения адресов в тексте в основном используются готовые библиотеки обработки естественного языка, такие как Spacy, Stanza, Pullenti, Natasha и др. Они могут распознавать различные типы именованных сущностей, включая географические объекты и адреса. Эти библиотеки используют модели, обученные на больших наборах данных, что увеличивает их точность. Также, они могут быть настроены для работы с разными языками и могут обрабатывать и анализировать неструктурированные тексты "как есть". Несмотря на высокие показатели распознаваемости, которые заявляют разработчики библиотек, часто, в зависимости от специфических требований задачи, полезно произвести обучение собственных моделей NER для выделения адресов. Сравнение результатов работы предобученных моделей «из коробки» позволит выявить слабые места готовых решений, понять специфику стратегий обозначения адресов в конкретном домене, послужившим источником формирования корпуса текстов.

Для изучения результатов обработки были выбраны 4 библиотеки - Spacy, Stanza, Pullenti и Natasha, две из которых изначально создавались для решения задач обработки русского языка, а одна имеет специализированный модуль обработки российский адресов.

2.1. Библиотека Natasha.

Проект Natasha — набор Python-библиотек для обработки текстов на естественном русском языке [9]. Библиотека предназначена для решения ряда задач обработки естественного языка: сегментация, морфология, синтаксис, NER. Разработчики позиционируют библиотеку, как предназначенную для коммерческой разработки с ориентаций на практичность и компактность. К особенностям следует отнести работу только с текстами на русском языке. Основной фреймворк – PyTorch.

Для поиска адресов и географический названий использовались два подхода:

1) с использованием модели SlovnetNER получаем список сущностей [10],

2) с использованием парсера Yargy получаем словарь извлеченный структурированной информации для объектов категории ‘LOC'.

Разработчики отмечают, что Slovnet NER модель представляет собой высокачественную модель для решения задач NER [11] с качеством на 1-2 % ниже, чем текущая SOTA модель от DeepPavlov, но в 60 раз компактнее и более быстрая на CPU.

Yargy-парсер использует правила для извлечения сущностей, описываемые с помощью контекстно-свободных грамматик и словарей [12]. Особо следует отметить, что есть возможность создания собственных правил для решения конкретных специфических задач извлечения сущностей.

2.2. Библиотека Stanza.

Stanza - это мощная библиотека для обработки естественного языка, разработанная университетом Стэнфорд [13]. Она включает в себя ряд инструментов для поиска именованных сущностей. Stanza обучается на больших наборах данных, что сильно увеличивает точность извлечения.

Stanza поддерживает большое количество языков, что позволяет обрабатывать и анализировать тексты на разных языках. На текущий момент доступны предобученные модели для 80 языков. Однако, для задачи NER доступы только модели для 29 языков. Библиотека Stanza также предоставляет возможность обучать собственные модели NER, что может быть полезно для уникальных или специфических задач.

Все вышеперечисленное делает Stanza универсальным инструментом для обработки данных. Библиотека использует фреймворк PyTorch.

2.3. Библиотека Spacy.

Spacy - это популярная библиотека для обработки естественного языка, которая широко используется для различных задач, включая извлечение именованных сущностей (NER) [14]. Она предоставляет мощные инструменты для создания моделей машинного обучения, обученных на больших наборах данных. Spacy поддерживает более 75 языков, в том числе русский, и может распознавать различные типы именованных сущностей, такие как имена людей, организаций, местоположений и дат. Для решения задачи поиска наименований и адресов использовалась предобученная модель для русского языка «ru_core_news_sm».

2.4. Библиотека Pullenti.

Pullenti - это высокоэффективная библиотека для обработки естественного языка, которая специализируется на извлечении именованных сущностей [15]. Изначально разработанная для работы с русским языком, Pullenti теперь поддерживает и другие языки.

Одной из ключевых особенностей Pullenti является ее способность распознавать широкий спектр типов сущностей. Она может идентифицировать имена, даты, организации, географические объекты и множество других типов сущностей, что делает ее универсальным инструментом для различных задач обработки языка.

