

Санкт–Петербургский государственный университет  
Кафедра математической теории игр и статистических решений

**Бельков Роман Андреевич**

**Выпускная квалификационная работа магистра**

**Интерактивная карта дорожных происшествий  
Санкт-Петербурга**

Направление 01.04.02

«Прикладная математика и информатика»

Основная образовательная программа ВМ.5504.2020 «Исследование  
операций и системный анализ»

Научный руководитель:

кандидат физ.-мат. наук, доцент  
кафедры компьютерного моделирования и  
многопроцессорных систем,  
Корхов Владимир Владиславович

Рецензент:

кандидат технических наук,  
ведущий программист  
ООО «Осенсус Арм»,  
Балян Сероб Гургенович

Санкт-Петербург

2022 г.

# Содержание

<b>Введение</b> . . . . .	3
<b>Постановка задачи</b> . . . . .	5
<b>Обзор литературы</b> . . . . .	6
<b>Глава 1. Алгоритмы и архитектуры</b> . . . . .	21
1.1. Методы распознавания именных сущностей . . . . .	21
1.1.1 Stanza . . . . .	21
1.1.2 Deep Pavlov . . . . .	24
1.1.3 Natasha . . . . .	28
1.2. Метрики оценки качества . . . . .	29
<b>Глава 2. Данные</b> . . . . .	31
2.1. Collection5 . . . . .	31
2.2. BSNLP-2019 . . . . .	32
2.3. FactRuEval 2016 . . . . .	33
<b>Глава 3. Реализация</b> . . . . .	35
3.1. Выбор технологий и инструментов для реализации . . . . .	35
3.2. Сравнение алгоритмов NER . . . . .	36
3.3. Сбор данных . . . . .	37
3.4. Выявление адреса в тексте . . . . .	38
3.5. Визуализация данных . . . . .	39
<b>Заключение</b> . . . . .	43
<b>Список литературы</b> . . . . .	45

## Введение

Из всех видов транспорта автомобильный является наиболее популярным, но также и наиболее опасным. Об этом свидетельствует статистика дорожно-транспортных происшествий в России. По состоянию на 2019 год в России на 1000 человек приходится 381 автомобиль, всего автомобилей насчитывается около 55.9 миллионов [1]. По данным МВД с января по ноябрь 2021 года в России произошло 96 314 ДТП с пострадавшими, погибли 10 516 человек и получили ранения 121 573 [2].

В условиях стремительной "автомобилизации" населения России и недостаточно высокого уровня культуры и правосознания от дорожно-транспортных происшествий ежегодно погибают или получают ранения десятки тысяч человек. Анализ статистических данных о количестве дорожно-транспортных происшествий и их последствиях свидетельствует о том, что уровень дорожно-транспортного травматизма в стране остается крайне высоким и имеет тенденцию к росту.

Масштабы социальных и экономических потерь от негативных последствий автомобилизации, по мнению ряда ученых, сопоставимы с самыми актуальными проблемами правоохранительной деятельности государства. И по сути, общество имеет дело с особым видом преступности.

Но в связи с тем, что дорожно-транспортная авария рассматривается в качестве явления, которое было вызвано некими случайными причинами, ее последствия не вызывают адекватного общественного резонанса, а следовательно, и адекватной реакции со стороны государства. Соответственно, строится и стратегия противостояния со стороны государства и общественности этому злу.

Любое дорожное происшествие возникает вследствие внешнего случайного стечения факторов, но в совокупности эти факторы образуют устойчивые связи и отношения. Они подчиняются строгим законам вероятностного вида.

В настоящее время со стремительным развитием технологий и увеличения количества данных есть возможность разработки информационного продукта, который бы автоматизировано проводил сбор данных о дорожно-транспортных происшествиях, формировал статистику, визуализировал информацию и проводил анализ. Такой продукт мог бы выявлять самые опасные участки дороги и формировать отчет по зависимости уровня аварийности на определенных участках от времени года, суток или погодных условий. Также, можно применять знания об аварийности в картах при построении маршрута.

## Постановка задачи

Проблема сильно недооценена обществом и воспринимается обыденностью и нормой. Чтобы переломить эту ситуацию хочется дать возможность всем желающим вовлечься в решение этой проблемы через исследование ДТП, выявление факторов аварийности, опасных участков в городе.

Для решения проблемы необходимо создать интерактивную карту дорожно-транспортных происшествий Санкт-Петербурга со ссылкой на источник, где описаны детали аварии.

Цель данной работы создать интерактивную карту, которая показывает в каком месте произошли события. Следовательно, необходимо выделять названия улиц, перекрестков и других данных из текста, которые помогут распознать местоположение.

Для достижения цели были поставлены следующие задачи:

1. Обзор существующих решений в области агрегации и визуализации данных о ДТП;
2. Поиск и сбор данных о ДТП;
3. Обзор и сравнение методов выявления местоположений в тексте;
4. Реализация собственного решения проблемы;
5. Сравнение полученных результатов с существующими методами.

## Обзор литературы

В работе «Street-based topological representations and analyses for predicting traffic flow in GIS» [3] решается задача прогнозирования транспортного потока Гонконга. В качестве источника информации о трафике используется AADT (среднегодовой ежедневный трафик), созданный на основе необработанных данных, собранных со счетных станций, где индуктивные петли и пневматические трубки установлены на проезжей части и подключены к придорожным автоматическим счетчикам [4]. Этот метод нам не подходит, так как таких данных для Санкт-Петербурга в открытом доступе нет и нам важны данные о ДТП, а не о трафике.

В статье «A computational approach to 'The Image of the City'» [5] предлагается вычислительный подход к формированию образа города. Методы, основанные на информационном подходе к изображению города, используются для определения ориентиров, интегрируя сложность точек отсчета в их визуальных, структурных и семантических компонентах, как это было концептуализировано Линчем и последовательными исследованиями. Образ города Линча [6] описывает, как люди воспринимают и запоминают особенности городских пространств. Наиболее характерные элементы городского ландшафта, классифицированные по путям, узлам, краям, районам и достопримечательностям, формируют ментальное представление людей о городе. Эти методы были применены к центральному району Бостона и построены с использованием свободно доступных пространственных наборов данных.

Узлы – это стратегические фокусы, в которые может войти наблюдатель, и которые являются интенсивными фокусами, к которым он движется и из которых он движется. Это могут быть в первую очередь перекрест-

ки, места остановки транспорта, пересечения или схождения путей В этой структуре топологические свойства степени, близости и взаимосвязи обычно вычисляются с использованием представления двойного графа, в котором сегменты улиц представлены вершинами, а перекрестки - ссылками.

Пути – это каналы, по которым обычно, время от времени или потенциально движется наблюдатель. Сегменты улиц преобразуются в вершины. Когда два сегмента улицы пересекают друг друга в дорожной сети, создается ссылка, соединяющая соответствующую вершину в двойном представлении. Амплитуда угла падения, образованного двумя сегментами улицы, присваивается соответствующей ссылке в качестве веса.

Районы – это относительно большие городские пространства, в которые наблюдатель может мысленно войти и которые имеют некоторый общий характер. Law (2017) [7] иллюстрирует процесс генерации подграфов из топологии улиц с применением методов обнаружения сообществ. Так называемые уличные локальные районы (SLA) (Turner, 2007) [8] - это регионы, внутренняя однородность которых имеет социальные и функциональные основы.

Достопримечательности – это точечные ориентиры, которые считаются внешними по отношению к наблюдателю. Настоящая работа следует концепции различения трех типов достопримечательностей: визуальных, структурных и когнитивных. Визуальные свойства включают высоту, площадь фасада и видимость. Культурное значение здания было получено путем подсчета количества перечисленных исторических элементов, расположенных в его границах.

Края – это линейные элементы, которые не рассматриваются как пути:

обычно они являются границами между двумя типами областей. В текущем анализе в качестве ребер были извлечены следующие линейные элементы с заданной минимальной длиной:

