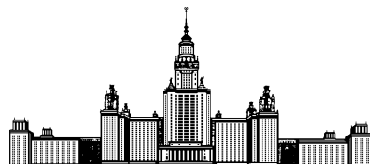


Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики
Кафедра Математических Методов Прогнозирования

ДИПЛОМНАЯ РАБОТА СТУДЕНТА 417 ГРУППЫ

«Построение поисковой системы для интернет магазина»

Выполнил:

студент 4 курса 417 группы

Федоров Илья Сергеевич

Научный руководитель:

д.ф-м.н., профессор

Дьяконов Александр Геннадьевич

Заведующий кафедрой

Математических Методов

Прогнозирования, академик РАН

_____ Ю. И. Журавлёв

К защите допускаю

«_____» _____ 2021 г.

К защите рекомендую

«_____» _____ 2021 г.

Москва, 2021

Содержание

| | | |
|----------|--|-----------|
| 1 | Введение | 3 |
| 2 | Постановка задачи | 3 |
| 3 | Данные | 6 |
| 3.1 | Особенности сбора данных | 6 |
| 3.2 | Описание собранных наборов данных | 7 |
| 4 | Обзор существующих работ | 9 |
| 4.1 | Подход на основе рекуррентных нейросетей | 10 |
| 4.2 | BERT | 14 |
| 5 | Предложенная архитектура | 19 |
| 6 | Обучение модели | 21 |
| 6.1 | Tokenizer | 21 |
| 6.2 | Pre-training | 22 |
| 6.3 | Fine-tuning | 23 |
| 7 | Результаты и примеры работы моделей | 23 |
| 8 | Заключение | 26 |
| 9 | Приложение | 27 |

Аннотация

Todo

1 Введение

Современный технологический прогресс неразрывно связан с быстрым доступом к информации. Ежедневно поисковые системы обрабатывают миллиарды запросов от колоссального количества людей по всему миру, а их базы данных исчисляются десятками миллионов терабайт. И хотя поисковые системы, безусловно, стоит считать одним из главных достижений человечества за последние десятилетия, задача получения данных по запросу возникает и в других приложениях меньших масштабах, к примеру, задача выдача подходящих под запрос пользователя товаров в интернет-магазине.

Основной целью данной работы является исследование подходов к построению поисковой системы для абстрактного интернет-магазина с использованием современных технологий глубокого обучения и нейронных сетей. Безусловно для её решения, существуют (и активно применяются на практике) методы, никак не использующие машинное обучение, однако рост интереса исследователей и бизнеса к глубокому обучению, а также активное развитие вычислительной техники, позволяющие обучать глубокие нейросети, способствуют развитию применения технологий интеллектуальной обработки естественного языка в данной области.

В исследовании рассмотрены некоторые из существующих подходов к приложению технологий глубокого обучения к задаче построения поисковой системы для интернет-магазина, описаны особенности сбора данных в контексте решаемой задачи, предложена архитектура поисковой системы, основанная на BERT, а также представлены результаты экспериментов и значения метрик для предложенной модели.

2 Постановка задачи

Существует множество подходов к построению поисковой системы для интернет-магазина. Во многом они зависят от способа представления товаров в базе данных. К примеру, они могут описываться большим количеством признаков (такими как название, бренд, страна-производитель, размеры, цвет и т.д.), а могут лишь одной текстовой строкой - описанием, в котором, как предполагается, содержится вся информация о товаре.

Основным случаем, рассмотренным в данной работе, является модель данных, в которой товары в базе магазина описываются тремя признаками: описание товара, название товара и название бренда. Примеры представлены в таблице 1.

| № | Описание товара | Название товара | Бренд |
|---|--|---------------------|--------------------|
| 1 | Смартфон Apple iPhone 5S 16Gb Space Gray (ME432RU/A) | Смартфон | Apple |
| 2 | Варенье Вкусвилл брусника - апельсин, 310 г | Варенье | Вкусвилл |
| 3 | Женские кроссовки CALVIN KLEIN JEANS | Кроссовки | CALVIN KLEIN JEANS |
| 4 | Гриль Bosch TFB3323V | Гриль | Bosch |
| 5 | TELEFUNKEN / Портативная колонка TF-1234B | Портативная колонка | TELEFUNKEN |

Таблица 1: Пример данных в базе интернет-магазина

Данная модель является некоторым промежуточным вариантом между тем случаем, когда товар описывается большим количеством различных признаков, и тем, когда товар описывается лишь одной текстовой строкой (то, что в рассматриваемой модели называется описанием товара). Мотивация выбора такой модели будет описана в следующем подразделе «Особенности сбора данных».

Будем считать, что интернет-магазин имеет доступ к базе данных с описаной выше моделью, а также некоторый сайт, который позволяет получать текстовые запросы от клиентов (пользователей). Пользователь задает свой запрос в текстовом виде на естественном языке. В ответ поисковая система должна предоставить некоторый список товаров из базы, которые являются наиболее релевантными запросу пользователя. Мы предполагаем, что пользователь вводит запрос, содержащий в себе слова, характеризующие товар. Безусловно, идеальным запросом являлось бы точно описание товара из базы данных магазина, тогда мы бы смогли произвести явный поиск по существующим наименованиям и выдать точный результат. Однако на практике пользователь никогда не будет знать, как именно нужный ему товар представлен в базе, а также довольно часто клиента интересует не конкретный товар, а некоторый каталог релевантных его запросу товаров. К примеру, его может интересовать чайник (без уточнения бренда) или все продукты компании Apple (без уточнения названия товара). Таким образом, будем предполагать, что запрос пользователя будет состоять из трех компонент, каждая из которых может как присутствовать, так и отсутствовать: название товара, название бренда и уточняющая информация. При-

меры запросов с соответствующим разделением слов по указанным трем компонента представлены в таблице 2.

| № | Запрос | Название товара | Бренд | Уточняющая информация |
|---|--------------------------------|-----------------|--------|-------------------------|
| 1 | чехол для iphone 6 | чехол | (нет) | для iphone 6 |
| 2 | смартфоны xiaomi | смартфоны | xiaomi | (нет) |
| 3 | чайник электрический маленький | чайник | (нет) | электрический маленький |
| 4 | apple iphone 7 | (нет) | apple | iphone 7 |
| 5 | футболки мужские с принтом | футболки | (нет) | мужские с принтом |

Таблица 2: Примеры запросов и соответствующих им смысловых частей

Исходя из описанной выше модели данных в базе магазина и предполагаемой структуры запроса пользователя, поставим задачу построения поисковой системы следующим образом. В запросе пользователя необходимо выделять следующие сущности: название товара, название бренда и уточняющую информацию. После выделения этих компонент, соответствующим образом ведется поиск по базе товаров магазина. К примеру, если выделен и бренд В, и название товара I, то пользователю выдается некоторая выборка товаров, имеющих бренд В и название товара I (это легко сделать, поскольку мы предположили, что данные признаки уже выделены в базе данных интернет-магазина). Если же выделено, к примеру, только название товара I, то стоит показать пользователю выборку товаров с названием I, вне зависимости от бренда. Данная задача давно известна в области обработки естественного языка (natural language processing, NLP) как распознавание именованных сущностей (named entity recognition, NER).