Pullenti особенно полезна для анализа неструктурированных текстов. В отличие от некоторых других библиотек обработки языка, которые могут потребовать предварительной подготовки или структурирования данных, Pullenti способна обрабатывать и анализировать данные "как есть". Это может значительно упростить процесс обработки больших объемов текста.

Кроме того, Pullenti может быть использована в различных областях, от информационного поиска до аналитики и автоматизации бизнес-процессов. Она предоставляет мощные инструменты для извлечения ключевой информации из текста. Используя Pullenti можно значительно улучшить эффективность и точность извлечения именованных сущностей.

Особо отметим, наличие специализированного компонента - SDK Pullenti Address, отвечающего за выделение из текста адресов и их привязки к объектам ГАР ФИАС.

Для решения поставленной задачи с использованием сторонних библиотек предполагалось реализация следующих мероприятий:

- сбор корпуса текстов с сообщениями об авариях и инцидентах на дорогах из телеграмм-каналов: необходимо собрать не менее 1000 сообщении из 3 и более телеграмм-каналов, в которых преимущественно публикуется информация о дорожно-транспортных происшествиях и других инцидентах на дорогах;
- ручное выделение элементов текста, относящихся к адресам, наименованиям локаций, по которым можно определить место инцидента;
- анализ корпуса сообщений;
- обработка корпуса текстов с использованием инструментов специализированных NLP библиотек и анализ результатов;
- выработка рекомендаций для формирования оптимального алгоритма для выделения адреса;
- выработка рекомендаций доработки алгоритма для решения прикладных задач.

Подробнее остановимся на методике ручной обработки сообщений. Для качественной оценки результатов работы библиотек следовало провести ручную обработку данных для выделения всех лексем, которые возможно использовать при определении адреса: названия элементов адреса (область, город, улица, дом и т.п.), собственные имена улиц, городов, областей. Дополнительно выделялись собственные имена локаций (ЖК «Орбита», ТЦ «Космос», станция «Киевская» и т. п.), наименование объектов улично-дорожной сети и дорожной инфраструктуры (перекресток, мост, эстакада и т.п.), собственные имена таких объектов (МКАД, КАД, ЦКАД и т.п.). В случае написания адреса или его элемента без пробела, напр., «ул.Лермонтова», - такой объект делился на два элемента: «ул.» и «Лермонтова». Из строки адреса исключались запятые.

3. Описание датасета.

Для проверки работы библиотек по распознавания адресов и географических наименований использовался собранный корпус текстов (сообщений) из следующих телеграмм-каналов: «ДТП и ЧП МОСКВА и МО», «ПИТЕР | ДТП | ЧП | ОЧЕВИДЕЦ», «ДТП и ЧП Воронеж», «ДТП и ЧП Ростов», «Брянск /новости/ ЧП/ДТП», охватывающий период с 01.02.2024 по 15.03.2024. Итоговый датасет содержит 1202 текстов-сообщений о различного рода инцидентах и происшествиях.

Датасет представляет собой файл в формате csv следующей структуры:

ID – идентификатор объекта, число,

Source – источник текстовой записи (название телеграмм-канала), строка

Datetime – дата и время текстовой записи, строка,

Text – текст сообщения о происшествии или инциденте, строка,

Text_address – обработанная строка текста только со словами на основании которых формируется географическая привязка к местности, строка,

Nat_NER – результат поиска объекта класса ‘LOC’ с помощью библиотеки Natasha, список строк,

Nat_addr - результат поиска объекта класса ‘Address’ с помощью библиотеки Natasha, список объектов,

Stanza - результат поиска объекта класса ‘LOC’ с помощью библиотеки Stanza, список строк,

Pullenti - результат поиска объекта класса ‘LOC’ с помощью библиотеки Pullenti, список словарей,

Spacy - результат поиска объекта класса ‘LOC’ с помощью библиотеки Spacy, список строк.

Распределение сообщений по городам:

Город	К-во сообщений	Доля
Брянск	138	11,48%
Воронеж	159	13,23%
Москва и МО	452	37,60%
Ростов	80	6,66%
Санкт-Петербург и ЛО	373	31,03%
Общий итог	**1202**	

Распределение сообщений по полноте указанного адреса:

В 1159 сообщениях есть информация о месте происшествия, как минимум отсылка на город и область.