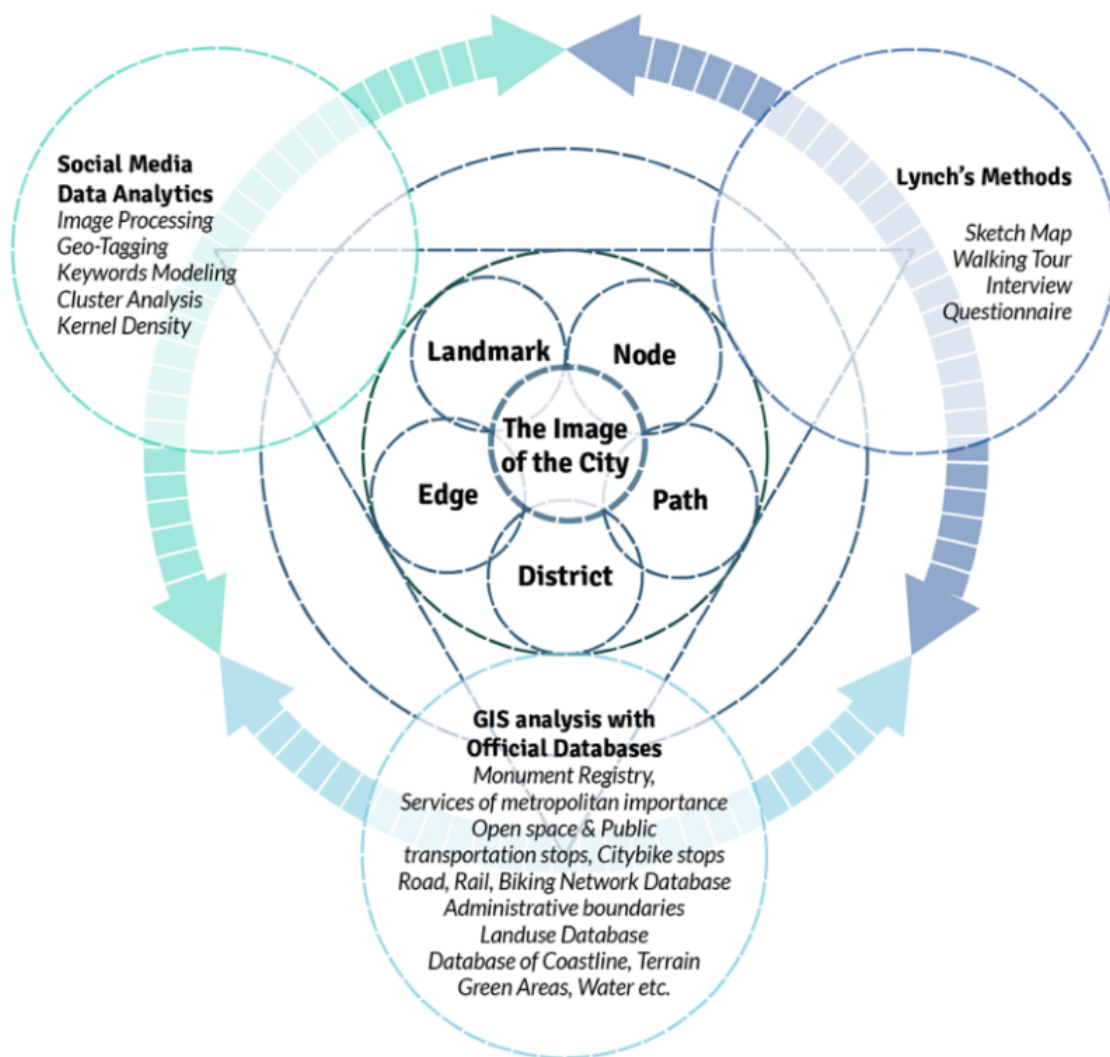
- Участки железнодорожных сооружений в качестве объездных путей или других видимых сооружений.
- Участки крупных дорог (например, дороги с двусторонним движением).
- Участки автомагистралей.
- Берега рек или общие водоемы (озера, морское побережье).

На основе этих данных формируется образ города, который визуализируется с помощью карты. Авторы утверждают, что этот инструмент может поддерживать решения по пространственному планированию в городском дизайне, предоставляя важную информацию, касающуюся пригодности для жизни в городе, качества жизни, адекватное сочетание видов землепользования, простота навигации и ориентации.

В статье «The image of the City on social media: A comparative study using 'Big Data' and 'Small Data' methods in the Tri-City Region in Poland» были применены три параллельных метода (рис. 1):

1. анализ социальных сетей, т.е. сбор данных, обработка изображений, кластерный анализ, оценка плотности ядра, анализ настроений;
2. оригинальные методы Линча [6];
3. официальная база данных ГИС о городе, памятниках, открытых пространствах, дорожной сети, границах районов.





**Рис. 1:** Концептуальная основа использования аналитики социальных сетей, оригинальных методов Линча и ГИС-анализа для триангуляции результатов по теориям изображения городов.

Данные, то есть изображения и текстовая информация, были собраны с двух популярных платформ социальных сетей: 1) Instagram, социальной сети для обмена фотографиями и видео, и 2) Twitter, в основном текстовой платформы, позволяющей пользователям обмениваться короткими сообщениями. Исходный набор данных состоит из текста, изображений, видео, если таковые имеются, метки времени, GPS-координат, языковых настроек, идентификатора пользователя и т.д. Повторяющиеся посты были удалены.

Обработка изображений была проведена с целью определения содержания изображений в Instagram и их соответствия воспринимаемым линчевским элементам городских изображений.

Например, если фотография посвящена знаковому зданию, она будет помечена как “достопримечательность” или “узел”, если речь идет об открытом пространстве или площади. В случае перекрытия, т.е. фотография содержит как знаковое здание, так и площадь, тогда изображение будет помечено как “достопримечательность” и “узел”.

Кластерный анализ был использован для оценки пространственной непрерывности и прерывности активности в социальных сетях, чтобы измерить понятие “район”. Результаты показали, относится ли сообщение в социальной сети к ядру (в пределах района), границе (на границе района) или является изолированным выбросом.

База данных ГИС была получена через государственные геодезические службы, транспортное управление и реестр памятников и достопримечательностей.

Аналитическая основа, разработанная на основе этого исследования, может быть использована для оценки и расширения классической теории планирования в эпоху цифровых.

Приведенные выше методы не подходят для решения задачи построения интерактивной карты дорожно-транспортных происшествий Санкт-Петербурга, так как в работах в основном описаны алгоритмы построения образа города и исследование трафика, но не делается упор на ДТП. Но можно воспользоваться, приведенными в статьях, методами получения данных для построения карты, такие как анализ социальных сетей и официальные

географические информационные системы.

Так как готового решения по построению карты ДТП не нашлось, рассмотрим составные части.

Для карты требуются данные о произошедших ДТП. Источники данных можно разделить на официальные и неофициальные. Официальные - это данные опубликованные государственными структурами: МВД, ГАИ, ГИБДД. Неофициальные - всевозможные источники, которые наполняются благодаря информации от самих людей.

Информация по дорожным происшествиям публикуются на официальном сайте статистики ГИБДД [stat.gibdd.ru](http://stat.gibdd.ru). Данные публикуются раз в год, группируются по регионам, а внутри региона по районам. На данный момент на сайте имеется статистика только за 2021 год и за 2004-2014 года. В архиве за 2004-2014 года имеется информация только о количестве ДТП, за 2021 год есть данные о конкретных происшествиях: координаты, дата и время, вид ДТП, информация о дорожных условиях и участниках происшествия. Но вся информация на сайте ГИБДД о ДТП, в которых имеются пострадавшие, данных о ДТП без пострадавших не публикуются.

Этот источник не подходит для выполнения поставленной задачи, так как данные появляются с задержкой и не дают полной картины по ДТП, так как многие происшествия не имеют пострадавших.

Также есть пользовательские метки о ДТП в Яндекс Навигаторе. Пользователи собственноручно могут на карте выставлять метки, обозначающие ДТП, и писать комментарии. К сожалению, доступ к этим данным Яндекс не предоставляет.

Чтобы данные были свежими, будем брать данные из открытых ис-

точников, например, из группы Вконтакте(ДТП и ЧП | Санкт-Петербург | Питер Онлайн | СПб), так как это крупнейшая группа посвященная ДТП Санкт-Петербурга. Любой пользователь может опубликовать здесь новость об аварии или происшествии, чтобы другие люди знали об этом. В этом сообществе публикуют новости об авариях каждый день, но нет визуального отображения на карте и зачастую неясно, где произошла авария, также информация не агрегирована и нет возможности получать количественную статистику.

Для отображения данных на карте, требуется иметь информацию о координатах ДТП. Координаты можно получить из адреса, а адрес нужно извлечь из текста сообщения в группе.

Впервые задача распознавания объектов в тексте была сформулирована на шестой конференции Message Understanding (MUC-6) в 1996 году. В процессе постановки задач в области извлечения информации появилась отдельная задача, которая заключалась в извлечении объектов из документов. Для определения объекта ввели термин «именованная сущность», а задачу назвали распознавание именованных сущностей (named entity recognition, NER), так её в области обработки естественного языка и называют до сих пор. Сейчас необходимо уметь распознавать в тексте различные сущности: имена, местоположения, названия организаций, числовые выражения, валюты, адреса, даты, время и многое другое. Не так давно на международной конференции по компьютерной лингвистике «Диалог 2016» проводилось соревнование по извлечению информации из текстов на русском языке [9]. Был подготовлен размеченный набор данных из новостных коллекций и решалась задача распознавания именованных сущностей. Необходимо было

распознать следующие классические именованные сущности:

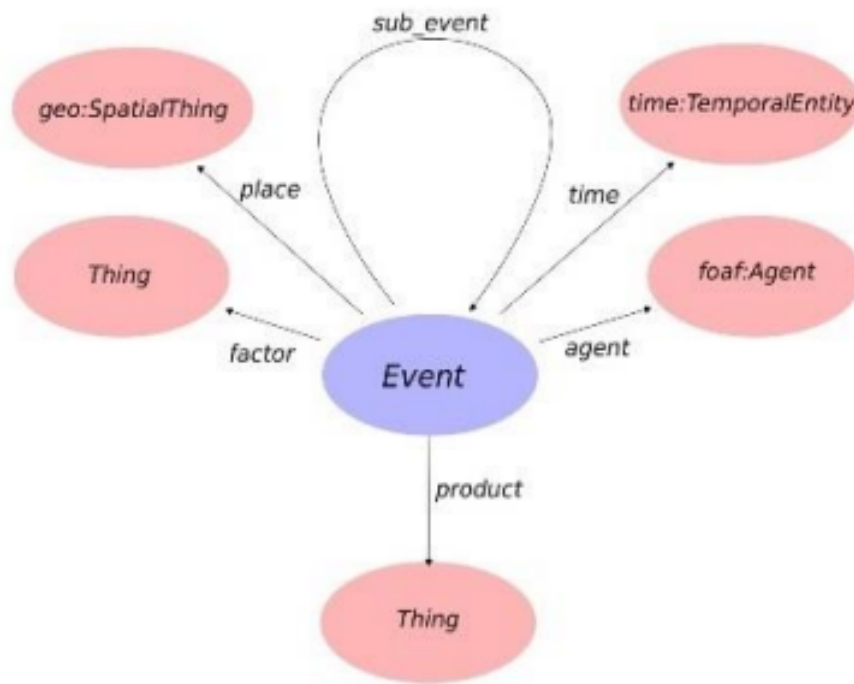
1. Персона (person, PER).
2. Местоположение (location, LOC).
3. Организация (organization, ORG).
4. Остальное (O).