Отдельного внимания заслуживает уточняющая информация. Поскольку данная компонента может изменяться в слишком широких масштабах (так как для разных категорий товаров подробности описания могут сильно отличаться, к примеру для чайников важен объем, а для планшетов – размер экрана), попытки привести её в унифицированный вид могут быть достаточно затруднительными (подробнее об этом будет сказано в следующем подразделе). В связи с этим данная информация должна учитываться в поисковой выдаче, которая была отфильтрована названием товара и бренда, некоторым особым образом.

3 Данные

3.1 Особенности сбора данных

Проведение некоммерческих исследований в рассматриваемой в данной работе области затрудняется в связи с высокой коммерциализацией. Необходимые для построения поисковых систем данные могут быть получены единственным образом: путем выгрузки данных из баз реальных интернет-магазинов. Чаще всего компании не стремятся делиться подобной информацией, поскольку она может быть использована конкурентами. Из постановки задачи следует, что для построения поисковой системы необходимы как минимум два набора данных: база товаров, а также примеры запросов пользователей. Опишем затруднения, которые возникли во время сбора указанных датасетов.

База товаров. На первый взгляд может показаться, что собрать такие данные довольно просто, ведь все интернет-магазины имеют каталоги товаров, а для каждого товара указаны как минимум название, описание, категория и чаще всего бренд. Однако на практике подавляющее большинство компаний стараются защитить свои сайты от автоматического сбора данных (скрапинга, парсинга). Безусловно, такие защиты можно обойти, но при этом они достаточно сильно затрудняют процесс получения необходимых датасетов. Кроме того, разные категории товаров описываются значительно отличающимися друг от друга наборами признаков. К примеру, для одежды важны такие параметры как размер, цвет, материал, бренд и страна-производитель, а для смартфона – мощность процессора, объем памяти, размеры экрана, бренд, модель и т.д. Таким образом, при попытке сбора максимально детализированной информации о товарах (то есть когда товар описывается большим количеством признаков), гипотетический исследователь может столкнуться с тем, что в собранных им данных количество признаков невероятно велико, и для получения хоть сколько-то репрезентативной выборки необходимо спарсить практически все товары с сайта магазина, что в условиях некоммерческой разработки просто невозможно (из-за описанных выше проблем с защитой сайтов). Именно этот факт мотивирует предложенную в предыдущем разделе модель данных для базы товаров. Однако, если бы построение поисковой системы вел уже существующий интернет-магазин с

большим количеством детализированных примеров, модель с более подробным описанием каждого товара была бы возможна (и, скорей всего, более успешна).

Запросы пользователей. Если сведения о товарах, хоть и с трудом, но можно получить из открытых источников, то текстовые запросы клиентов в поисковую строку интернет-магазина являются полностью закрытой для сторонних лиц информацией. Автор данной работы провел в поисках открытых данных подобного рода более двух недель, и ничего похожего найдено не было. Однако был найден способ добывать текстовые запросы, наиболее похожие на необходимые. Довольно часто пользователи ищут товары не непосредственно на сайтах интернет-магазинов, а вводят свой запрос в поисковые системы вроде Google или Яндекс. При этом, если взять запросы, по которым совершаются переходы на крупные интернет-магазины (или их агрегаторы), такие как Ozon, Яндекс.Маркет и т.д., то окажется, что почти все такие запросы представляют собой как раз те самые текстовые строки, которые нам нужны (быть может, с некоторыми дополнительными словами, к примеру "купить"). Однако существуют сервисы (к примеру, ahrefs.com), которые позволяют анализировать и предсказывать переходы на сайты с поисковых систем. При этом подобные сервисы часто продают подобные данные. Именно таким образом можно собрать некоторую базу поисковых запросов в интернет-магазинах, не обращаясь к компаниям напрямую.

3.2 Описание собранных наборов данных

База названий товаров. Часть данных была получена путем сбора названий, категорий и брендов с сайтов крупных интернет-магазинов Ozon, BERU, Яндекс.Маркет, Юлмарт. С указанных площадок было собрано порядка 450.000 наименований. Отметим, что для построения основной компоненты поисковой системы (а именно глубокой нейронной сети для задачи NER) будут использовать лишь текстовые строки, содержащие всю информацию о товаре (то, что выше было названо описанием товара). Значительная часть товаров принадлежит разделу «Электроника». Пример данных из данной части выборки – в таблице 3.

| № | Описание товара |
|---|--------------------------------------|
| 1 | Смартфон Sony Xperia Z2 D6503 Purple |
| 2 | Чайник Bosch TWK7808, золотистый |
| 3 | a4tech / Мышь Bloody P80 Pro |
| 4 | Игровая консоль PlayStation 5, белый |

Таблица 3: Примеры товаров с сайтов с Ozon, BERU, Яндекс.Маркет, Юлмарт

Помимо собранных данных из открытых источников, были использованы данные, предоставленные на соревновании DataFusion, проходившем на площадке boosters.pro с 27 января 2021 года по 27 марта 2021 года. В задаче требовалось определить бренд товара по его текстовому описанию в чеке. Всего в датасете были представлены 3.1 млн подобных текстовых строк. Основная часть наименований является названиями продуктов питания, однако встречаются товары и из других разделов. Некоторые примеры из этого набора данных представлены в таблице 4.

| № | Описание товара |
|---|--|
| 1 | 20 БРЮКИ МУЖ / J Lin / 72 MC 1 |
| 2 | "Русский Аппетит "Суп борщ 50 гр . |
| 3 | Йогурт Даниссимо фантазия 105 г хрустящие шарики |
| 4 | Штора рулонная "Декор белый 57 * 160 см |