130 сообщений включают полный адрес с указанием улицы и дома (город указывается не всегда, поскольку сообщения взяты из телеграм-каналов, посвящённых одному конкретному городу).

215 сообщений содержат названия шоссе и трасс, включая МКАД и КАД.

В 76 сообщениях вместо адреса или в дополнение к нему указаны имена собственные — названия жилых комплексов, торговых центров, железнодорожных станций, станций метро, мостов и путепроводов.

65 сообщений содержат слова «перекрёсток» или «пересечение».

Исходный корпус сообщений, ввиду специфического источника, имеет ряд особенностей, который существенно влияет на распознавание адресов и географических наименований:

- полные названия адресов встречаются крайне редко («Москва, Кантемировская улица»);
- имена улиц, городов, районов, трасс часто пишутся с прописной буквы («92 км кад»);
- отсутствует правильная пунктуация (« Москва «Марьино»»);

- отсутствуют пробелы между словами («Москва, ул.Подольских Курсантов 4 Сгорел автомобиль»);

- очень много сокращенных названий, локализмов и регионализмов (локальной топонимики) («Пожарные МЧС России предотвратили взрыв газовых баллонов на стройке на Левенцовке»);

- отсутствует информация о типе объекта в адресе – адресном предикате (улица, площадь, проспект, станция метро, дом и т.д.) («Сообразили на троих на Науке»).

Не меньшую роль на формирование географической привязки по адресу оказывает природа сообщений – уведомление об инциденте или дорожно-транспортном происшествии. Обозначение места ДТП редко имеет формализованный вид – название трассы, улицы, шоссе и т.д., указание конкретной точки – километровая отметка, адрес дома, локальная точка притяжения (ТЦ, памятник и т.п.). Гораздо чаще встречается указание на локальный топоним, на объект дорожной инфраструктуры (мост, путепровод, тоннель), просто на населенный пункт или даже район. Так как аварийность на перекрестках существенно выше, чем на перегонах, то во многих случаях для обозначения места происшествия используется названия двух улиц.

4. Эксперименты.

4.1.Описание метрик.

Для оценки качества решения задачи Named Entity Recognition (NER) используются различные метрики, среди которых наиболее распространенной является строгая f-мера. Эта метрика объединяет точность и полноту в одно значение, где точность отражает долю правильно идентифицированных сущностей среди всех выделенных, а полнота показывает долю верно выделенных сущностей среди всех сущностей в эталоне. Эти метрики позволяют количественно оценить эффективность алгоритмов NER и сравнить их между собой [16].

Единицей для расчета этих метрик считается сущность, а не слово, то есть точность определяется как отношение количества правильно распознанных сущностей и количества распознанных сущностей, а полнота – как отношение правильно распознанных сущностей и количества сущностей в золотом стандарте. F-мера объединяет в себе две предыдущие метрики.

Из-за сегментации оценка качества модели осложняется, поскольку при обучении моделей мы используем слова, а для оценки – целые сущности. По этой причине если в нашем корпусе есть выражение площадь Маршала Жукова, а система определяет только Маршала Жукова – это уже ошибка [17].

Однако на практике, существует примеры использования подходов модифицированных метрик Precision и Recall. Например, расширение границ, при котором засчитывается результат, даже если границы сущности указаны неправильно, при условии что сама сущность полностью найдена и правильно распознана, либо же засчитывание примера при неправильной классификации сущности, но при правильном определение границ сущности (span) [18].

Представляется, чтобы решить задачу обнаружения адресов, можно не стремиться к абсолютной точности в определении границ объекта, поскольку адреса в сообщениях из мессенджеров часто бывают неполными и ошибочными из-за специфики таких текстов. Более важно определить наличие самого признака адреса или места происшествия. Скорее всего, после этого потребуется дополнительная обработка человеком. Таким образом в качестве базовых метрик можно взять Precision, Recall и F1 меру, но при этом засчитывать в качестве правильных ответов не только те варианты, где правильно распознаны границы сущности, но и те, в которых границы совпадают частично, то есть адрес будет считаться обнаруженным, если зафиксирован хотя бы один из его элементов.

