

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение
высшего образования
Национальный исследовательский университет
«Высшая школа экономики»

Факультет
гуманитарных наук
Образовательная программа
«Компьютерная лингвистика»

Николаева Анна Михайловна
АВТОМАТИЧЕСКОЕ ОПРЕДЕЛЕНИЕ СМЕНЫ ИНТЕНТА
ПОЛЬЗОВАТЕЛЯ В ПОИСКОВЫХ ЗАПРОСАХ
METHODS FOR AUTOMATIC DETECTION OF USER INTENT
CHANGE IN SEARCH QUERIES

Выпускная квалификационная работа студента 2 курса магистратуры
группы МКЛ181

Академический руководитель
образовательной программы
канд. филологических наук, доц.
А.А. Бонч-Осмоловская

Научный руководитель
канд. технических наук, доц.
Э.С. Клышинский

« » _____ 2020 г.

Москва 2020

ОГЛАВЛЕНИЕ

Введение	4
Глава 1. Определение поискового запроса и поискового интента	6
Глава 2. Данные	8
2.1 Описание запросов на доске объявлений	8
2.1.1 Статистика по запросам	9
2.1.2 Орфография запросов	11
2.1.3 Частеречный состав запросов	13
2.2 Разметка тестовой выборки по поисковому интенту	14
Глава 3. Подходы к автоматическому определению смены интента	18
3.1 Методы анализа поисковых запросов	17
3.2. Определение семантической близости двух строк	22
3.2.1. Дистрибутивные модели	22
3.2.2. Машинное и глубокое обучение с учителем	24
Глава 4. Эксперименты	29
4.1 Постановка задачи	29
4.2 Разметка обучающей выборки	29
4.2.2 Разметка методами символьного и лексического расстояния	29
4.2.3 Разметка в Яндекс.Толоке	30
4.3 Обучение дистрибутивной модели	32
4.4 Бинарная классификация	33
4.4.1 Описание моделей	33
4.4.2 Результаты	38
4.4.3 Анализ ошибок	40
4.5 Классификация на 3 класса	42

4.5.1 Описание моделей	42
4.5.2 Результаты	43
4.5.3 Анализ ошибок	43
4.6 Общие выводы	44
 Заключение	 47
 Литература	
 Приложение	

Введение

Эффективные поисковые системы становятся неотъемлемой частью интернет-сервисов с большим числом пользователей. Трудно представить себе онлайн-магазин или сайт услуг без возможности поиска желаемого продукта. При этом большинство подобных сервисов обладает уникальной спецификой в виду разных факторов, таких как тип онлайн-площадки, вид предоставляемой информации в поисковой выдаче, механики взаимодействия с пользователем. Улучшение качества поиска в таких условиях требует нетривиальных решений, в том числе с использованием методов обработки естественного языка.

В задаче улучшения качества поиска особое внимание уделяется анализу потока текстовых запросов пользователей, в соответствии с которыми формируется и ранжируется поисковая выдача. Такой анализ позволяет не просто понимать, что именно ищут пользователи (т.е. определять поисковой *интент*), но и получать больше информации о пользовательском поведении – нашел ли он то, что искал, и насколько быстро. Одним из перспективных направлений исследования является распознавание моментов, когда пользователь по какой-то причине решил изменить предмет поиска. Это знание можно использовать для оптимизации ранжирования в поисковой выдаче, улучшения рекомендательной системы, а также для создания дополнительных бизнес-метрик, на основе которых можно оценить качество поисковой системы.

В зарубежной практике существует множество подходов к определению смены предмета поиска в поисковой сессии или в паре идущих друг за другом запросов. Многие из них ориентируются на экстралингвистические признаки - разница во времени между запросами, пересечения ключевых слов документов из поисковой выдачи, тематика и категория запроса, клики на ссылки по запросу, - и почти не учитывают лингвистическую природу запросов.

С другой стороны, за последние десятилетия бурно развивались методы автоматической обработки языка, позволяющие сравнивать текстовые фрагменты между собой по внешней форме и семантике, определять наличие в них семантического сдвига. К сожалению, практически нет современных исследований,

которые рассматривают в качестве текстовых фрагментов коммерческие запросы, обладающие своей грамматической, орфографической, семантической спецификой. Мы намерены ликвидировать это упущение и положить начало подобным исследованиям в российской практике.

Данная работа посвящена анализу методов автоматического определения смены интенга пользователя в коротких коммерческих запросах. В качестве данных используются поисковые логи российской онлайн-площадки для размещения объявлений о товарах, вакансиях, услугах.

Цель работы – оценить перспективы разных методов обработки естественного языка для задачи определения смены интенга, учитывая лингвистическую специфику данных.

Поставленную цель можно декомпозировать на несколько **задач**:

1. Определить основные лингвистические характеристики поискового запроса и поискового интенга;
2. Охарактеризовать данные, с которыми нам предстоит работать;
3. Рассмотреть и оценить на практике эффективность методов автоматического определения смены интенга в запросах. Мы остановились на 1) методах, основанных на орфографическом сходстве запросов; 2) методах дистрибутивной семантики с агрегацией векторов слов; 3) машинном и глубоком обучении с механизмом внимания и функциями пословного выравнивания. Выбор этих методов обусловлен их широким использованием в зарубежной практике для задач, связанных с сопоставлением двух текстов, и особенностью данных.

Диссертация состоит из введения, где обоснована актуальность исследования, сформулированы цель и задачи работы; в первой главе с лингвистических позиций раскрываются понятия поискового запроса и интенга; во второй главе объясняется специфика данных и интенга в них; в третьей главе представлен обзор литературы по теме; в четвертой главе описывается выбранная методология и эксперименты, а также предложены направления по улучшению качества моделей; в заключении сформулированы общие выводы по работе.

Глава 1. Определение поискового запроса и поискового интента

Ключевыми терминами, к которым мы будем обращаться в данной работе, являются поисковой запрос и поисковой интент.

Поисковой запрос в общем понимании – это слово, фраза, выражение, вводимое пользователем в поисковую строку поисковой системы с целью получения информации в виде списка сайтов, объявлений, изображений и видеороликов, товаров и услуг. Специфика запроса и тип данных в поисковой выдаче зависит от назначения онлайн-площадки, на которой расположен поиск.

С лингвистических позиций любой запрос можно рассматривать как коммуникацию пользователя с поисковой системой, что позволяет выделить в запросе уровни речевого акта (Остин 1999: 94-107). На локутивном (поверхностном) уровне запрос состоит из словесной, грамматической формы и смысла, т.е. существования в природе запрашиваемого контента. На иллокутивном (скрытом) уровне в любом запросе содержится директива к поисковой системе выдать этот контент. Таким образом, запрос, объединяющий в себе локутивное и иллокутивное содержание, является в некотором смысле минимальной коммуникативной единицей в общении человека с поисковой системой, также как предложение – в общении между людьми.

Структура большинства поисковых запросов русского языка отличается от структуры предложения, в частности, специфическими правилами грамматики и отсутствием связи на синтаксическом уровне (Бонч-Осмоловская 2014: 303). Среди таких правил в запросах русского языка можно выделить инверсию (*‘duck stories слова песни скачать’*), отсутствие маркирования зависимого слова (*‘журнал Финансы телефон’*), пропуск предлога или сочинительного союза (*‘перила ограждения’*). Эти и другие структурные особенности обуславливаются еще одним неотъемлемым компонентом запроса – *интендом*.

Поисковой интент (от англ. *intent* – «намерение, цель») – это устоявшийся термин в области оптимизации поиска, обозначающий цель, которая движет человеком, когда он вводит запрос в поисковую строку. Для понимания интента форма выражения цели важна не менее, чем сама цель, поскольку от того, как

формируется запрос, зависит тип ресурса, запрашиваемый человеком у поисковой системы (Jansen, Booth 2010).

В широком смысле интенды поисковых запросов делят на информационные, навигационные и транзакционные (Broder 2002). С помощью информационного запроса пользователь хочет собрать информацию о чем-либо или лучше изучить товар перед покупкой, просмотреть при этом несколько ресурсов (например, *‘портретная съемка’, ‘лучшие портретные объективы’*)¹. Такие запросы часто содержат вопросительные слова: как, почему, зачем, где, что делать². Запросы могут быть просто познавательными (например, *‘почему идет снег’*) или иметь коммерческий характер. С помощью навигационного запроса пользователь стремится немедленно попасть на определенный сайт (*‘мвидео каталог товаров’, ‘сервисный центр apple воронеж’, ‘вконтакте’*). При транзакционном запросе пользователь планирует совершить в интернете какое-либо действие, транзакцию, на коммерческом веб-сайте. Такой запрос часто включает в себя глагольные формы (например, *‘купить’, ‘скачать’*), и слово *‘цена’*: *‘купить кресло маркус москва’, ‘фотоаппарат зенит 122 цена’*. Однако такая классификация является довольно общей, не учитывает тип онлайн-площадки и не объясняет семантическую структуру запроса.

С точки зрения семантики исследователи в области обработки естественного языка относят интенд запроса к определенному концептуальному классу, например ФИЛЬМ, ЛЕКАРСТВО или ПРОДУКТЫ (Li 2010, Hassan 2013). Так, в запросе *«the side effect of aspirin»* интендом является концепт ЛЕКАРСТВО. В свою очередь, у класса интенда выделяют два основных атрибута: вершину (*intent head*) и модификаторы (*intent modifier*). Вершина интенда отвечает за наименование класса, в случае примера 1 это *‘аспирин’*. Модификаторы же накладывают ограничения на атрибут вершины. В частности, пользователя в примере 1 интересует именно побочное действие лекарства, помимо его остальных свойств. При этом существуют запросы без вершины или без модификатора, что будет продемонстрировано в следующей главе.

¹ <https://impulse.guru/blog/intent-poiskovogo-zaprosa/>

² <https://www.ashmanov.com/education/articles/typy-poiskovyh-zaprosov/>

А. А. Бонч-Осмоловская (Бонч-Осмоловская 2014), анализируя поисковые запросы на русском языке, выделяет целую группу атрибутов вершины, которые могут оставаться незаполненными:

- 1) название объекта (*‘охота на лис фильм’*);
- 2) место (*‘изумрудный город пенза’*);
- 3) дата (*‘выставка кошек петербург 2009’*);
- 4) рестрикторы (*‘портальная автомойка для грузовых машин’*);
- 5) модификатор в виде глагола или наречия (*‘якитория заказать’*) и т.д.

Исследователь отмечает, что атрибутивный состав запроса определяется экстралингвистическими знаниями пользователя о свойствах данной категории объектов.

Подытоживая вышесказанное, поисковой запрос является коммуникативной единицей в общении пользователя с поисковой системой. Запрос имплицитно содержит в себе директиву выдать какой-либо контент, специфика которого зависит от поискового интента пользователя. Поисковой интент, в свою очередь, состоит из концептуальной категории и ее атрибутов. В следующих главах мы проанализируем специфику запросов и интента пользователей на онлайн-площадке с объявлениями.

Глава 2. Данные

2.1 Описание запросов на доске объявлений

В качестве материала для данного исследования использовались логи поисковых запросов интернет-сервиса для размещения объявлений (классифайда). Данные содержат следующие поля:

1. *session_hash* - ID сессии пользователя;
2. *prev_query* - текст предыдущего запроса пользователя;
3. *query* - текст текущего запроса пользователя;
4. *prev_date* - дата и время предыдущего поискового запроса;
5. *event_date* - дата и время текущего поискового запроса;
6. *time_diff* - разница между *event_date* и *prev_date* в секундах.

Логи датируются последней неделей февраля 2020 года и содержат почти 30 млн строк с парами запросов. Это такие пары, в которых текущий запрос пользователя (*query*) следует за предыдущим запросом (*prev_query*) по времени (см. примеры в Таблице 1). При этом в паре запросов ни один не является подстрокой другого. Мы заранее исключили эти случаи, поскольку наличие общей подстроки позволяет заблаговременно сделать вывод об одинаковом интене в подобной паре запросов.

Таблица 1. Примеры пар запросов из поисковых логов

<i>prev_query</i>	<i>query</i>
радиатор ваз 2110	радиатор ваз 2107
работа в кфс	помощник повара
кровать испанский размер 160 200	кровать нметцкая размер 160 200
горелка газовая для ювелира	горелка для ювелира
айкос	iqos

Основная часть запросов соотносится с несколькими глобальными категориями объявлений, среди которых *автомобили*, *недвижимость*, *работа*, *услуги*, *для бизнеса*, *для дома и дачи*, *хобби и отдых*, *бытовая техника*, *личные вещи*, *животные*. Принадлежность запроса к той или иной категории очевидна не всегда. Встречаются и неопределенные запросы, такие как ‘*отдам даром*’, ‘*бесплатно*’. Это означает, что пользователь хочет найти объявления, предлагающие вещи безвозмездно. При этом возможно, что человек уже находится в желаемой категории на сайте, вводя данный запрос.