Таблица 4: Примеры товаров из данных с соревнования DataFusion

Запросы пользователей. Изначально, запросы собирались с помощью метода описанного в предыдущем подразделе. Таким образом было собрано порядка 50.000 запросов среднего качества. Однако, к счастью, автору работы удалось связаться с тим лидом интернет-магазина KazanExpress (<https://kazanexpress.ru>) Юрием Гаврилиным, который на безвозмездной основе передал для проведения некоммерческих исследований датасет запросов пользователей размером 3.7 млн записей. Автор приносит благодарность компании KazanExpress и, в частности, Юрию Гаврилину, за оказанную поддержку. Примеры запросов – в таблице 5.

| № | Запрос |
|---|-------------------------------------|
| 1 | ботинки рабочие |
| 2 | чехол для iphone |
| 3 | redmi note 9 |
| 4 | сменная головка для бритвы philips |
| 5 | капсулы для кофемашин bosch tassimo |

Таблица 5: Примеры запросов в поисковую систему KazanExpress

Для решения задачи распознавания именованных сущностей на представленных запросах необходима разметка: для каждого слова в обучающем датасете нужно указать, является ли слово брендом, названием товара или не относится к этим классам. Как было отмечено, в открытом доступ подобных данных нет, поэтому для выполнения разметки автор обратился в компанию, предоставляющую подобные услуги. Поскольку данное исследование является некоммерческим проектом, бюджет был существенно ограничен, поэтому итоговый размер обучающего набор составил 10000 сэмплов. Автор выражает благодарность компании LabelMe за льготные условия при выполнении данной задачи. Пример размеченных данных - в таблице 6. После каждого слова в скобках указана метка класса. Обозначения: О - отсутствие класса, I - название товара, В - название бренда.

| № | Запрос |
|---|------------------------------|
| 1 | четырёхгранный(О) ключ(I) |
| 2 | подвеска(I) с(О) цепочкой(О) |
| 3 | кроссовки(I) найк(В) |
| 4 | natura(В) siberica(В) |
| 5 | xiaomi(В) mi(О) очки(I) |

Таблица 6: Примеры размеченных запросов

4 Обзор существующих работ

Приведем краткий обзор некоторых работ, связанных с данным исследованием.

4.1 Подход на основе рекуррентных нейросетей

Наиболее близкой по задаче и методу решения является статья [1], выпущенная командой сотрудников компании The Home Depot (<https://www.homedepot.com/>), которая занимается продажей товаров для дома, ремонта и строительства. Авторы рассматривают в точности такую же постановку задачи, как и в данной работе (классификация слов в запросе пользователя на название товаров, бренд и прочие слова). В рамках статьи они представили цельное (end-to-end) решение. Было предложено, как оптимальным образом организовать создание обучающих датасетов, была обучена нейросеть для распознавания именованных сущностей, а также представлены некоторые показатели метрик качества. Рассмотрим подробнее каждый из этих этапов.

Подготовка данных. С учетом того факта, что в качестве модели классификации будет использоваться глубокая нейросеть, авторы статьи выдвигают три требования к обучающему датасету.

1. Большое количество данных
2. Высокое качество разметки
3. Широкое покрытие меток классов (названий товар и брендов)

Идеальным решением для такого перечня требований стал бы большой набор данных, размеченных вручную. Однако разметка текстовых данных (особенно с учетом того факта, что иногда потребуется дополнительно проводить поиск: является ли некоторое слово брендом или неизвестным разметчику товаров) является достаточно сложной и дорогостоящей процедурой. В связи с этим, авторы предлагают составить обучающих датасет из трех компонент:

1. Размеченные автоматическими эвристическими методами данные
2. Размеченные вручную запросы
3. Запросы, состоящие только из бренда или названия товара

В первом компоненте для автоматической разметки используются детерминированные алгоритмы, основанные на сопоставлении в запросах пользователей слов с названиями брендов и товаров. Вторая часть, как и следует из названия, состоит из размеченных человеком данных, однако, как уже было отмечено, этот процесс трудозатратен, поэтому данная компонента сравнительно невелика в размерах. Третья часть призвана отвечать третьему требованию, а именно пополнять «кругозор» модели редко встречающимися товарами и брендами. Подробнее о том, как именно будут использоваться три указанные поднабора данных, будет сказано ниже.

Модель. В ходе экспериментов авторы тестировали различные нейросетевые архитектуры, в основе которых лежат рекуррентные нейросети (recurrent neural networks, RNN). Ещё с конца прошлого века было известно, что классические RNN обладают рядом недостатков: при их обучении стохастические градиенты часто, проходя через слои сети, начинают быстро затухать (в связи с чем модель «забывает» контекст очередного входа довольно быстро), а иногда, наоборот, «взрываются», когда модуль градиента экспоненциально увеличивается на каждом шаге алгоритма обратного распространения. В связи с обозначенными проблемами, на практике чаще всего применяются модификации RNN: LSTM (Long-Short Term Memory) [4] и GRU (Gated Recurrent Unit) [2]. В задачах, в которых заранее целиком известна входная последовательность (например, NER, но не предсказание временных рядов), рекуррентные модели также часто модифицируют до их двунаправленных версий: BiLSTM (Bidirectional LSTM) и BiGRU (Bidirectional GRU). Основным отличием является наличие в архитектуре двух нейросетей, в одну из которых входная последовательность поступает в прямом порядке, а в другую — в обратном. Это способствует улавливанию модели двухстороннего контекста очередного элемента входа.

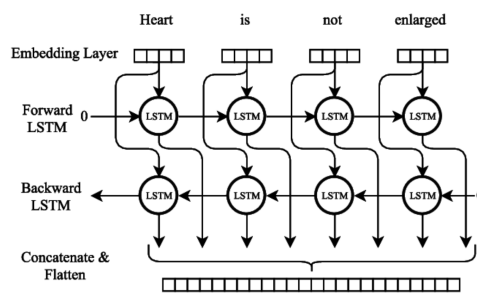


Рис. 1: BiLSTM

Для построения финальной модели, авторы рассматриваемой статьи комбинируют указанные выше двунаправленные архитектуры. А именно, для составления векторных представлений слов (эмбеддингов) в предложении используется посимвольная BiLSTM (рис. 2), а для решения задачи распознавания именованных сущностей BiGRU с CRF[5] (Conditional Random Field) слоем (рис. 3), на вход которой подается последовательность токенов, полученных из слов после прогонки через первую нейросеть.

