Содержание

1. Обзор

Решение задачи по установлению взаимосвязи между твитами и новостными статьями в общем случае представляет собой решение задачи определения семантического сходства между короткими текстами. Методы естественной обработки языка не позволяет с высокой степенью точности определить семантическое сходство между короткими текстами, поэтому установление связей между твитами и новостями должно опираться на дополнительную информацию о предметной области.

1.1. Терминология

Решение задачи предполагает использование наработок различных дисциплин, таких как: обработка естественного языка, машинное обучение, информационный поиск, а основным источником данных выступают интернет ресурсы. Поэтому в работе используется специфичная терминология.

Твиттер — социальная сеть для публичного обмена короткими (до 140 символов) сообщениями при помощи веб-интерфейса, SMS, средств мгновенного обмена сообщениями или сторонних программ-клиентов.

Твит — термин сервиса микроблоггинга Твиттер, обозначающий сообщение, публикуемое пользователем в его твиттере. Особенностью твита является его длина, которая не может быть больше 140 знаков.

Ретвит — сообщение, целиком состоящее из цитирования сообщения одного пользователя Твиттера другим.

Новость — оперативное информационное сообщение, которое представляет политический, социальный или экономический интерес для аудитории в своей свежести, то есть сообщение о событиях произошедших недавно или происходящих в данный момент.

Хэштег — слово или фраза, которым предшествует символ #, используется в различных социальных сетях (Twitter, Facebook, Instagram) для объединения группы сообщений по теме или типу. Например: #искусство, #техника, #смешное, #анекдоты и т.д.

URL (от англ. Uniform Resource Locator — единый указатель ресурса) — единообразный определитель местонахождения ресурса. URL служит стандартизированным способом записи адреса ресурса в сети Интернет.

Обработка естественного языка (англ. Natural language processing) — направление математической лингвистики, которое изучает проблемы компьютерного анализа и синтеза естественных языков.

Именованная сущность — последовательность слов, являющаяся именем, названием, идентификатором, временным, денежным или процентным выражением.

Аннотирование текста — краткое представление содержания текста в виде аннотации (обзорного реферата).

Информационный поиск — процесс поиска неструктурированной документальной информации, удовлетворяющей информационные потребности, и наука об этом поиске.

TF-IDF (от англ. TF — term frequency, IDF — inverse document frequency) — статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов или корпуса. Вес некоторого слова пропорционален количеству употребления этого слова в документе, и обратно пропорционален частоте употребления слова в других документах коллекции.

TF-IDF матрица — матрица, строки которой соответствуют словам из корпуса, а столбцы текстам. Значение ячейки матрицы (i, j) равно значению метрики tf-idf для слова, соответствующего строчке i, и текста, соответствующего столбцу j.

WTMF — метод машинного обучения, применяемый для анализа схожести между короткими текстами [?].

Тематическое моделирование — способ построения модели коллекции текстовых документов, которая определяет, к каким темам относится каждый из документов.

LDA (от англ. Latent Dirichlet allocation — Латентное размещение Дирихле) — методов тематического моделирования, позволяющий объяснять результаты наблюдений с помощью неявных групп.

1.2. Существующие подходы к решению задачи

Задача автоматического установления связей между твитами и новостными статьями до сих пор не имеет устоявшегося решения. В рамках предварительного исследования были отобраны наиболее перспективные подходы к решению задачи, а именно:

- метод WTMT-G, представляющий собой доработку метода WTMF, которая позволила учитывать информацию о связях между текстами;
- обобщённый метод, позволяющий по новости находить относящиеся к ней высказывания из социальных медиа;
- связывание твитов с новостями на основе словарей соответствий;

Также рассматривается классическое решение задачи определения схожести текстов на основе частотности употребления слов. Ниже представлен краткий обзор выбранных методов.

Стоит также ввести несколько определений употребляемых в дальнейшем: под связью *текст-текст* подразумевается определение двух текстов как схожих на основе некоторой дополнительной информации; под связью *текст-слово* подразумевается определение двух текстов как схожих только на основе слов, из которых состоит текст.

1.2.1. Определение схожести текстов на основе частотности употребления слов

Наиболее простым и очевидным подходом к решению задачи связывания твитов с новостными статьями, является связывание текстов, наиболее близких по частотности употребления слов. Способ основан на сравнении значений метрики TF-IDF, поэтому в дальнейшем будем называть этот спосок TF-IDF методом.

Решение задачи связывания твитов с новостными статьями на основе частотности употребления слов можно представить в виде небольшого алгоритма:

1. объединить тексты всех твитов и тексты всех новостей — (для новости текст это конкатенация заголовка и краткого изложения);
2. в качестве корпуса использовать объединение начальных форм всех слов, используемых в текстах, за вычетом списка стоп-слов (под списком стоп-слов подразумевается набор часто употребимых слов языка, которые вне контекста не несут смысловой нагрузки, к примеру, предлоги);
3. по множеству текстов и построенному корпусу построить TF-IDF матрицу;
4. каждому тексту сопоставить столбец TF-IDF матрицы, соответствующий тексту (вектор для сравнения);
5. рассматривая вектор для сравнения в качестве координат в метрическом пространстве, для каждого твита найти список наиболее похожих на него новостей.

В работе в качестве меры близости в метрическом пространстве используется косинусная мера близости — мера численно равная косинусу угла между векторами. В дальнейшем каждый раз, когда говориться о схожести или близости двух векторов, подразумевается близость согласно косинусной мере.

1.2.2. Обобщённый метод, сопоставляющий новостной статье высказывания из социальных медиа

В рамках метода решается следующая задача: по новости необходимо найти высказывания в социальных сетях, которые на неё неявно ссылаются. Метод был предложен в статье Linking Online News and Social Media [?]. Поставленная задача решается в три этапа:

1. по заданной новостной статье формируется несколько моделей запросов, которые создаются как на основе структуры статьи, так и на основе явно связанных со статьей высказываний из социальных медиа.

2. построенные модели используются для получения высказываний из индекса целевого социального медиа, результатом являются несколько ранжированных списков;

3. полученные списки объединяются с использованием особой техники слияния данных.

Авторы также предлагают способ, созданный для борьбы с дрейфом запроса (порождение менее подходящего запроса), который возникает при большом объёме используемого текста. Способ основан на выборе дополнительных отличительных условий.

Для экспериментальной оценки используются данные из различных медиа, таких как Twitter, Digg, Delicious, the New York Times Community, Wikipedia, а так же из блогов.

В результате работы показано, что модели запросов, основанные на различных источниках данных, повышают точность выявления высказываний из социальных медиа; методы слияния ранжированных списков приводят к значительному повышению производительности в сравнении с другими подходами.

1.2.3. Связывание твитов с новостями на основе словарей соответствий

Метод связывания твитов с новостными статья, основанный на словарях соответствий, предложен в статье Bridging Vocabularies to Link Tweets and News [?]. *Словарь соответствий* — множество слов, которые встречаются только в твитах и, соответственно, не встречаются в новостях. Авторы предложили способ автоматического установление связи между множеством твитов и множеством новостей определённой темы. Темы извлекаются из новостей на основе методов тематического моделирования.

Значительную сложность при решении проблемы связывания твитов с новостями вызывают малый размер твита и различия в словарях: в твитах используются аббревиатуры, неформальный язык, сленг; в новостях, напротив, используется литературный язык. В частности, твиты могут вообще не нести смысловой нагрузки.

Твиттер предлагает хэштэги, как механизм для категоризации твитов. Но этот подход обладает рядом недостатков, таких как: не все записи содержат хэштеги, хэштег не содержит информацию о событии, хэштег сформулирован в слишком общей форме, твит содержит несколько хэштегов. Следовательно использование одних хэштегов приведёт к низкому качеству связывания твитов с новостями.

Для решения задачи и преодоления описанных выше проблем, авторами работы предлагается следующий подход:

1. С помощью метода LDA из множества новостей извлекается набор тем. Тема характеризуется распределением частот слов, характерных для этой темы.

2. Каждой полученной теме сопоставляется множество наиболее близких к ней твитов.

3. Из полученных твитов извлекаются слова, которые дополняют характеристику рассматриваемой темы.

4. Полученные слова образуют словарь соответствий и служат «мостом» к другим твитам.

В результате работы продемонстрирован способ установления связей между множеством твитов и множеством новостей с использованием словарей соответствий.