Для оценки качества распознавания адресов также можно использовать две дополнительные метрики: Bleu (в варианте sacrebleu) [19] и Bertscore [20], причем Bertscore также в качестве результата выдает Precision, Recall и F1.

Метрика BLEU (Bilingual Evaluation Understudy) является одним из основных инструментов для оценки качества решения NLP задач. Она позволяет оценить, насколько близок итоговый текст к человеческому, сравнивая их на уровне отдельных слов и фраз. Основная идея BLEU заключается в том, что чем лучше сгенерированный текст, тем больше он должен быть похож на человеческий. Метрика работает лучше на уровне больших текстов, поскольку на уровне отдельных предложений она может давать некорректные результаты.

SacreBLEU - это модификация метрики BLEU, разработанная для улучшения её способности оценивать качество генерации текстов. Она использует модифицированный алгоритм для вычисления весов n-грамм, что позволяет более точно отражать качество генерации. Эта метрика широко используется в исследованиях и практике машинного перевода благодаря своей эффективности и гибкости.

Метрика BLEU (Bilingual Evaluation Understudy) изначально была разработана для оценки качества машинного перевода, но она также может быть адаптирована для использования в задачах Named Entity Recognition (NER). BLEU измеряет сходство между предсказаниями модели и эталонными аннотациями, используя n-граммные совпадения и взвешивание.

В контексте NER, BLEU может быть применена для оценки качества распознавания именованных сущностей, сравнивая предсказанные сущности с истинными. Это позволяет оценить, насколько хорошо модель способна распознавать и классифицировать именованные сущности в тексте.

Однако, следует отметить, что BLEU не всегда является идеальным выбором для оценки NER, поскольку она может недооценивать качество распознавания сущностей, особенно если модель делает ошибки в порядке или количестве сущностей. В таких случаях, использование ранее упомянутых метрик, таких как F-мера или Precision/Recall, может быть более подходящим.

Метрика Bertscore была предложена в 2019 [21] году для оценки качества генерируемого текста. Она основана на использовании предобученной нейросетевой модели BERT для вычисления близости контекстных эмбеддингов между сгенерированным текстом и эталонным.

Для расчета Bertscore сначала необходимо получить контекстные эмбеддинги для каждого слова в обоих предложениях (сгенерированном и эталонном) с помощью модели BERT. Затем вычисляется косинусное подобие между каждым словом в одном предложении и каждым словом в другом предложении. Сумма этих косинусных подобий и составляет значение Bertscore.

Bertscore имеет несколько преимуществ перед другими метриками. Во-первых, она учитывает контекст слов, что позволяет более точно оценить качество текста. Во-вторых, она легко интерпретируется, так как ее значение находится в диапазоне от 0 до 1, где 1 означает полное соответствие между сгенерированным и эталонным текстом.

Однако у Bertscore есть и некоторые ограничения. Она может быть чувствительна к длине текста и структуре предложений, что может повлиять на ее точность. Кроме того, Bertscore не учитывает семантическую близость слов, что может привести к неправильным оценкам в некоторых случаях.

Использование метрики Bertscore для решения задачи поиска именованных сущностей (Named Entity Recognition, NER) может быть эффективным способом оценки качества моделей, особенно когда речь идет о моделях, основанных на архитектуре BERT. Bertscore позволяет оценить, насколько хорошо модель

способна распознавать и классифицировать именованные сущности в тексте, учитывая контекст и семантику слов.

Применение Bertscore для NER включает следующие шаги:

Подготовка данных: Необходимо иметь набор данных с размеченными именованными сущностями. Эти данные могут быть использованы для обучения модели NER и для оценки ее производительности.

Обучение модели NER: Модель обучается на подготовленном наборе данных, используя архитектуру BERT или другую подходящую модель.

Оценка модели: После обучения модели, она применяется к тестовому набору данных, и результаты ее работы сравниваются с эталонными аннотациями.

Расчет Bertscore: Для каждой пары сгенерированных и эталонных аннотаций рассчитывается значение Bertscore (Precision, Recall, F1). Эти значения отражают степень сходства между сгенерированными и эталонными аннотациями.

Интерпретация результатов: Значения Bertscore находится в диапазоне от 0 до 1, где 1 означает полное соответствие между сгенерированными и эталонными аннотациями. Чем ближе значение к 1, тем лучше модель справляется с задачей NER.

