

1. Вопросы

Не лучше ли "поверхностную" постановку задачи расположить во введении, а подробное описание сделать в начале главы реализация?

Перевод bridging vocabulare

**МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ Н. Э. БАУМАНА**
Факультет информатики и систем управления
Кафедра теоретической информатики и компьютерных технологий

Наброски дипломного проекта

«Автоматическое установление связей между сообщениями твиттера и
новостными статьями»

Выполнил:
студент ИУ9-101
Выборнов А. И.
Руководитель:
Лукашевич Н.В.

Москва 2016

!Аннотация

Раздел будет заполнен после завершения записки.

В данной работе рассматривается применение методов машинного обучения к решению задачи ...

Проведен обзор ...

Реализовано ...

В организационно-экономической части представлено технико-экономическое обоснование разработки, проанализирована структура затрат проекта и построен календарный план-график проекта.

Пояснительная записка к работе содержит текст на ... листах формата А4, ... таблиц, ... блок-схем и ... рисунков, список литературы из ... библиографических ссылок. К данной работе разработана презентация в формате *.pdf из ... слайдов.

Содержание

1. Вопросы	1
Введение	7
2. !Обзор	8
2.1. !Терминология	8
2.2. Существующие подходы к решению задачи	8
2.2.1. Поиск похожих текстов на основе частотности употребления слов	9
2.2.2. WTMF-G	9
2.2.3. Обобщённый метод, сопоставляющий новостной статье выска- зывания из социальных медиа	11
2.2.4. Связывание твитов с новостями на основе bridging словарей . .	11
2.3. Подробный разбор выбранного подхода?!	12
2.3.1. Построение датасетов	12
2.3.2. WTMF	12
2.3.3. Построение связей текст-текст	14
2.3.4. WTMF-G	14
2.4. Оценка качества	15
2.4.1. RR	15
2.4.2. TOP10	15
2.4.3. АТОР	15
3. Постановка задачи	16
4. Реализация	17
4.1. Архитектура?! мб Проектирование	17
4.2. Получение данных	17
4.2.1. Консьюмер твиттера	17
4.2.2. Разрешение ссылок	18
4.2.3. Поиск подходящего источника новостей	18
4.2.4. Консьюмер новостей	20
4.2.5. Обработка данных	20
4.3. Формирование датасетов	20
4.3.1. Автоматическое построение датасета	21
4.3.2. Ручная разметка датасета	21
4.3.3. Построение связей текст-текст	21

4.3.4.	Сформированные датасеты	21
4.4.	Реализация WTMF	21
4.4.1.	Модель данных	21
4.4.2.	Обучение	21
4.4.3.	Apply	21
4.4.4.	Оптимизация	21
4.5.	Реализация WTMF-G	22
4.5.1.	Модель данных	22
4.5.2.	Обучение	22
4.5.3.	Apply	22
4.6.	Nature Language Processing	22
4.6.1.	Лемматизация	22
4.6.2.	Извлечение имён собственных	22
5.	Руководство пользователя	24
5.1.	Пакет twnews_consumer	24
5.1.1.	Установка	26
5.1.2.	Использование	26
5.2.	Пакет twnews	26
5.2.1.	Установка	28
5.2.2.	Использование	28
6.	Тестирование	30
6.1.	Оптимизация параметров WTMF	30
6.2.	Оптимизация параметров WTMF-G	30
6.3.	Сравнительные результаты	30
7.	Технико-экономическое обоснование	31
7.1.	Трудоемкость разработки программной продукции	33
7.1.1.	Трудоемкость разработки технического задания	33
7.1.2.	Трудоемкость разработки эскизного проекта	34
7.1.3.	Трудоемкость разработки технического проекта	35
7.1.4.	Трудоемкость разработки рабочего проекта	37
7.1.5.	Трудоемкость выполнения стадии «Внедрение»	39
7.2.	Расчет количества исполнителей	40
7.3.	Ленточный график выполнения работ	41
7.4.	Определение себестоимости программной продукции	42