1.2.4. Метод WTMF

Метод WTMF предназначен для определения семантической близости коротких текстов. Этот метод учитывает отсутствующие в тексте слова в виде признаков короткого текста. Под отсутствующими словами подразумеваются все слова из корпуса, составленного из всех текстов, за исключением слов из рассматриваемого короткого текста, то есть отсутствующие слова можно трактовать как негативный сигнал.

Работа метода WTMF основана на разложении TF-IDF матрицы X в произведение двух матриц P и Q:

$$X \sim P^T Q.$$

На рисунке 1 показано как матрица X приближается произведением двух матриц P^T размера $M \times K$ и Q размера $K \times N$.

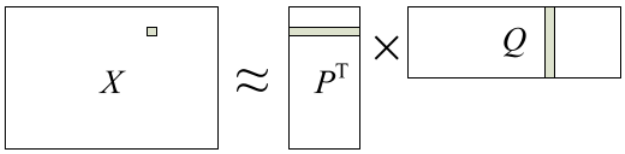

Рисунок 1 — Разложение TF-IDF матрицы (X) на произведение матриц P и Q

Каждый текст s_j представлен в виде вектора $Q_{\cdot,j}$ размерности K, каждое слово w_i представлено в виде вектор $P_{i,\cdot}$. Если $X_{ij} = (P_{i,\cdot}, Q_{\cdot,j})$ близко к нулю, то это трактуется как отсутствующее слово.

Задачей метода является минимизация целевой функции (λ - регуляризирующий член, матрица W определяет вес элементов матрицы X):

$$\sum_i \sum_j W_{ij}(P_{i,\cdot} \cdot Q_{\cdot,j} - X_{ij})^2 + \lambda||P||_2^2 + \lambda||Q||_2^2.$$

Для получения векторов $P_{i,\cdot}$ и $Q_{\cdot,j}$ используется алгоритм описанный в статье [?]. Сначала P и Q инициализируются случайными числами. Затем запускается итеративный пересчёт P и Q по следующим формулам (эффективный способ расчёта описан в [?]):

$$P_{i,\cdot} = (QW_i'Q^T + \lambda I)^{-1}QW_i'X_{i,\cdot}^T,$$

$$Q_{\cdot,j} = (PW_j'P^T + \lambda I)^{-1}PW_j'X_{j,\cdot}.$$

Здесь $W_i' = diag(W_{i,\cdot})$ - диагональная матрица полученная из i-ой строчки матрицы W, аналогично $W_j' = diag(W_{\cdot,j})$ - диагональная матрица полученная из j-ого столбца матрицы W. Матрица W определяется следующим образом:

$$W_{ij} = \begin{cases} 1, & if\ X_{ij} \neq 0, \\ w_m, & otherwise. \end{cases},$$

где w_m положительно и $w_m << 1$.

Столбцы построенной матрицы Q представляют собой вектора для сравнения текстов между собой. Тексту, на основе которого построена i-я строка TF-IDF матрицы X, в соответствие ставится i-й столбей матрицы Q.

В статье предложен подход для поиска семантической близости текстов, который на небольших текстах работает лучше чем подход, основанный на частотности слов.

1.2.5. Метод WTMF-G

Метод WTMG-G решает задачу установления связей между твитами и новостными статьями, путём построения модели, которая учитывает неявные связи между текстами. Метод был предложен в статье Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media [?].

Метод WTMF-G (WTMF on Graphs) представляет собой доработанный метод WTMF, позволяющий хорошо моделировать семантику коротких текстов, но не учитывающий некоторые специфичные для твитов и новостей характеристики, которыми обладает исходная выборка и которые взаимосвязаны с семантической близостью текстов:

1. хештеги, которые являются прямым указанием на смысл твита;
2. именованные сущности, которые с высокой точностью можно извлекать из новостей;
3. информацию о времени публикации твитов и новостей.

Метод WTMF-G расширяет возможности метода WTMF, путём учёта взаимосвязи текстов на основе специфичных для твитов и новостных статей характеристик, то есть позволяет учесть информацию о взаимосвязи текст-текст.

Для решения задачи необходимо иметь эталонный набор данных, на котором будет производится оценка качества полученного решения. Сначала за общий период времени собираются твиты и новости. Для твита помимо текста хранится информация о времени публикации и авторе работы. Для новости хранится время публикации, заколовок, краткое изложение и URL.

На основе собранной информации строится набор данных, который состоит из трёх частей:

1. множество новостей — все собранные новости;
2. множество связей твит-новость, под связью подразумевается явное указание URL новости в тексте твита;
3. множество твитов – все твиты, имеющие связь с одной из собранных новостей.

К построенному набору данных применяется метод WTMF-G — то есть метод WTMF, расширенный путём добавления связей текст-текст. Добавление связей

текст-текст происходит путём модификации регуляризирующего члена *lambda*. Для каждой пары связанных текстов j_1 и j_2:

$$\lambda = \delta \cdot (\frac{Q_{\cdot,j_1} \cdot Q_{\cdot,j_2}}{|Q_{\cdot,j_1}||Q_{\cdot,j_2}|} - 1)^2,$$

коэффициент δ задаёт степень влияния связей текст-текст.

Так как новый регуляризирующий член *lambda* зависит от $|Q_{\cdot,j}|$, который меняется во время итерации, вводим упрощение: длина вектора $Q_{\cdot,j}$ не изменяется во время итерации. Также необходимо модифицировать итеративный процесс построения матриц P и Q следующим образом:

$$P_{i,\cdot} = (QW_i'Q^T + \lambda I)^{-1}QW_i'X_{i,\cdot}^T,$$

$$Q_{\cdot,j} = (PW_j'P^T + \lambda I + \delta L_j^2 Q_{\cdot,n(j)} diag(L_{n(j)}^2) Q_{\cdot,n(j)}^T)^{-1}(PW_j'X_{j,\cdot} + \delta L_j Q_{\cdot,n(j)} L_{n(j)}).$$

В этих формулах $n(j)$ — список связанных текстов с текстом j. $Q_{\cdot,n(j)}$ — матрица, состоящая из связанных векторов для $Q_{\cdot,j}$. L_j - длина вектора Q_j на начало итерации, $L_n(j)$ — вектор длин векторов связанных с j, то есть $Q_{\cdot,n(j)}$, полученный на начало итерации.

В статье показано, что добавление информации о взаимосвязи текст-текст позволяет повысить качество установления связей между твитами и новостными статьями. Качество метода WTMF-G, измеренное с использованием метрики MRR (метрика описана в главе 4.1.1), в сравнении с такими популярными подходами как: TF-IDF, LDA, WTMF, показано в таблице 1.

Таблица 1: Значение метрики MRR для алгоритма WMTF-G в сравнении с другими подходами.

Алгоритм	TF-IDF	LDA	WTMF	WTMF-G
Значение MRR	0.4602	0.1313	0.4531	0.4791

В таблице 1 показано, что алгоритм WTMF-G даёт лучшее качество, чем прочие подходы.

1.3. Выбор подхода для решения задачи

В качестве основного подхода, на основе которого строится решение задачи по установлению связей между твитами и новостями, был выбран WTMF-G. Основной причиной подобного выбора является то, что большинство подходов учитывают

только статистические зависимости вида текст-слово; метод WTMF-G, напротив, не ограничивается зависимостями текст-слово, а позволяет учесть взаимосвязь текст-текст, что, как ожидается, даст прирост качества в решении задачи установления связей.

Также, в рамках работы задача по установления связей между твитами и новостями решена классическим подходом для установления связей между текстами — определение схожести текстов на основе частотности употребления слов. Этот подход даёт хорошие результаты на больших текстах. Результаты этого метода помогут оценить влияние связей вида текст-текст в методе WTMF-G на качество полученного решения.

2. Получение и разметка данных

Для работы с алгоритмами автоматического связывания твитов и новостей был построен набор данных, который состоит из четырёх частей:

1. множество новостей — все собранные новости;
2. множество связей твит-новость, под связью подразумевается пара твит-новость, где в твите говорится о описанной в статье новости;
3. множество твитов – все твиты, имеющие связь с одной из собранных новостей.
4. множество связей текст-текст.

Для построения набор данных необходимо: собрать твиты и новости за длительный промежуток времени; предобработать полученные данные; выявить (разметить) связи твит-новость; построить множество связей текст-текст.

2.1. Получение данных

Получение данных включает в себя не только скачивание информации (твиты и новости), но и выявление корректных источников данных. В рамках получения данных сделано:

1. реализовано скачивание твитов и новостей из разных новостных источников;
2. получены твиты за небольшой промежуток времени;
3. расшифрованы сокращённые URL;
4. определён список наиболее популярных новостных источников в твиттере;
5. в течение длительного времени собраны данные как с твиттера, так и с новостных источников.