2.1.1 Статистика по запросам

Всего в данных содержится почти 60 миллионов запросов, из которых почти 10,4 миллионов – уникальные. Только 8% имеют частоту 1, у 75% запросов частота 5 и больше. Если говорить о парах запросов, уникальными является 62% пар из почти 30 миллионов.

В Таблице 2 приведены параметры распределения числа токенов (слов, разделенных пробелоподобными символами и очищенными от знаков препинания) в запросах.

Таблица 2. Параметры распределения числа токенов в запросах

Параметр	Все запросы (около 60 млн)	Уникальные запросы (10,4 млн)
Медиана	2 токена	3 токена
Среднее	2.3 токенов	3 токена
Стандартное отклонение	1.5 токенов	2.1 токен

Уникальные запросы содержат в среднем 3 токена против 2-х во всех запросах и обладают большим разбросом. В Рис. 1 и Рис. 2 для наглядности приведены гистограммы частот токенов во всем множестве запросов и в подмножестве уникальных запросов.

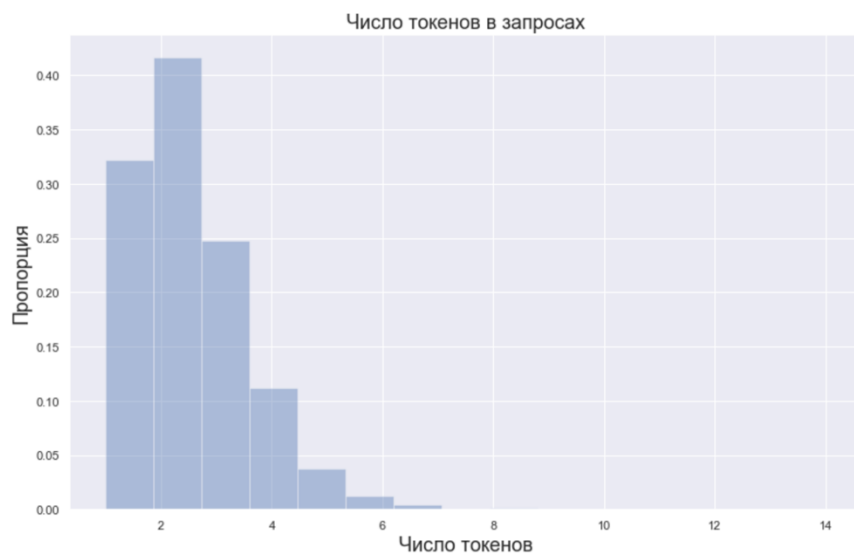


Рисунок 1: Распределение токенов в запросах

На основании приведенных данных можно сделать вывод о том, что мы имеем дело с очень короткими запросами из 2-3 слов, и что чем частотнее запрос, тем он короче. При этом пропорция запросов, состоящих из более чем 10 слов, чрезвычайно мала.

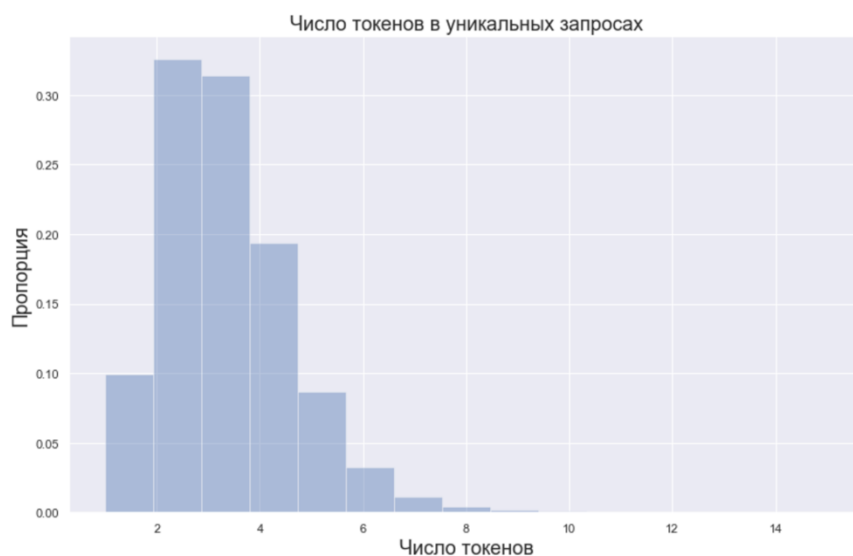


Рисунок 2: Распределение токенов в уникальных запросах

2.1.2 Орфография запросов

Чтобы более полно понять особенности данных запросов, рассмотрим состав их орфографии. В Таблице 3 приведено соотношение кириллицы, латиницы и цифр, а также их комбинаций в запросах.

Таблица 3. Доля кириллицы, латиницы в парах запросов

Орфография запроса	Пропорция пар запросов с данным написанием
C-C	0.49
C-CD	0.06
LD-LD	0.06
C-L	0.05
L-L	0.05
CD-CD	0.05
C-CL	0.04
CL-CL	0.03
L-LD	0.03
Остальные комбинации	каждая < 0.03

Примечание. С - кириллица, D - цифры, L - латиница. Дефис разделяет типы написания в одном и другом запросе, порядок не имеет значения.

Мы можем наблюдать в парах запросов 49% кириллицы, 6% – переход кириллицы в кириллицу и цифры и наоборот, 6% – латиница и цифры, 5% – переход из кириллицы в латиницу и наоборот, 5% – латиница и т.д. Если не учитывать пары, кириллица присутствует в 79% всех запросов, латиница – в 32%, цифры – в 23% запросов.

Закономерно, что большинство запросов написано именно кириллицей: основная аудитория интернет-ресурса – русскоязычные пользователи. В свою очередь, значительная доля запросов с латиницей и цифрами (32 и 23% соответственно) объясняется тем, что пользователи совершают поиск товаров конкретных брендов и моделей, имеющих латинское написание, и указывают размер одежды, габаритов мебели и других числовых параметров.

Еще одна отличительная черта данного массива запросов заключается в разнообразии вариантов написания одного и того же слова, особенно это касается иностранных брендов. Например, автомобиль Mitsubishi может быть введен в поиск как *мицубиси*, *мицубиши*, *мецубиси*, *mitsubishi*. Кроме того, данные изобилуют опечатками, которые требуют особого внимания во время автоматической обработки.

В данных присутствуют не только запросы, введенные пользователем напрямую, но также и языковые подсказки (саджесты). Это запросы, которые предлагает поисковая система при вводе пользовательского текста в поисковую строку. В саджестах, на которые кликнул пользователь, совершив при этом поисковое действие, могут исправляться опечатки или написание бренда переводится с кириллицы в латиницу. Поэтому неудивительно, что как минимум в 9% пар запросов происходит переход от написания кириллицей в полное или частичное написание латиницей. Однако действительно ли предыдущий запрос является переформулировкой текущего запроса в данных случаях, нам только предстоит выяснить в ходе экспериментов.

2.1.3 Частеречный состав запросов

Частеречный состав запросов также обладает своей спецификой (см. Таблицу 4). Для анализа использовался морфологический модуль pymystem3³.

Таблица 4. Частеречный состав токенов на кириллице

#	Часть речи	Доля
1	Существительное	73%
2	Прилагательное	15%
3	Предлоги	8%
4	Глагол или причастие	2%
5	Наречия	0.7%

Среди токенов с латиницей практически все – имена собственные, являющиеся наименованиями брендов и их моделей: *iphone, samsung, xiaomi, pro, sony, galaxy, bmw, gtx, mercedes, toyota, plus, apple* и т.д.

Подобный частеречный состав отличается от того, что можно ожидать от поисковых запросов в поисковых системах Google или Yandex. Основное отличие заключается в подавляющем присутствии именной группы (существительное и прилагательное) и небольшим семантическим разнообразием глаголов (“*купить*”, “*продам*”, “*сдам*”, “*требуется*” и т.д.). Это обусловлено типом онлайн-площадки, на которой основная мотивация пользователя – это найти объявление о каком-либо товаре, вакансии, жилье или услуге. Фактически пользователи осуществляют поиск по ключевым словам, указывая класс желаемого объекта (например, ‘*телевизор*’) и его атрибуты (‘*samsung 32 дюйма*’). Атрибуты, иначе модификаторы, могут быть самыми разными: марка, модель, цвет, технические характеристики, габариты, локация. Лишь в редких случаях пользователь находит необходимым использование глагола: (“*срочно сниму 1-комнатную квартиру на ул рождественской*”).

³ <https://pypi.org/project/pymystem3/>

2.2 Разметка тестовой выборки по поисковому интену

Чтобы охарактеризовать пары запросов на предмет смены интенга, стоит вспомнить нашу основную мотивацию - получить новое знание о пользовательском поведении. В связи с этим мы сформулировали несколько интересующих нас вопросов:

1. Сколько запросов наберет пользователь, прежде чем решит изменить поисковой интенг?
2. Если поисковой интенг не меняется во время сессии, то как пользователь модифицирует запрос, чтобы достичь цели поиска?
3. Если поисковой интенг изменился, то почему? Потому что пользователь нашел, что искал, или же не нашел и решил искать что-то другое?

Чтобы приблизиться к ответам на эти вопросы, мы разметили небольшую случайную выборку пар запросов. Было выбрано по 100 пар запросов за каждый из 7 дней с весами, согласованными с их частотным распределением. В полученной выборке были удалены дубликаты, была осуществлена ее ручная разметка. Итоговая выборка состоит из 675 пар запросов.

Прежде всего мы разметили примеры по бинарному признаку, где класс 1 означает, что в обоих запросах интенг одинаковый, а класс 0 - что интенг разный (см. Таблицу 5). Данная разметка позволит нам отвечать на первый поставленный вопрос. Мы руководствовались следующими правилами. Если пользователь в двух запросах искал предметы разного назначения или разного функционала (т.е. запросы относятся к разным категориям или, выражаясь точнее, имеют разные вершины), то поисковой интенг различается, и наоборот – если категория объекта поиска совпадает, то запросы имеют одинаковый интенг. В спорных моментах мы пытались встать на сторону пользователя и думали о том, хотел бы он видеть в одной поисковой выдаче одновременно объявления от первого и второго запроса при наборе второго запроса, или нет. Если нет, то эти запросы относились к запросам с разным интенгом.

В классе 1 можно пронаблюдать как минимум 3 типа отношений между запросами.

Во-первых, это практически полное совпадение запросов по лексическому составу и написанию, но различающееся наличием опечатки, порядком слов, способом написания (кириллица vs латиница), либо второй запрос является тем же, что и первый, но записан другими словами. Например: *‘детская кровать – детская кроватка’*, *‘айфон 8 – iphone 8’*, *‘гид – экскурсовод’*, *‘детский стол и стул – стол детский’*.

Во-вторых, это «горизонтальные» синонимичные запросы, которые имеют общую вершину, выраженную эксплицитно или имплицитно. Например, в запросах *‘cobalt - gentra’* скрытой вершиной является категория «автомобиль».

В-третьих, это запросы, соответствующие онтологическим отношениям ЧАСТЬ – ЦЕЛОЕ, ГИПОНИМ – ГИПЕРОНИМ. К этому типу относятся запросы вида *‘водосточная система - изделия из жести’*, поскольку водосточные системы также изготавливают из жести. Здесь первый запрос является подклассом второго запроса, или даже гипонимом, находясь со вторым запросом в «вертикальных» отношениях.

Таблица 5. Бинарная разметка пар запросов в тестовой выборке

Класс	Доля	N	Примеры запроса 1	Примеры запроса 2
1	70%	473	1. детская кровать 2. айфон 8 3. gtx 1070 4. cobalt 5. диски toyota 6. работник в типографию 7. гид 8. пуховик uniqlo 9. водосточная система	1. детская кроватка 2. iphone 8 3. 1070ti 4. gentra 5. диски lexus 6. типография 7. экскурсовод 8. пуховик дутовый 9. изделия из жести
0	30%	202	1. золотые серьги калачи 2. оперативная память 2gb samsung 3. kiturami 4. вешалка передвижная 5. xiaomi pocophone f1	1. золотые часы 2. телевизор плазма 50 3. тележка инструментальная 4. часы ориент 5. ps vita

Класс 0 также обладает своей спецификой. Запросы с разным интендом могут оставаться похожими по тематике. Например, *‘золотые серьги калачи – золотые*

часы'. И серьги, и часы являются аксессуарами, но с точки зрения функционального назначения это разные предметы.