7.5. Определение стоимости программной продукции	43
7.6. Расчет экономической эффективности	43
7.7. Результаты	44
8. Заключение	46
Список литературы	47

Введение

В современном мире всё больший вес приобретают социальные медиа (преимущественно социальные сети). Их главное отличие от традиционных медиа (газеты, тв) заключается в том, что контент порождается тысячами и миллионами людей. Социальные медиа не заменяют традиционные новостные источники, а дополняют их. Они могут служить полезным социальным датчиком того, насколько популярна история (тема) и как долго. Часто обсуждения в социальных медиа основаны на событиях из новостей и, наоборот, социальные медиа влияют на новостные события.

Одной из самых популярных социальных сетей является Twitter - социальная сеть для публичного обмена сообщениями. Главной особенностью Twitter является малый размер сообщений (140 символов), называемых твитами. Часто твиты являются собой описание, происходящего прямо сейчас события, отклик на него.

...

Выявление связи между сообщениями твиттера (твитов) и новостями позволит как расширить информативность твитов, так и обогатить новости.

...

Преимущества расширения новости с помощью твитов: определение отношения аудитории к новости, дополнительные признаки для тематической классификации новостей, дополнительная информация для аннотирования новостей.

Современные методы обработки естественного языка хорошо работают, используя большой массив текста в качестве входных данных, однако, они становятся неэффективными, когда применяются на коротких текстах, таких как твиты. Существенным преимуществом расширения твита с помощью новости является появляющаяся возможность использования большого количества методов обработки естественного языка (Natural Language Processing).

...

Данная работа ставит целью исследование и разработку методов автоматического установления связей между сообщениями твиттера и новостными статьями.

Не существует стандартных решений. и есть считанное количество статей. На основе этих статей будет сделана попытка построить pipeline для получения подобной взаимосвязи.

2. !Обзор

В двух словах о решаемой задаче.

О том, что задача похожа на поиск семантического сходства между короткими текстами. Она плохо решается.

2.1. !Терминология

Глава будет пополняться терминами в течение написания записки

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.

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.

Tf-idf —

WTMF —

Именованная сущность —

2.2. Существующие подходы к решению задачи

Задача автоматического установления связей между твитами и новостными статьями пока 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. Предложен в статье [1].
- Обобщённый метод, позволяющий по новости находить относящиеся к ней высказывания из социальных медиа. Предложен в статье [2].
- Связывание твитов с новостями на основе bridging словарей (нужен перевод слова bridging, в контексте это примерно значит: связующие словари или перекрывающие). Предложен в статье [3].

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

2.2.1. Поиск похожих текстов на основе частотности употребления слов

описание рекомендаций на основе tfidf

2.2.2. WTMF-G

по статье Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media

В двух-трёх абзацах описание подхода, описание задачи убрать.

Перевод аннотации Многие современные методы обработки естественного языка (NLP¹) хорошо работают с большой массив текста в качестве входных данных. Однако они очень неэффективны при работе с короткими текстами (к примеру твиты). Преодоление этой проблемы мы видим в нахождении соответствующего твиту новостного документа. Решение этой задачи требует хорошего моделирования семантики коротких текстов.

Основной вклад статьи двойной:

1. представлено решение задачи нахождения взаимосвязи между твитами и новостями, из этого могут извлечь выгоду многие NLP задачи;
2. в отличие от предыдущих исследований, которые фокусируются на лексических особенностях коротких текстов (информация о связи текст-слово), мы предлагаем взаимосвязь, основанную на модели скрытой переменной, которая моделирует корреляцию между короткими текстами (информация о связи текст-текст). Необходимость этого обоснована наблюдением: твит обычно покрывает только один аспект события.

Мы покажем, что с помощью особенных признаков твита (хэштегов) и особых признаков новостей (именованных сущностей²) а также временных ограничений, мы можем получить взаимосвязь текст-текст, и, таким образом, дополнить семантическую картину короткого текста. Наши эксперименты показывают значительное преимущество нашей новой модели над baseline³.