Для решения задачи распознавания именованных сущностей можно использовать подходы, основанные на рукописных правилах, например регулярные выражения. При таком подходе достаточно легко разобраться в причинах ошибки и исправить её, однако, процесс подбора исчерпывающего подбора правил очень ресурсоёмкий. Придумывать правила легче всего для классических именованных сущностей. Однако, правила для всех ситуаций не придумаешь.

Также решать задачу NER можно с помощью онтологий (рис. 2). Онтология — это структура данных, содержащая понятия и объекты, правила и отношения между ними.

Онтология представляется в виде графа, вершины которого это сущности, а ребра – отношения между сущностями. Текст на естественном языке можно представить в виде простых предложений, из которых вы можете извлекать сущность и отношения между ними. Как правило, процесс создания онтологий осуществляется вручную, экспертами и специалистами.

Применение онтологий дает хороший результат в распознавании именованных сущностей, но такие системы совсем не имеют никакой обобщающей способности, их применение в другой доменной области или с новыми возникающими объектами не принесёт никаких результатов.



**Рис. 2:** Онтологии.

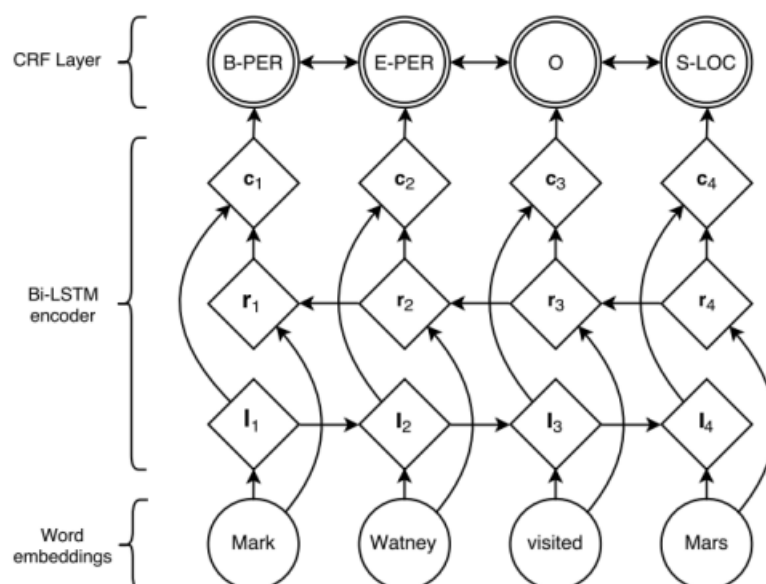
Также при решении задачи NER часто применяют алгоритмы машинного обучения. Признаки для таких алгоритмов нужно придумывать вручную. Классическим алгоритмом для решения NER могут быть деревья принятия решений – альтернатива рукописным правилам, алгоритм сам на основе признаков выучивает наилучшие правила. Использование таких методов подразумевает разбиение текста на токены и представление их в векторном пространстве. Сами вектора уже классифицируются. Такие алгоритмы воспринимают каждый токен в отрыве от всего текста, не учитывают контекст и порядок следования.

Популярным статистическими моделями являются скрытые марковские цепи или условные случайные поля. УСП - ненаправленная графическая модель, используемая для оценки условных вероятностей событий, соответствующих выходным вершинам некоторого графа, при наступлении некото-

рых событий, соответствующих входным вершинам. В линейной цепи УСП цепь образуют выходные вершины. Проблемы таких систем заключается в частом переобучении и сложности извлечения хороших признаков.

В настоящее время очень популярны решения, основанные на нейронных сетях. Открытым стоит вопрос выбора признаков для нейронной сети: ручные правила, морфологические, синтаксические и семантические. Неотъемлемой частью задач обработки естественного языка стало использование векторного представления слов, полученного из различных обученных моделей языка. Как известно, нейронная сеть умеет хорошо автоматически извлекать репрезентативные признаки. Также рекуррентная нейронная сеть позволяет получить из текста контекстно-чувствительное представление. Однако, для качественной работы нейронной сети требуется большая обучающая выборка. Для решения задачи распознавания именованных сущностей современная архитектура нейронной сети состоит обычно из нескольких двунаправленных рекуррентных слоёв [10]. Далее признаки с последнего слоя нейронной сети можно подать в другой классический алгоритм [11]. В примере ниже представлен слой условных случайных полей (CRF Layer), в качестве признаков к которому подаются веса с выхода последнего слоя двунаправленной нейронной сети (рис. 3)

В работе «Pattern-Based Extraction of Addresses from Web Page Content» [12] решается задача извлечения адресов с веб-сайтов. Проблема решается довольно примитивным способом. Стандартизация форматов слов. Многие слова и названия имеют разные написания, сокращения и варианты названий или синонимы. Программа поиска знаний сначала стандартизирует или преобразует различные варианты в их предварительно выбранные форматы.



**Рис. 3:** Пример шаблонного решения на примере нейронной сети.

Многие названия мест написаны последовательно без какого-либо разделительного символа. Средство поиска знаний разделяет элементы на более тонкий уровень на основе предварительно накопленных знаний в базе знаний. Устранение двусмысленности. Поисковик знаний пытается сопоставить название места как можно глубже. Например, если Сент-Люсия уже сохранена в базе данных как уникальное имя, тогда любое непосредственное совпадение Сент, а затем Люсия будет рассматриваться как "Сент-Люсия" вместо Сент <Триггер> + Люсия <Неизвестное слово>. Названия мест собираются на основе адресов, признанных в предыдущей части. Накопитель знаний оценивает, можно ли рассматривать неизвестное слово или фразу (несколько слов) как название места или нет, и если да, то к какому типу места его следует отнести и какой показатель достоверности ему следует присвоить.

Некоторые исследователи [13] склонны использовать онтологию предметной области в качестве концептуальной структуры для извлечения понятий путем объединения человеческих знаний и вычислительных технологий.



Сопоставление слов с набором predetermined понятий (знаний) - это один из способов обнаружить лежащие в основе текста значения. В этом контексте онтология определяет концептуализацию конкретной области знаний, в которой каждое понятие формализуется как узел в онтологии, а связь между понятиями представлена ребром, связывающим узлы понятий. Некоторые подходы предлагают применять модель концептуального пространства для представления документов. Каждый документ нормализуется к вектору концептуальных элементов, и значение каждого элемента определяется частотой концептов. Онтология или тезаурус используются для объединения синонимов и другого связанного синтаксиса. Однако общий вектор не может показать взаимосвязь между понятиями.

Представление документа на основе графов позволяет легко сохранять неявные знания в контексте текстов. Однако преобразование простых текстов в графы - непростая задача, поскольку понятия или их взаимосвязи должны определяться автоматически. Сопоставление графов также является сложной задачей (NP-полной). Эти два недостатка препятствуют применению графовых структур при поиске информации.

Эти традиционные методы разработаны в предположении, что данные не содержат ошибок, что снижает адаптивность архитектуры в реальных сценариях. Кроме того, на их качество влияет наличие нестандартизированных адресов.

В статье «Address entities extraction using named entity recognition» [14] используются более современные методы извлечения адресов, такие как нейронные сети. Чтобы обучить данные распознаванию именованных объектов, нужно, чтобы слова из набора данных были переведены в формат, который

читается и понятен компьютеру. По этой причине слова, представляющие части адресов, преобразуются в векторы с помощью неконтролируемого алгоритма обучения GloVe [15]. Эта процедура называется моделью “слово в вектор”. Используя Tensorflow, библиотеку машинного обучения на Python, эти векторы обучаются с помощью рекуррентной нейронной сети. Тип используемой нейронной сети - это LSTM [21], поскольку она подходит для захвата контекста в тексте и часто используется также при переводе языка.

Но самым современным и эффективным решением задачи распознавания именованных сущностей в тексте является нейронная сеть с архитектурой трансформер. В статье BERT (Bidirectional Encoder Representations from Transformers) [18], опубликованной исследователями Google AI Language, описывается алгоритм обучения нейронной сети, а также проводится эксперимент для сравнения с существующими методами NER. Метрикой качества была выбрана мера F1. По итогам тестов BERT показывала наилучший результат среди всех своих конкурентов.

Задача NER традиционна и хорошо изучена, особенно для английского языка. Существует большое количество как коммерческих так и открытых решений, например, NLTK, Spacy, Stanford NER, OpenNLP и другие. Для русского языка тоже существует довольно много инструментов, но почти все они являются коммерческими (DaData, Abbyy Infoextractor, Dictum).

SDK Pullenti [16] - библиотека для C#, в последствии адаптированная на другие языки программирования, предназначенная для анализа неструктурированной информации на русском и украинском языках. Помимо лингвистического, морфологического и семантического анализа, SDK может выделять именованные сущности в тексте. Pullenti может выделять более 10

типов сущностей, помимо стандартных PER, ORG, LOC, это могут быть даты, номера телефонов, валюты. Библиотека имеет отдельную компоненту для выявления адресов. SDK Pullenti Address - выделение из текста адресов, привязка их к объектам ГАР ФИАС. Для своей работы SDK обращается не к внешнему ресурсу, а к локальному индексу, в который преобразуются объекты ГАР ФИАС (из файлов формата xml). Также SDK предлагает разные поисковые возможности по объектам ГАР - по реквизитам, части именованных и пр. Через SDK можно получить полную иерархию всех объектов. Библиотека является закрытой и полное использование SDK возможно только в коммерческой версии.