2.1.1. Получение твитов

Для получения данных твиттера используется Twitter Streaming API — сервис, предоставляющий разработчикам возможность в реальном времени получить поток данных твиттера. С помощью Twitter Streaming API можно бесплатно получить 1% от всех публичной информации твиттера: публикация и удаления твитов.

Для работы с Twitter Streaming API на сайте https://apps.twitter.com/ зарегистрировано новое приложение и получен набор секретных ключей, которые требуются для авторизации. Для упрощения работы с Twitter Streaming API использована библиотека tweepy, предоставляющая удобный интерфейс на языке Python.

Twitter Streaming API предоставляет данные в формате JSON (от англ. JavaScript Object Notation) — текстовый формат обмена данными, удобный для чтения человек, первоначально создавался как формат на текстового описания и сериализации объектов языка программирования JavaScript.

В рамках работы использована информация о публикации твитов. Каждое событие о создании твита описывается в виде большого количества параметров (несколько десятков), в работе использованы следующие:

1. created_at — дата создания,
2. id — уникальный идентификатор,
3. retweeted_status — существует, только если твит является ретвитом, содержит информацию о ретвитнутом твите,
4. lang — язык,
5. entities — информация о хэштегах и ссылках, которые упоминаются в твите,
6. text — текст твита.

Каждое событие о создании твита обрабатывается. Результатом обработки является структура данных, в которой содержится вся необходимая информация о созданном твите. Обработка происходит следующим образом:

1. Если твит не на русском языке, он отбрасывается.
2. Если твит является ретвитом, то взводится специальный флажок и дальнейшая работа происходит с исходным ретвитнутым твитом. Это делается для того, чтобы получить полноценный текст исходного твита.
3. Из поля entities извлекается информация о хэштегах и ссылках, встречаемых в твите.

Вся полученная информация помещена в хранилище твитов. Хранилище твитов реализованно использованием Python библиотеки shelve.

2.1.2. Получение новостных статей

Получение новостных статей происходит через RSS потоки — специальное API, предоставляемое интернет ресурсами и позволяющее получать информацию в формате RSS. *RSS* — семейство XML-форматов, предназначенных для описания лент новостей, анонсов статей, изменений в блогах и т.п. Формат RSS выбран ввиду его поддержки всеми популярными новостными источниками. Для работы с RSS потоками использована Python библиотека feedparser, позволяющая скачивать и анализировать данные в формате RSS.

RSS поток представляет собой переодически обновляемый список статей. Каждая статья обладает рядом параметров в работе использованы следующие:

1. published — дата создания,
2. summary — краткое изложение новостной статьи,
3. link — URL, который ведёт на описываемую новостную стать.,
4. title — заголовок новостной статьи,

Скачивание RSS потоков происходит следующим образом: переодически получается актуальное состояние всех RSS потоков, из них вычленяются все новые статьи, которые предобрабатываются и добавляются в хранилище новостей. Хранилище новостей реализованно с использованием Python библиотеки shelve.

2.1.3. Расшифровка сокращённых URL

Сокращение URL — это сервис, предоставляемый разными компаниями, заключающийся в создании дополнительного, в общем случае более короткого URL, введущего на искомый адрес. Обычно применяется с целью экономии длины сообщения или для предотвращения непреднамеренно искажения URL. В общем случае механизм сокращений реализовывается путём переадресации короткого URL на искомый.

В твиттере все ссылки автоматически сокращается с помощью сервиса *t.co*. Также многие ссылки добавляются в твиттер уже сокращёнными через сторонние сервисы. Для автоматического выявления связей между твитами и новостями с целью построения тестового набора данных необходимо уметь по сокращённому URL получить исходный.

Расшифровка сокращённых URL — процесс получения по сокращённому URL исходного адреса. На практике часто встречается применение сокращения URL кас-

кадом: сокращение уже сокращённого URL, в таком случае расшифровка заключается в получении исходного URL, который не является сокращённой ссылкой. Можно трактовать задачу расшифровки следующим образом: необходимо получить URL адрес на котором завершится процесс переадресации.

Расматриваемая задача требует обработки большого количества твитов и следовательно большого количества расшифровок сокращённых URL (в главе 2.2 получено, что количество ссылок, требуемых для анализа превышает 10^5). Поэтому возникает требование к повышенной эффективности решения.

В качестве базового решения используется стандартный API языка Python, позволяющий получить содержимое веб-страницы по URL, а следовательно адрес целевой страницы на которую вела сокращённая ссылка. Случаи в которых исходный URL не был получен, будем называть ошибочными. Базовое решение было оптимизировано следующим образом:

1. Работа только с заголовками ответа. Это позволило снизить количество данных пересылаемых по сети. Работа с заголовками требует логики для принятия решения об остановке — то есть выявления искомого URL.

2. Использование многопоточности. Так как большую часть времени код, получающий заголовок страницы ждёт ответа сервера, то асинхронность позволит значительно увеличить быстродействие.

3. Использование «воронки» данных. При увеличении количества потоков стало появляться большее количество ошибок, ввиду того, что загруженность интернет-канала повышает время ответа http-запросов. Для их снижения было выбран подход «воронки» данных с последующей коррекцией ошибок. Данный подход на первом этапе обрабатывает все ссылки в N потоков, на втором этапе все ошибочные ссылки полученные на первом обрабатываются в $\frac{N}{10}$ потоков и так далее, вплоть до 1 потока на итерацию.

2.1.4. Выявление источников новостей

Задача выявления источников новостей требует статистического исследования ссылок, которые встречаются в твитах. Для определения ссылок ведущих на новостные источники из всех URL извлекалось полное доменное имя (в дальнейшем доменное имя). Также стоит отметить, что новостные агрегаторы (к примеру Яндекс-новости, Рамблер-новости) не рассматривались ввиду того, что они агрегируют очень большое количество новостных статей с множества разнородных источников. То есть

информацию с новостных агрегаторов очень сложно собрать и в дальнейшем дорого обрабатывать.

Для грубой оценки использована выборка 1, содержащая 35704 твитов, 13670 ссылок, 12510 уникальных ссылок. Статистика по 20 наиболее часто встречаемым доменным именам в выборке 1 представленна в таблице 2.

Таблица 2: 20 наиболее часто вречаемых доменных имён в выборке 1 (всего 12510 уникальных ссылок)

Доменное имя	**Количество ссылок**	**Процент от общего числа ссылок**	**Новостной источник**
twitter.com	3521	25.76	нет
www.facebook.com	1418	10.37	нет
t.co	405	2.96	нет
www.youtube.com	315	2.30	нет
news.yandex.ru	239	1.75	нет
su.epeak.in	214	1.57	нет
www.instagram.com	198	1.45	нет
www.periscope.tv	191	1.40	нет
l.ask.fm	121	0.89	нет
lifenews.ru	109	0.80	да
ria.ru	108	0.79	да
vk.com	93	0.68	нет
news.7crime.com	82	0.60	нет
lenta.ru	74	0.54	да
russian.rt.com	61	0.45	да
linkis.com	57	0.42	нет
www.gazeta.ru	53	0.39	да
tass.ru	43	0.31	да
www.swarmapp.com	42	0.31	нет
pi2.17bullets.com	36	0.26	нет

Как видно из таблицы 2 популярные новостные агенства составляют лишь малую долю от общего количества используемых ссылок (3.3%). Для получения более точной количественной информации за неделю собрана выборка 2, содержащая 341863 твитов, 134945 ссылок, 115940 уникальных ссылок. Статистика по 20 наиболее часто используемым доменным именам в выборке 2 представленна в таблице 3.

Как видно из таблицы 3 среди твитов, собранных на довольно большом промежутке времени (неделя), популярные новостные источники составляют лишь малую долю от общего числа употребляемых ссылок (3%).

В работе одновременно использовано 5 самых популярных новостных источ-

Таблица 3: 20 наиболее часто вречаемых доменных имён в выборке 2 (всего 115940 уникальных ссылок)

Доменное имя	**Количество ссылок**	**Процент от общего числа ссылок**	**Новостной источник**
twitter.com	36807	31.75	нет
apps.facebook.com	6234	5.38	нет
www.youtube.com	3659	3.16	нет
m.vk.com	2400	2.07	нет
www.periscope.tv	2215	1.91	нет
news.yandex.ru	2041	1.76	нет
www.instagram.com	1798	1.55	нет
su.epeak.in	1624	1.4	нет
www.facebook.com	1406	1.21	нет
lifenews.ru	888	0.77	да
ria.ru	863	0.74	да
l.ask.fm	803	0.69	нет
vk.com	696	0.6	нет
lenta.ru	647	0.56	да
pi2.17bullets.com	577	0.5	нет
news.7crime.com	567	0.49	нет
russian.rt.com	564	0.49	да
www.gazeta.ru	523	0.45	да
linkis.com	485	0.42	нет
ask.fm	430	0.37	нет

ников, а именно: `ria.ru`, `lifenews.ru`, `lenta.ru`, `russian.rt.com`, `www.gazeta.ru`.