¹Natural Language Processing

²Какой-то кривой перевод, найдо найти получше. In data mining, a named entity is a phrase that clearly identifies one item from a set of other items that have similar attributes.

³Как перевести?

Идея статьи Современные методы обработки естественного языка плохо работают с короткими текстами. Для преоболения этого к твитам привязываются соответствующие новости.

Для формирования обучающей выборки, были выбраны твиты, которые имели ссылки на новости, опубликованные новостными агентствами (CNN или NYT) в тот же период.

Как показано в статье [5], добавление к твиту содержимого веб-страницы, ссылка на которую включена в этот твит, повышает **purity score** их кластеризации с 0.280 до 0.392.

Модели со скрытой переменной хорошо подходят для отображения коротких текстов в плотный малоразмерный вектор. В рамках решения задачи была применена модель со скрытой переменной, которая называется WTMF (Weighted Textual Matrix Factorization, подробное описание[6]), к твитам и к новостям. Модель была протестирована на двух схожих наборах данных из небольших сообщений. Как результат - используемая модель с большим запасом превзошла и LSA (Latent Semantic Analysis) и LDA (Latent Dirichelet Allocation). Эта модель позволила добавить информацию об отсутствующих словах в твит (модель WTMF добавляет более 1000 фичей к твиту, LDA лишь 14). Недостатком WTMF является то, что порождается только связь текст-слово, без учёта взаимосвязи между короткими текстами.

Ввиду разреженности исходных данных, возникает ещё одна проблема: твит обычно отражает, только один аспект события.

Полученный подход не учитывает следующих характеристик, которым обладает исходная выборка:

1. Хэштеги, которые являются прямым указанием на смысл твита.
2. **Named entities** новостей. Из новостей можно с высокой точностью извлекать **named entities**, используя инструменты для NER (Named Entity Recognition). Если несколько текстов содержат схожие **named entities** они наверняка описывают одно и то же событие.
3. Информация о времени публикации для твитов и новостей. Если несколько текстов опубликованы примерно в одно и то же время, то велик шанс, что они описывают одно и то же событие

В статье описывается решение проблемы поиска взаимосвязи между текстами, с использованием описанных выше характеристик. Два связанных текста, должны иметь схожий скрытый вектор (семантическая модель твита достраивается из схожих твитов).

Эта дополнительная информация была добавлена в модель WTMF. Было также показано различное влияние на связь текст-текст жанра твита и жанра новости. Был получен на порядок более лучший результат чем при использовании исходной WTMF модель.

2.2.3. 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. Используется трехступенчатый подход:

1. по заданной новостной статье формируется несколько моделей запросов, которые создаются как на основе структуры статьи, так и на основе явно связанных со статьёй 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 запроса¹ при большого 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 также из блогов.

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

2.2.4. Связывание твитов с новостями на основе bridging словарей

Статья Bridging Vocabularies to Link Tweets and News

В двух-трёх абзацах описание подхода, описание задачи убрать.

¹Под дрейфом запроса подразумевается порождение менее подходящего запроса.

Основная идея Значительную сложность при решении проблемы связывания твитов с новостями преимущественно вызывают малый размер твита и различия в словарях: в твитах используются аббревиатуры, неформальный язык, сленг, в новостях, напротив, используется литературный язык. Также твиты очень зашумлены и не содержат полезного содержимого.

Твиттер предлагает хештеги, как механизм для категоризации твитов. Но этот подход далеко не совершенен, так как не только далеко не все записи содержат хештеги, но и записи содержащие хештеги обладают рядом проблем. Такими как: хештег не содержит информацию о событии, хештег сформулирован в слишком общей форме, твит содержит несколько хештегов. Из этого делается вывод, что использование только хештегов приведёт к низкому качеству связывания твитов с новостями.

Предлагается следующий подход: Используется LDA для построения моделей тем поверх новостей. Затем среди твитов ищутся наиболее близкие к конкретному топiku. Из полученных твитов извлекаются слова, которые служат “мостом” к другим твитам.