Чтобы определить динамику пользовательского поискового поведения и ответить на второй вопрос, мы сделали еще одну разметку выборки – на 3 класса (см. Таблицу 6). В классе 0 остаются запросы с разным интендом. Бинарный класс 1 мы разделили на 2 подкласса: теперь к классу 1 будет относиться практически полное совпадение запросов или же их переформулировка другими словами. К классу 2 мы отнесли запросы с синонимичными отношениями – горизонтальными и вертикальными. Условно назовем класс 1 «переформулировкой», класс 2 «синонимами».

Таблица 6. Разметка пар запросов на 3 класса

Класс	Доля	N	Примеры запроса 1	Примеры запроса 2
0	30%	202	1. золотые серьги калачи 2. оперативная память 2gb samsung 3. kiturami 4. вешалка передвижная 5. xiaomi pocophone f1	1. золотые часы 2. телевизор плазма 50 3. тележка инструментальная 4. часы ориент 5. ps vita
1	46%	308	1. детская кровать 2. айфон 8 3. работник в типографию 4. гид 5. пуховик uniqlo	1. детская кроватка 2. iphone 8 3. типография 4. экскурсовод 5. пуховик дутовый
2	24%	165	1. cobalt 2. диски toyota 3. gtx 1070 4. водосточная система	1. gentra 2. диски lexus 3. 1070ti 4. изделия из жести

Мотивация к разделению первого класса из бинарной разметки на 2 подвиды следующая: если человек практически не меняет основной запрос, а только переформулирует его несколько раз, то это может свидетельствовать о недостаточной адаптации поисковой системы к особенностям пользовательского ввода. При этом человек хочет увидеть в выдаче фактически один и тот же контент. А если человек набирает синонимичные запросы, то это указывает на то, что либо он не определился с выбором конкретной марки/модели/услуги и выбирает из нескольких вариантов, либо он не нашел то, что ему нужно, при первом запросе, и решил поискать что-то похожее во втором. Более ясную картину можно будет получить, проанализировав другие метрики пользовательского поведения (клики на

объявления и т.д.). В совокупности эти знания помогут ответить на второй и третий поставленные вопросы.

В этой главе мы описали специфику данных, с которыми мы будем работать; сделали разметку выборки на 2 и 3 класса согласно особенностям поискового интента в запросах; сформулировали основные вопросы о пользовательском поведении, на которые должна помочь ответить наша модель, автоматически определяющая смену интента. На основе проведенного анализа выделим несколько основных требований, которым должна отвечать модель:

1. Модель должна хорошо работать на коротких запросах из нескольких слов;
2. Модель должна устанавливать закономерности между одними и теми же сущностями, написанными на кириллице и латинице (*‘айфон – iphone’*);
3. Модель должна уметь работать с грязными данными (в частности, с опечатками);
4. Модель должна уметь распознавать синонимы (*‘гид – экскурсовод’*);
5. При принятии решения о смене интента модель должна распознавать разницу между вершиной и модификаторами в запросах, придавая больший вес вершине.

В следующей главе мы расскажем о методах определения смены интента, существующих в компьютерной лингвистике, которые соотносятся с нашим пониманием задачи.

Глава 3. Подходы к автоматическому определению смены интента

3.1. Методы анализа поисковых запросов

Существует не так много исследований, целиком посвященных автоматическому определению смены интента в поисковых запросах. Это связано, во-первых, с тем, что большинство поисковых систем являются коммерческими, и компании не часто публикуют свои решения. Во-вторых, задача определения смены интента является довольно узкой по сравнению с более широкими задачами классификации запросов по семантике или теме интента (Jansen, Booth 2010; Hernández et al. 2012; Hashemi et al. 2016), задачей автоматического расширения запроса – query refinement (Nogueira, Cho 2017; Diaz 2016). Тем не менее можно выделить несколько направлений исследований, близких к нашей задаче.

В 2000-е было предложено несколько подходов к определению того, является ли очередной запрос перефразированием предыдущего. В рамках первого из них задача сводилась к нахождению границы пользовательской сессии (session boundary detection). Под пользовательской сессией понималась серия запросов, направленных на одну информационную потребность (Jansen et al. 2007, Jones, Klinkner 2008). На первых порах при принятии решения о близости двух запросов в расчет брались только экстралингвистические признаки, такие как временная разница между запросами, позиция запроса в сессии пользователя, совпадение документов в поисковой выдаче.

В последствии фокус сместился на подходы, основанные на посимвольном и пословном расстоянии запросов (Jones, Klinkner 2008; Lucchese et al. 2011). Среди посимвольных расстояний широко использовалось расстояние Левенштейна и его модификации (Levenshtein 1965; Levenshtein 1966), мера сходства Джаро-Винклера, а также измеряющие фонетическое сходство строк алгоритмы Saundex и Phonex. Говоря о пословном сопоставлении, каждый запрос представлялся в виде мешка слов (Bag of Words), затем считалось пересечение слов в двух запросах. В качестве комбинации пословного и посимвольного расстояния можно выделить алгоритм мэтчинга Mongue-Elkan (Jimenez et al. 2009), в котором мерой близости двух строк служит средняя посимвольная близость между ближайшими парами токенов в двух

запросах. Преимущество данного метода в том, что он позволяет решить проблему измененного порядка слов. Подробнее о разных методах сопоставления строк можно прочесть в статье (Christen 2006), анализирующей их эффективность на примере имен собственных.

В работе (Hassan 2013) отмечается главная проблема данных подходов: решения о схожести интента, принятые на основе посимвольной и лексической близости строк, могут давать ложноположительный результат, т.е. говорить об одинаковом интенте, когда на самом деле он разный. Например, в случае пары *'hotels in new york city'* и *'wheather in new york city'* мы имеем практически одинаковые запросы на уровне токенов и символов, но интенты запросов разные и зависят от несовпадающего токена.

Авторы данной работы были одними из первых, кто исследовал именно пары поисковых запросов и попытался решить задачу автоматического предсказания того, является ли запрос Q2 перефразированием запроса Q1. Они адаптировали метрику PMI (Pointwise Mutual Information; Church, Hanks 1990), чтобы выделить внутри запросов ключевые фразы-коллокации. Затем был применен синтаксический парсер, чтобы выбрать в запросе главную фразу (вершину) и зависимые (модификаторы), которые объединялись в один концепт. Для каждой ключевой фразы и концепта были рассчитаны вектора признаков, учитывающие нормализованное расстояние Левенштейна, совпадение токенов и их лемм, контекстную вероятность перехода токенов одного запроса в токены другого (translation probability).

Получившиеся вектора признаков использовались для классификации пар запросов на 2 класса (является ли один запрос переформулировкой другого) и на 4 класса:

1. Генерализация: второй запрос является более общим по сравнению с первым;
2. Конкретизация: второй запрос конкретизирует первый, более общий запрос;
3. Исправление опечаток;
4. Одинаковый интент: второй запрос отличается от первого порядком слов или их частичной заменой.

В качестве модели использовался градиентный бустинг над решающими деревьями и перекрестная проверка на 10 блоках. В качестве данных был использован

размеченный датасет из 6000 запросов коммерческой поисковой системы. Предложенный подход превзошел предыдущие, где признаки не рассчитывались отдельно для вершин и модификаторов запросов.

Исследование имеет много общего с нашим с точки зрения постановки задачи (бинарная и многоклассовая классификация пар запросов по типу реформулировки), однако оно устарело по методам обработки текста. В работе не говорится о том, как сопоставлять запросы с разной орфографией (кириллица vs латиница) и как автоматически выделить вершину и модификатор без синтаксических маркеров.

Существуют другие подходы к выявлению смены интента пользователя, но все они очень зависят от типа онлайн-площадки. В работе (Hienert, Kern 2019) исследователи пытались определять моменты, когда пользователь меняет тему в течение одной поисковой сессии в электронной библиотеке. Было предложено сначала определять тему поиска в каждом запросе, а затем группировать запросы с одинаковыми темами в рамках сессии. Тема запроса выводилась из вероятностного распределения ключевых слов документов по запросу. Однако в данной электронной библиотеке каждому документу изначально соответствовал набор ключевых слов, что свело задачу классификации к простой линейной функции. Также здесь использовались больше экстралингвистические признаки (документы из поисковой выдачи), нежели свойства самих запросов.

Модель, предложенная в (Hienert, Kern 2019), позволяла определять единство темы у синонимичных запросов (*'facebook – instagram'*), но ее недостаток заключался в том, что каждый запрос ассоциировался только с одной темой. Мы решили, что выделять в наших запросах конкретную тему излишне, поскольку каждая вершина запроса фактически и есть тема (*'тойота королла фары'*, *'телевизор samsung бу'*), а, значит, их слишком много.

Еще одно исследование, на которое хотелось бы обратить внимание в контексте анализа запросов, посвящено определению смены интента в серии запросов поисковой системы широкого назначения с помощью кластеризации (Wang, Chen 2011). В качестве данных использовались поисковые логи Майкрософта (MSN Search Query

Log⁴, 2009 г.), состоящие 1) из пользовательских запросов; 2) ссылок, на которые кликнул пользователь по данным запросам; 3) даты и времени поискового действия. Анализировались только те запросы, соответствующие ссылки которых были в Открытом каталоге ODP⁵ – самой большой на тот момент таксономии веб-сайтов с иерархической разметкой на категории (например, веб-сайт <http://www.sinica.edu.tw/> соответствовал категориям *Top/Regional/Asia/Taiwan/Education*).

Авторы статьи кластеризовали поисковые сессии с общим интендом по следующим признакам: токены в запросах, ссылки на веб-сайты, средний и полный «путь из категорий» ODP. Качество кластеризации проверялось на тестовой выборке из сессий, в каждой из которой было по 10 запросов: запросы Q₁-Q₅ соответствовали одному интенду, Q₆-Q₁₀ - другому. Для определения запроса, на который приходится смена интенда, поочередно замерялось Евклидово расстояние каждого запроса до ближайшего кластера, и если оно увеличивалось на очередном запросе на более, чем установленный порог, то считалось, что происходила смена поискового интенда. Порядковый номер запроса, с которого начинался новый поисковой интенд, затем сопоставлялся с реальным порядковым номером.

Несмотря на то, что описанная выше работа использует не так много лингвистических признаков, она демонстрирует возможность анализа всей последовательности запросов из поисковой сессии, а не только текущей пары запросов.

В предыдущих статьях мы увидели, что для анализа смены интенда в поисковых запросах используются как лингвистические (расстояние между символами и токенами в двух запросах), так и экстралингвистические признаки (временная разница между запросами, ключевые слова из поисковой выдачи, клики на ссылки и т.д.). Наверняка их комбинация даст наилучшие результаты в задаче определения смены интенда в поисковой сессии. Однако в нашей работе мы хотели бы сфокусироваться именно на лингвистических свойствах запросов, оставив другие подходы для будущих исследований.

⁴ <http://www.sobigdata.eu/content/query-log-msn-rfp-2006>

⁵ https://ru.wikipedia.org/wiki/Open_Directory_Project

3.2. Определение семантической близости двух строк

Задачу определения смены интента в паре запросов можно рассматривать более широко, как частный случай классификации двух текстовых фрагментов на предмет их сходства или различия. Современные подходы к данной задаче можно широко разделить на два вида:

1. Дистрибутивные модели: вычисление косинусного расстояния между векторными репрезентациями слов, фраз, предложений;
2. Машинное и глубокое обучение с учителем с использованием векторных репрезентаций в качестве признаков.

Для начала рассмотрим дистрибутивные модели.

3.2.1. Дистрибутивные модели

Наиболее популярный способ сравнить 2 слова по смыслу – поместить их в общее векторное пространство размерности d с другими словами так, чтобы вектора слов, похожих по контексту, были расположены близко друг к другу, а далеких по контексту слов – далеко. Именно этот подход был реализован в нейросетевых моделях вида **word2vec** (Mikolov et al. 2013), которые учатся устанавливать зависимости между близкими по контексту словами на большой коллекции текстовых документов. Данные модели реализуют основную идею дистрибутивной семантики, что лингвистические единицы, встречающиеся в схожих контекстах, имеют близкие значения.