Из открытых инструментов отметим: Deep Pavlov [17] - это фреймворк с открытым исходным кодом для разработки чат-ботов и виртуальных помощников. Он обладает всеобъемлющими и гибкими инструментами, которые позволяют разработчикам и исследователям NLP создавать готовые к работе разговорные навыки и сложные разговорные помощники с несколькими навыками. Deep Pavlov для задачи NER использует архитектуру Google BERT [18]. BERT (Bidirectional Encoder Representations from Transformers) — языковая модель, основанная на архитектуре трансформер, предназначенная для предобучения языковых представлений с целью их последующего применения в широком спектре задач обработки естественного языка.

Библиотека Natasha [19] решает базовые задачи обработки естественного русского языка: сегментация на токены и предложения, морфологический и синтаксический анализ, лемматизация, извлечение именованных сущностей. Является аналогом DeepPavlov и представляет из себя BERT NER + дистилляция через синтетическую разметку в WordCNN-CRF с кван-

тованными эмбедингами + движок для инференса на NumPy.

Stanza [20] - это набор точных и эффективных инструментов для лингвистического анализа многих естественных языков. Начиная с необработанного текста и заканчивая синтаксическим анализом и распознаванием сущностей, Stanza предлагает самые современные модели NLP для языков по вашему выбору. Для NER используется контекстуализированный последовательный теггер. Сначала обучается языковая модель LSTM прямого и обратного уровня символов для формирования эмбедингов, а после эмбединги передаются в модель Bi-LSTM с декодером CRF.

# Глава 1. Алгоритмы и архитектуры

## 1.1 Методы распознавания именных сущностей

### 1.1.1 Stanza

Основой алгоритма Stanza является нейронная сеть архитектуры LSTM [21]. LSTM — особая разновидность архитектуры рекуррентных нейронных сетей (рис. 4), способная к обучению долговременным зависимостям. Рекуррентные нейронные сети добавляют память к искусственным нейронным сетям, но реализуемая память получается короткой — на каждом шаге обучения информация в памяти смешивается с новой и через несколько итераций полностью перезаписывается.

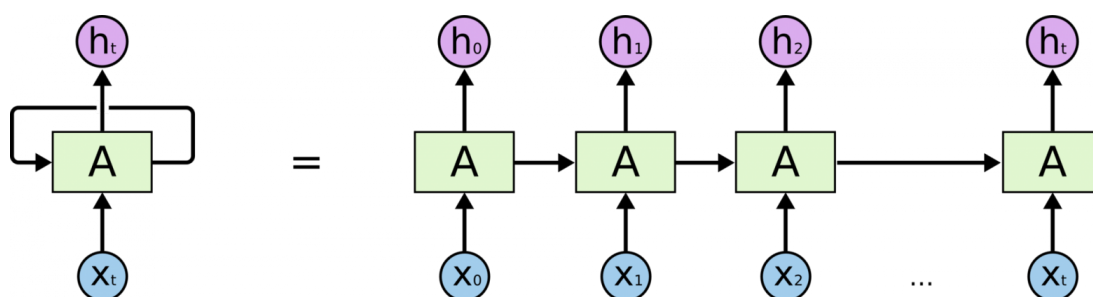


Рис. 4: Рекуррентная нейронная сеть.

LSTM-модули (рис. 5) разработаны специально, чтобы избежать проблемы долговременной зависимости, запоминая значения как на короткие, так и на длинные промежутки времени. Это объясняется тем, что LSTM-модуль не использует функцию активации внутри своих рекуррентных компонентов. Таким образом, хранимое значение не размывается во времени и градиент не исчезает при использовании метода обратного распространения ошибки во времени при тренировке сети.

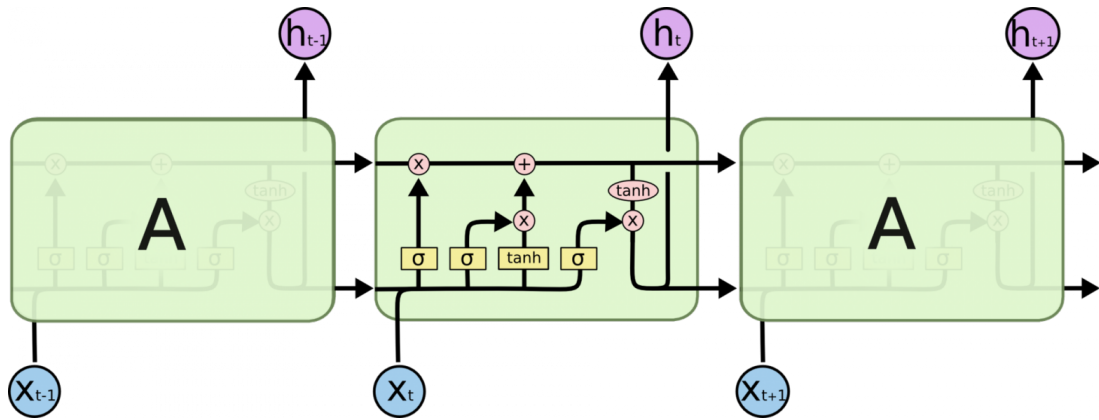


Рис. 5: LSTM.

Текст рассматривается как последовательность символов. Символы используются в качестве атомарных единиц языкового моделирования. LSTM в каждой точке последовательности обучается предсказывать следующий символ. Начиная с первого символа, на вход модели подается сам символ  $x_t$  и значения предыдущего выхода  $h_{t-1}$ .

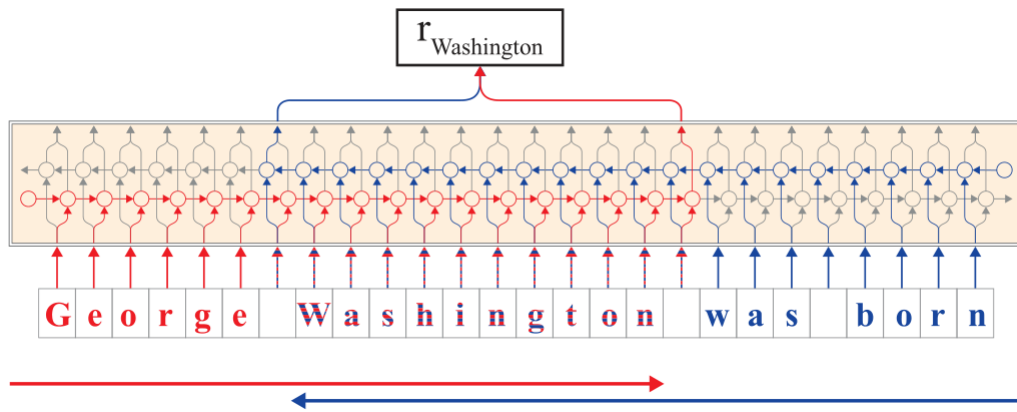
В Stanza обучаются две модели LSTM прямой направленности и обратной [22]. Обратная направленная LSTM принимает на вход последовательность символов в обратном порядке.

Далее определяются контекстные строковые эмбединги  $w_i^{CharLM}$  как вектор из двух значений (рис. 6).

$$w_i^{CharLM} := \begin{bmatrix} h_{t_{i+1}-1}^f \\ h_{t_i-1}^b \end{bmatrix}$$

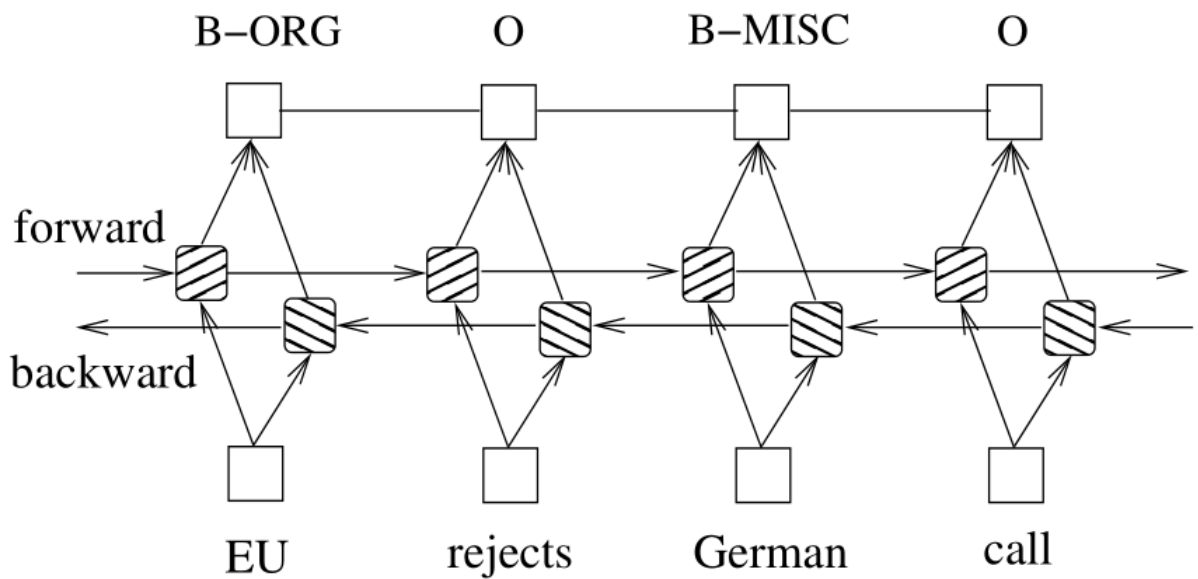
$h_{t_{i+1}-1}^f$  - значение выхода сети LSTM прямой направленности для последнего символа слова.

$h_{t_i-1}^b$  - значение выхода сети LSTM обратной направленности для символа перед первым символом слова.