2.2. Описание собранных данных

Набор данных сформирован на основе заранее собранной и предподготовленной информации. Для формирования набора данных использовалась информация собранная с 06.04.2016 по 17.04.2016. Было получено множество, основные характеристики которого приведены в таблице 4.

Таблица 4: Сводная характеристика по собранному множеству твитов и новостей за период с 06.04.2016 по 17.04.2016

Метрика	**Значение**
Количество твитов	495552
Количество новостей	13711
Количество твитов, содержащих ссылку	150510
Количество уникальных ссылок, встречаемых в твитах	101017
Количество твитов, содержащих ссылку на новости из рассматриваемых новостных источников	4324
Количество уникальных ссылок на новости из рассматриваемых новостных источников	2979

В дальнейшей под собранными данными будет подразумеваться описанное в таблице 4 множество.

2.3. Разметка наборов данных

Для построения требуемого в работе набора данных необходимо найти множество связей твит-новость. Процесс поиска связей твит-новость называется *разметкой набора данных*. В работе использовано два способа разметки:

1. автоматическая разметка набора данных;
2. ручная разметка набора данных.

В результате разметки получено большое количество пар твит-новость, в которых твит, практически полностью совпадает с заголовком новости. В дальнейшем, для удобства обозначение подобных пар, будем называть связь твит-новость *тривиальной*, если в заголовке статьи встречается менее половины слов из твита.

2.3.1. Автоматическая разметка набора данных

Автоматически построенный набор данных состоит из всех собранных новостей и твитов, которые содержат ссылку на одну из собранных новостей. Автоматически разметив полученную информация, построили множество состоящее из 4324 твитов, 13711 новостей, а также 4324 связей между ними.

Полученный набор данных был проанализирован, с целью выявления количества нетривиальных связей. Для каждого твита был взят его текст, для каждой новости — заголовок. Все полученные тексты поэтапно преобразованы согласно алгоритму, который сопоставляет тексту множество слов и состоит из следующей последовательность действий:

1. текст конвертируется в формат unicode;
2. текст разбивается на токены;
3. полученное множество токенов очищается от токенов, не являющихся словами;
4. из множества удаляются все слова входящие в словарь стопслов;
5. из множества удаляются все дубли.

На основе предподготовленных данных для каждой пары твит, связанная с твитом новость измерялось две метрики: длина пересечения слов твита и слов новости, нормализованная по длине новости; количество слов в твите, которые не встречаются в новости.

На рисунке 2 изображена зависимость количества пар твит-новость от длины пересечения слов твита и слов новости, нормализованной по длине новости. Как видно из рисунка 2 слова в подавляющем большинстве твитов полностью совпадают со словами в соответствующей новости. Среди 4324 пар твит-новость в 3082 парах твит полностью совпадает с заголовком новости. Остаётся 1242 пар, которые не являются просто копией заголовка, среди этих пар нас интересуют те, в которых твит не является обрезанной частью заголовка статьи.

Для выявления количества пар, где твит содержит информацию не содержащуюся в заголовке статьи посмотрим на зависимость количества пар твит-новость от процента уникальных слов в твите, эта зависимость изображена на рисунке 3. Как видно из рисунка 3 количество пар твит-новость, образующих нетривиальную связь, достаточно мало.

На основе исследования зависимости можно получить грубую оценку количества нетривиальных связей. В исследуемом наборе данных таких связей порядка 500-1000, что очень мало и составляет примерно 12-23% от общего числа пар.

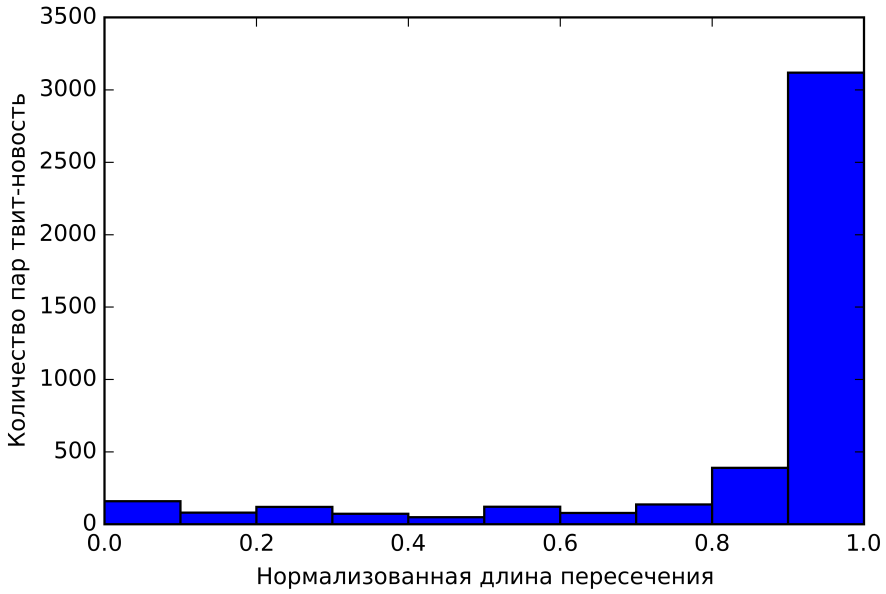

Рисунок 2 — Зависимость количества пар твит-новости от нормализованный длины пересечения множества слов (автоматически размеченный набор данных).

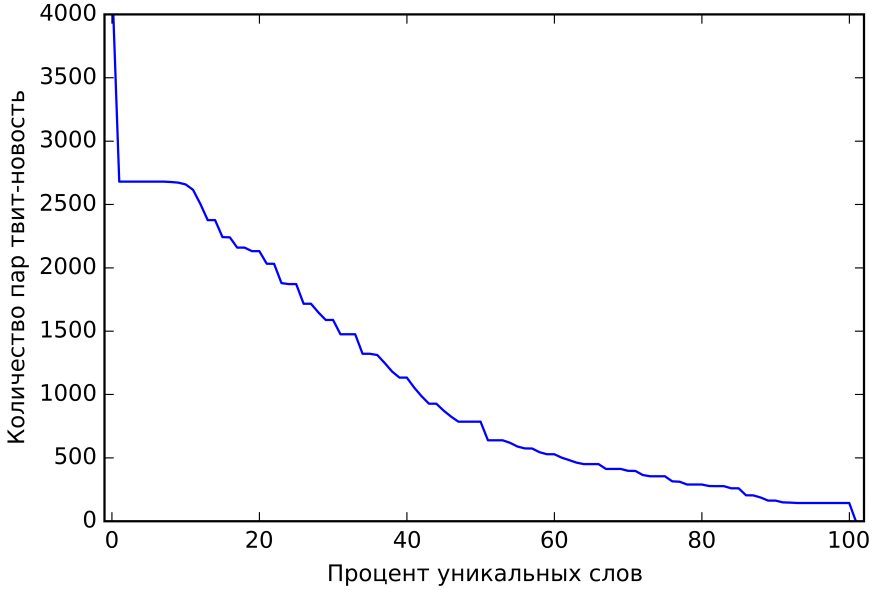

Рисунок 3 — Зависимость количества пар твит-новости от процента уникальных слов в твите (автоматически размеченный набор данных).

2.3.2. Ручная разметка набор данных

Для получения большего количества нетривиальных пар твит-новость была предпринята ручная разметка набора данных. Для ручной разметки были предпод-

готовлены данные. Основные этапы предподготовки данных:

1. на основе множества новостей строится список именованных сущностей L;
2. случайным образом берётся подмножество T множества твитов;
3. из полученного множества T удаляются все твиты, которые удолетворяют следующим правилам:
   а) твит является ретвитом,
   б) твит содержит ссылку на URL с плохим доменным именем (под *плохим доменным именем* подразумевается доменное имя, которое достаточно популярно и не ведёт на новостной источник, в работе использовался следующий список плохих доменных имён: `apps.facebook.com`, `ask.fm`, `twitter.com`, `apps.facebook.com`, `www.instagram.com`, `vk.cc`),
   в) в приведённом к нормальной форме (определение нормальной формы находится в главе**??**) тексте твита содержится менее 2 слов из списка именованных сущностей L;
4. с помощью метода определения схожести текстов на основе частотности употребления слов каждому твиту сопоставляется 10 наиболее схожих с ним новостей.