2.3. Подробный разбор выбранного подхода?!

Надо подумать куда это лучше поместить
Обоснование почему был выбран WTMF-G;
tfidf в качестве baseline

2.3.1. Построение датасетов

За один и тот же промежуток выкачиваем твиты с помощью stream api, новости с помощью rss.

Твит задаётся кортежем: (time, author, text). Новость задаётся кортежем: (time, title, summary, url).

Train/test множества формируем из твитов, которые содержат единственную ссылку на новость, из выкаченных нами ранее, и не совпадают с заголовком новости.

2.3.2. WTMF

WTMF - модель применяемая для анализа схожести между короткими текстами [6]. Модель рассматривает отсутствующие в тексте слова как признаки короткого текста. Отсутствующие слова это все слова корпуса рассматриваемых текстов за исключением слов из рассматриваемого короткого текста. Отсутствующие слова являются негативным сигналом для смысла коротких текстов.

WTMF похож на SVD, но использует не разложение, а непосредственный расчёт каждой ячейки. Модель раскладывает матрицу $X \sim P^T Q$.

Корпус рассматривается как матрица X размера $M \times N$: строки - это слова (всего M), столбцы - короткие тексты (всего N), ячейки - мера tf-idf. Как показано на рисунке 1 матрица X приближается произведением двух матриц P размера $M \times K$ и Q размера $K \times N$.

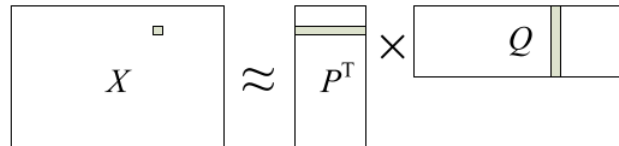


Рисунок 1 — wtmf

Каждый текст s_j представлен в виде вектора $Q_{\cdot,j}$ размерности K , каждое слово w_i представлено в виде вектор $P_{i,\cdot}$. Когда их скалярное произведение X_{ij} близко к нулю, то мы считаем, что это отсутствующее слово.

Задачей модели является минимизация целевой функции (λ - **регуляризующий член**, матрица W определяет вес каждого элемента 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 [8]. Сначала P и Q инициализируются случайными числами. Затем запускается итеративный пересчёт P и Q по следующим формулам (эффективный способ расчёта описан в [7]):

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

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

Здесь $W'_i = \text{diag}(W_{i,\cdot})$ - диагональная матрица полученная из i -ой строчки матрицы W , аналогично $W'_j = \text{diag}(W_{\cdot,j})$ - диагональная матрица полученная из j -ого столбца.

Определим матрицу W следующим образом:

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

где w_m положительно и $w_m \ll 1$.

2.3.3. Построение связей текст-текст

Твиты связываются с помощью хэштегов, named entities и времени.

Связь твитов с помощью хэштегов. Сначала извлекаем все хэштеги из твитов, затем превращаем в хэштеги все слова во всех твитах, которые совпали с ранее извлечёнными хэштегами. Для каждого твита и для каждого хэштега извлекаем k твитов, которые содержат этот хэштег, если хэштег появлялся в более чем k твитах берём k твитов наиболее близких во времени к исходному.

Связь твитов с помощью named entities. Применяем методы NER к новостным summary и получаем множество named entities. Затем применяем тот же подход, что и к хэштегам, сначала превращаем в NE слова из твитов, которые совпали с полученными NE, а затем получаем k связей для каждого твита.

Связь твитов с помощью времени. Аналогично вышеописанному для каждого твита выбираем k связей с наиболее схожими твитами в окрестности 24 часов. Наиболее близкие находятся с помощью косинусной меры, рассчитываемой для векторов из таблицы X .

Новости связываются только по времени.

2.3.4. WTMF-G

Добавление связей текст-текст в WTMF происходит с помощью влияния на regularization term. Для каждой пары связанных текстов j_1 и j_2 :

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

коэффициент δ задаёт степень влияния связей текст-текст.