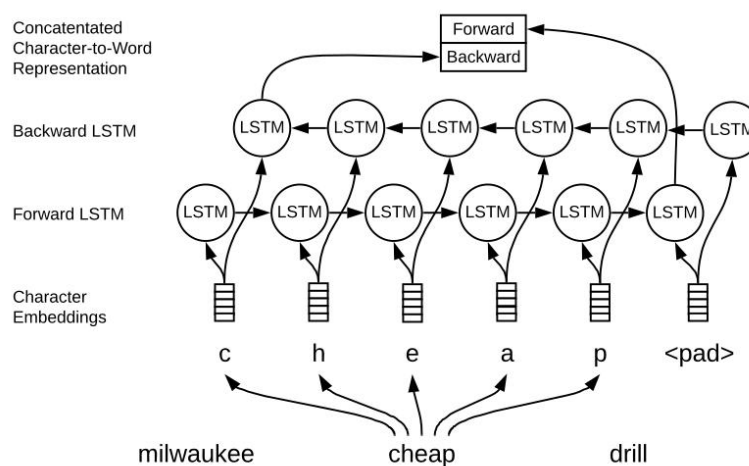


Рис. 2: Посимвольная BiLSTM нейросеть для получения эмбеддингов

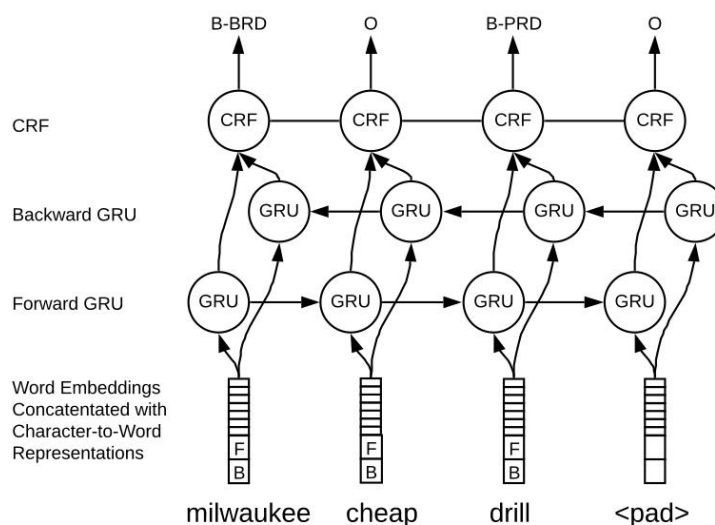


Рис. 3: BiGRU-CRF для NER

Итеративный процесс обучения. После того как были описаны входные датасеты и модель, мы можем рассмотреть предлагаемый авторами статьи процесс обучения итоговой системы. Все её компоненты представлены на рисунке 4.

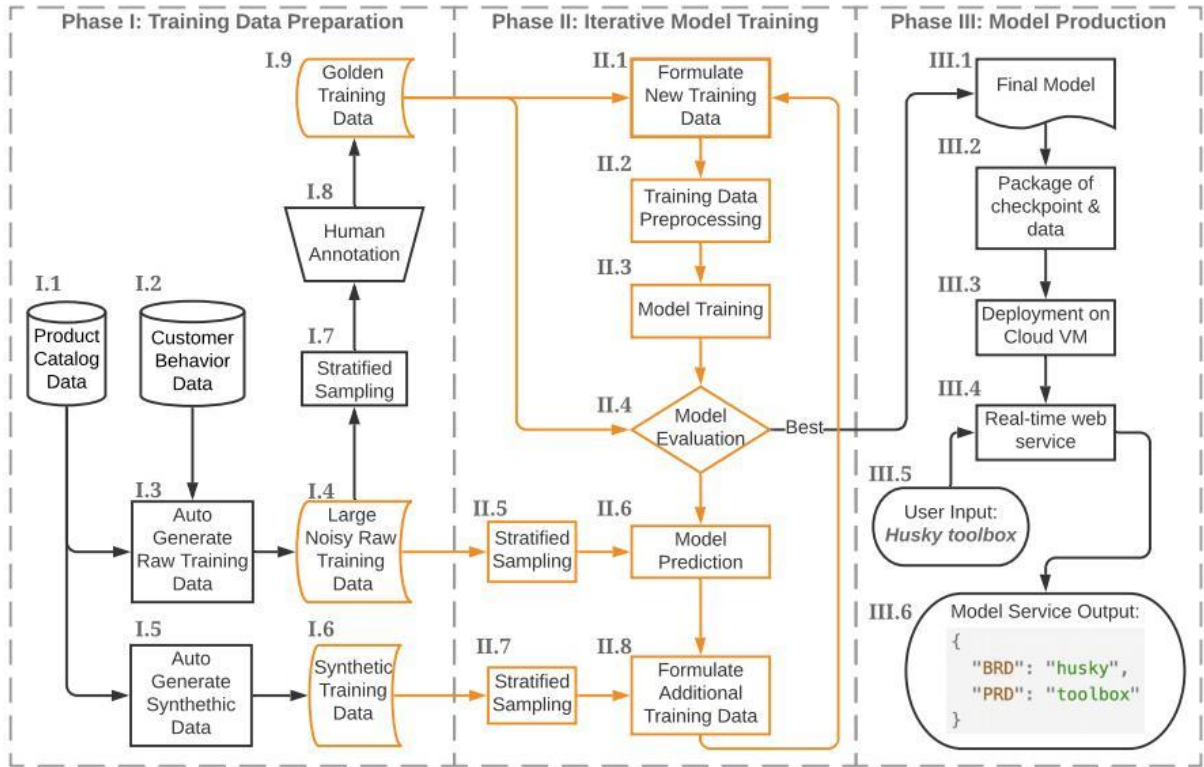


Рис. 4: Левый столбец — датасеты. Центральный — цикл обучения. Правый — особенности развертки модели в бизнес.

Первая итерация начинается с вручную размеченного датасета («золотого» датасета, так как данные размечать дорого) (I.9). 15% от этих данных случайным образом откладываются в качестве тестовой выборки; остальные данные случайным образом разделяются на обучающие (90% оставшихся данных) и валидационные («dev») данные (10%). Затем данные для обучения предобрабатываются (II.2). На следующем этапе (II.3) модель обучается до тех пор, пока F1-мера качества не перестанет улучшаться на валидационной выборке, либо пока не будет достигнуто максимальное число итераций (эпох). Далее выполняется оценка качества на тестовой выборке (II.4). Если выполнен некоторый критерий остановки (достигнуто максимальное число итераций, или модель достигла необходимого качества), то процесс завершается. Иначе алгоритм переходит на шаги расширения обучающего датасета (II.6 и II.7).