**Рис. 6:** Контекстный строковый эмбединг.

На следующем этапе, для маркирования последовательностей эмбедингов слов передаются в BiLSTM-CRF сеть (рис. 7) [23].



**Рис. 7:** BiLSTM-CRF.

Подавая на вход в BiLSTM эмбединги слов  $w_0, \dots, w_n$ , на выходе получаем:

$$r_i := \begin{bmatrix} r_i^f \\ r_i^b \end{bmatrix}$$

Где  $r_i^f$  и  $r_i^b$  являются прямыми и обратными выходными состояниями BiLSTM.

Затем конечная вероятность последовательности задается CRF по возможным меткам последовательности  $y$ :

$$\hat{P}(y_{0:n}|r_{0:n}) \propto \prod_{i=1}^n \psi_i(y_{i-1}, y_i, r_i)$$

Где:

$$\psi_i(y', y, r) = \exp(W_{y',y}r + b_{y',y})$$

### 1.1.2 Deep Pavlov

Внутри фреймворка Deep Pavlov [17] есть два решения задачи NER. Первое является комбинированной нейронной сетью, состоящей из BiLSTM, сверточной нейронной сети (CNN) и условных случайных полей (CRF) [24]. В основе второго лежит нейронная сеть от Google под названием BERT [18]. DeepPavlov дообучают BERT для задачи распознавания именных сущностей на русском языке с помощью коллекции Collection5 [25].

Ключевым техническим новшеством BERT является применение двунаправленного обучения архитектуры Transformer, популярной модели внимания, к моделированию языка.

В модели есть два этапа: предварительная подготовка и тонкая настройка. Во время предварительного обучения модель обучается на немаркированных данных по различным задачам предварительного обучения. Для



точной настройки модель BERT сначала инициализируется предварительно обученными параметрами, и все параметры настраиваются с использованием помеченных данных из последующих задач. Каждая последующая задача имеет отдельные точно настроенные модели, даже если они инициализируются с одними и теми же предварительно подготовленными параметрами. Отличительной особенностью BERT является его унифицированная архитектура для различных задач. Существует минимальная разница между предварительно обученной архитектурой и окончательной нисходящей архитектурой.

Архитектура модели BERT представляет собой многослойный двунаправленный энкодер трансформера, основанный на оригинальной реализации, описанной в [26]. Для того, чтобы BERT мог выполнять множество задач, входное представление способно однозначно представлять как одно предложение, так и пару предложений в одной последовательности токенов. “Предложение” может быть произвольным отрезком непрерывного текста, а не фактическим лингвистическим предложением. “Последовательность” относится к последовательности входных токенов для BERT, которая может быть одним предложением или двумя предложениями, упакованными вместе. В модели используются эмбединги WordPiece [27] со словарем из 30 000 токенов. Первый токен каждой последовательности всегда является специальным классификационным токеном ([CLS]). Конечное скрытое состояние, соответствующее этому токenu, используется в качестве представления совокупной последовательности для задач классификации. Пары предложений упакованы вместе в единый эмбединг. Предложения различаются двумя способами. Сначала разделяем их с помощью специального

токена ([SEP]). Во-вторых, добавляем изученный эмбединг к каждому токeну, указывающее, принадлежит ли он предложению А или предложению В. Обозначаем эмбединг ввода как  $E$ , конечный скрытый вектор специального токeна [CLS] как  $C \in \mathbb{R}^H$ , а конечный скрытый вектор для  $i$ -го входного токeна как  $T_i \in \mathbb{R}^H$ . Для данного токeна его входное представление строится путем суммирования соответствующих эмбедингов токeна, сегмента и позиции. Визуализацию этой конструкции можно увидеть на рисунке 8.

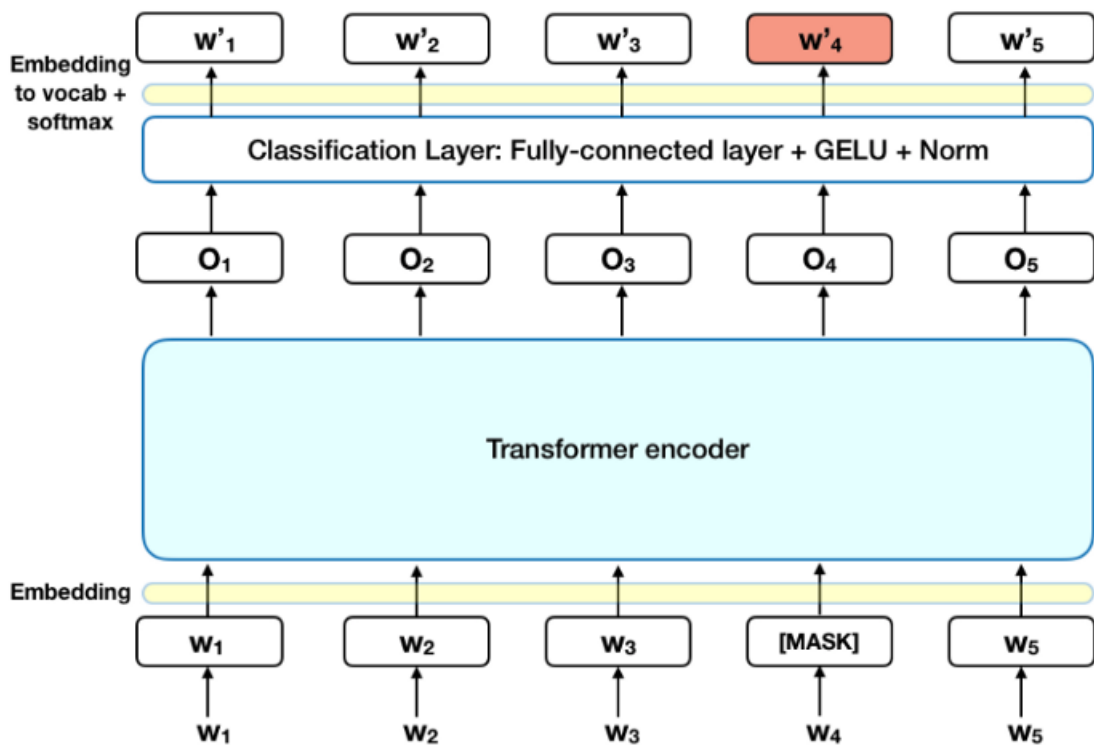


Рис. 8: Трансформаторный энкодер.

Предварительная подготовка модели состоит из двух задач.

Задача 1: Masked LM. Чтобы обучить глубокое двунаправленное представление, просто случайным образом маскируется некоторый процент входных токeнов, а затем прогнозируются эти замаскированные токeны. В этом случае конечные скрытые векторы, соответствующие маркерам маски, по-

даются в выходной softmax поверх словаря, как в стандартном LM. Случайным образом маскируется 15% всех лексем фрагментов слов в каждой последовательности. Прогнозируются только замаскированные слова, а не восстанавливается весь ввод. Хотя это позволяет получить двунаправленную предварительно обученную модель, недостатком является то, что создается несоответствие между предварительным обучением и точной настройкой, поскольку маркер [MASK] не появляется во время точной настройки. Чтобы смягчить это, заменяется  $i$ -й токен токеном [MASK] в 80% случаев, случайным токеном в 10% случаев, не изменяется в 10% случаев.

Задача 2: Предсказание следующего предложения (NSP). Чтобы обучить модель, которая понимает взаимосвязи предложений, предварительно готовимся к бинаризованной задаче прогнозирования следующего предложения, которая может быть тривиально сгенерирована из любого одноязычного корпуса. В частности, при выборе предложений A и B для каждого примера предварительного обучения в 50% случаев B является фактическим следующим предложением, которое следует за A (помечено как IsNext), и в 50% случаев это случайное предложение из корпуса (помечено как NotNext).

Для предтренировочного корпуса используется Books Corpus (800 млн слов) [28] и английская Википедия (2500 млн слов). Для Википедии извлекаются только текстовые фрагменты и игнорируются списки, таблицы и заголовки.

Точная настройка проста, поскольку механизм самоконтроля в Transformer позволяет BERT моделировать множество последующих задач - независимо от того, связаны ли они с одним текстом или текстовыми парами — путем замены соответствующих входных и выходных данных. Используя BERT,

модель NER может быть обучена путем подачи выходного вектора каждого токена в слой классификации, который предсказывает метку NER.

### 1.1.3 Natasha

Идея библиотеки Natasha заключается в получении функционала BERT, но существенно снизив размер модели и увеличив скорость ее работы, при этом не потеряв в качестве.

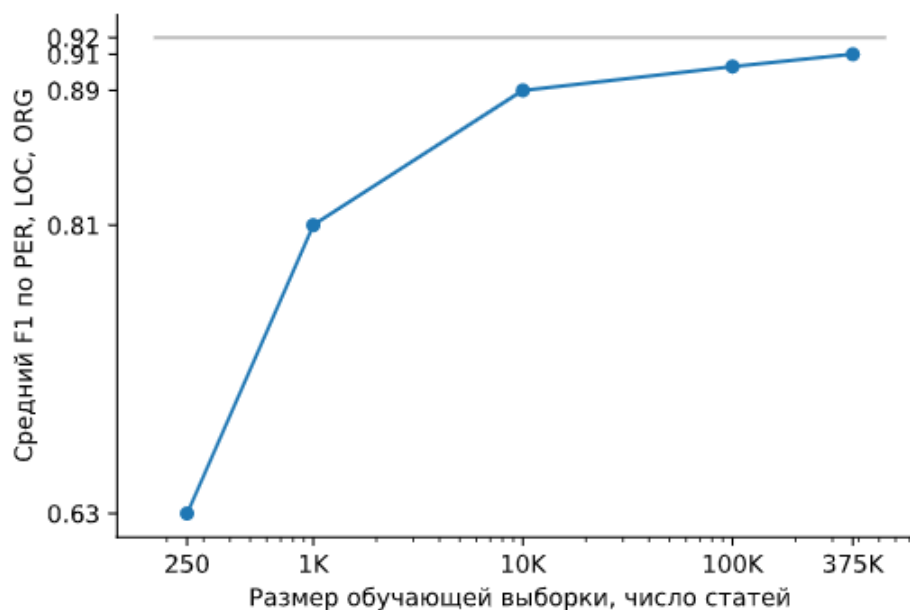
В части предварительно подготовки BERT были применены техники из статьи RoBERTA [29]. Используются большие агрегированные батчи, применяется динамическая маска, отказ от предсказания следующего предложения (NSP). Динамическая маска заключается в том, что одно предложение используется в модели 10 раз и каждый раз маска на новом месте. Также предварительная подготовка теперь проводится на корпусе русскоязычных новостей объемом 12 Гб [30].