Полученная модель и называется WTMF-G (WTMG on graphs).

Alternating Least Square используемый в [7] не применим из-за нового regularization term, который зависит от $|Q_{\cdot, j}|$ (по-хорошему нужно понять почему). Для того, чтобы мы могли применить ALS мы вводим упрощение: длина вектора $Q_{\cdot, j}$ не изменяется во время итерации. Получаем уравнения:

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

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

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 i.e. $Q_{\cdot, n(j)}$, полученный на начало

итерации.

2.4. Оценка качества

формулирование того как переходим к рекомендациям и что вообще в них меряем

2.4.1. RR

описание метрики

2.4.2. TOP10

описание метрики

2.4.3. АТОР

описание метрики и обоснование почему она не очень

Используем метрику *АТОР* (метрика подробно описана в [7]). Рассмотрим что означает эта метрика в применении к нашей задаче (я немного модифицировал метрику, для более простого описания, полученная метрика полностью совпадает с описанной метрикой). Пусть T - это множество твитов, $N \in \mathbb{N}$ - размер рассматриваемого топа новостей для твита (могут быть все новости вообще), $k < N \in \mathbb{N}$. $TOPK_t(k) = 1$, если твит $t \in T$ соответствует хотя бы одной новости в top-k результатов, иначе $TOPK_t(k) = 0$

$$TOPK(k) = \frac{\sum_{t \in T} TOPK_t(k)}{|T|},$$

$$АТОР = \frac{\sum_{k=1}^N TOPK_t(k)}{N} = \frac{1}{|T| * N} \sum_{k=\overline{1, N}, t \in T} TOPK_t(k).$$

Значения метрики *АТОР* лежат на отрезке $[0, 1]$. Чем ближе *АТОР* к 1 тем лучше.

3. Постановка задачи

постановка задачи - что нужно было сделать (на полстранички)

4. Реализация

Вступление, где общими словами описываются основные моменты связанные с реализацией

состав работы: 1) собрать данные и разметить их,
2) реализовать и т.п. - здесь же и особенности реализации
получаем рекомендательную систему

4.1. Архитектура?! мб Проектирование

Описание структуры предполагаемой рекомендательной системы

4.2. Получение данных

1. Выбрать источники данных.
2. Выбрать формат хранения.
3. Выбрать БД, для записи результата.
4. Написать отказоустойчивый консьюмер.
5. В течение длительного времени собрать данные.

4.2.1. Консьюмер твиттера

Лог работы в raw формате :) будет много копиясты с используемых мануалов

1. Step 1: Getting Twitter API keys

In order to access Twitter Streaming API, we need to get 4 pieces of information from Twitter: API key, API secret, Access token and Access token secret. Follow the steps below to get all 4 elements:

Create a twitter account if you do not already have one. Go to <https://apps.twitter.com/> and log in with your twitter credentials. Click "Create New App" Fill out the form, agree to the terms, and click "Create your Twitter application" In the next page, click on "API keys" tab, and copy your "API key" and "API secret". Scroll down and click "Create my access token and copy your "Access token" and "Access token secret".

- 2.

4.2.2. Разрешение ссылок

описание проблемы, пути решения, benchmark разных подходов

4.2.3. Поиск подходящего источника новостей

Пока только письмо, потом оформлю

Провёл небольшое исследование, получил достаточно странные результаты.

Суть исследования была в выявлении наиболее популярных новостных источников среди русскоязычных твитов. Был написан код, который умеет находить финальную страничку среди сервисов сокращателей ссылок (в твиттере все ссылки сокращаются). Всего твиттов, в анализируемой выборке: 35704 Ссылок в анализируемой выборке: 13670 Уникальных ссылок: 12510

Топ20 результатов по частоте в формате кортежей: домен, количество ссылок ведущих на домен, процент от общего числа ссылок. В скобках некоторые предположения о том, что конкретная ссылка означает.