Использование Bertscore позволяет учесть контекст и семантику слов при оценке качества моделей NER, что делает эту метрику особенно полезной для задач, где важно точное распознавание именованных сущностей.

Значения метрики Bertscore будет интересно сравнить с используемыми нами модифицированными метриками (Precision, Recall, F1).

4.2. Описание экспериментов.

На первом этапе были проведены эксперименты по извлечению адресов из собранного датасета стандартными средствами библиотек. В библиотеки Natasha использовались два встроенных механизма: NER tagger и Address extractor. С помощью NER tagger были извлечены именованные сущности категории ‘LOC’. Результат работы NER tagger был сохранен в виде списка строк. Далее корпус был обработан с помощью Address extractor, который потребовал предварительную лемматизацию текстов с помощью MorphVocab. Результатом обработки стал список объектов с указанием наименования (value) и адресного предиката (type).
Для библиотеки Stanza необходимо было разбить текст на предложения и далее из полученного списка токенов выбрать токены категории ‘LOC’. Результатом работы стал сохранённый список строк с выявленными названиями адресов. Для библиотеки Pullenti модель была настроена на извлечение географических наименований и адресов. Результат работы стал список словарей, где ключом выступает значение предикат адреса, а значением – наименование города, улицы и т.д. Извлечение адреса с помощью библиотеки Spacy также сводиться к выбору токенов с лейблом 'LOC’ и сохранением их в виде списка строк.

Итогом работы библиотеки Natasha(NER) стал следующий результат: выделяются в основном слова с большой буквы, при этом наблюдаются проблемы с выделением адресных предикатов, если они стоят перед названием, то есть «Варшавское шоссе» распознается как локация, а «улица Подольских Курсантов» выделиться в сокращённом виде – «Подольских Курсантов». Наблюдаются проблемы с сокращениями – «ул.», «г.», «г.о.» и т.п. Из-за проблем с адресными предикатами не распознаются такие названия как «улица Дмитрия Ульянова». Также, практически не выделяются: 1) адрес в конструкциях «номер километра + название автодороги» 2) второе название улицы в конструкциях типа «на пересечении ул. Первой и ул. Второй», 3) номера домов, строений, 4) коды автодорог.

Результаты работы библиотеки Natasha(Address_Extractor) ожидаемо невысоки в неформализованных конструкциях. Но удивительное другое, что библиотека не очень хорошо обрабатывает и тексты, содержащие полные адреса, но без адресных предикатов. Например, «Москва» и «город Москва» распознается одинаково хорошо, но уже «Варшавское шоссе 21» распознается только как «Варшавское шоссе». Отсутствие таких маркеров как «дом», «корпус» обрезает распознаваемый адрес. Также, наблюдаются проблемы в распознавании с наличием двух слов/словосочетаний с большой буквы перед и после адресного предиката, например, «Варшавское шоссе Наркологическая клиника» или «улица Братеевская Автомобиль сбил пешехода». В данном случае распознается адрес «шоссе Наркологическая клиника» и «улица Братеевская Автомобиль». Отметим, что не распознаются неформальные названия локаций, название районов и регионов, название станций метро или железнодорожных станций, некрупных городов, поселков и т.д.

Интересно, проанализировать работу библиотеки Pullenti. Данная библиотека содержит компонент привязки адреса из текста к объектам ГАР ФИАС. Поэтому ожидается лучшее и более точное распознавание адресов. Действительно для данной библиотеки мы наблюдаем хорошее распознавание полных адресов, даже если они не содержат всех адресных предикатов («дом», «корпус», «квартира»). В большинстве случаев хорошо обрабатываются названия районов, городов, посёлков и сел. Обрабатываются названия автодорог, станций метро. Однако сохраняется проблемы с распознаванием при наличии нескольких названий с заглавной буквы, ошибках в названии, например, «Подмосковье, г.о.Раменское Новорязанское шоссе деревня Становое», «Подмосковье, Наро-Фоминский г.о. Эстакада ЦКАД / Киевское шоссе» распознаются как «область Московская», а «Москва, метро «Зюзино» Человек прыгнул под поезд» - «город МОСКВА метро ЗЮЗИНО ЧЕЛОВЕК». Отметим нестабильную работу при выделении наименований автодорог, МКАД и А101 распознаются, а ЦКАД – нет:
«Москва, 35-км МКАД (внеш.) Шашечник влетел в несколько машин и перевернулся» - «город МОСКВА автодорога МОСКОВСКАЯ КОЛЬЦЕВАЯ»;
«Подмосковье, ЦКАД Опрокинулся грузовик» - «область МОСКОВСКАЯ».
При распознавании дорог не выделяется номер километра, даже если он есть в сообщении. Для данной библиотеки характерно приведении всех собственных названий к написанию заглавными буквами, что безусловно повлияло на результаты оценки некоторыми метриками.