В качестве результата предподготовки получили множество пар твит-ранжированный список новостей.

Предподготовленные данные были размечены. Разметка заключается в записи специальной отметки рядом с подходящей новости из предложенного списка для каждого твита. Для каждого твита отмечалась только одна, наиболее подходящая новость, или не отмечалось ни одной.

Было сформировано множество из 7373 пар твит-ранжированный список новостей. В нём было выявлено 1600 связей твит-новость. Получаем набор данных, состоящий из 1600 твитов, 13711 новостей, а также 1600 связей между ними.

Для сравнения вручную построенного набора данных с набором данных, полученным автоматически были построены две зависимости — зависимость количества пар твит-новость от длины пересечения слов твита и слов новости и зависимость количества пар твит-новости от процента уникальных слов в твите. На рисунке 4 изображена зависимость количества пар твит-новость от длины пересечения слов твита и слов новости, нормализованная по длине новости. Как видно из рисунка 4

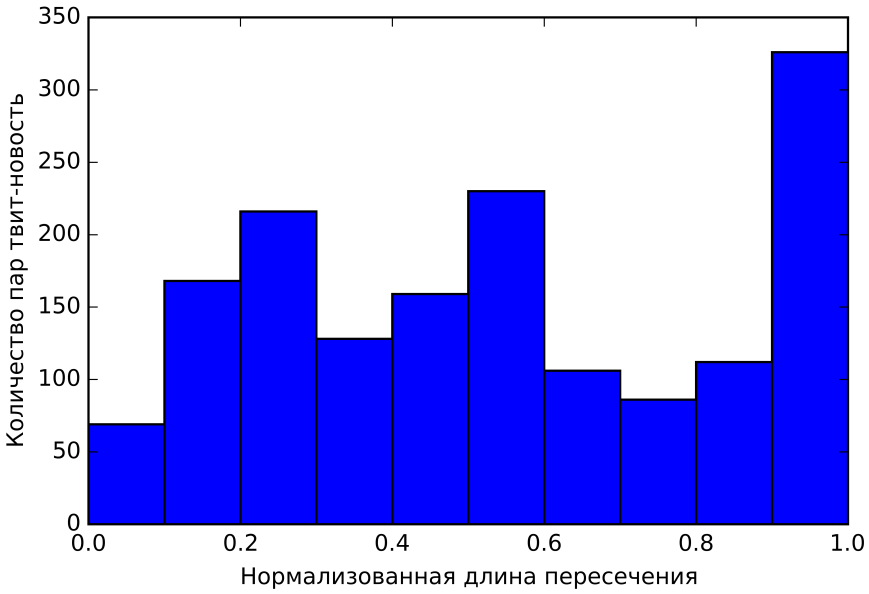

Рисунок 4 — Зависимость количества пар твит-новости от нормализованный длины пересечения множества слов (вручную размеченный набор данных).

было получено распределение намного более близкое к равномерному, чем в случае автоматически размеченного набора данных.

На рисунке 5 изображена зависимость количества пар твит-новость от процента уникальных слов в твите. Как видно из рисунка 5 количество твитов более с чем половиной уникальных слов сравнимо с аналогичным количеством в автоматически размеченном наборе данных, несмотря на то, что автоматически размеченный набор данных почти в три раза больше, чем вручную размеченный набор данных. Количественные значения полученных метрик приведены в таблице 5.

Таблица 5: Сравнение количества твитов

Метрика	**Автоматически размеченный набор данных**	**Вручную размеченный набор данных**
Количество связей	4324	1600
Количество нетривиальных связей	746	976
Процент нетривиальных связей от общего числа связей (%)	17.25	61.00

Как видно из таблицы 5 вручную собранный набор данных намного более качественный, чем автоматический. Но как в ручном, так и в автоматическом наборе

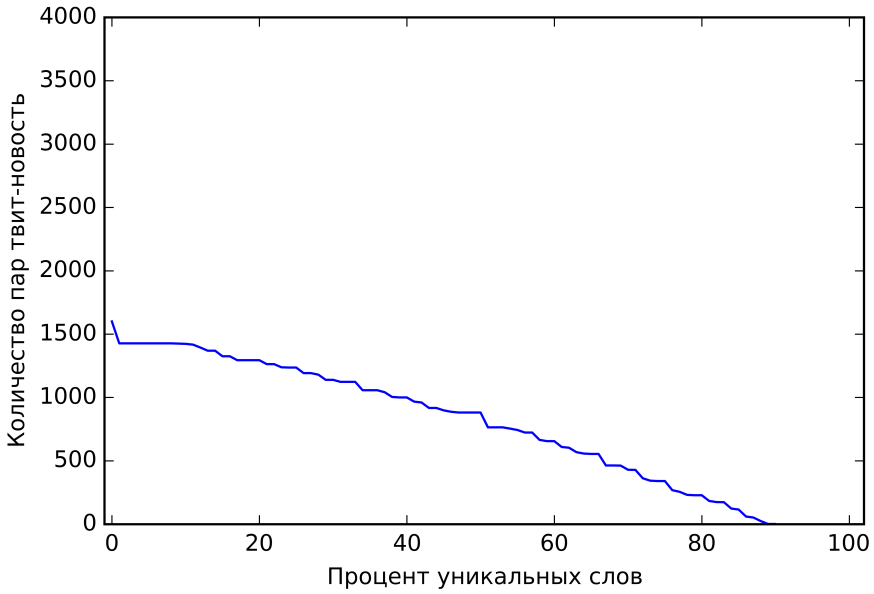

Рисунок 5 — Зависимость количества пар твит-новости от процента уникальных слов в твите (вручную размеченный набор данных).

данных содержится очень мало нетривиальных связей твит-новость (в сравнении с количеством новостей).

2.4. Построение связей текст-текст

Построение связей текст-текст предполагает поиск потенциально семантически близких текстов. При построении связей текст-текст было использовано три способа:

1. построение связей на основе общих хэштегов,
2. построение связей на основе общих именованных сущностей,
3. построение связей на основе близости по времени.

Связь твитов с помощью хэштегов. Из твитов извлекаются все хэшетеги. Затем в хэштеги превращаются все слова во всех твитах, которые совпали с ранее извлечёнными хэштегами. Для каждого твита и для каждого хэштега извлекается k твитов, которые содержат этот хэштег. Если хэштег появлялся в более чем k твитах, то берём k твитов наиболее близких во времени к исходному.

Связь твитов с помощью именованных сущностей. К краткому изложению новостей применяются методы извлечения именованных сущностей. Для каждо-

го твита, содержащего именованную сущность в виде отдельного слова извлекается k твитов, которые содержат эту же именованную сущность. Если именованная сущность содержалась более чем в k твитах, то берём k твитов наиболее близких во времени к исходному.

Связь твитов и новостей на основе близости по времени Для каждого твита (новости) выбираем k связей с наиболее схожими твитами (новостями) в окрестности 24 часов.

Построение связей текст-текст было реализовано для $k = 10$. Для избежания появления большого количества «лишних» записей использовался набор эвристических ограничений:

1. удаление слишком популярных хэштегов (слишком популярным считаем хэштег, который встретился более чем в 10 твитах из обучающей выборки);
2. твиты, считаются связанными когда содержат не менее 2 общих хэштегов или именованных сущностей;
3. связь не устанавливается если тексты слишком похожи (если косинусная мера близости текстов больше 0.99);
4. в случае установления связей на основе времени публикации и схожести текстов, слишком не похожие тексты отбрасываются (слишком не похожие тексты это тексты с мерой близости меньше чем 0.3).

2.5. Сформированные наборы данных

На основе собранной информации было сформировано несколько базовых эталонных наборов, а именно:

1. auto — автоматически размеченный набор данных;
2. manual – вручную размеченный набор данных;
3. total — набор данных состоящий из объединения всех размеченных связей (то есть объединение auto и manual);
4. cutted — набор данных, основанный на наборе total, в котором количество новостей сравнимо с количеством твитов (набор данных создавался для изучения влияния соотношения количества новостей и твитов на качество установления связей).