twitter.com 3521 25.76%%	(внутренний контент твиттера, в
www.facebook.com 1418 10.37%%	(ссылки на facebook, преимущественно не п
t.co 405 2.96%%	(наверняка это погрешность)
www.youtube.com 315 2.30%%	(видео контент)
news.yandex.ru 239 1.75%%	(популярный в россии новостной агрегатор)
su.epeak.in 214 1.57%%	(популярный на украине новостной ресурс)
www.instagram.com 198 1.45%%	(фото контент)
www.periscope.tv 191 1.40%%	(трансляции)
l.ask.fm 121 0.89%%	(сервис анонимных мнений)
lifenews.ru 109 0.80%%	(новостной ресурс)
ria.ru 108 0.79%%	(новостной ресурс)
vk.com 93 0.68%%	(преимущественно ссылки на п
news.7crime.com 82 0.60%%	(украинские новости)
lenta.ru 74 0.54%%	(новостной ресурс)
russian.rt.com 61 0.45%%	(новостной ресурс)
linkis.com 57 0.42%%	(сервис сокращающий ссылки, т
www.gazeta.ru 53 0.39%%	(новостной ресурс)
tass.ru 43 0.31%%	(новостной ресурс)
www.swarmapp.com 42 0.31%%	(какие-то игрушки на телефон)
pi2.17bullets.com 36 0.26%%	(возможно, погрешность метода)

Итого в топ20 результатов от общего количества ссылок:

Всего ссылок в топ20 : 54%%.

Ссылок на новостные ресурсы: 3.3%% (при это самый популярный только 0.8%%.

Ссылок на новостные агрегаторы: 1.75%%

Ссылок с ошибками в топ20: 2.5%%

Вывод: Популярные новостные агенства составляют слишком малую долю в общем количестве ссылок. Даже суммарный объём крайне мал. При этом хотелось бы сосредоточиться на одном-двух источниках информации, но судя по всему не получится.

Было решено использовать lifenews и ria новости.

На данных за неделю было получено:

341863 134945 115940

TOP20 by 115940 rows

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

total_percent in TOP20: 56.26

Мало
решение

Изменения для консьюмера:

- собирать resolve_url который отдаёт twitter
- собирать больше новостных источников
- собирать английские твиты
- собирать английские новости
- собирать хештеги
- решить проблему с нераспознанным RT

4.2.4. Консьюмер новостей

lifenews - <http://lifenews.ru/xml/feed.xml>

ria - <http://ria.ru/export/rss2/index.xml>

4.2.5. Обработка данных

Для твиттера и новостей

Известные проблемы: * Ретвиты обрезаются.

Берём поля: * retweet - parsed_tweet['retweeted'] or text.startswith('RT @') * text.
* created_at - время по гринвичу. * timestamp_ms - timestamp по местному времени
(в нашем случае +0300). * lang.

Ретвиты скипаем, ввиду описанной выше проблемы. Надо разобраться как
брать для ретвита исходный твит.

Формируем tsv по lang: timestamp_ms, text

Пример строения типичного сообщения твиттера, получаемого с помощью
Twitter Stream API.

4.3. Формирование датасетов

в двух словах что мы подразумеваем под датасетом

4.3.1. Автоматическое построение датасета

так-то собрали
мало данных
аналитика
не очень хороший результат

4.3.2. Ручная разметка датасета

так-то собрали
аналитика
лучше чем автоматический

4.3.3. Построение связей текст-текст

всё о построение этих связей

4.3.4. Сформированные датасеты

список из всех сформированных датасетов с описанием каждого из них

4.4. Реализация WTMF

4.4.1. Модель данных

какие параметры
от чего зависит построение модели
какой результат

4.4.2. Обучение

4.4.3. Apply

4.4.4. Оптимизация

Как установить numpy и scipy <https://hunseblog.wordpress.com/2014/09/15/installing-numpy-and-openblas/>

Ускорение формулы 4 из статьи WTMF, примерное время за 100 итераций: В одну итерацию входит перемножение нескольких больших матриц и взятие обратной матрицы.