Мерой близости двух векторных репрезентаций слов A и B может служить косинусное расстояние между векторами, рассчитываемое по формуле:

$$d = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

Иногда используется геометрическое расстояние Евклида:

$$d = \sqrt{\sum_{i=1}^N (A_i - B_i)^2}$$

Чаще используется косинусное расстояние на нормализованных векторах, поскольку оно не чувствительно к длине вектора.

Чтобы посчитать векторную близость двух фраз, состоящих из нескольких слов, каждую фразу нужно представить в виде единого вектора. Вектора отдельных токенов размерности L можно объединить в один с помощью одной из агрегирующих функций (Shen et al. 2018):

1. Поэлементная сумма или усреднение нескольких векторов (Wieting et al, 2015; Adi et al., 2016). Результатом будет вектор такой же размерности. Иногда перед усреднением применяется L1- или L2-регуляризация.
2. Max-pooling. Это метод агрегации, позаимствованный из сверточных нейронных сетей, в котором мы оставляем на каждой позиции в финальном векторе максимальное значение по всем токенам. Таким образом мы сохраняем в одном векторе информацию о всей фразе, выбирая самые отличительные признаки. Размерность финального вектора – L .
3. Конкатенация результата поэлементного усреднения и Max-pooling. Размерность финального вектора – $2L$.
4. Hierarchical pooling. В данном методе мы проходимся окном размера n по векторам слов, идущих один за другим, и для каждого окна подсчитываем усредненный вектор, затем применяем Max-pooling на получившихся векторах. Метод позволяет учитывать порядок слов.

Полученные данными способами векторные репрезентации можно использовать для подсчета косинусного и Евклидова расстояния между двумя фразами. Окончательное решение об их близости принимается в зависимости от подобранного числового порога: например, если две фразы имеют близость 0.7 или больше (где 1.0 означает совпадение двух слов), то можно сделать вывод о том, они семантически близки.

Существуют разные модификации модели *word2vec*, наиболее известные из них – *Glove* (Pennington et al. 2014) и *fastText* (Bojanowski et al. 2017). При обучении *Glove* используется другая функция оптимизации, в которую закладывается рассчитанная заранее вероятность встретить слово и его контекст вместе; это помогает добиться лучших результатов на задаче составления языковых аналогий и определения синонимичных отношений. В свою очередь, модель *fastText* умеет игнорировать опечатки и формировать вектора слов, отсутствующих в обучающем корпусе, поскольку она обучается не только на полных словах из корпуса, но и на их

символьных n-граммах. Если слово не представлено в корпусе, то его *fastText*-вектор получается через усреднение векторов его n-грамм.

Важно отметить, что большой вклад в успешную работу дистрибутивной модели на конкретной задаче вносит то, на каких данных она обучается. Принято подбирать данные так, чтобы они соответствовали предметной области реального мира, в которой модель будет использоваться.

Одна из проблем данных моделей в том, что каждому слову соответствует только один вектор. В случае многозначного слова (например, «ключ») вектор будет содержать в себе часть всех смыслов слова понемногу, что может негативно отразиться на результатах сопоставления двух текстов. Проблему «статичности» векторов решает другая модификация векторных репрезентаций – контекстуальные вектора слов, такие как BERT (Devlin et al. 2018). Здесь вектора слов формируются динамически в зависимости от контекста, т.е. одно и то же слово в разных контекстах будет иметь разный вектор. Хотя векторные репрезентации из BERT бьют рекорды по качеству на самых разных задачах обработки текста, данная модель подходит не для всех задач и типов данных. Дело в том, что обучение нейросетевой модели BERT осуществляется сразу с двумя целями: предсказание пропущенного слова в предложении и определение того, является ли следующее предложение продолжением предыдущего. Это требует большого числа текстов из нескольких предложений, идущих один за другим. А поскольку тексты запросов являются очень короткими и мы не имеем точного знания о том, вытекает ли текущий запрос из предыдущего в поисковой сессии (по сути, именно это мы должны определить), мы считаем нецелесообразным использовать контекстуальные вектора в рамках задачи определения смены интента по крайней мере на данном, начальном этапе исследования.

3.2.2. Машинное и глубокое обучение с учителем

Агрегирующие операции над векторами с последующим расчетом косинусной близости могут дать ответ только на вопрос о близости двух строк. Они считаются достаточно сильным бейзлайном в задачах обработки естественного языка, где сопоставляются 2 текста: идентификация парафраза, *natural language inference*, *question answering* (Shen et al. 2018). Однако для установления более сложных

зависимостей между текстами и для классификации текстов на 3 и более класса используются более сложные модели машинного и глубокого обучения.

Алгоритмов машинного обучения с учителем для задачи классификации существует достаточно много (логистическая регрессия, мультиномиальный Байесовский классификатор, метод опорных векторов, случайный лес и др.) и их несложно реализовать технически. В этой связи основной задачей исследователя становится подбор признаков для создания тренировочных данных. Если нужно присвоить класс одному тексту, его обычно представляют мешком слов, весами TF-IDF, а также добавляются грамматические и семантические признаки. Однако в случае с сопоставлением пары запросов выбор признаков сужается, ведь они должны отражать не только и не столько смысловое наполнение, сколько отношения между семантикой слов в двух запросов. Например, в запросах *‘тойота королла бампер’* и *‘тойота королла фары’* нужно, чтобы признаки учли семантическую разницу между “бампером” и “фарами”, чтобы посчитать ее значительной для вынесения решения о разном интенте.

С этой точки зрения интересна работа (Joshi 2016). Здесь для определения сарказма предложение представляется в виде матрицы косинусных расстояний между word2vec-векторами слов того же предложения, и затем в качестве признаков выбираются ближайшие и самые дальние расстояния между словами из самых близких и самых дальних пар слов.

На Рис.3 слова *‘man’* и *‘woman’* являются наиболее близкими друг к другу, поэтому в вектор признаков добавляется значение 0.766; также для слова *‘man’* добавляется признак 0.078 как самое минимальное расстояние для этого слова среди других расстояний. То же самое делается для минимального расстояния в матрице: выбирается минимальное значение 0.022 во всей матрице, которое выпадает на слово *‘needs’*, и выбирается максимальное расстояние для этого слова - 0.078. Ничто не мешает рассчитать данные признаки для пар запросов, где слова первого запроса – это строки, а второго – столбцы (или наоборот). Это может указать на скрытые взаимосвязи между словами в двух запросах.

	man	woman	fish	needs	bicycle
man	-	0.766	0.151	0.078	0.229
woman	0.766	-	0.084	0.060	0.229
fish	0.151	0.084	-	0.022	0.130
needs	0.078	0.060	0.022	-	0.060
bicycle	0.229	0.229	0.130	0.060	-

Table 1: Similarity scores for all pairs of content words in 'A woman needs a man like a fish needs bicycle'

Рисунок 3. Извлечение признаков из матрицы косинусных расстояний (Joshi 2016)

Тем не менее в последние годы векторные представления слов показали высокую эффективность в связке именно с глубоким обучением. В задаче сопоставления двух текстов с предсказыванием двух и более классов выделяют два основных типа нейросетевой архитектуры (Wang et al. 2017a).

Первый – сиамские нейронные сети. Их особенностью является то, что один и тот же энкодер на основе сверточных (CNN) или рекуррентных сетей (RNN) (Hochreiter, Kepler 1997) применяется к первой и второй последовательности векторов токенов таким образом, чтобы оба текста были закодированы в одном векторном пространстве. Фактически это те же агрегирующие функции векторов нескольких слов в один, только более сложные: CNN учитывает порядок слов и словосочетаний относительно друг друга и агрегирует зависимости на разных уровнях в зависимости от размера сверточных фильтров, RNN же учитывает больше контекстуальные зависимости. Например, благодаря механизму памяти в RNN единый стационарный вектор слова из дистрибутивной модели меняется в зависимости от того, какие слова его окружают в данном тексте. На основе векторов, агрегированных с помощью RNN и CNN, осуществляется классификация двух текстов (Tan et al. 2015).

Недостаток данного типа модели в том, что во время процедуры кодирования слова из двух текстов не взаимодействуют друг с другом, и мы теряем важную информацию об отношениях между ними. Преодолеть данный недостаток позволяет другая архитектура нейронной сети - matching-aggregation (Wang et al. 2017b). В данном подходе сначала происходит сопоставление слов двух текстовых фрагментов (matching, выравнивание), и затем результат агрегируется с помощью CNN или LSTM (разновидность RNN) в общий вектор для принятия финального решения о

принадлежности пары строк к какому-либо классу. Данный подход сложнее с точки зрения имплементации из-за необходимости выбора подходящей функции пословного выравнивания, но он показал большую эффективность на задачах, частично похожих на нашу: идентификации парафраз (Wang et al. 2017a) и textual entailment (определение того, является ли второе предложение продолжением к первому) (Wang et al. 2017b).

В работе (Wang et al. 2017b) авторы предлагают базовую архитектуру нейронной модели matching-aggregation:

1. Энкодер на основе LSTM, который обогащает вектора слов контекстной информацией;
2. Механизм внимания (Bahdanau et al. 2014; Vaswani et al., 2017), заимствованный из машинного перевода. В ходе операций над матрицами векторов слов из двух текстов (перемножение, косинусная близость, линейные функции) он запоминает то, как и какие слова из первого текста связаны со словами из второго;
3. Одна из шести функций сравнения векторов из механизма внимания;
4. Агрегирующий слой CNN.

Исследователи тестируют 6 функций сравнения векторов слов двух текстов, получаемых на выходе из механизма внимания:

1. Полносвязный линейный слой с нелинейной функцией активации;
2. Тензорная нейронная сеть;
3. Конкатенация значений Евклидова и косинусного расстояния;
4. Поэлементное возведение в квадрат разницы между векторами;
5. Поэлементное произведение векторов;
6. Конкатенация результатов функций 4 и 5 с последующим применением полносвязного линейного слоя.

Последняя функция показала наилучший результат на задаче text entailment – классификации пар текстов на 3 класса: один текст следует из другого; один текст противоречит другому; один текст не связан с другим.

Существует много других работ, так или иначе реализующих подход matching-aggregation внутри нейронных сетей. В (Wang et al. 2017a) оценивается еще несколько функций сравнения, основанных на косинусной близости с использованием механизма внимания и max-pooling, при этом сравнение происходит в обе стороны (первое предложение по отношению ко второму и второе предложение по отношению к первому). В (He, Lin 2016) используется алгоритм попарного сравнения слов и n-грамм двух текстов (Pairwise Word Interaction Model), концептуально напоминающий механизм внимания. Не менее интересно исследование (Wang et al. 2016), где вектора слов в парах предложений раскладываются на схожий и противоречащий компоненты (similar and dissimilar components) с помощью косинусного и геометрического расстояния.

В окончание обзора литературы стоит отметить, что нейросети не всегда побеждают более простые методы - косинусную близость между векторами в случае двухклассовой классификации и традиционное машинное обучение в случае трехклассовой. Наша гипотеза состоит в том, что мэтчинг слов в двух коротких запросах с дополнительно обучаемыми параметрами (помимо векторных репрезентаций) улучшит качество модели. Подтвердится гипотеза или нет, покажут эксперименты.

Глава 4. Эксперименты

4.1 Постановка задачи

В предыдущей главе мы осветили возможные методы автоматического сопоставления двух запросов. На наш взгляд, многие из них являются релевантными для построения базовых моделей, определяющих смену поискового интента.

Отталкиваясь от похожих исследований по определению типа реформулировки запроса и задачи *textual entailment*, мы поставили перед собой задачу классифицировать пары текстовых запросов на 2 и 3 класса: «одинаковый интент – разный интент», «переформулировка – синоним – разный интент». В ходе экспериментов протестированы две основные парадигмы моделей: без учителя (дистрибутивные модели) и с учителем (машинное и глубокое обучение).

Все модели вскоре будут доступны в репозитории проекта⁶.

4.2. Разметка обучающей выборки

4.2.2. Разметка методами символьного и лексического расстояния

Для обучения моделей с учителем требуется разметка как минимум нескольких тысяч пар запросов на 2 и 3 класса. Мы выбрали около 10 тыс. пар из всей совокупности запросов в качестве тренировочных данных. Прежде всего мы решили применить методы посимвольного расстояния между двумя строками, чтобы автоматически разметить часть данных для бинарной классификации.

В качестве методов мы использовали нормализованное расстояние Дамерау-Левенштейна, метод Mongue-Elkan с расстоянием Джаро-Винклера, пересечение слов в запросах. Эти методы позволяют находить одинаковый интент в паре запросов, если оба являются почти идентичными и в них либо допущены опечатки, либо присутствует замена символов, либо слова переставлены местами.