На этапе II.6 из составленной с помощью эвристических алгоритмов компоненты (I.4) данных случайным образом сэплируется некоторая подвыборка. Текущая модель делает предсказание меток классов. Если метки, полученный эвристическим алгоритмом, а также метки модели совпадают, то такой сэмпл добавляется в обучающий датасет. На этапе II.7 к нему случайным образом дополнительно добавляются сэмплы, состоящие только из бренда или названия товара.

Результаты. На рис. 5 изображена таблица, представленная авторами в статье. Как было отмечено выше, лучшие результаты показала архитектура, использующая 2 нейросети: посимвольная BiLSTM для построения векторных представлений слов и BiGRU-CRF для непосредственного распознавания именованных сущностей. В качестве метрики используется F1-мера (среднее гармоническое между точностью и полнотой классификации слов в запросе, усредненное по всем запросам в соответствующем наборе данных).

| models | char emb. | dev F1 | test F1 |
|------------|------------|--------------|--------------|
| BiLSTM | No | 85.77 | 85.05 |
| | Yes | 86.99 | 86.23 |
| BiLSTM-CRF | No | 87.69 | 86.72 |
| | Yes | 88.57 | 88.44 |
| BiGRU | No | 85.42 | 85.57 |
| | Yes | 86.53 | 87.09 |
| BiGRU-CRF | No | 87.12 | 87.04 |
| | Yes | 88.71 | 88.82 |

Рис. 5: Результаты авторов статьи

4.2 BERT

Ключевым отличием данной работы от исследования, описанного в предыдущем разделе, является использование более современной нейросетевой архитектуры, которая называется BERT (Bidirectional Encoder Representations from Transformers). Архитектура была предложена[3] в 2018 году и стала активно применяться во многих задачах обработки естественного языка, для решения которых используются нейросети. Поскольку эта модель лежит в основе предлагаемого в данной работе решения, кратко опишем её ключевые принципы работы.

Архитектура Transformer. Революционной для области глубокого обучения в обработке естественного языка стала статья [7], вышедшая в 2017 году. В статье была описана архитектура Transformer — первая успешная модель для задачи машинного перевода, не использующая рекуррентные нейросети. Рассмотрим более подробно данную архитектуру, поскольку она тесно связана с нужным нам BERT’ом.

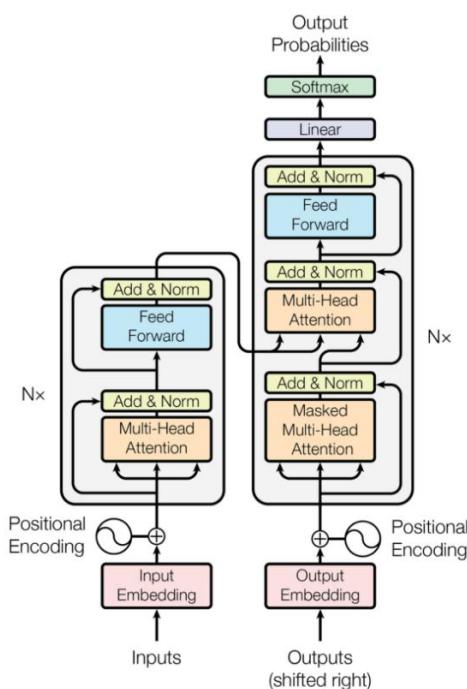


Рис. 6: Архитектура Transformer

На рисунке 6 изображена схема трансформера: левый блок представляет собой один слой энкодера, а правый — один слой декодера. Для понимания BERT нас будет интересовать только энкодер. Опишем, из чего он состоит.

Механизм внимания (Attention). Идея Attention довольно проста: имея наборы векторов-ключей и соответствующих им векторов-значений (K, V) , а также вектор-запрос q , требуется найти в K вектор, наиболее похожий на q , и выдать соответствующий ему вектор-значение из V . На практике в качестве меры схожести используется скалярное произведение. Поскольку данный механизм хочется использовать в нейронных сетях, операцию взятия ключа по значению заменяют на суммирование всех значений с некоторыми весами, чтобы данная операция была дифференцируема. В свою очередь указанные веса получаются применением функции softmax к

скалярным произведениям векторов из K и q . Описанный механизм легко записать в матричном виде:

$$Attention(q, K, V) = (qK^T) V,$$

где в K и V векторы записаны по строкам, а q — это запроса-строка. На практике обычно в «головку» внимания приходит не один вектор-запрос, а несколько, поэтому их удобно также собрать в единую матрицу Q . Также дополнительно будем нормировать аргумент softmax на некоторую константу β (на практике $\beta = \sqrt{d}$, где d — размерность ключей), для стабильности вычислений. В итоге получим формулу:

$$Attention(Q, K, V) = \left(\frac{QK^T}{\beta} \right) V.$$

В трансформере используется модификация данного механизма: многоголовое внимание (MultiHead Attention). Основным отличием «многоголового» внимания является наличие нескольких параллельно работающих головок Attention.

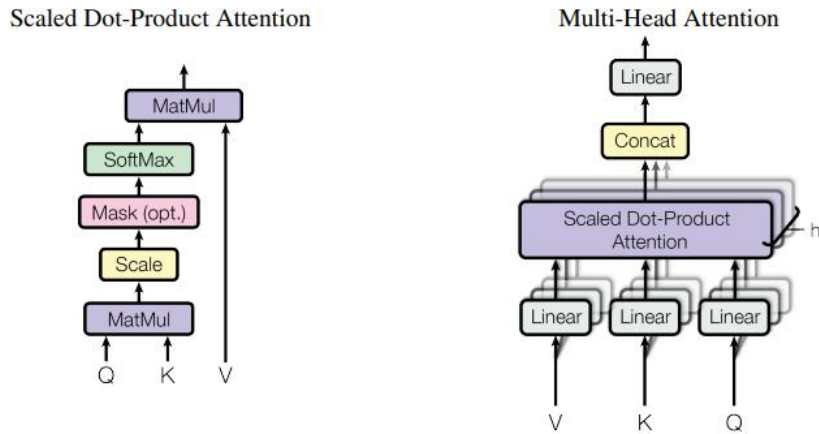


Рис. 7: Схема механизма внимания (слева) и многоголового внимания (справа) [7]

Для того, чтобы получать разные результаты, исходные матрицы Q , K и V линейным образом преобразуются. После этого для каждой тройки таких матриц применяется обычная головка Attention. Далее полученные результаты конкатенируются, и снова преобразуются линейным образом. Все перечисленные этапы можно описать несколькими простыми формулами:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

$$\text{where } \text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Где матрицы W^O , W_i^Q , W_i^K , W_i^V — обучаемые параметры.