Точная настройка модели происходит также, как и в DeepPavlov, с помощью коллекции Collection5 [25]. Эта модель получила название Slovnet BERT NER.

Теперь с помощью полученной модели, формируется синтетический датасет [31] из 700 000 новостей из корпуса Lenta.ru [32]. На этих данных обучается более простая нейронная сеть с архитектурой (WordEmbedding + ShapeEmbedding)-WordCNN-CRF. Модель назвали Slovnet NER

В статье указывается, что значение метрики F1 для модели Slovnet NER - 0.91, а Slovnet BERT NER - 0.92.

Также утверждается, что размер модели уменьшен в 10 раз, а скорость работы увеличена в 3 раза.



**Рис. 9:** Зависимость качества модели от размера обучающей выборки.

## 1.2 Метрики оценки качества

При оценке модели извлечения пользовательских сущностей используются следующие метрики: Точность – это процент правильных именованных объектов системой NER от всех именованных объектов системой NER, а полнота – это процент правильных именованных объектов системой NER от фактического числа существующих сущностей.

Наиболее подходящей метрикой качества модели распознавания именованных сущностей, является F1-мера (1). Впервые метрика была предложена на конференции Conference on Computational Natural Language Learning [33] и в дальнейшем была принята за стандарт в оценке качества моделей NER.

Именованная сущность считается правильно выделенной, если правильно определен ее класс и границы.

$$F1 = \frac{2 * precision * recall}{precision + recall} \quad (1)$$

Где

$$precision = \frac{\text{количество верно выделенных сущностей}}{\text{количество верно и неверно выделенных сущностей}}$$

$$recall = \frac{\text{количество верно выделенных сущностей}}{\text{количество сущностей в корпусе}}$$

## Глава 2. Данные

### 2.1 Collection5

Размеченная коллекция Collection5 [25] создана для оценки качества алгоритмов автоматического извлечения именованных сущностей из текстов на русском языке.

В качестве основы для разметки этих коллекций взята коллекция Persons-1000, подготовленная Исследовательским центром Искусственного интеллекта Института программных систем РАН. В коллекции Persons-1000 размечены только имена собственные.

В коллекции Collection5 размечены следующие типы именованных сущностей:

1. Имена людей (Per)
2. Места (Loc)
3. Государства (Geopolit)
4. Организации (Org)
5. Источники информации (Media)

Данные представляют из себя 1000 текстов (файлы \*.txt). Для каждого текста есть файл (\*.ann) с указанием для каждой именованной сущности ее тип, границы и сама сущность.

## 2.2 BSNLP-2019

BSNLP-2019 [34] – 2-е издание общей задачи по распознаванию многоязычных именованных объектов. Издание общей задачи 2019 года охватывает четыре языка:

1. Болгарский
2. Чешский
3. Польский
4. Русский

и фокусируется на распознавании пяти типов именованных объектов, включая:

1. Люди
2. Места
3. Организации
4. Продукты
5. События

Задача фокусируется на извлечении именованных объектов на уровне межъязыкового документа, т.е. системы должны распознавать, классифицировать и извлекать все упоминания именованных объектов в документе, но определение положения каждого упоминания именованных объектов в тексте не требуется. Кроме того, упоминания именованных объектов должны



быть лемматизированы, а упоминания, относящиеся к одному и тому же объекту реального мира, должны быть связаны между документами и языками. Коллекция входного текста состоит из наборов новостных статей из онлайн-СМИ, каждая коллекция вращается вокруг определенного объекта или события. Корпус был получен путем обхода веб-страниц и синтаксического анализа HTML соответствующих документов.

## 2.3 FactRuEval 2016

FactRuEval 2016 [35] – соревнование по выделению именованных сущностей, выделению именованных сущностей с атрибутами и извлечению фактов.

Файлы коллекции:

1. \*.txt - тексты документов
2. \*.tokens - деление на токены и предложения
3. \*.spans - спаны (первый слой разметки)
4. \*.objects - упоминания объектов (второй слой разметки)
5. \*.coref - кореференция и идентификация (третий слой разметки)

Упоминания объектов (\*.objects) Каждая строка - одно упоминание объекта. Разделитель полей - пробел.

Поля:

1. id упоминания
2. тип упоминания

3. список идентификаторов входящих в упоминание спанов
4. текст всех входящих в упоминание объекта спанов

## Глава 3. Реализация

### 3.1 Выбор технологий и инструментов для реализации

Для программной реализации интерактивной карты ДТП Санкт-Петербурга были выбраны следующие инструменты:

- Python3 – скриптовый динамический язык программирования, выбран для реализации методов и использования готовых библиотек.
- Jupyter Notebook – веб-приложение для написания кода на языке Python с возможностью написания и запуска кода блоками, а не целиком.
- Pandas [36] – это инструмент для анализа и манипулирования данными с открытым исходным кодом, построен на основе язык а программирования Python.
- DeepPavlov [37] – это библиотека разговорного искусственного интеллекта с открытым исходным кодом, построенная на TensorFlow, Keras и PyTorch.
- Stanza [20] – это пакет Python анализа естественного языка. Модули построены поверх библиотеки PyTorch.
- Natasha [19] – библиотека Python с реализацией основных задач NLP для русского языка: токенизация, сегментация предложений, встраивание слов, маркировка морфологии, лемматизация, нормализация фраз, синтаксический анализ, NER-теги, извлечение фактов.
- Flask [38] – фреймворк для создания веб-приложений на языке программирования Python.

- Requests [39] – это элегантная и простая HTTP-библиотека для Python.
- API ВКонтакте [40] – это интерфейс, который позволяет получать информацию из базы данных vk.com с помощью http-запросов к специальному серверу.
- Yandex Геокодер [41] – API для перевода географических координат в адрес и наоборот.
- Folium [42] позволяет легко визуализировать данные, обработанные на Python, на интерактивной карте.

### 3.2 Сравнение алгоритмов NER

Для сравнения методов была выбрана F1-мера, использовались реализации алгоритмов на Python и данные описанные в Главе 2.

Алгоритмы проверялись на качество выявления сущностей: люди, организации, локации.

PER	Collection5	BSNLP-2019	FactRuEval 2016
Slovnet NER	0.984	0.944	0.959
Slovnet BERT NER	0.996	0.960	0.973
DeepPavlov BERT	0.997	0.954	0.971
stanza	0.923	0.938	0.943
pullenti	0.952	0.900	0.905

ORG	Collection5	BSNLP-2019	FactRuEval 2016
Slovnet NER	0.951	0.718	0.825
Slovnet BERT NER	0.976	0.733	0.831
DeepPavlov BERT	0.976	0.741	0.825
stanza	0.734	0.724	0.687
pullenti	0.683	0.566	0.686

LOC	Collection5	BSNLP-2019	FactRuEval 2016
Slovnet NER	0.973	0.834	0.915
Slovnet BERT NER	0.989	0.838	0.928
DeepPavlov BERT	0.990	0.840	0.928
stanza	0.753	0.838	0.865
pullenti	0.862	0.769	0.814

В большинстве случаев, наилучший результат показывает модель Deep Pavlov BERT.

### 3.3 Сбор данных

Для получения данных о ДТП использовалась группа Вконтакте (ДТП и ЧП | Санкт-Петербург | Питер Онлайн | СПб). С помощью API Вконтакте передается get запрос на список записей из группы. Далее результат запроса передается парсеру, который был написан на языке Python, Парсер выявляет такие данные, как id записи, дата публикации и текст публикации. Таким образом получился датасет размером 145 020 записей, первая запись датируется 2014-03-20, последняя 2022-04-07.

### 3.4 Выявление адреса в тексте

Для выявления адреса в тексте использовался фреймворк DeepPavlov [17] на основе языковой модели BERT [18] предобученной на Collection3 датасете для русского языка.

Collection3 является частью датасета Collection5 [25], но с только тремя размеченными типами сущностей.

Для задачи определения сущностей BERT умеет определять 3 типа сущностей: ORG - компании, организации, учреждения; LOC - не геополитические локации, горные и водные объекты; PER - люди; O - отсутствие сущности.

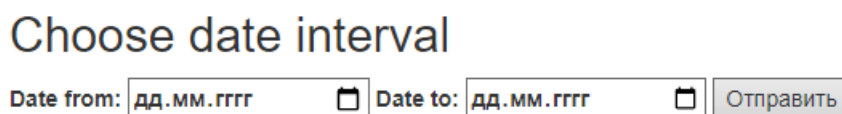
Пример работы на предложении: "Фольксваген и маршрутка столкнулись на перекрёстке Кондратьевского и Минеральной. Гайцы на месте"

Фольксваген	O
и	O
маршрутка	O
столкнулись	O
на	O
перекрёстке	O
Кондратьевского	B-LOC
и	O
Минеральной	B-LOC
.	O
Гайцы	O
на	O
месте	O

Алгоритм применяется на собранный датасет, извлекая сущности относящиеся только к локации. В датасет добавляется еще одно поле, в котором хранятся части текста публикации, определенные как локация.