Пороги для принятия решения о близости интента в двух запросах подбирались эмпирически отдельно для каждого расстояния, в случае расстояния Левенштейна учитывалась длина запроса в символах и число слов в запросе. Например, для запросов из двух вида *‘кожаная сумка – кожаная юбка’* мы не можем сделать вывод об одинаковом интенте, даже если расстояние Левенштейна между ними меньше 0.5,

⁶ <https://github.com/annnyway/automatic-intent-shift-detection.git>

так как объект поиска разный. В Таблице 7 приведены подобранные пороги для разных методов.

Таблица 7. Пороги для определения символьной и лексической близости

<i>Метод</i>	<i>Если 1 токен и в нем больше 5 символов</i>	<i>Если 2-3 токена</i>	<i>Если больше 3 токенов</i>
Дамерау-Левенштейн	0.4	0.2	0.3
Mongue-Elkan	-	0.15	0.15
Пересечение слов	0.25	0.25	0.25

Перед подсчетом расстояний для пар запросов, где есть переход из кириллицы в латиницу, к латинице применялся специально разработанный нами конвертер, переводящий слово с латинским написанием в кириллицу согласно его фонетическому составу. Мы использовали модуль Epitran⁷ (Mortensen et al. 2018), чтобы перевести латинское слово в его транскрипцию; затем срабатывали наши правила о переводе каждого символа транскрипции в символы кириллицы, наиболее подходящие по звучанию. В результате получался приближенный перевод:

1. *'lego duplo'* → *'лего дупло'*,
2. *'nexia'* → *'нексия'*,
3. *'monster high'* → *'монстер хай'*,
4. *'мультиварка redmond'* → *'мультиварка редменд'*,
5. *'bratz сваровски'* → *'бретс сваровски'*.

Конечно, наиболее эффективным методом было бы использование переводчика, но мы не нашли бесплатных и быстро работающих решений.

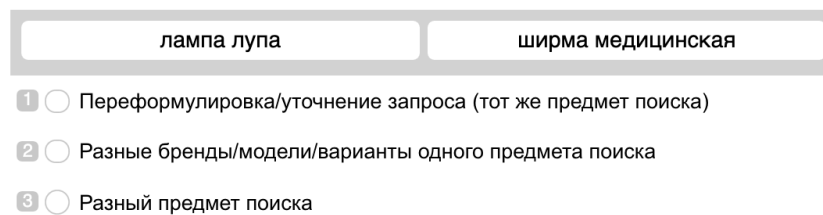
Данными методами мы смогли разметить только 18% подвыборки из 10 тыс. пар запросов, присудив данным запросам класс 1 (одинаковый интент). Это достаточно большой процент запросов с высокой частотой.

4.2.2 Разметка в Яндекс.Толоке

⁷ <https://pypi.org/project/epitran/>

Для разметки остальных запросов из 10 тыс. мы обратились к Яндекс.Толоке⁸ – онлайн-площадке для быстрой разметки большого количества данных, которые затем используются для машинного обучения и совершенствования поисковых алгоритмов.

Мы подготовили для «толокеров» инструкцию по разметке запросов на 3 класса (см. Приложение 1). Задание включало в себя предварительное обучение на 16-ти парах запросов, содержащих сложные случаи, в том числе вертикальные синонимы. На одной странице задания размещалось 20 пар запросов с 3-мя вариантами ответа, 6 из которых контрольные. На каждую страницу отводилось по 20 минут (~1 минута на каждую пару). В случае затруднения при принятии решения разметчики могли кликнуть на запросы и перейти в поисковую систему Google или на доску объявлений и посмотреть поисковую выдачу на данные запросы. Пример интерфейса задания можно увидеть на Рис. 4.



лампа лупа	ширма медицинская
------------	-------------------

1 ☐ Переформулировка/уточнение запроса (тот же предмет поиска)

2 ☐ Разные бренды/модели/варианты одного предмета поиска

3 ☐ Разный предмет поиска

Рисунок 4. Пример интерфейса разметки пар запросов на Яндекс.Толоке

Настройки основного пула: 50% лучших толокеров; перекрытие – 5 человек; блокировка толокеров на несколько дней по проценту неправильных ответов на задания, за слишком быстрые ответы, слишком большой заработок за сутки, пропущенные страницы. Такая блокировка позволяет быстро отсеять тех, кто размечает плохо. Однако мы позволяли толокерам ошибаться в одном из шести контрольных заданий, поскольку понимали, что задача достаточно сложная, и существуют спорные моменты.

Толокеры разметили около 7,5 тыс. пар запросов с 80%-уверенностью, т.е. в среднем на каждый пример 4 из 5 разметчиков ответили одинаково. В итоговую тренировочную выборку для трехклассовой классификации мы включили только

6273 пар запросов, которым общий класс проставили 4 или 5 человек из 5-ти. Соотношение размеченных классов приведено в Таблице 8.

Таблица 8. Распределение разметки с помощью Яндекс.Толоки по 3 классам

Класс	Название класса	Абсолютное число пар	Доля класса
1	Переформулировка	1970	0.44
2	Синонимы	1568	0.25
3	Разные запросы	2735	0.31
Всего		6273	1.0

Мы не включали в разметку на Яндекс.Толоке пары запросов, уже размеченные с помощью посимвольного и лексического расстояния. Для бинарной классификации мы просто объединили их с примерами, размеченными толкерами, в пропорции 18/100. Именно 18-ти процентам запросов присваивался класс 1 (одинаковый интент) в реальных данных. В итоге в обучающей выборке получилось 8973 пары запросов для бинарной классификации.

4.3 Обучение дистрибутивной модели

На основании обзора литературы по теме мы выбрали в качестве основных признаков запросов вектора слов, полученные из дистрибутивной модели *fastText*. Выбор именно этой модели обусловлен тем, что она обучается не только на полных словах, но и на символьных *n*-граммах, что позволяет получать очень близкие вектора для слов с опечатками и предсказывать вектора для языковых единиц, которые модель никогда не видела. Такие вектора являются суммой векторов символьных *n*-грамм (3-граммы, 4-граммы, 5-граммы), из которых состоит данное слово.

Мы не использовали «готовые» модели *fastText*, заранее обученные на текстовых корпусах⁹, не имеющих отношения к нашим поисковым запросам. Отталкиваясь от специфики данных, мы сочли целесообразным обучить собственную модель *fastText* на заголовках объявлений и адресах с той же онлайн-площадки. Заголовки были выбраны потому, что в них, с одной стороны, наиболее широко представлены все возможные объекты поиска, а с другой – их много (десятки

⁹ <https://rusvectors.org/ru/models/>

миллионов) и они состоят в среднем из 4-5 слов, что достаточно для установления контекстных зависимостей между словами. Адреса также были выбраны не случайно: пользователи часто ищут объявления о продаже или сдаче в аренду недвижимости, осуществляя поиск по улицам или районам, которых нет в заголовках. От использования текстов объявлений было решено отказаться, поскольку, на наш взгляд, они могут содержать слишком специфическую информацию о предмете поиска, тогда как нас интересуют скорее ключевые слова. Более того, в одном объявлении могут предлагаться сразу несколько объектов поиска, относящихся к разным типам интента, что может негативно отразиться на качестве векторных представлений.

Мы попробовали обучить две модели *fastText*. В обеих моделях размерность векторов была выбрана 200, обучение заняло 20 эпох, размер контекстного окна – 7 (фактически весь заголовок). Модели различались только одним гиперпараметром: одна обучалась на всех токенах (2 485 668 уникальных токенов), другая – на токенах с частотностью 3 и выше (992 867 уникальных токенов).

Мы сравнили модели на задаче предсказания топ-30 слов, наиболее близких к целевому слову по косинусному расстоянию. Модель с минимальной частотой токенов, равной 3, выдает более четкие и разнообразные предсказания с точки зрения лексического состава, тогда как модель, обученная на всех токенах, чаще предсказывает одинаковые слова и фрагменты слов с опечатками. В то же время на модель *fastText* выучила некоторые различия в семантике одного и того же слова «дворник» в зависимости от того, в каком контексте оно употребляется в единственном и множественном числе («дворник» как работа и «дворники» как стеклоочиститель).

Значения косинусного расстояния между запросами из тестовой выборки показали, что модель, обученная с минимальной частотой 3, на 1 п.п. лучше справляется с задачей бинарной классификации. Это было предсказуемо, ведь из-за более узкого лексического состава она может устанавливать связи с более широким множеством языковых сущностей. Поэтому мы выбрали вторую модель для основных экспериментов.

4.4 Бинарная классификация

4.4.1 Описание моделей

В данном параграфе мы расскажем о типах моделей для бинарной классификации, которые мы тестировали.

Первый тип – **методы агрегации векторных представлений** каждого запроса с последующим подсчетом косинусной близости между запросами. Мы использовали следующие методы агрегации:

1. Усреднение векторов, разделенных на их норму (базовая модель);
2. Max-pooling;
3. Конкатенация результата поэлементного усреднения и Max-pooling;
4. Hierarchical pooling;
5. Усреднение векторов, взвешенных с помощью IDF.

В качестве бейзлайна мы выбрали самый первый метод – эта агрегация встроена в модель *fastText*. Про 3 следующих метода можно прочесть подробнее на с.23-24. Кратко расскажем о методе 5. Оригинальная метрика IDF для слова – это логарифм от числа документов с этим словом, деленного на число документов в коллекции. При расчете IDF мы брали в качестве документов заголовки объявлений и вычислили IDF как отрицательный логарифм отношения числа объявлений (x), в которых встречается слово, ко всем объявлениям (y):

$$IDF = -\log \frac{x}{y}$$

Полученное значение IDF мы затем домножали на все размерности вектора данного слова. Мотивация использования IDF для взвешивания заключалась в том, что более редкие слова будут иметь более яркие отличительные признаки.

После агрегации мы посчитали косинусную близость между векторами запросов на тренировочной выборке, чтобы определить порог, максимизирующий вероятность правильного ответа. Порог замерялся с шагом 0.5 на промежутке [0.3; 0.6] и оказался равным 0.4 для всех методов. Затем мы посчитали косинусное расстояние между запросами из тестовой выборки и отнесли к классу 0 пары запросов, у которых косинусная близость была меньше 0.4, и к классу 1 – все остальные.

Следующий тип модели – это **классическое машинное обучение**. В качестве алгоритма обучения был использован случайный лес (Breiman, 2001) состоящий из

множества решающих деревьев. Решение модели о том или ином классе принимается голосованием деревьев по большинству, при этом все деревья строятся независимо.

При выборе признаков для машинного обучения, характеризующих наши данные, мы отталкивались от основной цели работы – подобрать модель, которая, с одной стороны, смогла бы различать главные и зависимые слова в запросах (вершины и модификаторы), с другой – устанавливая связи между ними в двух запросах. Мы воспользовались идеей из работы (Joshi 2016), где предложение представлялось матрицей смежности из косинусных расстояний между словами, а затем из матрицы выбирались ближайшие и дальнейшие расстояния для близких и дальних пар слов. Мы адаптировали данный подход к нашей задаче и рассчитали для каждой пары запросов один вектор из следующих признаков:

1. Максимальная косинусная близость между словами двух запросов;
2. Минимальная косинусная близость между словами двух запросов;
3. Минимальная косинусная близость для слова с максимальной косинусной близостью из первого запроса;
4. Максимальная косинусная близость для слова с минимальной косинусной близостью из первого запроса.

По тому же принципу мы извлекли признаки из матрицы попарных разниц значений IDF для каждого слова (чем больше разница между IDF двух слов, тем они более различны по степени специфичности).

Также мы добавили в качестве признаков косинусное расстояние между усредненными векторами запросов, Евклидово расстояние, расстояние Дамерау-Левенштейна между запросами, расстояние Mongue-Elkan, расстояние по пересечению слов, число слов в первом запросе, число слов во втором запросе. Всего получилось 15 признаков для каждой пары запросов.

Мы использовали модель *Random Forest Classifier* из библиотеки *scikit-learn* с гиперпараметрами: *max_depth* = 20, *n_estimators* = 100, *class_weight* = {0:0.3, 1:0.7}, *bootstrap* = True.