что сделали	текущее время	во сколько раз стало быстрее
наивная реализация	205с	1
перемножение с помощью OpenBlass	55с	3.73
вынесение общих множителей	15.15с	3.63
переход к работе с sparse матрицами	0.75с	20.2
удаление ненужного приведения матрицы к python list	0.63с	1.21

итого пришли к коду, который работает в 325 раз быстрее наивной реализации
идеи по до оптимизации: CUDA почему быстрее на ноуте? на котором не работает распараллеливание?

4.5. Реализация WTMF-G

4.5.1. Модель данных

какие параметры
от чего зависит построение модели
какой результат

4.5.2. Обучение

4.5.3. Apply

почему сложно написать apply

4.6. Nature Language Processing

Всё что связано с обработкой текста

4.6.1. Лемматизация

что это
как используется
<https://pythonprogramming.net/stop-words-nltk-tutorial/?completed=/tokenizing-words-sentences-nltk-tutorial/>

4.6.2. Извлечение имён собственных

что это
Обзор подходов

объяснение используемого

5. Руководство пользователя

Проект состоит из двух пакетов, каждый из которых устанавливается и используется в отдельности.

- `twnews_consumer` — консьюмер, который позволяет выкачивать твиты с твиттера и новости с rss каналов.
- `twnews` — пакет, позволяющий по твитам и новостям, произвести все необходимые преобразования данных и на основе полученных признаков произвести обучение и оценку модели.

Оба пакета ориентированы на работу в операционных системах из семейства linux. Для начала работы необходимо иметь установленный менеджер пакетов для языка Python — `pip`, а также установить `setuptools`:

```
$ pip install setuptools
```

Также необходимо выкачать git-репозиторий: <https://github.com/art-vybor/twnews.git>. Если установлен пакет `git`, то это можно сделать следующим образом:

```
$ git clone https://github.com/art-vybor/twnews.git
```

Для корректной работы пакетов, все указываемые в конфигурации директории должны быть заранее созданы.

5.1. Пакет `twnews_consumer`

Пакет `twnews_consumer` располагается в папке `consumer` в корне репозитория. Он позволяет выкачивать и сохранять в формате, удобном для дальнейшей работы пакета `twnews`, твиты и новости.

Конфигурирование пакета производится в файле `twnews_consumer/defaults.py`. Описание задаваемых параметров находится в таблице 1.

Результатом работы пакета является множество новостей и твитов, выкаченных за время работы программы. Для новостей сохраняются заголовок, краткое описание, ссылка на новость, время публикации и имя ресурса, на котором новость была опубликована. Для твитов сохраняются текст, время публикации и информация о том, является ли он ретвитом (ретвит — твит, представляющий собой, ссылку на ранее созданный твит).

Таблица 1: Описание конфигурации пакета twnews_consumer

Имя параметра	Пример значения	Описание
LOG_FILE	'/var/log/twnews_consumer.log'	Путь до файла с логом
LOG_LEVEL	logging.INFO	Уровень подробности лога
TWNEWS_DATA_PATH	'/home/avybornov/twnews_data/'	Путь до директории, в которую будут сохранены данные
RSS_FEEDS	{'ria': {'rss_url': 'http://ria.ru/export/rss2/index.xml'}, 'lifenews': {'rss_url': 'http://lifenews.ru/xml/feed.xml'}}}	Новостные источники, которые требуется выкачать
TWEETS_LANGUAGES	['ru']	Список языков, твиты с использованием которых выкачиваются из твиттера

5.1.1. Установка

Для установки, необходимо зайти в папку `consumer`, находящуюся в корне репозитория и выполнить команду:

```
$ make install
```

Во время установки, нужно будет ввести пароль, для распаковки секретного ключа, который необходим для работы с API твиттера.