### 3.5 Визуализация данных

На языке Python с помощью библиотеки Flask было разработано веб-приложение. Приложение можно запустить локально на компьютере, но также приложение было развернуто на хостинге и находится по ссылке [traffic-accident-map.xuz/](http://traffic-accident-map.xuz/). Приложение представляет из себя два поля для ввода дат (рис. 10). Даты обозначают временной промежуток, за который будет сгенерирована карта.



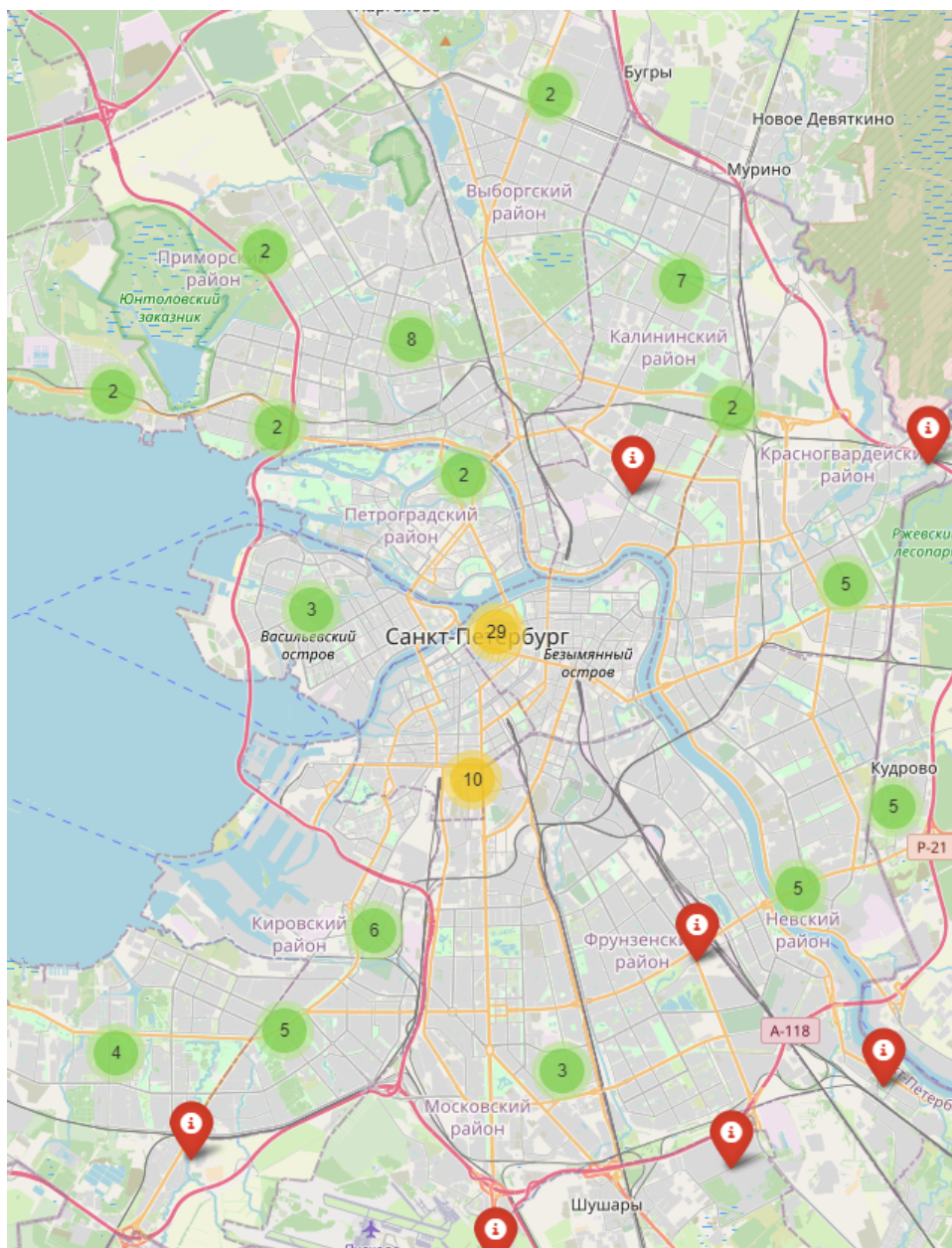
The image shows a web form with the title "Choose date interval". Below the title, there are two input fields for dates. The first is labeled "Date from:" and the second is labeled "Date to:". Both fields contain the placeholder text "дд.мм.гггг" and have a small calendar icon to their right. To the right of the second date field is a button labeled "Отправить" (Submit).

**Рис. 10:** Интерфейс выбора дат.

Датасет фильтруется по полученным из интерфейса датам.

Далее, в API `yangex geocoder` [41] передается `get` запрос, содержащий выявленные сущности, обозначающие локации, а из ответа запроса мы получаем географические координаты (ширина и долгота).

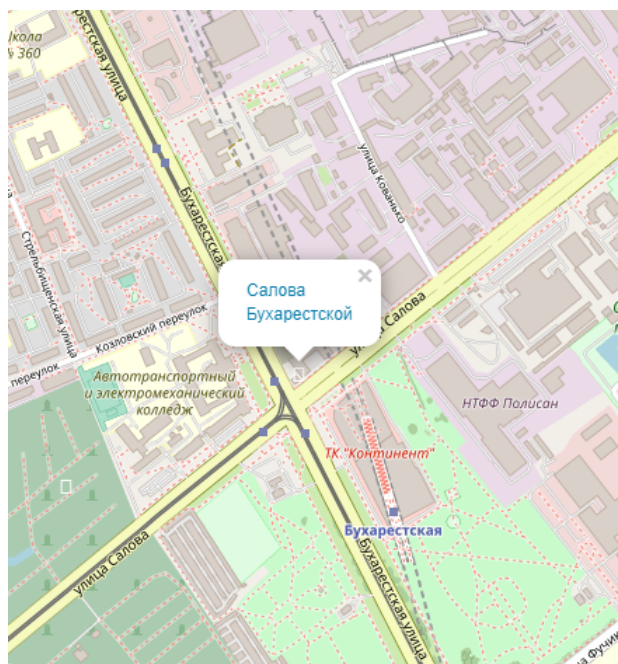
На последнем этапе, с помощью библиотеки `folium` [42] для Python генерируется карта `OpenStreetMap` [43] (рис. 11).



**Рис. 11:** Карта с отметками ДТП.

В библиотечную функцию передаются координаты для отображения меток ДТП. Для каждой координаты на карте добавляется метка. При уменьшении масштаба карты, метки группируются. Также на при нажатии на метку отображается сущности, которые алгоритм определил как локацию и ссылка на новость в группе Вконтакте, к которой относится метка (рис. 12).

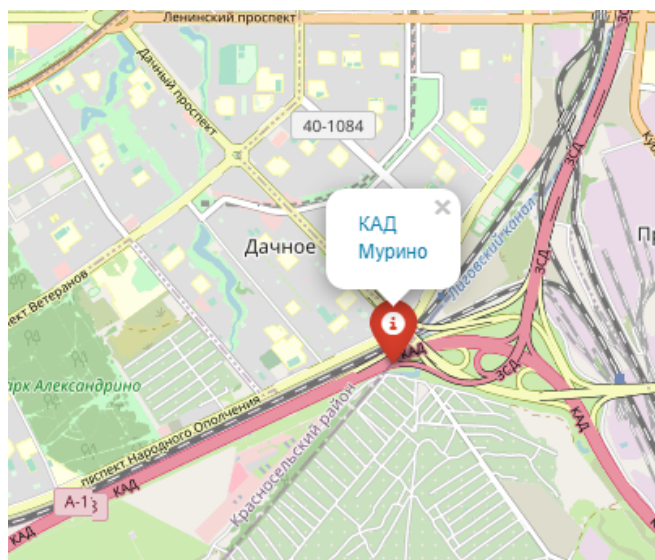




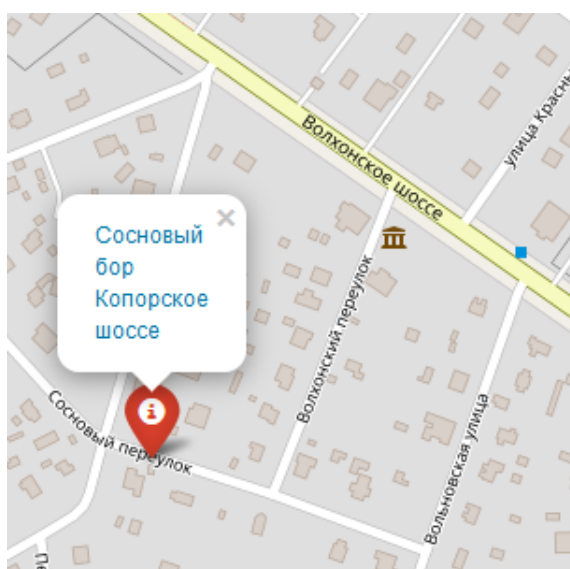
**Рис. 12:** Именованная сущность и ссылка на новость.

Карта получилась рабочей, но бесплатная версия API yandex geocoder предоставляет только 1000 запросов в сутки, что достаточно мало, при генерации карты за большой промежуток времени.