Третий вид модели бинарной классификации – **нейронные сети**. В ходе экспериментов мы комбинировали разные архитектуры сетей, двигаясь от более простых к более сложным. Перечислим их:

1. *LSTM + CNN + Linear*: векторы *fastText*, энкодер LSTM и последующая агрегация CNN с линейным слоем;
2. *Annt + Linear*: Векторы *fastText*, механизм внимания с линейным слоем;
3. *Attn + Max-pooling + Linear*: векторы *fastText*, механизм внимания, Max-pooling и линейный слой;
4. *BiAttn + Linear1 + Linear2*: векторы *fastText*, двусторонний механизм внимания (Q1 по отношению Q2 и Q2 по отношению к Q1) с конкатенацией результата и применением двух линейных слоев с нелинейной функцией активации между ними;
5. *LSTM + Attn + Linear*: векторы *fastText*, LSTM, механизм внимания, линейный слой;
6. *BiAttn + Orthogonal + CNN + Linear*: векторы *fastText*, двусторонний механизм внимания, функция мэтчинга векторов внимания с помощью ортогональной проекции (Wang et al. 2016), CNN, линейный слой.
7. BIMPM: адаптированная под наши данные модель из работы (Wang et al. 2017a). В ней используются векторы *fastText*, сконкатенированные с векторами посимвольных n-грамм, которые обучаются вместе с моделью. Векторы слов подаются в BiLSTM-слой для получения контекстных представлений. Затем контекстные представления слов двух запросов передаются в специальный слой для мэтчинга, в котором каждое представление из одного запроса сопоставляется со всеми представлениями в другом запросе, и наоборот. Для сопоставления использовались 4 функции сравнения на основе косинусной близости (см. Рис.5), затем их результат конкатенировался, агрегировался с помощью BiLSTM, и к ним применялся линейный слой.

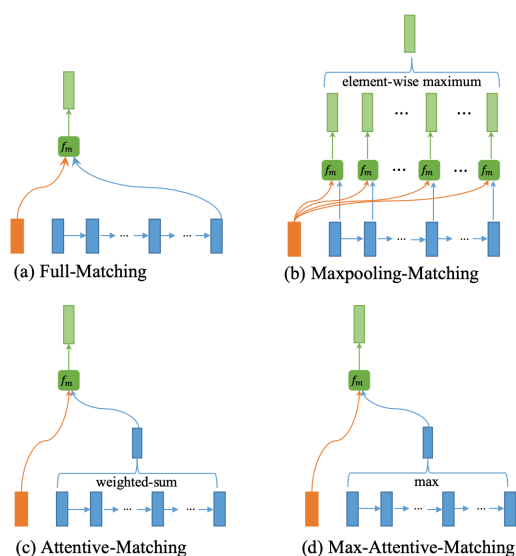


Рисунок 5. Функции мэтчинга из (Wang et al. 2017a).

Вышеперечисленные нейронные модели кроме последней писались специально для настоящей работы с помощью фреймворка *Pytorch*.

В первых 6-ти моделях мы использовали в качестве функции потерь бинарную кросс-энтропию, в 7-й – функцию отрицательной логарифмической вероятности (Negative Log-Likelihood). В качестве оптимизатора использовался алгоритм Adam. Модели тренировались на батчах размера 32 или 64. Максимальное число слов в запросе – 10. Такие гиперпараметры, как скорость обучения и размерность скрытых слоев, выбирались исходя из архитектуры моделей: в одних моделях мало параметров и был риск их перетренировать с высокой скоростью обучения (например, 0.001), тогда как в случае LSTM, где число параметров увеличивалось на порядок, наоборот, требовалось увеличить скорость обучения. Иными словами, мы старались найти компромисс между скоростью обучения и размерностями слоев.

Стоит признать, что около 9 тысяч размеченных данных для нейросети – это не так много, и они представляют собой очень короткие фразы. В этом причина того, что для некоторых моделей было достаточно 2-3 эпох, чтобы добиться наилучшего результата на тестовой выборке. В целом обучение моделей занимало на GPU 1-5 эпох.

4.4.2 Результаты

Для оценки моделей бинарной классификации мы использовали 5 метрик:

1. *Precision*;
2. *Recall*;

3. *F-score* - F-мера для каждого класса;
4. *F-score (macro)* - F-мера для двух классов с макроусреднением, т.е. с одинаковыми весами для каждого класса;
5. *F-score (weighted)* - F-мера для двух классов, взвешенная долей каждого класса в тестовой выборке.

Мы оценили все вышеперечисленные модели на тестовой выборке – 675 примерах, которые не были включены в обучение. Эксперименты показали, что далеко не все модели преодолели базовую (см. Таблицу 9).

Если говорить о методах, основанных на агрегации векторов, то лучше всего себя показали модели с Hierarchical Pooling и IDF-взвешиванием: распознавание классов 0 и 1 увеличилось на 1-2 пп. Еще лучше показали себя признаки из векторных представлений с алгоритмом случайного леса – качество определения целевого класса 0 (разный интент) повысилось на 5 п.п., макро-F-мера и взвешенная F-мера – на 3 п.п. Сопоставимые результаты среди нейронных моделей показали те, в которых использовались специальные функции мэтчинга поверх механизма внимания, что и было ожидаемо.

Таблица 9. Результаты бинарной классификации на тестовой выборке

Метод	Класс	<i>Precision</i>	<i>Recall</i>	<i>F-score</i>	<i>F-score (macro)</i>	<i>F-score (weighted)</i>
<i>Average + L2 + Cosine (baseline)</i>	0	0.70	0.73	<u>0.71</u>	<u>0.79</u>	<u>0.82</u>
	1	0.88	0.86	<u>0.87</u>		
<i>Max-pooling + Cosine</i>	0	0.61	0.54	0.57	0.70	0.75
	1	0.81	0.85	0.83		
<i>[Average, Max-pooling] + Cosine</i>	0	0.61	0.54	0.58	0.71	0.76
	1	0.81	0.85	0.83		
<i>Hierarchical-pooling + Cosine</i>	0	0.72	0.74	0.73	0.80	0.83
	1	0.89	0.88	0.88		
<i>IDF + Average + Cosine</i>	0	0.73	0.71	0.72	0.80	0.83
	1	0.88	0.89	0.88		
<i>RandomForest</i>	0	0.70	0.83	0.76	0.82	0.85
	1	0.92	0.85	0.88		
<i>LSTM + CNN + Linear</i>	0	0.57	0.55	0.56	0.69	0.74
	1	0.81	0.82	0.82		
<i>Attn + Linear</i>	0	0.64	0.80	0.71	0.78	0.81
	1	0.90	0.81	0.85		
<i>Attn + Max-pooling + Linear</i>	0	0.65	0.55	0.60	0.70	0.70
	1	0.75	0.78	0.76		
<i>LSTM + Attn + Linear</i>	0	0.54	0.41	0.47	0.64	0.71
	1	0.77	0.85	0.81		
<i>BiAttn + Linear1 + Linear2</i>	0	0.68	0.83	0.75	0.81	0.84
	1	0.92	0.83	0.87		
<i>BiAttn + Orthogonal + CNN + Linear</i>	0	0.74	0.77	0.75	0.82	0.85
	1	0.90	0.89	0.89		
<i>BIMPM</i>	0	0.76	0.79	0.77	0.84	0.86
	1	0.91	0.89	0.90		

Можно сказать, что механизм внимания вносит частичный вклад в улучшение качества модели, несмотря на небольшое количество тренировочных данных. Двустороннее внимание с линейными слоями превзошло базовую модель на 4 п.п. в определении разного интенга.

Хуже всего себя показали модели с Max-pooling. На основании этого можно сделать вывод, что в определении смены интенга в двух запросах, по крайней мере коммерческих, порядок слов роли не играет, и что подавлять менее выраженные элементы векторов не стоит, поскольку они имеют ценность в принятии решения. И это, в свою очередь, подтвердило IDF-взвешивание.

Сложно сказать что-либо определенное о вкладе LSTM и CNN в работу моделей. Вероятно, число слов в запросах слишком мало, чтобы эти архитектуры могли значительно повлиять на качество.

4.4.3 Анализ ошибок

Теперь посмотрим на то, в каких парах запросов интенга распознаются хорошо, а в каких – плохо. Для этого сравним результаты базовой модели, Hierarchical-pooling, случайного леса и BIMPM.

Все 4 модели ошиблись в 45 одинаковых тестовых примерах из 675. Среди причин ошибок в классе «одинаковый интенга» можно выделить:

1. Случаи вертикальной синонимии:

- (1) *“водосточная система - изделия из жести”*
- (2) *“2110 - 4 болтовые диски”* (подразумевается «ваз 2110», отношения «часть-целое»)

2. Пары запросов с многозначными словами и грамматической омонимией:

- (3) *‘серебро – xbox’*, («серебро»)
- (4) *‘мыло ручной работы - ручной сканер’*, («ручной»)
- (5) *‘асбестовый лист – прокладки’* («прокладки»)

3. Спорные случаи, чаще всего связанные с одеждой и аксессуарами, когда в одном из запросов есть бренд одежды, в другом – предмет одежды, в т.ч. с другим схожим по тематике брендом. Мы относили такие случаи к синонимам (если разные бренды), или к переформулировке (если предмет одежды – бренд), т.е. к классу 1, но модели часто не могли его определить:

(6) *'jack wolfskin – рюкзак thule crossover 2'*

(7) *'майкл корс – джинсы женские 29'*

(8) *'платье zimmerman оригинал - free people'*

4. Спорные случаи, где даже человеку трудно определиться с правильным классом:

(1) *'гироскутер бу - yamaha sr400'* (глобально интендом является средство передвижения, хотя гироскутер и мотоцикл совсем разные)

(2) *'кирпич – спецтехника'* (даже если это переформулировка, то довольно общая)

5. Пары с редкими неопределенными запросами, которых недостаточно в тренировочных данных, но которые могут быть переформулировкой:

(9) *'cobalt - под выкуп'*

5. Пары коротких запросов с аббревиатурой, указанием марок товара и т.п.

(10) *'sfx -i3 9100f'*

У класса «разный интенд» были следующие причины ошибок:

1. Повторяющиеся слова, являющиеся атрибутом интенда:

(11) *'детские коляски – детские игрушки'*

(12) *'золотые серьги калачи – золотые часы'*

(13) *'тормозные диски lancer 9 – бампер lancer 9'*

2. Запросы из одной сферы, но с разным интендом:

(13) *'кабуки chanel – помада givenchy'*

(14) *'кружево платье белое – джинсовый корсет'*

(15) *'миксер kitchenaid – кофемашина wmf'*

Часть этих ошибок говорит о негативном влиянии «статичности» векторов слов (многозначные слова и омонимы, короткие запросы), другая часть – о трудностях разметки и ее возможном несовершенстве. Оказалось, что модели часто ошибаются с классом, если вертикальные или горизонтальные синонимичные отношения между запросами неочевидны.

У каждой модели можно выделить ошибки, характерные только для нее. Например, в случае Hierarchical Pooling ошибки возникали из-за порядка слов: одинаковый интенд не распознал в запросах *'заднее колесо 26 - 26 jmjkl'*. Также модель не определяет смену интенда, если один запрос значительно короче другого:

(16) *'обувь женская 39 размер ботинки - дома из бруса'*

(17) *'мужские плавки серые камуфляжные clever severo swimsuit brief 069111 - samsung galaxy s8'*

В модели со случайным лесом можно выделить типичную ошибку: она не может установить синонимичную связь в парах запросов, если один из них состоит из одного слова:

- (18) *'5c айфон - a10 битый'*,
- (19) *'armani junior – florens'*,
- (20) *'art audio lab – gryphon'*,
- (21) *'braccialini - twin set'*,
- (22) *'сторож - мясник'*

Это закономерно, ведь на очень коротких запросах матрицы расстояний, из которых мы выбирали признаки для случайного леса, не работают. Другими словами, недостатком модели является то, что она никак не учитывает тематику запросов, уделяя внимание только разнице и сходству между ними.