Библиотека Spacy выделяет как сущности практически все слова, написанные с заглавной буквы, если перед ними нет знаков препинания. Она неплохо справляется со стандартными адресными предикатами, но испытывает трудности с сокращениями, особенно с «г.о». Кроме того, в некоторых текстах названия предикатов не распознаются как часть адреса. Библиотека выдаёт неоднозначные результаты: в похожих контекстах локации могут выделяться по-разному или вообще не выделяться. Несмотря на то, что оценки метрики Bertscore неплохи, но другая метрика (BLEU) выдало для данной библиотеки нулевое значение, что подтверждает наши наблюдения.

Библиотека Stanza показывает самые высокие оценки при выделении объектов категории «LOC». Это связано с тем, что библиотека стабильно выделяет все слова, начинающиеся с заглавной буквы, сокращения («МКАД», «ЖК», «ТЦ»), слова и словосочетания в кавычках, большинство цифровых обозначений («35-км МКАД», «Рязанский проспект 14к2»), адресные предикаты. Безусловно при таком подходе в результат попадают и часть лишней информации. Вместе с тем, библиотека обладает рядом типовых недостатков – плохо обрабатывает тексты с пунктуационными ошибками, частично теряет цифры в адресах, не знает сокращения типа «г.о.», обрезает адрес в случаях похожих названий(«Москва, посёлок Киевский Киевское шоссе» - выдаст «Москва Киевское шоссе», то есть Киевский/Киевское будет использовано только 1 раз)

На втором этапе был проведен анализ результатов в сравнении с выделенными адресами при ручной обработке. При этом адрес засчитывался как найденный, даже если стандартными инструментами библиотек был найден только один его элемент (подобнее см. раздел 4.1. Описание метрик). Результаты показали неплохие метрики для библиотек Natasha и Stanza, которые можно использовать для предварительной оценки наличия адреса в сообщении и последующей его обработки (см. Таблицы 1 и 2)

Таблица 1.

	Обнаружено адресов (TP+FP)	Из их ошибочно (FP)	Не обнаружено адресов (TN+FN)	Из них ошибочно (FN)
Natasha(NER)	1097	69	105	55
Natasha(Adress_extractor)	860	8	342	238

Stanza	1079	62	123	61
Pullenti	934	37	268	181
Spacy	994	36	208	120

Таблица 2.

	Precision	Recall	F1	Bleu	Bertscore (P)	Bertscore (R)	Bertscore (F1)
Natasha(NER)	0,9371	**0,9492**	**0,9431**	0.00	0.8353	0.8191	0.8248
Natasha (Address_extractor)	**0,9907**	0,7817	0,8738	37.99	0.8141	0.7652	0.7874
Stanza	0,9425	0,9434	**0,943**	**80.91**	**0.8689**	**0.8630**	**0.8647**
Pullenti	0,9604	0,8321	0,8917	8.64	0.6229	0.6664	0.6422
Spacy	0,9638	0,8887	0,9247	0.00	0.8546	0.8180	0.8344

Самые лучшие значения F1 показали сразу две библиотеки – Natasha(NER) и Stanza. Далее была проведена оценка результатов при помощи метрики Bleu (sacrebleu). Бесспорным лидером с существенным отрывом снова стало решение от Стэндфордского Университета – Stanza. Отметим, что метрика не сработала для анализа результатов библиотек Natasha(NER) и Spacy. Интерпретация отрицательных результатов для этих библиотек требует отельного изучения. Возможно, это связано с тем, что метрика в первую очередь предполагалась для оценки других типов задач. Также велик разрыв между значением метрик для лидера и двух других библиотек.