Энкодер. Энкодер состоит из нескольких (в оригинальной статье $N = 6$) идентичных слоев. Каждый из них имеет архитектуру, изображенную слева на рисунке 6. В свою очередь каждый слой имеет два подслоя. Первым из них является Multi-Head Attention, а вторым — обычная полносвязная нейросеть. Выход каждого слоя складывается со своим входом, а к полученной сумме применяется нормализация по слою (Layer Normalization). Данный механизм (который называется skip-connections) нужен для решения проблемы затухающих градиентов.

Архитектура BERT. В сущности, BERT[3] представляет собой энкодер от трансформера. Входная последовательность проходит через слои, каждый слой выполняет над эмбедингами некоторое преобразование. На выходе сети получают трансформированные эмбединги для входной последовательности токенов.

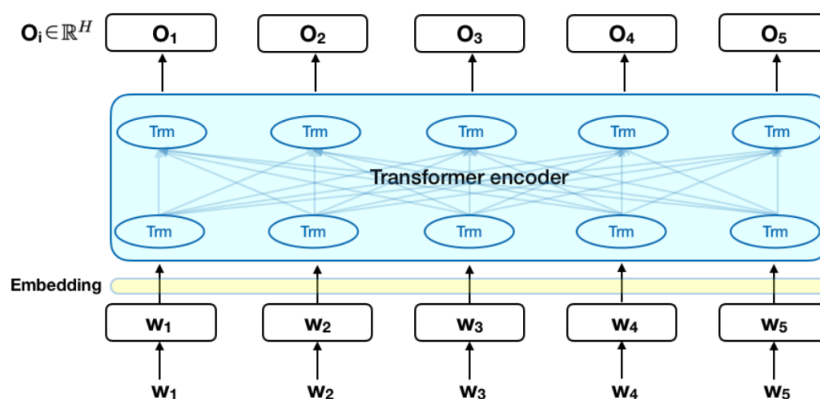


Рис. 8: Преобразование токенов через BERT, изображение с сайта lyqn.ai, статья от Rani Horev

Ключевой особенностью данной модели является возможность её использования для многочисленных задач NLP. Этап обучения разбивается на 2 этапа:

1. Предобучение (pre-train). На данном этапе модель обучается на большом корпусе неразмеченных текстов.
2. Дообучение (fine-tuning). Предобученный на предыдущем этапе BERT дообучается на конкретную задачу (к примеру, классификация текстов или NER).

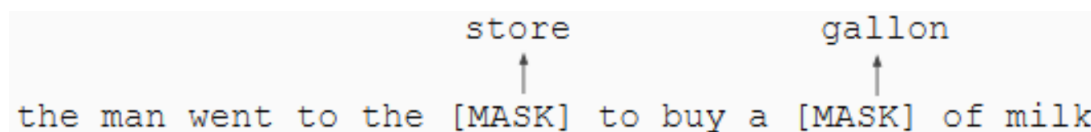
Данная парадигма называется transfer learning. Как показывают многочисленные эксперименты, модель, предварительно обученная на некоторые задачи на большом

неразмеченном наборе текстов, а потом дообученная на конкретную (downstream) задачу, показывает гораздо лучшие результаты, чем модель, обученная только на некоторую специфичную задачу без предобучения. Рассмотрим более подробно данные этапы для BERT.

Предобучение (pre-train). Предобучение BERT на большом корпусе неразмеченных данных выполняется с помощью решения двух задач:

1. Предсказание замаскированных слов (Masked Language Modelling, MLM)
2. Классификация: являются ли два предложения идущими друг за другом в тексте (Next Sentence Prediction, NSP).

Предсказание замаскированных слов. Классическая проблема в однонаправленных языковых моделях (к примеру, в рекуррентных нейросетях): модель видит контекст слова только с одной стороны. Если же использовать двунаправленный подход, то возникает проблема переобучения, поскольку информация о предсказываемом слове содержится в одном из контекстов (левом или правом). Авторы BERT предлагают следующий подход: некоторая доля слов в предложениях закрываются масками, а модель должна их предсказать:



store gallon

↑ ↑

the man went to the [MASK] to buy a [MASK] of milk

Рис. 9: Задача определения замаскированных слов

Классификация: является ли предложение А продолжением предложения Б. Смысл следует из названия.

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence

Рис. 10: Задача NSP (Next Sentence Prediction)

Обученный для решения этих задач BERT оказывается достаточно интеллектуальной системой: задача восстановления замаскированных слов позволяет ей «понимать» связи между словами в предложении, а задача NSP позволяет ей устанавливать связи между целыми предложениями. Важно отметить, что данные, на которых выполняется pre-train Берта не требуют разметки, что позволяет обучать его на огромном количестве данных. В связи с этим BERT активно применяется для решения множества задач.

Замечание. Как показали эксперименты в оригинальной статье, задача NSP не несет существенного вклада в качество модели, поэтому часто на практике на этапе предобучения её пропускают.

Дообучение (fine-tuning). После того как модель предобучена на большом корпусе текстов, архитектура берта незначительно изменяется для решения необходимых задач. В частности, для распознавания именованных сущностей (то есть, по сути, классификации токенов), достаточно добавить полносвязный линейный слой и слой softmax, который принимают на вход прошедшие через BERT входные токены. Помимо NER, BERT часто используется для следующих задач:

- Классификация входных последовательностей
- Ответы на вопросы (QA)
- Natural language inference (NLI) — классификация: следует ли из предложения-посылки предложение-вывод
- и т.д.

5 Предложенная архитектура

В качестве основной модели для решения задачи распознавания именованных сущностей в запросах пользователей будем использовать BERT. Конкретная конфигурация следующая: в качестве основы взят был BERT-Base (6 слоев, 12 головок self-attention, 768 нейронов в полносвязном слое), однако число головок внимания

было сокращено до 6. Мотивацией для такой модификации послужило априорное знание о том, что размер входных предложений довольно мал. Если обрезать распределения количества символов в запросах и количества слов в запросах по 0.99-му квантилю, то получим следующие распределения:

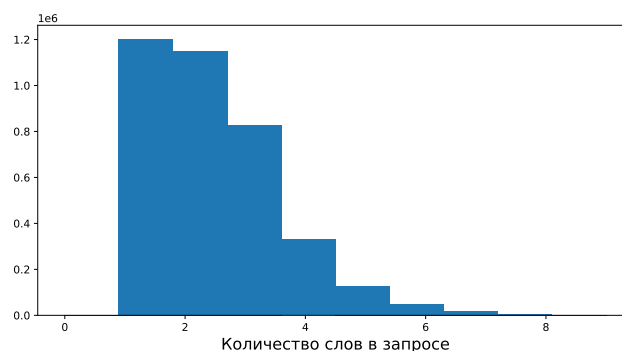


Рис. 11: Распределение числа слов в запросе

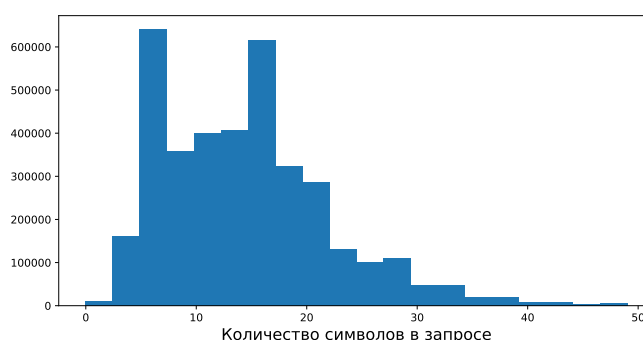


Рис. 12: Распределение числа символов в запросе

Поскольку длины входных последовательностей невелики, мы можем уменьшить число головок внимания, так как их будет достаточно для того, чтобы модель смогла учитывать контекст. Кроме того, уменьшение числа параметров способствует более быстрому обучению и снижает эффект переобучения. После стадии предобучения, к модели также добавляет линейный слой и softmax слой для дообучения на непосредственно распознавание именованных сущностей. Полученная модель имеет порядка 66 млн. параметров.

В качестве токенизатора будем использовать ставший для BERT классическим WordPiece[6]. Данный алгоритм токенизации разбивает входные слова на буквы, а затем, используя статистики по корпусу, начинает их сливать, до того момента, пока в словаре не окажется заранее выбранное число токенов. В итоге чаще всего многие слова оказываются разбитыми на более мелкие подслова (к примеру, слово *токенизатор* может быть разбито на слова *токен*, *##изатор*. В этом случае две решетки в начале подслова означают, что данный токен встречается внутри какого-то слова, но не является его началом). Данный подход (разбиение слов на более мелкие подслова) оказывается весьма полезным в контексте нашей задачи. Дело в том, что пользователи часто могут допустить опечатки в своем запросе. В случае, если бы токенизатор использовал все слово целиком в качестве самостоятельной единицы, чаще всего модель бы не нашла его в своем словаре. Однако когда в качестве токенов используются подслова, в случае ошибки, пользователь чаще всего ошибается лишь в одном из подслов, что позволяет модели относительно качественно отработать в данном случае.

6 Обучение модели

6.1 Tokenizer

В описанной в предыдущих разделах статье [1] было установлено, что стандартные предобученные токенизаторы в рамках рассматриваемой задачи покрывают достаточно малое количество токенов:

| embedding | similar words | vocab coverage | F1 |
|------------------------|------------------------------------|----------------|--------------|
| pre-trained GloVe | chicago detroit minneapolis | 39.8% | 87.56 |
| pre-trained Word2vec | springfield harvey wisconsin | 15.5% | 83.99 |
| custom GloVe | m18 dewalt drill | 99.9% | 87.58 |
| custom Word2vec | dewalt makita ridgid | 99.9% | 88.82 |

Рис. 13: Покрытие словаря различными токенизаторами [1]

Как видно из таблицы, предобученные эмбединги покрывают лишь порядка 15-40% встречающихся токенов. Этот факт, а также мотивация из предыдущего раздела (про опечатки) спродвигли автора на обучение собственного токенизатора WordPiece. Токенизатор обучался на всех имеющихся данных, его итоговый размер составил 30.000 токенов.

6.2 Pre-training

Предобучение проводилось на задаче предсказания маскированных слов (MLM). В качестве обучающего датасета использовалась база названий товаров, составленная из нескольких компонент, описанных в соответствующем разделе. После препроцессинг и удаления дубликатов размер набора данных составил 3.268.226 сэмплов. Из них 10.000 примеров были отложены в качестве валидационной выборки. Обучение велось в течение 12 часов на графическом процессоре NVIDIA RTX 3090. За это время модель прошла около 8 эпох. Функция ошибки (кросс-энтропия) на обучающем и валидационном наборах данных представлены на рисунке 13. Как видно, большой размер обучающих данных позволил модели постоянно уменьшать значения функции ошибки как на обучающем датасете, так и на валидации, что говорит об отсутствии эффекта переобучения в данном случае. При этом графики выходят на некоторое плато, дальнейшее обучение несущественно влияет на качество полученной нейросети.

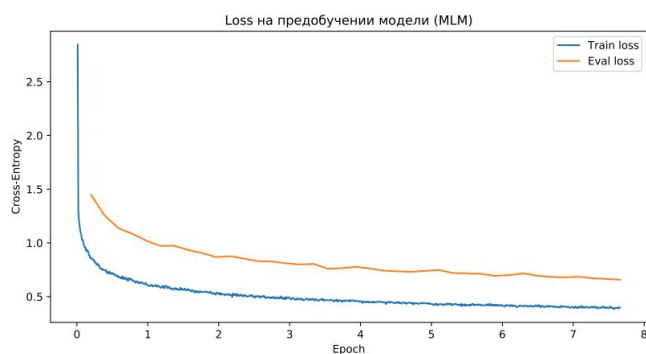


Рис. 14: Функция ошибки на этапе предобучения

6.3 Fine-tuning

Дообучение велось на размеченных людьми данных. В качестве обучающей выборки было выделено 9500 сэмплов, оставшиеся 500 были отложены для валидации. Во время проведения экспериментов был обнаружено, что такого количества примеров недостаточно для стабильного обучения модели в течение большого количества эпох. В связи с этим в процесс обучения были внесены поправки: вероятность dropout была увеличена до 0.2 (в оригинальном BERT - 0.1), коэффициент L2-регуляризации (в контексте нейросетей известный как Weight Decay) был увеличен до 0.1 (был 0.0). Эти модификации позволили относительно успешно тренировать модель на протяжении трех эпох.