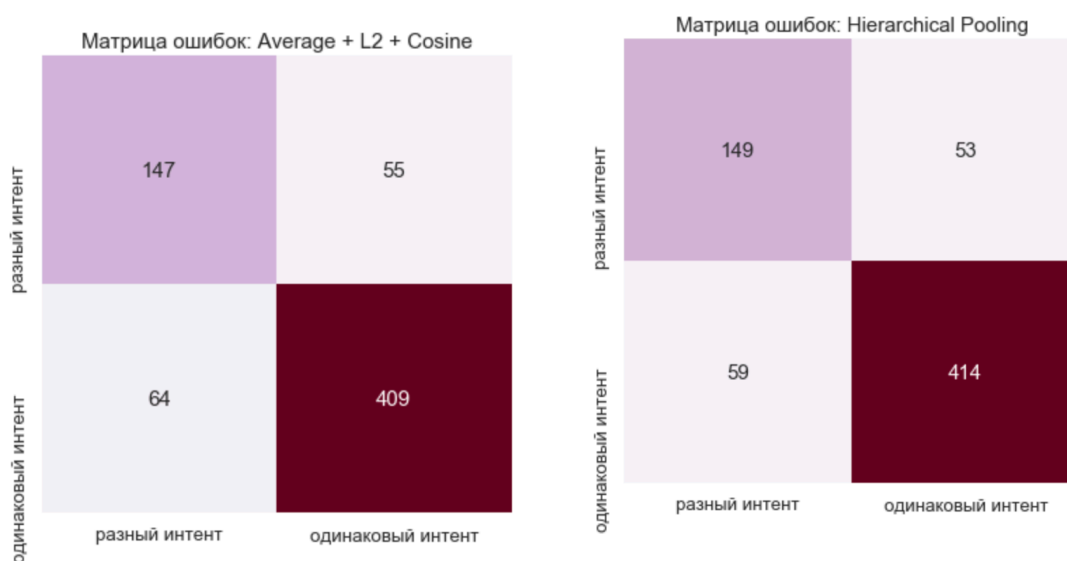
Что касается модели BIMPМ, показавшей самый лучший результат, она не уловила разницу между локацией и поиском работы: *'галина – продавец'*, а также не сочла одинаковым интендом сложные переформулировки:

- (23) *'лошади - советский тяжеловоз'*
- (24) *'мини маслaбойка - машина ажим семичка'*
- (25) *'для пекарни - пончик аппарат'*.

И не распознала несколько синонимов:

- (26) *'раскладушка – кушетка'*
- (27) *'эквалайзер - акустика'*
- (28) *'шапка зверюшка - knitwits'*.

Количественный анализ ошибок моделей можно наблюдать на Рисунке 6.



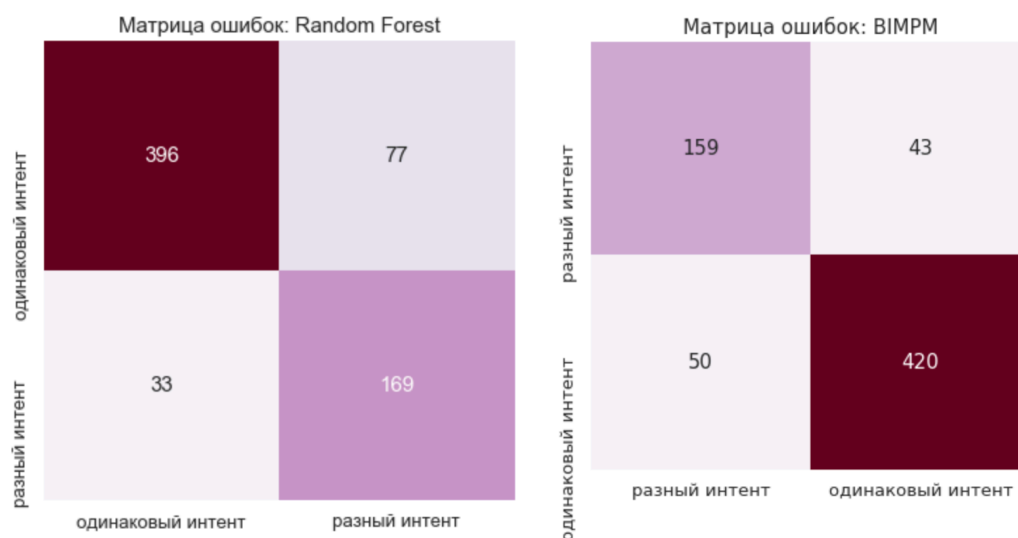


Рисунок 6. Матрицы ошибок моделей бинарной классификации

4.5 Классификация на 3 класса

4.5.1 Описание моделей

Для классификации на 3 класса были выбраны 3 модели машинного и глубокого обучения, хорошо показавшие себя на бинарной классификации. Это случайный лес, *BiAttn + Orthogonal + CNN + Linear* (пункт 6 на с.36), BIMP. В этот раз мы хотим проверить, насколько точно модель может распознать разный интент (0), наиболее сильную связь между запросами (переформулировка, 1), и промежуточную ситуацию (вертикальные и горизонтальные синонимы, 2). В тренировочных данных эти классы представлены в соотношении [0.44, 0.31, 0.25].

4.5.3 Результаты

Модели 3-классовой классификации оценивались по тем же метрикам: *Precision*, *Recall*, *F-score (macro)*, *F-score (weighted)*. Также мы добавили метрику *F-score (micro)*, в которой перед подсчетом средней F-меры учитывается вес каждого класса. Эта метрика используется в мультиклассовой классификации при дисбалансе классов.

В Таблице 9 приведены результаты 3-классовой классификации. Лучше всего по метрикам F-меры сработала модель BIMP. Можно объяснить это тем, что в модели тренируется в разы больше параметров, чем в остальных, за счет чего она может устанавливать более сложные зависимости между запросами.

Разный интент лучше всего определяется моделью с алгоритмом случайного леса. Мы полагаем, что признаки, которые мы подобрали для машинного обучения,

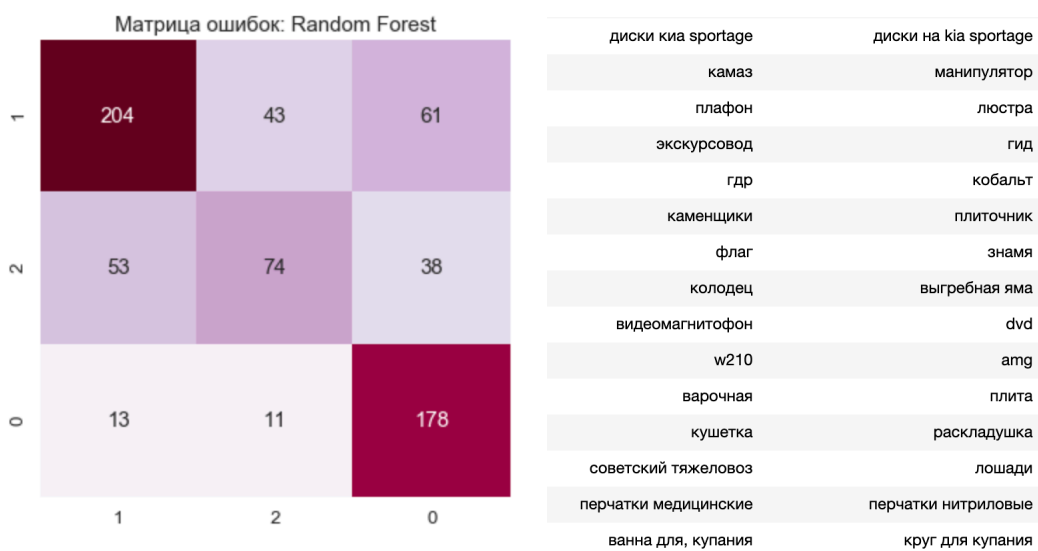
хорошо ловят именно различия в двух запросах. В свою очередь, нейронные модели лучше справились с другими классами, уравнивая F-score по трем классам. По этой причине их матрицы ошибок (Рис.6) выглядят более гармоничными.

Таблица 9. Результаты классификации на 3 класса

Метод	Класс	<i>Precision</i>	<i>Recall</i>	<i>F-score</i>	<i>F-score (micro)</i>	<i>F-score (macro)</i>	<i>F-score (weighted)</i>
<i>RandomForest</i>	0	0.64	0.88	0.74	0.68	0.65	0.67
	1	0.76	0.66	0.71			
	2	0.58	0.45	0.51			
<i>BiAttn + Orthogonal + CNN + Linear</i>	0	0.73	0.58	0.65	0.66	0.66	0.66
	1	0.66	0.76	0.70			
	2	0.56	0.68	0.62			
<i>BIMPM</i>	0	0.70	0.73	0.72	0.71	0.70	0.71
	1	0.78	0.75	0.76			
	2	0.62	0.61	0.61			

4.5.4 Анализ ошибок

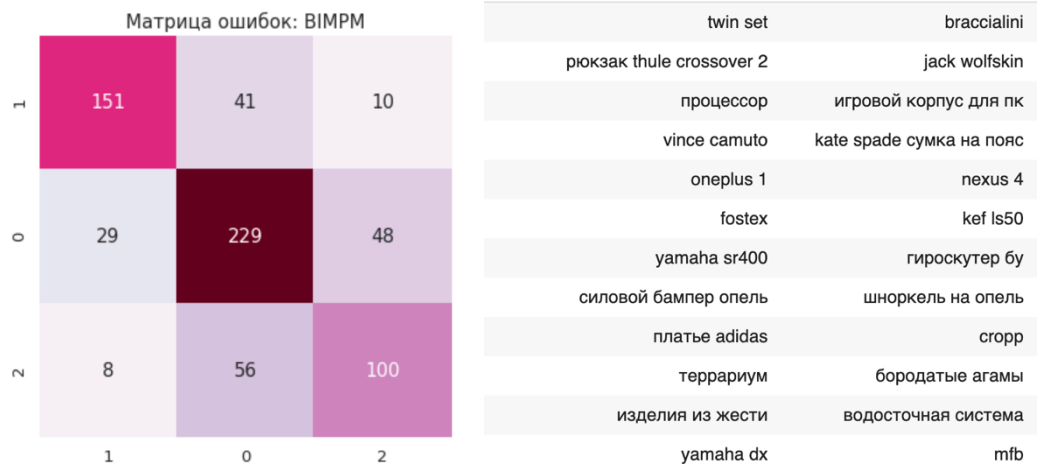
Как мы видим на Рис. 7-8, в модели со случайным лесом класс 2 (синонимы) пересекался с классом 1 (переформулировка) – 43 пары запросов. Т.е. то, что было принято за синонимы, на самом деле – переформулировка. Это, как и в случае бинарной классификации, короткие слова, запросы из 1-2 слов. Модели просто не хватает признаков для определения класса двух запросов. При этом она неплохо справляется с отделением синонимов от запросов с разным интендом.



Рисунки 7-8. Матрица и примеры ошибок случайного леса.

Примечание к матрице: по горизонтали – теги в тестовой выборке, по вертикали – предсказания.

Лучше всего синонимы определяет модель 2 с геометрической функцией мэтчинга (113 правильных ответов против 74 из предыдущей модели). Последняя модель, показавшая наилучшие результаты, путает класс 0 с классом 2 (см. Рис.9-10).



Рисунки 9-10. Матрица и примеры ошибок модели BIMPM. Примечание к матрице: по горизонтали – теги в тестовой выборке, по вертикали – предсказания.

4.6 Общие выводы

В данной главе мы рассказали о 3 типах моделей, которые можно использовать для определения смены интента. В задаче бинарной классификации мы с помощью нейросетей улучшили метрики на 3-6 пп. по сравнению с простой агрегацией векторов слов, которая в силу специфики данных оказалась сильным бейзлайном. Модель с алгоритмом случайного леса с меньшим числом обучаемых параметров показала сопоставимые результаты на запросах из 3 слов и более. Мы считаем это обнадеживающим результатом, учитывая то, что у нас было не очень много размеченных данных.

С задачей трехклассовой классификации модели справились значительно хуже. Причина – в «серой зоне» пар запросов, находящейся между явной переформулировкой и совсем разными интентами. Изначально мы надеялись, что механизм внимания и функции сопоставления слов в двух запросах позволят объединить синонимы, гипонимы, гиперонимы, классы и его экземпляры в некий отдельный класс. Однако мы не учли все паттерны запросов и спорные моменты, когда сам человек сомневается в отнесении запросов к тому или иному классу.

На основании вышесказанного мы предлагаем несколько направлений улучшения моделей для определения смены интента в поисковых запросах:

1. Обучение векторов слов на более широком корпусе, где устанавливаются зависимости между общими словами. Например, можно добавить к заголовкам объявлений тексты из русскоязычной Википедии. Это нужно для того, чтобы слова «юбка» и «сумка» не казались модели синонимами. Также можно попробовать обучить контекстуальные вектора слов, чтобы модель не ошибалась в случаях многозначности и грамматической омонимии.

2. Продумать классификацию на большее число классов. Можно принять во внимание следующие классы:

- 1) Переформулировка – включает только изменение порядка/добавление/удаление слов, опечатку, исправление кириллицы на латиницу;
- 2) Явный синоним – запросы вида *'dvd – видеомаягнитофон'*, *'гид – путевоаитель'*;
- 3) Спецификация или генерализация запроса – например, в одном запросе есть бренд, во втором – уточнение товара бренда: *'куртка – massimo dutti'*;
- 4) Гипонимы и гиперонимы, отношения «часть – целое». Например, *'ваз – летние шины'*;
- 5) Пары с неопределенным запросом – когда один из запросов является общим: *'отдам даром'*, *'под выкуп'*;
- 6) Горизонтальные синонимы – *'iphone – samsung'*;
- 7) Общая тема: *'изаеия из жести – воаосточная система'*;
- 8) Совсем разный интент.