Также геокодер иногда допускает ошибки, например, когда передается несколько названий улиц, которые не имеют общего пересечения или геокодер его не нашел, в качестве ответа будут отправлены координаты одной из этих улиц. На рис. 13 геокодер не нашел пересечения Мурино и КАДа, и точка была поставлена на КАД в красносельском районе. А иногда геокодер находит улицу с похожим названием. Так на рис. 14 точка поставлена на Сосновом переулке, хотя в новости имеется ввиду Копорское шоссе в Сосновом бору.



**Рис. 13:** Не нашлось пересечения КАД и Мурино и точка поставлена неправильно.



**Рис. 14:** Точка поставлена на схожее название улицы.

## **Заключение**

### **Итоги работы**

В рамках работы были выполнены следующие задачи:

- Обзор существующих методов решения проблемы;
- Обзор литературы на тему распознавания именных сущностей;
- Формирование коллекции данных о ДТП;
- Сравнение методов распознавания именных сущностей.
- Реализация интерактивной карты Санкт-Петербурга;

Цель работы достигнута, реализовано веб-приложение, которое получает данные из открытого источника, группы Вконтакте, выявляет сущности обозначающие местоположения и строит визуальное отображение данных на карте.

### **Практическое применение**

Реализованный продукт можно применить для аналитики и выявления опасных участков дороги, анализ зависимости аварийности на определенных участках от времени суток, времени года и погодных условий. Также после анализа аварийности участков можно оптимизировать построение маршрутов в навигаторах.

## **Дальнейшее развитие**

Требуется улучшить работу геокодера. Найти открытый геокодер с большим количеством бесплатных запросов или реализовать собственный геокодер.

В дальнейшем можно расширить проект на другие города. Также можно попробовать обратиться к государственным структурам, с целью получения доступа к официальным данным по дорожно-транспортным происшествиям. Возможно написание мобильного приложения, либо сотрудничество с существующими приложениями карт, с целью интеграции функционала карты ДТП.

## Список литературы

- [1] Единая межведомственная информационно-статистическая система: [fedstat.ru/indicator/36228](http://fedstat.ru/indicator/36228).
- [2] Статистика аварийности за 2021 год от МВД: [media.mvd.ru/files/embed/2256058](http://media.mvd.ru/files/embed/2256058).
- [3] B. JIANG, C. LIU «Street-based topological representations and analyses for predicting traffic flow in GIS». International Journal of Geographical Information Science Vol. 23, No. 9, September 2009, 1119–1137.
- [4] S.C. LEE «Road traffic monitoring in Hong Kong». Proceedings of the Second International Conference on Road Traffic Monitoring, London, UK, 1989, pp. 14–18.
- [5] Gabriele Filomena, Judith A. Versteegen, Ed Manley «A computational approach to 'The Image of the City'». Cities Volume 89, June 2019, Pages 14-25.
- [6] K. Lynch «The image of the city». Cambridge, MA: MIT Press 1960.
- [7] S. Law «Defining street-based local area and measuring its effect on house price using a hedonic price approach: The case study of Metropolitan London». Cities Volume 60, 2017, 166–179.
- [8] A. Turner «From axial to road-centre lines: A new representation for Space Syntax and a new model of route choice for transport network analysis». Environment and Planning B: Planning and Design, 34(3), 2007, 539–555.

- [9] Международная конференция по компьютерной лингвистике «Диалог 2016»: [github.com/dialogue-evaluation/factRuEval-2016](https://github.com/dialogue-evaluation/factRuEval-2016).
- [10] Lample G. et al. «Neural architectures for named entity recognition». arXiv preprint arXiv:1603.01360. – 2016.
- [11] Lafferty J., McCallum A., Pereira F. C. N «Conditional random fields: Probabilistic models for segmenting and labeling sequence data». – 2001.
- [12] Saeid Asadi, Guowei Yang, Xiaofang Zhou, Yuan Shi, Boxuan Zhai, and Wendy Wen-Rong Jiang «Pattern-Based Extraction of Addresses from Web Page Content». Lecture Notes in Computer Science, 407–418.
- [13] Wentao Cai, Shengrui Wang, and Qingshan Jiang «Address Extraction: Extraction of Location-Based Information from the Web». APWeb 2005, LNCS 3399, pp. 925–937, 2005.
- [14] Kanita Krdžalić-Korić, Emine Yaman «Address entities extraction using named entity recognition». INTERNATIONAL JOURNAL OF COMPUTERS Volume 13, 2019.
- [15] Jeffrey Pennington, Richard Socher, and Christopher D. Manning «GloVe: Global Vectors for Word Representation». 2014 [nlp.stanford.edu/projects/glove/](http://nlp.stanford.edu/projects/glove/).
- [16] Официальная сайт с документацией SDK Pullenti: [www.pullenti.ru/Document](http://www.pullenti.ru/Document).

- [17] Mikhail Burtsev, Alex Seliverstov et al. «DeepPavlov: Open-Source Library for Dialogue Systems». Moscow Institute of Physics and Technology / 9 Institutskiy per., Dolgoprudny, 141701, Russian Federation.
- [18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». Google AI Language arXiv:1810.04805.
- [19] Статья о работе библиотеки Natasha для извлечения именованных сущностей: [natasha.github.io/ner/](https://natasha.github.io/ner/).
- [20] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, Christopher D. Manning «Stanza: A Python Natural Language Processing Toolkit for Many Human Languages». ACL2020 System Demonstration. First two authors contribute equally. arXiv:2003.07082.
- [21] Sepp Hochreiter, Jürgen Schmidhuber «Long Short-term Memory». Neural computation. 1997. 9. 1735-80.
- [22] Alan Akbik, Duncan Blythe, Roland Vollgraf «Contextual String Embeddings for Sequence Labeling». Proceedings of the 27th International Conference on Computational Linguistics. 1638–1649.
- [23] Zhiheng Huang, Wei Xu, Kai Yu «Bidirectional LSTM-CRF Models for Sequence Tagging». arXiv:1508.01991.
- [24] The Anh Le, Mikhail S. Burtsev «A Deep Neural Network Model for the Task of Named Entity Recognition». International Journal of Machine Learning and Computing, Vol. 9, No. 1, February 2019.

- [25] Можарова В.А., Лукашевич Н.В. «Двухэтапный подход к извлечению именованных сущностей». Труды конференции по искусственному интеллекту КИИ-2016, т.2., 2016. - С.81-88.
- [26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin «Attention Is All You Need». *Advances in Neural Information Processing Systems*, pages 6000–6010.
- [27] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. «Google’s neural machine translation system: Bridging the gap between human and machine translation». arXiv:1609.08144.
- [28] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. «Aligning books and movies: Towards story-like visual explanations by watching movies and reading books». *Proceedings of the IEEE international conference on computer vision*, 2015, pages 19–27.
- [29] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov «RoBERTa: A Robustly Optimized BERT Pretraining Approach». arXiv:1907.11692.
- [30] Коллекция ссылок на публичные русскоязычные датасеты: [github.com/natasha/corus](https://github.com/natasha/corus).
- [31] Большой корпус с автоматической разметкой именованных сущностей, морфологии и синтаксиса: [github.com/natasha/nerus](https://github.com/natasha/nerus).



- [32] Репозиторий с новостями с сайта Lenta.ru: [github.com/yutkin/Lenta.Ru-News-Dataset](https://github.com/yutkin/Lenta.Ru-News-Dataset).
- [33] Erik F. Tjong Kim Sang and Fien De Meulder «Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition». HLT-NAACL 2003 - Volume 4, May 2003, 142–147.
- [34] Piskorski, Jakub and Laskova, Laska and Marcińczuk, Michał and Pivovarov, Lidia and Přibáň, Pavel and Steinberger, Josef and Yangarber, Roman «The Second Cross-Lingual Challenge on Recognition, Normalization, Classification, and Linking of Named Entities across Slavic Languages». Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference «Dialogue 2016».
- [35] Starostin A. S., Bocharov V. V., Alexeeva S. V., Bodrova A. A., Chuchunkov A. S., Dzhumaev S. S., Efimenko I. V., Granovsky D. V., Khoroshevsky V. F., Krylova I. V., Nikolaeva M. A., Smurov I. M., Toldova S. Y. «FactRuEval 2016: Evaluation of Named Entity Recognition and Fact Extraction Systems for Russian». Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing.
- [36] The pandas development team pandas: «powerful Python data analysis toolkit». [pandas.pydata.org/docs/pandas.pdf](https://pandas.pydata.org/docs/pandas.pdf).
- [37] Официальная документация на библиотеку DeepPavlov для Python: [docs.deeppavlov.ai/en/master](https://docs.deeppavlov.ai/en/master).
- [38] Официальная документация на библиотеку Flask для Python: [flask.palletsprojects.com/en/2.1.x](https://flask.palletsprojects.com/en/2.1.x).

- [39] Официальная документация на библиотеку requests для Python:  
[docs.python-requests.org/en/latest](https://docs.python-requests.org/en/latest).
- [40] Официальный сайт API V Kontakte: [dev.vk.com/reference](https://dev.vk.com/reference).
- [41] Официальная документация API yandex map:  
[yandex.ru/dev/maps/geocoder/doc/desc/concepts/about.html](https://yandex.ru/dev/maps/geocoder/doc/desc/concepts/about.html).
- [42] Официальная документация библиотеки folium: [python-visualization.github.io/folium](https://python-visualization.github.io/folium).
- [43] Wiki OpenStreetMap: [wiki.osmfoundation.org/wiki/Terms\\_of\\_Use](https://wiki.osmfoundation.org/wiki/Terms_of_Use).
- [44] Репозиторий с исходным кодом текущей работы:  
[github.com/roma0398/traffic\\_accident\\_map](https://github.com/roma0398/traffic_accident_map).