Метрика Bertscore подтверждает результаты предыдущих оценок. Снова бесспорным лидером является Stanza, на втором месте Spacy, далее Natasha(NER).

Результаты анализа метрик подтвердили, что при извлечении адреса из неформализованных сообщений, которыми являются тексты из телеграмм-каналов, использование инструментов типа Pullenti и Natasha(Address_Extractor) дает худшие результаты, чем использование библиотек решающих задачу NER в общем виде. Полные адреса встречаются только в 11% сообщений, поэтому работа специализированных библиотек не дает им высокий прирост метрик качества. К тому же, накладывают дополнительные ограничения сам подход к оценке – использование метрик BLEU и Bertscore, которые сравнивают результат с выделенным человеком сущностью. При этом у Pullenti, например, все элементы адреса начинаются с большой буквы и имеют адресные предикаты, что безусловно снижает итоговую оценку.

В сообщениях без четкого адреса лучше всего отрабатывает библиотека Stanza. При этом она выделяет имена собственные, такие как ЖК, название организаций, ТЦ, объектов инфраструктуры. Таким образом – формируется максимальная привязка к местности. Вместе с тем, отметим и большое количество «мусорных» элементов, например, слов с большой буквы, которые появляются в начале предложений или из-за орфографических и пунктуационных ошибок в сообщениях.

Таким образом, стандартные инструменты библиотек не способны качественно решать задачу выявления адреса среди сообщений об инцидентах и ДТП в сообщениях ТГ-каналов без вторичной ручной проверки результатов. В качестве дальнейшего развитие настоящей работы, возможно рассмотреть дообучение собственной NER-модели, например на базе трансформера T5.

Выводы.

1. Извлечение адресной информации и информации о местоположении является актуальной задачей для разных предметных областей. При практическом решении данной задачи активно используются готовые инструменты (библиотеки с предобученными языковыми моделями), разрабатываются специализированные алгоритмы и механизмы обработки текстов, активно используется итерационный подход к разработке.
2. Исходный корпус сообщений, имеет ряд особенностей, который существенно влияет на распознавание адресов и географических наименований: обозначение места ДТП редко имеет формализованный вид; полные адреса встречаются редко, названия улиц, городов, районов, трасс часто пишутся с прописной буквы, отсутствует правильная пунктуация отсутствуют пробелы между; очень много сокращенных названий, локализмов и регионализмов отсутствует информация о типе объекта в адресе.
3. Для поиска и выделения адресов в тексте в основном используются готовые библиотеки обработки естественного языка, такие как Spacy, Stanza, Pullenti, Natasha и др. Они могут распознавать различные типы именованных сущностей, включая географические объекты и адреса.
4. Для оценки качества решения задачи Named Entity Recognition (NER) используются различные метрики, среди которых наиболее распространенной является F1-мера, Precision, Recall, Bertscore, с ограничениями возможно применения BLEU.
5. При извлечении адреса из неформализованных сообщений использование инструментов типа Pullenti и Natasha(Address_Extractor) дает худшие результаты, чем использование библиотек решающих задачу NER в общем виде. В сообщениях без четкого адреса лучше всего отрабатывает библиотека Stanza. При этом она выделяет имена собственные, такие как ЖК, название организаций, ТЦ, объектов инфраструктуры.
6. Существующие инструменты библиотек не могут эффективно определять адреса в сообщениях Telegram-каналов об инцидентах и ДТП без дополнительной ручной проверки результатов. Для дальнейшего развития данной работы можно рассмотреть возможность дообучения собственной NER-модели, например, на основе трансформера T5.

Библиографический список.