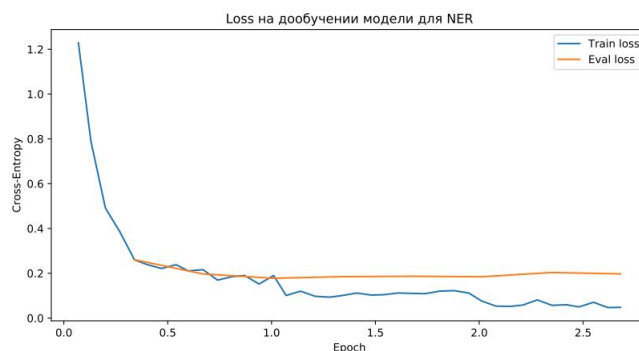


Рис. 15: Функция ошибки на этапе NER

Как можно заметить, качество на валидации к концу первой эпохи выходит на минимум. Далее уменьшается лишь качество на обучающей выборке. Если продолжить тренировать модель, то начнется переобучение (качество на обучении будет повышаться, а на валидации - падать). Это происходит из-за того, что обучающая выборка достаточно мала, поскольку данное исследование является некоммерческим, а разметка большого количества запросов является финансово затратной процедурой.

7 Результаты и примеры работы моделей

В результате обучения описанных в предыдущем разделе моделей, были получены 2 предобученные нейросети: BERT для предсказания замаскированных слов, а

также BERT для распознавания именованных сущностей. На изображениях ниже — примеры их работы.

```
make_prediction('купить чайник BOSCH'.split())
[('купить', '0'), ('чайник', 'Товар'), ('BOSCH', 'Бренд')]
```

```
make_prediction('apple iphone'.split())
[('apple', 'Бренд'), ('iphone', 'Товар')]
```

```
make_prediction('купить смартфон'.split())
[('купить', '0'), ('смартфон', 'Товар')]
```

В данном случае samsung - это телефон, для которого покупается чехол, то есть не бренд чехла

```
make_prediction('чехол для samsung galaxy'.split())
[('чехол', 'Товар'), ('для', '0'), ('samsung', '0'), ('galaxy', '0')]
```

А тут уже samsung - бренд

```
make_prediction('samsung смартфон'.split())
[('samsung', 'Бренд'), ('смартфон', 'Товар')]
```

```
make_prediction('посуда'.split())
[('посуда', 'Товар')]
```

```
make_prediction('mercedes'.split())
[('mercedes', 'Бренд')]
```

```
make_prediction('носки мужские adidas'.split())
[('носки', 'Товар'), ('мужские', '0'), ('adidas', 'Бренд')]
```

Рис. 16: Примеры распознавания именованных сущностей.

Можно заметить, как модель использует контекст: в 4-м примере Samsung является брендом телефона, для которого покупается чехол, а не самом брендом чехла, поэтому данное слово не классифицируется как бренд. А в 5-м, наоборот, Samsung является брендом желаемого смартфона.

```

get_examples('чехол для [MASK] galaxy')
[('чехол для samsung galaxy', 0.5545173287391663),
 ('чехол для телефона galaxy', 0.061687879264354706)]

get_examples('купить [MASK] для iphone 6 ')
[('купить чехол для iphone 6', 0.9618306756019592),
 ('купить стекло для iphone 6', 0.0038562484551221132),
 ('купить наклейки для iphone 6', 0.0036110864020884037)]

get_examples('купить [MASK] bosch 2 литра')
[('купить чайник bosch 2 литра', 0.07487241923809052)]

get_examples('молоко [MASK] ультрапастеризованное')
[('молоко питьевое ультрапастеризованное', 0.12975719571113586),
 ('молоко фруктояня ультрапастеризованное', 0.04602883383631706),
 ('молоко свитлогорье ультрапастеризованное', 0.043549809604883194),
 ('молоко вкуснотеево ультрапастеризованное', 0.029766695573925972)]

```

Рис. 17: Примеры предсказаний замаскированных слов. Число рядом с предсказанием — его вероятность.

Помимо использования модели MLM в качестве базовой модели, которая будет дообучаться для NER, в контексте рассматриваемой задачи её также можно использовать для предложения следующего слова в запросе пользователя. Для этого необходимо указывать в качестве маскируемого слова последний (виртуальный) токен, после уже написанного пользователем куска запроса.

```

get_examples('iphone [MASK]')
[('iphone 7', 0.02028769813477993),
 ('iphone 6', 0.019341209903359413),
 ('iphone 4', 0.018780456855893135)]

get_examples('купить яблоки [MASK]')
[('купить яблоки вес', 0.08910977840423584),
 ('купить яблоки кг', 0.054261304438114166),
 ('купить яблоки зеленые', 0.033656924962997437)]

get_examples('смартфон samsung[MASK]')
[('смартфон samsung galaxy', 0.07469146698713303),
 ('смартфон samsung note', 0.05042911320924759),
 ('смартфон samsung s', 0.040562234818935394)]

```

Рис. 18: Предложение следующего слова при вводе пользователем запроса.

8 Заключение

В данном исследовании:

- Была рассмотрена задача построение поисковой системы для абстрактного интернет-магазина с использованием алгоритмов глубокого обучения
- Были указаны ключевые сложности со сбором данных, которые возникнут у гипотетического исследователя, который захочет заниматься данной областью
- Был предложен метод сбора запросов пользователей в интернет-магазины через посредников — общие поисковые системы (Яндекс, Google и т.д.)
- Был собран обучающий набор данных, часть из них была вручную размечена
- Был проведен обзор релевантный данной тематике статей и подходов
- Была предложена нейросетевая архитектура, основанная на BERT, для решения поставленной задачи
- Был предложен метод токенизации входных запросов, который способствует более качественной работе модели в случае, когда пользователь допустил опечатку в запросе
- Были построены прототипы моделей, решающих поставленные задачи, представлены соответствующие им графики качества работы
- Был предложен метод предсказания следующего слова в запросе пользователя с помощью BERT, обученного на задачу MLM

Безусловно, построенные модели являются лишь прототипами. Для внедрения их в реальные бизнес-процессы потребуются дополнительная работа. В частности, для повышения качества работы необходимо будет разметить больший набор запросов, а также дополнить обучающий датасет синтетическими данными, как это было описано в [1].

9 Приложение

Весь код для препроцессинга входных данных, предобучения модели, дообучения и выполнения предсказаний можно найти на GitHub репозитории автора: https://github.com/Sorrow321/cmc_seminar/

Помимо этого, по указанной ссылке можно найти обученные в ходе работы модели, что позволит читателю самостоятельно провести интересные его дополнительные эксперименты, а также убедиться в полученных результатах.

Список литературы

- [1] Xiang Cheng, Mitchell Bowden, Bhushan Ramesh Bhange, Priyanka Goyal, Thomas Packer, and Faizan Javed. An end-to-end solution for named entity recognition in ecommerce search. 2020.
- [2] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 2014.
- [3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.
- [4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.
- [5] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. 2015.
- [6] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152, 2012.
- [7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.