Также рассматриваются эталонные наборы данных без тривиальных связей, подобный набор образуется путём удаления из базового эталонного набора твитов, создающих тривиальную связь. Обозначим эталонные наборы данных с удалёнными тривиальными связями как auto_nt, manual_nt, total_nt и cutted_nt, полученные путём удаления тривиальных связей из базовых эталонных наборов auto, manual, total и cutted, соответственно.

Ключевая информация характеризующая эталонные наборы представлена в таблице 6.

Таблица 6: Сводная таблица по эталонным наборам данных

Набор данных	**Количество твитов**	**Количество новостей**
manual	1600	13711
auto	4324	13711
total	5798	13711
cutted	5798	6011
manual_nt	976	13711
auto_nt	746	13711
total_nt	1709	13711
cutted_nt	1709	6011

3. Установления взаимосвязей между новостями и твитами

Задача автоматического установления связей между твитами и новостями решена посредством написания программного комплекса, который обладает следующими возможностями:

1. сбор необходимой для решения задачи информации;
2. построение наборов данных;
3. применение к наборам данных методов машинного обучения;
4. получение рекомендаций новостей для произвольных твитов;
5. вариативность в выборе метода для построения рекомендаций;
6. возможность получить информацию о качестве используемого метода.

Программный комплекс реализован с использование языка программирования Python версии 2.7.

Ниже приводится описание архитектуры программного комплекса, а также разбор отдельных моментов.

3.1. Архитектура

Программный комплекс состоит из набора подсистем, которые выполняют следующий набор функций:

1. получение данных из твиттера;
2. получение данных из новостной rss-ленты;
3. расшифровка коротких URL;
4. автоматическое построение набора данных;
5. построение набора данных на основе вручную построенных заготовок;
6. построение моделей для методов WTMF и WTMF-G;
7. построение рекомендаций для методов WTMF, WTMF-G и поиска схожести на основе частнотности употребления слов (TF-IDF);

8. оценка качества рекомендаций;
9. получение результатов рекомендаций в пригодном для чтения формате;

подробное описание архитектуры системы приведённой на flowchart

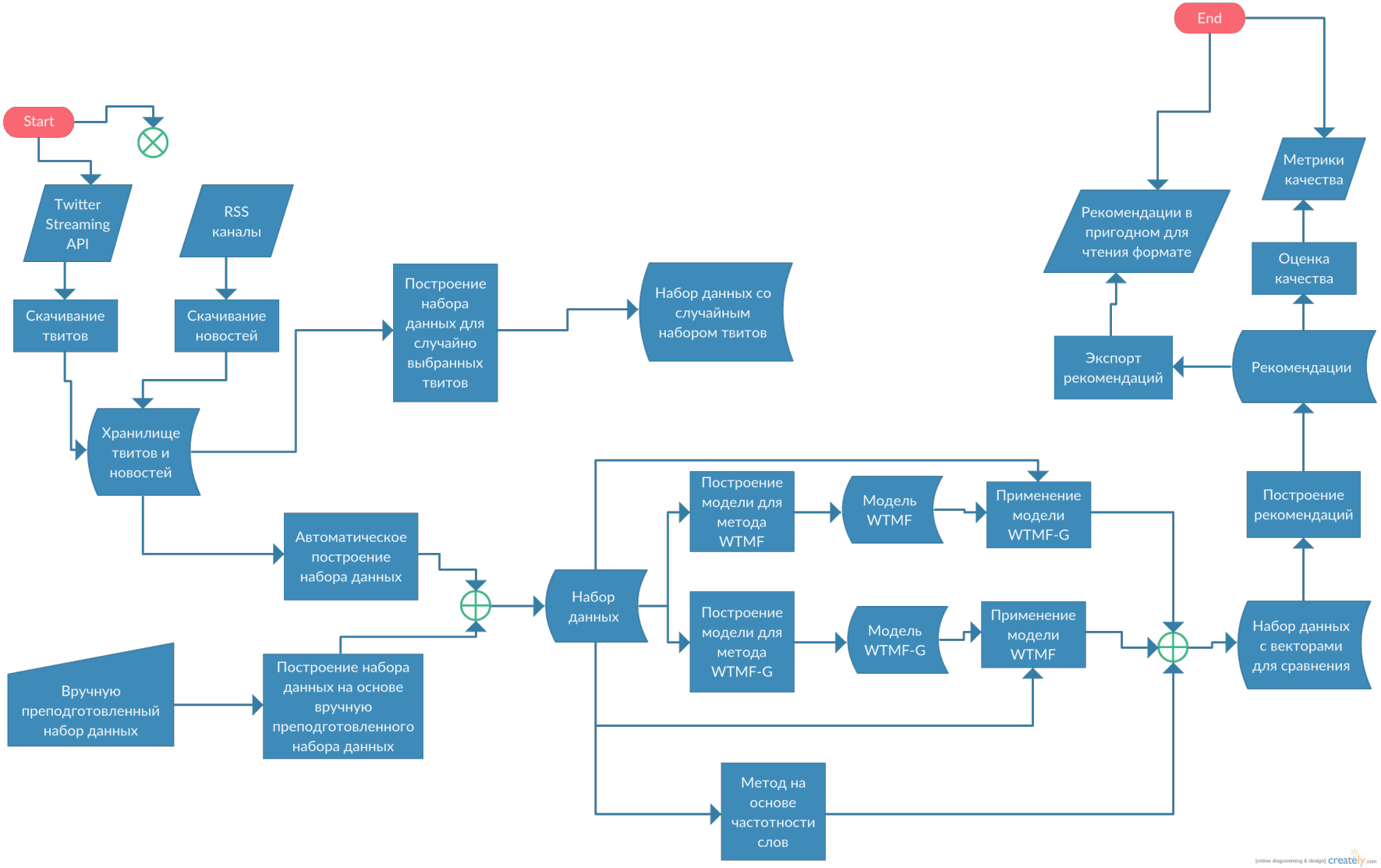

Рисунок 6 — flow chart

3.2. Обработка естественного языка

Работа посвящена поиску семантической близости текстов, поэтому в ней имеет место использование решений таких задач обработки естественного языка, как:

1. токенизация — разбиение предложения на слова;
2. лемматизация — процесс приведения словоформы к лемме;
3. извлечение именованных сущностей.

Описанные выше задачи решены с использованием набора сторонних библиотек для языка Python, а именно:

1. ntlk — платформа, для написания приложений на языке Python, обрабатывающих естественный язык;
2. pymorphy2 — морфологический анализатор;
3. polyglot — библиотека, позволяющая извлекать именованные сущности из текстов на разных языках.

Для решения задачи токенезации используется стандартный токенизатор, реализованный в ntlk. Задача лемматизации решается в случае русского языка с помощью морфологического анализатора pymorphy2, в случае английского языка с помощью морфологического анализатора WordNet, реализованного в ntlk.

Извлечение именованных сущностей происходит с помощью библиотеки polyglot. В используемой библиотеке реализуется выявление именованных сущностей на основе заранее сформированного и размеченного корпуса именованных сущностей. Корпус формируется на основе данных из Википедии.

3.3. Метод WTMF

Модель для метода WTMF построена на основе мнзаранее подготовленного набора данных. В контексте работы набор данных состоит из множества новостей и твитов, из которых в процессе работы извлекается набор текстов (для твита — текст твита, для новости — конкатенация заголовка и краткого изложения статьи).

По множеству текстов, которые получены из набора данных, построена модель, пригодная для сериализации, состоящая из матрицы P (здесь и далее используются обозначения введённые в главе 1.2.4). Построение модели зависит от четырёх констант:

1. K — размерность вектора, по которому производится сравнение (если TF-IDF матрица X была размера $M \times N$, то по завершении работы алгоритма будут получены две матрицы P размера $K \times M$ и Q размера $K \times N$);

2. I — число итераций алгоритма построения модели;

3. w_M — коэффициент, задающий вес негативного сигнала при построении матрицы весов W;

4. λ — регуляризирующий член.

Применение полученной модели на множество твитов представляет собой следующий процесс: сначала строится TF-IDF матрица X для новостей из набора данных и множества твитов, затем на основе новой матрицы X строится весовая матрица W, и наконец на основе построенных матриц X и W и посчитанной на этапе обучения матрицы P выполняется половина итерации алгоритма обучения, а именно получение матрицы Q по матрице P:

$$Q_{\cdot,j} = (PW'_jP^T + \lambda I)^{-1}PW'_jX_{j,\cdot}.$$

В результате получаем вектора для сравнения твитов из заданного множества.