1. Мазалов А. Сравниваем работу open source Python — библиотек для распознавания именованных сущностей. Блог Александра Мазалова. [Электронный ресурс]. URL: https://habr.com/ru/articles/502366/ (дата обращения: 13.05.2024).
2. NLP. Проект по распознаванию адресов. Natasha, Pullenti, Stanza. Блог NewTechAudit. [Электронный ресурс]. URL: https://habr.com/ru/articles/667442/ (дата обращения: 13.05.2024).
3. Камов С. Как найти что-то в тексте. Блог Сергея Камова. [Электронный ресурс]. URL: https://habr.com/ru/articles/530878/ (дата обращения: 13.05.2024).
4. Как мы распознавали бренды в покупках целевой аудитории. Блог ГК ЛАНИТ. [Электронный ресурс]. URL: https://habr.com/ru/companies/lanit/articles/708414/ (дата обращения: 13.05.2024).
5. Макаров О.С., Куляшова Н.М. АЛГОРИТМЫ РАСПОЗНАВАНИЯ ПОЧТОВЫХ АДРЕСОВ // E-Scio. 2020. №7 (46). URL: https://cyberleninka.ru/article/n/algoritmy-raspoznavaniya-pochtovyh-adresov (дата обращения: 13.05.2024).
6. ПОИСК АДРЕСОВ В «ИСПОРЧЕННЫХ» ДАННЫХ. Блог NewTechAudit. [Электронный ресурс]. URL: https://vc.ru/dev/286644-poisk-adresov-v-isporchennyh-dannyh (дата обращения: 13.05.2024).
7. Обработка NL адресов для создания эффективных поисковых запросов при помощи набора инструментов проекта Natasha. Блог NewTechAudit. [Электронный ресурс]. URL: https://newtechaudit.ru/obrabotka-nl-adresov-dlya-sozdaniya-effektivnyh-poiskovyh-zaprosov/(дата обращения: 13.05.2024).
8. NER: Как мы обучали собственную модель для определения брендов. Часть 2. Блог ГК ЛАНИТ. [Электронный ресурс]. URL: https://habr.com/ru/companies/lanit/articles/725960/ (дата обращения: 13.05.2024).
9. Проект Natasha — набор Python-библиотек для обработки текстов на естественном русском языке. [Электронный ресурс]. URL: https://natasha.github.io/ (дата обращения: 13.05.2024).
10. SlovNet. [Электронный ресурс]. URL: https://github.com/natasha/slovnet#ner (дата обращения: 13.05.2024).
11. SlovNet. Evaluation. [Электронный ресурс] URL: https://github.com/natasha/slovnet#evaluation (дата обращения: 13.05.2024).
12. Yargy. [Электронный ресурс]. URL: https://github.com/natasha/yargy (дата обращения: 13.05.2024).
13. Stanza – A Python NLP Package for Many Human Languages. [Электронный ресурс]. URL: https://stanfordnlp.github.io/stanza/ (дата обращения: 13.05.2024).
14. spaCy. Industrial-Strength Natural Language Processing. [Электронный ресурс]. URL: https://spacy.io/ (дата обращения: 13.05.2024).
15. Pullenti. [Электронный ресурс]. URL: https://www.pullenti.ru/ (дата обращения: 13.05.2024).
16. NLP. Основы. Техники. Саморазвитие. Часть 2: NER. Блог Content AI. [Электронный ресурс] URL: https://habr.com/ru/companies/contentai/articles/449514/ (дата обращения: 13.05.2024).
17. Васильев В. Простым языком об извлечении текстовой информации и NERю Блог Владимира Васильева. [Электронный ресурс] URL: https://vc.ru/u/1200058-vladimir-vasilev/456811-prostym-yazykom-ob-izvlechenii-tekstovoy-informacii-i-ner (дата обращения: 13.05.2024).
18. Abhigyan Raman. NER Report. [Электронный ресурс]. URL: https://wandb.ai/abhigyan/Named%20Entity%20Recognition/reports/NER-Report--Vmlldzo0MDM0NjE (дата обращения: 13.05.2024).
19. sacreBLEU. [Электронный ресурс]. URL: https://github.com/mjpost/sacreBLEU(дата обращения: 13.05.2024) .
20. BERTScore. [Электронный ресурс]. URL: https://github.com/Tiiiger/bert_score#readme (дата обращения: 13.05.2024).
21. Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, Yoav Artzi. BERTScore: Evaluating Text Generation with BERT [Электронный ресурс]. URL: https://arxiv.org/abs/1904.09675 (дата обращения: 13.05.2024).