3. Декомпозиция задачи определения смены интента на 2 части. На первом шаге можно отнести сам запрос к какому-нибудь классу согласно его паттерну, например: запрос с брендом / запрос с брендом и общим словом / запросы с грамматическим согласованием (*'ящики для овощей'*) / локация и т.д. Это не тематический класс, а, скорее, стилистический. На следующем шаге классифицировать пару запросов, отталкиваясь от их классов, с использованием дистрибутивных моделей.

4. В качестве компонента модели можно собрать словарь возможных вершин запросов и определить их синонимы. Затем выносить решение о смене интента, если одна вершина не является синонимом другой. Также можно связать вершины запросов и категории, в которые попадает пользователь по этому запросу, т.е. добавить экстралингвистические признаки.

Поскольку поток запросов на подобных онлайн-площадках большой, цена ошибок очень велика и может обернуться потерей пользователей. Поэтому, помимо усовершенствования самих моделей, можно пересмотреть отношение к задаче как таковой. Если модель не до конца уверена насчет сохранения интента в части запросов, можно не принимать их во внимание при оценке качества поиска.

Заключение

В настоящем исследовании мы описали основные характеристики поискового запроса и интента; познакомили читателя со спецификой данных с онлайн-площадки по поиску и размещению объявлений; подошли с разных сторон к изучению методов автоматического определения смены интента; протестировали на практике несколько передовых методов обработки пар текстовых запросов, реализовав разные виды моделей классификации пар запросов на два и три класса.

По итогам анализа ошибок мы предложили пути по улучшению наших моделей и выполнили поставленную цель - оценить перспективы разных методов обработки естественного языка для задачи определения смены интента, учитывая лингвистическую специфику данных. Предложенные модели с некоторой доработкой можно использовать для оценки качества поиска. На основе того, сохраняется ли интент пользователя, можно построить самые разные поисковые метрики (например, среднее число запросов до смены интента в определенной категории объявлений).

До сих пор в российской академической среде практически не было подобных исследований. Мы надеемся, что наша работа станет хорошим заделом для их развития в будущем.

Литература

Бонч-Осмоловская 2014 — А. А. Бонч-Осмоловская. Кормить свинью online бесплатно: язык запросов как лингвистический объект // Я. Э. Ахапкина, Е. В. Рахилина (ред.). *Современный русский язык в интернете*. М.: Языки славянской культуры, 2014. С. 297-309.

Остин 1999 — Остин Джон. *Избранное* / пер. с англ. Л. Б. Макеевой, В. П. Руднева. М.: Идея-Пресс: Дом интеллектуальной книги, 1999.

Adi et al. 2016 — Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg. *Fine-grained analysis of sentence embeddings using auxiliary prediction tasks*. ICLR, 2016.

Bahdanau et al. 2014 — D. Bahdanau, K. Cho, and Y. Bengio. *Neural machine translation by jointly learning to align and translate* // Proceedings of the International Conference on Learning Representations, 2014.

Bojanowski et al. 2017 — P. Bojanowski, E. Grave, A. Joulin, T. Mikolov. *Enriching Word Vectors with Subword Information*. Transactions of the Association for Computational Linguistics, 2017

Breiman, 2001 — L. Breiman. *Random Forests* // Machine Learning, 45. 2001. P. 5-32.

Broder 2002 — Andrei Z. Broder. *A Taxonomy of Web Search*. Article in ACM SIGIR Forum, September 2002.

Christen 2006 — P. Christen. *A Comparison of Personal Name Matching: Techniques and Practical Issues*. ICDMW '06: Proceedings of the Sixth IEEE International Conference on Data Mining, December 2006. P. 290-294

Church, Hanks 1990 — K.W. Church, P. Hanks. *Word association norms, mutual information, and lexicography*. Computational Linguistics 16(1), 1990. P. 22–29

Devlin et al. 2018 — J. Devlin, M. Chang, K. Lee, K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Preprint. <https://arxiv.org/pdf/1810.04805.pdf>. October, 2018.

Diaz 2016 — Fernando Diaz. *Pseudo-Query Reformulation* // European Conference on Information Retrieval: Advances in Information Retrieval, 2016. P. 521-532.

Jansen et al. 2007 — J. Jansen, M. Zhang, A. Spink. *Patterns and transitions of query reformulation during web searching* // International Journal of Web Information Systems, December 2007.

Jansen, Booth 2010 — B. J. Jansen, D. Booth. *Classifying Web Queries by Topic and User Intent*. // Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Extended Abstracts Volume, Atlanta, Georgia, USA, April 10-15, 2010.

Jimenez et al. 2009 — S. Jimenez, C. Becerra, A. Gelbukh, and F. Gonzalez. *Generalized Mongue-Elkan Method for Approximate Text String Comparison* // Computational Linguistics and Intelligent Text Processing: 10th International Conference, Mexico City, Mexico, March 1-7, 2009. P. 559-570.

Jones, Klinkner 2008 — R. Jones and K. Klinkner. *Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs* // Proceedings of ACM 17th Conference on Information and Knowledge Management (CIKM 2008).

Joshi 2016 — A. Joshi, V. Tripathi, K. Patel, P. Bhattacharyya, M. Carman. *Are Word Embedding-based Features Useful for Sarcasm Detection?* // Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2016. P. 1006–1011

Hashemi et al. 2016 — H. B. Hashemi, A. Asiaee, R. Kraft. Query Intent Detection using Convolutional Neural Networks. WSDM QRUMS 2016 Workshop, 2016.

Hassan 2013 — Ahmed Hassan. *Identifying Web Search Query Reformulation using Concept based Matching* // Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle, Washington, USA, 18-21 October 2013. P 1000–1010.

He, Lin 2016 — H. He, J. Lin. Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. Proceedings of NAACL-HLT 2016, San Diego, California, June 12-17, 2016. P. 937–948

Hernández et al. 2012 — D. I. Hernández, P. Gupta, P. Rosso, M. Rocha. A Simple Model for Classifying Web Queries by User Intent. In: 2nd Spanish Conference on Information Retrieval, CERI 2012. P. 235-240.

Hienert, Kern 2019 — D. Hienert, D. Kern. *Recognizing Topic Change in Search Sessions of Digital Libraries Based on Thesaurus and Classification System*. 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL), June 2019.

Hochreiter, Kepler 1997 — S. Hochreiter, J. Kepler. *Long Short-term Memory* // Neural Computation, December 1997.

LeCun 1989 — Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel. *Backpropagation Applied to Handwritten Zip Code Recognition* // Neural Computation, 1(4):541-551, Winter 1989.

Levenshtein 1965 — V. Levenshtein. *Binary codes capable of correcting spurious insertions and deletions of ones* // Problems of Information Transmission, 1965.

Levenshtein 1966 — V. Levenshtein. *Binary Codes Capable of Correcting Deletions, Insertions and Reversals* // Soviet Physics Doklady, Vol. 10. February 1966. P. 707

Li 2010 — Xiao Li. *Understanding the Semantic Structure of Noun Phrase Queries*. // Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11-16 July 2010. P 1337–1345.

Lucchese et al. 2011 — C. Lucchese, S. Orlando, R. Perego, F. Silvestri, and G. Tolomei. *Identifying task-based sessions in search engine query logs*. // Proceedings of ACM Conference on Web Search and Data Mining (WSDM 2011).

Mikolov et al. 2013 — T. Mikolov, K. Chen, G. Corrado, J. Dean. *Efficient Estimation of Word Representations in Vector Space*. Proceedings of the International Conference on Learning Representations (ICLR), 2013

Mortensen et al. 2018 — D. R. Mortensen, S. Dalmia, P. Littell. *Epitran: Precision G2P for Many Languages*. LREC, 2018

Nogueira, Cho 2017 — Rodrigo Nogueira, Kyunghyun Cho. *Task-Oriented Query Reformulation with Reinforcement Learning* // EMNLP, 2017.

Pennington et al. 2014 — J. Pennington, R. Socher, C. Manning. *Glove: Global Vectors for Word Representation*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, October 2014. P. 1532–1543.

Shen et al. 2018 — D. Shen, G. Wang, W. Wang, M. Renqiang Min, Q. Su, Y. Zhang, C. Li, R. Henao, L. Carin. *Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms*. ACL, May 2018.

Tan et al. 2015 — M. Tan, C. dos Santos, B. Xiang, B. Zhou. *LSTM-based deep learning models for non-factoid answer selection*. arXiv preprint arXiv:1511.04108, 2015.

Vaswani et al., 2017 — A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin. *Attention is all you need* // 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Wang et al. 2016 — Z. Wang, H. Mi, A. Ittycheriah. *Sentence Similarity Learning by Lexical Decomposition and Composition*. Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11-17 December, 2016. P. 1340–1349.

Wang, Chen 2011 — Chieh-Jen Wang, Hsin-Hsi Chen. *Intent Shift Detection Using Search Query Logs*. Computational Linguistics and Chinese Language Processing, Vol. 16, No. 3-4, September/December 2011. P. 61-76

Wang et al. 2017a — Z. Wang, W. Hamza, R. Florian. Bilateral Multi-Perspective Matching for Natural Language Sentences. Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), 2017.

Wang at al. 2017b — S. Wang, J. Jiang. A Compare-Aggregate Model for Matching Text Sequences. ICLR, 2017.

Wieting et al, 2015 — J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. *Towards universal paraphrastic sentence embeddings*. ICLR, 2015.

Приложение. Инструкция для разметки данных на Яндекс.Толоке

Даны пары поисковых запросов.

Запросы могут относиться к разным категориям: Автомобили, Недвижимость, Работа, Услуги, Для бизнеса, Для дома и дачи, Хобби и отдых, Бытовая техника, Личные вещи, Животные.

Встречаются и неопределенные запросы, такие как "отдам даром". Это означает, что пользователь хочет найти объявления, предлагающие вещи безвозмездно. При этом возможно, что человек уже находится в желаемой категории на сайте, вводя данный запрос.

Для каждой пары запросов выберите один вариант из трех

1. Переформулировка/уточнение одного и того же запроса

Примеры:

<i>айфон</i>	<i>айфон 6</i>
<i>nissan sentra</i>	<i>nissan sentra</i>
<i>работа на дому</i>	<i>вакансии</i>
<i>dvd</i>	<i>видеомагнитофон бу</i>
<i>подработка</i>	<i>курьер</i>
<i>гидроманипулятор</i>	<i>урал</i>
<i>котятa</i>	<i>отдам даром</i>

Предмет поиска во втором запросе остался тем же самым, но описывается другими словами / уточняется. Второй запрос не противоречит первому (например, работа курьером может быть подработкой; существует лесовоз Урал с гидроманипулятором).

Также просим относить к данной категории пары с запросом вида "отдам даром".

2. Разные бренды/модели/варианты одного предмета поиска

Примеры:

<i>iphone</i>	<i>android</i>
<i>айфон 7</i>	<i>айфон 8</i>
<i>грант чероки</i>	<i>уаз буханка</i>
<i>корги</i>	<i>лабрадор</i>

Один запрос противоречит другому - например, "айфон 7" не может быть "айфоном 8" - но общий предмет поиска остался тот же - телефон.

Также просьба относить к этой категории пары запросов вида:

<i>ул космопольская</i>	<i>ул ленина</i> (поиск жилья - ищем объявления по нескольким адресам)
<i>охранник</i>	<i>грузчик</i> (поиск работы)

3. Разные предметы поиска

Примеры:

<i>холодильник</i>	<i>диван</i>
--------------------	--------------

<i>airpods</i>	<i>apple watch</i>
<i>чехлы для телефонов</i>	<i>весы</i>
<i>xiaomi roborphone f1</i>	<i>ps4</i>
<i>детские коляски</i>	<i>детские игрушки</i>
<i>курьер</i>	<i>стол</i>

Выбирайте этот вариант, когда человек пытается найти совсем разные по функционалу вещи, разные услуги - то есть, человек уже не ищет то, что указал в первом запросе.

Обратите внимание на пару *детские коляски* - *детские игрушки*: хотя тема запроса одна (что-то для ребенка), предметы поиска разные.

Если значение запроса непонятно, можете кликнуть на него и перейти в поиск. Также можно кликнуть на "Поиск в Google". Могут помочь с ответом картинки в Яндекс или Google по запросу, Википедия.

Заранее спасибо!