3.4. Метод WTMF-G

Построение модели для метода WTMF-G основывается на построение модели метода WTMF. Набор данных состоит из множества новостей и твитов и связей вида текст-текст, из которых, в процессе работы извлекается набор текстов. (для твита — текст твита, для новости — конкатенация заголовка и краткого изложения статьи).

По множеству текстов, которые получены из набора данных, построена пригодная для сериализации модель, представляющая собой матрицу P. Построение модели зависит от четырёх констант:

1. K — размерность вектора, по которому производится сравнение (если TF-IDF матрица X была размера $M \times N$, то по завершении работы алгоритма будут получены две матрицы P размера $K \times M$ и Q размера $K \times N$);

2. I — число итераций алгоритма построения модели;

3. w_M — коэффициент, задающий вес негативного сигнала при построении матрицы весов W;

4. δ — коэффициент, задающий степень влияния связей вида текст-текст.

Применение полученной модели на множество твитов производится аналогично применению модели для метода WTMF за исключением двух моментов: во-первых, необходимо на основе новостей из набора данных и множества твитов перестроить связи текст-текст, во-вторых получение матрицы Q происходит по следующей формуле:

$$Q_{\cdot,j} = (PW'_jP^T + \lambda I + \delta L_j^2 Q_{\cdot,n(j)} diag(L^2_{n(j)}) Q^T_{\cdot,n(j)})^{-1}(PW'_jX_{j,\cdot} + \delta L_j Q_{\cdot,n(j)} L_{n(j)}).$$

В результате получаем вектора для сравнения твитов из заданного множества.

3.5. Эффективная работа с матрицами

Построение и применение моделей WTMF и WTMF-G требует большого количества операций над матрицами, что на практике занимает продолжительное время. Поэтому актуальна задача по повышению эффективности работы с матрицами.

Для эффективной работы с матрицами используются программные библиотеки для языка Python numpy и scipy (базируется на библиотеке numpy и расширяет её функционал).

Повышение производительности при работе с матрицами производится на примере оптимизации времени расчёта формулы получения строк матрицы P, которая используется при построении моделей WTMF и WTMF-G. На каждой итерации построения модели происходит многократное выполнение формулы (число выполнений порядка 10^4, зависит от размера корпуса):

$$P_{i,\cdot} = (QW'_iQ^T + \lambda I)^{-1} QW'_iX^T_{i,\cdot}.$$

В начале была написана наивная реализация алгоритма, которая показала производительность, не приемлемую в рамках решения задачи. Затем наивная реализация оптимизировалась следующим образом:

1. переход к перемножению матриц с использованием высокопроизводительной библиотеки для языка C OpenBlass (в библиотеке numpy существует возможность перейти к использованию для работы с матрицами некоторых библиотек, написанных на языке C [**?**]);

2. сохранение в отдельной переменной переиспользуемых результатов вычислений над матрицами;

3. переписывание кода для работы с разреженными матрицами;

4. удаление лишних приведений матриц к формату python list и обратно.

Результаты оптимизации приведены в таблице 7.

Таблица 7: Оптимизация работы с матрицами

Добавленная оптимизация	**Время за 100 итераций (с)**	**Прирост производительности (раз)**
Наивная реализация	205	1
Перемножение с помощью OpenBlass	55	3.73
Переиспользование результатов	15.15	3.63
Работа с разреженными матрицами	0.75	20.2
Сокращение количества приведений типов	0.63	1.21

Получили, что оптимизированное решение работает в 325 раз быстрее наивной реализации. Дальнейшая оптимизация не производилась, так как получено решение работающее за приемлемое время.

4. Эксперименты

Главное целью проведения экспериментов является сравнение двух реализованных методов автоматического установления связей между твитами и новостными статьями: метод основанный на частотности употребления слов и WTMF-G. Для исследования влияния на качество добавления информации о взаимосвязях вида текст-текст также производится сравнительное тестирование методов WTMF и WTMF-G.

Ввиду малого числа твитов в наборах данных тестирование производится на тех же выборках, на которых производится обучение.

4.1. Методы оценки качества

Решение задачи установления связей между твитами и новостными статьями неоднозначно. Как твиту может соответствовать несколько новостей, так и новостной статье может соотвествовать несколько твитов. Но в эталонном наборе данных для каждого твита существует связь с единственной новостью. В данном случае для оценки качества применимы метрики принятые в информационном поиске.

Мы рассматривает твит как запрос в терминологии информационного поиска, а список новостей как ответом. То есть для каждого твита мы получаем список новостей, ранжированный по мере убывания их схожести.

4.1.1. Метрика качества MRR

MRR (от англ. Mean reciprocal rank) — статистическая метрика, используемая для измерения качества алгоритмов информационного поиска. Пусть $rank_i$ — позиция первого правильного ответа в i-м запросе, n — общее количество запросов. Тогда значение MRR можно получить по формуле:

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{rank_i}.$$

4.1.2. Метрика качества TOP_I

TOP_I — группа метрик, используемых для оценки качества алгоритмов информационного поиска. Значение метрики TOP_I численно равно проценту запросов с правильным ответом, входящим в первые I ответов. Пусть n — общее количество запросов, $Q_I(i)$ — равно 1, если правильный ответ на i-й запрос входит в первые I предложенных ответов, 0 — в противном случае. Тогда значение TOP_I можно полу-

чить по формуле:

$$TOP_I = \frac{1}{n}\sum_{i=1}^{n} Q_I(i).$$

В дальшейшем будут рассматриваться следующие три метрики из группы метрик TOP_I: TOP_1, TOP_3, TOP_{10}.

4.2. Оптимизация качества WTMF, путём варьирования параметров

Оптимизация параметров модели для метода WTMF будет производится на наборе данных cutted, используя метрику MRR. Модель WTMF зависит от четырёх параметров: K, I, λ, w_m. Параметры K и I влияют на время построения модели, а параметры λ и w_m не влияют на время построения модели.

В качестве начального приближения берутся значения параметров, которое использовали авторы работы [**?**], а именно: $K = 30$, $I = 3$, $\lambda = 20$, $w_m = 0.1$.

Оптимизируются параметры, не влияющие на время работы алгоритма: λ и w_m. Для этого фиксируются остальные параметры: $I = 1$, $K = 30$. Для начала находится оптимальный порядок значений начального приближения. Результаты занесены в таблицу 8.

Таблица 8: Качество работы алгоритма WMTF для различных значений λ и w_m при фиксированных значениях $I = 1$, $K = 30$.

$\lambda \backslash w_m$	**0.001**	**0.01**	**0.1**	**1**	**10**	**100**
0.2	0.6855	0.6877	0.7482	0.3651	0.1526	0.1485
2	0.7000	0.7015	0.7173	0.7525	0.3707	0.1605
20	0.6964	0.7081	0.7149	0.7308	0.7507	0.3784
200	0.7075	0.6991	0.7010	0.7016	0.7146	0.7448
2000	0.6970	0.7070	0.6991	0.7114	0.6994	0.7044

Как видно из таблицы 8 в целом получена достаточно однородная картина для всех порядков λ и w_m. Заметное снижение качества происходит при большом порядке w_m и малом порядке λ. Максимальное значение метрики достигнуто при $\lambda = 2$ и $w_m = 1$. Для уточнения значения коэффициентов, производится исследование качества работы алгоритма в окрестностях максимального значения метрики. Результаты приведены в таблице 9.

Из таблицы 9 получаем оптимальные значения коэффициентов $\lambda = 0.95$ и $w_m = 1.95$.

Таблица 9: Качество работы алгоритма WMTF для различных значений λ и w_m при фиксированных значениях $I = 1$, $K = 30$.

$\lambda \backslash w_m$	**0.9**	**0.95**	**1**	**1.1**	**1.2**
1.9	0.7442	0.7451	0.7536	0.7542	0.7544
1.95	0.7447	0.7554	0.7452	0.7439	0.7504
2	0.7507	0.7528	0.7504	0.7515	0.7566
2.05	0.7413	0.7505	0.7424	0.7525	0.7479
2.1	0.7405	0.7484	0.7485	0.7502	0.7501

Оптимизируются параметры, влияющие на время работы алгоритма: K и I. Для этого фиксируются остальные параметры: $\lambda = 0.95$, $w_m = 1.95$. Для начала находится примерное значение коэффициента K и оптимальное значение I. Результаты занесены в таблицу 10.

Таблица 10: Качество работы алгоритма WMTF для различных значений K и I при фиксированных значениях $\lambda = 0.95$, $w_m = 1.95$.

$K \backslash I$	**1**	**2**	**3**
5	0.1232	0.1593	0.1838
10	0.3521	0.4102	0.4437
30	0.7426	0.7422	0.7158
60	0.8326	0.8117	0.7620

Как видно из таблицы 10 увеличение K приводит к значительному улучшению качества работы алгоритма, увеличении I приводит к улучшению качества алгоритма только при малых значениях параметра K, при больших значениях K увеличение параметра I приводит к ухудшению качества. Максимальное значение метрики достигнуто при $K = 60$ и $I = 1$. Для уточнения значения коэффициента K, производится исследование качества работы алгоритма при фиксированном значении коэффициента I. Результаты приведены в таблице 11.

Из таблицы 11 получаем оптимальные значения коэффициента $K = 90$

В итоге оптимизации качества рекомендаций на основе алгоритма WMTF были получены оптимальные параметры: $K = 90$, $I = 1$, $\lambda = 0.95$, $w_m = 1.95$.

Таблица 11: Качество работы алгоритма WMTF для различных значений K при фиксированных значениях $I = 1$, $\lambda = 0.95$, $w_m = 1.95$.

K	**Значение метрики RR**
10	0.3595
20	0.6460
30	0.7496
40	0.8003
50	0.8220
60	0.8424
70	0.8472
80	0.8535
82	0.8549
84	0.8597
86	0.8592
88	0.8572
90	0.8675
92	0.8580
94	0.8604
96	0.8612
98	0.8644
100	0.8655
110	0.8627

4.3. Оптимизация качества WTMF-G, путём варьирования параметров

Датасет cutted_0.0

Начальное приближение $\delta = 0.1, w_m = 1.95, K = 90, I = 1, \lambda = 0.95$.

Таблица 12: Качество работы алгоритма WMTF-G для различных значений λ и δ при фиксированных значениях $I = 1$, $K = 30$, $w_m = 1.95$.

$\lambda \backslash \delta$	**0.001**	**0.01**	**0.1**	**1**	**10**
0.01	0.3889	0.3842	0.3924	0.3900	0.3895
0.1	0.4895	0.4875	0.4886	0.4850	0.4847
1	0.8227	0.8256	0.8242	0.8225	0.8212
10	0.8477	0.8440	0.8496	0.8454	0.8495
100	0.8294	0.8318	0.8283	0.8240	0.8243

максимум при $\lambda = 10$, $\delta = 0.1$ рассмотрим окрестности.

Таблица 13: Качество работы алгоритма WMTF-G для различных значений λ и δ при фиксированных значениях $I = 1$, $K = 30$, $w_m = 1.95$.

$\lambda \backslash \delta$	**0.06**	**0.08**	**0.1**	**0.12**	**0.14**
6	0.8589	0.8524	0.8511	0.8580	0.8493
8	0.8483	0.8528	0.8539	0.8439	0.8498
10	0.8504	0.8455	0.8416	0.8453	0.8408
12	0.8453	0.8398	0.8472	0.8376	0.8415
14	0.8462	0.8456	0.8387	0.8398	0.8377

рассмотрим максимум при $\lambda = 6$, $\delta = 0.06$

вариьируем w_m

0.01 0.8283 0.05 0.8296 0.1 0.8285 0.5 0.8359 1 0.8442 5 0.8639 10 0.8391 50 0.6094 100 0.5035

1.5 0.8474 2.0 0.8507 2.5 0.8563 3.0 0.8592 3.5 0.8585 4.0 0.8594 4.5 0.8603 5.0 0.8597 5.5 0.8591 6.0 0.8586 6.5 0.8574 7.5 0.8536

берём w_m=5, варьируем K/I

Оптимизация параметров ещё не завершена, существующая и очень, очень грубая оценка приведена ниже

Оптимизация параметров модели для метода WTMF-G будет производится на наборе данных auto_cleared, используя метрику MRR. Модель WTMF зависит от четырёх параметров: K, I, δ, w_m. Параметры K и I влияют на время построения модели, а параметры λ и w_m не влияют на время построения модели.

Таблица 14: Качество работы алгоритма WMTF-G для различных значений K и I при фиксированных значениях $w\_m = 5$, $\lambda = 6$, $\delta = 0.06$.

for delta in [0.06, 0.08, 0.1, 0.12, 0.14]: for lmbd in [6, 8, 10, 12, 14]:

$K \backslash I$	1	2	3	4	5
50	0.8269	0.8349	0.7834	0.6830	0.5801
60	0.8419	0.8450	0.7984	0.7056	0.6006
70	0.8557	0.8466	0.7977	0.7036	0.6002
80	0.8614	0.8511	0.7990	0.7032	0.5957
90	0.8606	0.8522	0.8039	0.7088	0.6038
100	0.8606	0.8527	0.8022	0.7089	0.6021
110	0.8686	0.8553	0.8074	0.7123	0.6065
120	0.8693	0.8579	0.8097	0.7174	0.6085

В качестве начального приближения параметров взяты оптимальные параметры для метода WTMF, а именно $K = 90$, $I = 1$, $w_m = 1.95$. В качестве начального приближения параметра δ вы берем значение 0.1

Оптимизируется параметр δ. Для этого фиксируются остальные параметры: $K = 90$, $I = 1$, $w_m = 1.95$. Для начала находится оптимальный порядок значений начального приближения. Результаты занесены в таблицу **??**. Как видно из таблицы **??**

Таблица 15: Качество работы алгоритма WMTF-G для различных значений δ при фиксированных значениях $K = 90$, $I = 1$, $w_m = 1.95$.

δ	**Значение метрики RR**
0.001	0.5508
0.01	0.5307
0.1	0.5695
1	0.5311
10	0.5303
100	0.5203

максимальное значение метрики получено при $\delta = 0.1$. Для уточнения значения коэффициента δ, производится исследование качества работы алгоритма в окрестностях максимального значения метрики. Результаты приведены в таблице **??**. Из таблицы **??** получаем оптимальные значения коэффициента $\delta = 0.1$.

В итоге оптимизации качества рекомендаций на основе алгоритма WMTF-G были получены оптимальные параметры: $K = 90$, $I = 1$, $\delta = 0.1$, $w_m = 1.95$.

Таблица 16: Качество работы алгоритма WMTF-G для различных значений δ при фиксированных значениях $K = 90$, $I = 1$, $w_m = 1.95$.

δ	**Значение метрики RR**
0.05	0.5340
0.1	0.5695
0.15	0.5380
0.25	0.5533
0.3	0.5195
0.35	0.5329

4.4. Сравнительные результаты

Для выявления влияния добавления связей текст-текст на результаты работы метода WMTF-G производится сравнительное тестирование алгоритма WTMF и WTMF-G. Результаты тестирования приведены в таблице **??**.

Таблица 17: Сравнительное тестирование алгоритмов WTMF и WTMF-G.

Набор данных	**Метрика MRR**		**Метрика** TOP_1		**Метрика** TOP_3	
	WTMF	**WTMF-G**	**WTMF**	**WTMF-G**	**WTMF**	**WTMF-G**
manual	0.7293	0.	0.	0.	0.	0.
auto	0.8640	0.	0.	0.	0.	0.
total	0.8196	0.	0.	0.	0.	0.
cutted	0.8630	0.	0.	0.	0.	0.
manual_nt	0.6194	0.	0.	0.	0.	0.
auto_nt	0.5297	0.5695	0.	0.	0.	0.
total_nt	0.5729	0.	0.	0.	0.	0.
cutted_nt	0.6495	0.	0.	0.	0.	0.

Как видно из таблицы **??** ...

Сравним метод основанный на частотности употребления слов и WTMF-G. Метод основанный на частотности употребления слов обозначим как TF-IDF. Результаты тестирования приведены в таблице **??**.

Как видно из таблицы **??** ...

объяснение влияния различных датасетов, специфики русского языка и сравнение с результатами статьи.

Таблица 18: Сравнительное тестирование алгоритмов TF-IDF и WTMF-G.

Набор данных	**Метрика MRR**		**Метрика** TOP_1		**Метрика** TOP_3	
	TF-IDF	**WTMF-G**	**TF-IDF**	**WTMF-G**	**TF-IDF**	**WTMF-G**
manual	0.8336	0.	0.	0.	0.	0.
auto	0.8817	0.	0.	0.	0.	0.
total	0.8610	0.	0.	0.	0.	0.
cutted	0.9075	0.	0.	0.	0.	0.
manual_nt	0.7565	0.	0.	0.	0.	0.
auto_nt	0.6048	0.5695	0.	0.	0.	0.
total_nt	0.6914	0.	0.	0.	0.	0.
cutted_nt	0.7485	0.	0.	0.	0.	0.