5.1.2. Использование

Для того, чтобы начать выкачивать новости, необходимо запустить команду:

```
$ twnews_consumer download --news
```

Для того, чтобы начать выкачивать сообщения твиттера, необходимо запустить команду:

```
$ twnews_consumer download --tweets
```

Узнать информацию о работе программы можно из файла лога. Пример:

```
$ tail -f /var/log/twnews_consumer.log
2016-04-05 11:37:14: RSS> Start consume rss feeds
2016-04-05 11:37:17: TWITTER> Starting write to /mnt/yandex.disk/
twnews_data/logs/tweets.shelve
2016-04-05 11:37:17: TWITTER> Starting to consume twitter
2016-04-05 12:33:32: TWITTER> ('Connection broken: IncompleteRead
(0 bytes read, 512 more expected)', IncompleteRead(0 bytes
read, 512 more expected))
```

5.2. Пакет twnews

Пакет `twnews` располагается в папке `core` в корне репозитория. Он позволяет обрабатывать данные полученные с помощью консьюмера с целью построения и оценки качества модели WTMF.

Конфигурирование пакета производится в файле `twnews/defaults.py`. Описание задаваемых параметров находится в таблице 2.

Результатом работы пакета является построенная модель WTMF, для которой измерено её качество. Перед построением модели, необходимо выполнить команду, которая разрешает ссылки (может занять длительное время).

Таблица 2: Описание конфигурации пакета twnews

Имя параметра	Пример значения	Описание
LOG_FILE	'/var/log/twnews.log'	Путь до файла с логом
LOG_LEVEL	logging.INFO	Уровень подробности лога
TWNEWS_DATA_PATH	'/home/avybornov/twnews_data/'	Путь до рабочей директории в которой лежат выкаченные с помощью консьюмера данные
DATASET_FRACTION	1.0	Часть датасета, которая будет использована для обучения модели
TMP_FILE_DIRECTORY	'/tmp/twnews/'	Путь до директории в которую будут сохранены временные данные

5.2.1. Установка

Для установки, необходимо зайти в папку `core`, находящуюся в корне репозитория и выполнить команду:

```
$ make install
```

Для повышения производительности рекомендуется вручную собрать пакет `numpy` с использованием математической библиотеки `OpenBLAS` [9].

5.2.2. Использование

Для того, чтобы разрешить ссылки, необходимо запустить команду:

```
$ twnews_consumer --resolve
```

Для того, чтобы посмотреть статистику по упомянутым в коллекции твитов ссылкам нужно выполнить команду:

```
$ twnews_consumer download --analyse_urls
```

Для построения модели необходимо запустить команду:

```
$ twnews_consumer download --run_pipe
```

Узнать информацию о работе программы можно из файла лога. Пример:

```
$ tail -f /var/log/twnews.log
INFO:root:2016-04-08 10:20:08.256411: News successfully loaded
INFO:root:2016-04-08 10:20:23.006948: Function iteration started
    with time measure
INFO:root:2016-04-08 10:33:32.930520: Function iteration finished
    in 13m9.9234058857s
INFO:root:2016-04-08 10:33:32.940666: Function
    find_topk_sim_news_to_tweets started with time measure
INFO:root:2016-04-08 10:34:42.360326: Function
    find_topk_sim_news_to_tweets finished in 1m9.41950583458s
INFO:root:2016-04-08 10:34:42.587674: Function iteration started
    with time measure
INFO:root:2016-04-08 10:48:04.453983: Function iteration finished
    in 13m21.8661620617s
INFO:root:2016-04-08 10:48:04.466846: Function
    find_topk_sim_news_to_tweets started with time measure
INFO:root:2016-04-08 10:49:18.958096: Function
    find_topk_sim_news_to_tweets finished in 1m14.4910538197s
```

INFO:root:2016-04-08 10:49:19.171160: Function iteration started
with time measure

6. Тестирование

возможно стоит куда-нибудь вставить время работы
из-за малого количества данных все тесты на dev

6.1. Оптимизация параметров WTMF

как получили оптимальные параметры

6.2. Оптимизация параметров WTMF-G

как получили оптимальные параметры

6.3. Сравнительные результаты

несколько таблиц обобщающих все датасеты и все алгоритмы
для каждой таблицы объяснение полученных результатов

7. Техничко-экономическое обоснование

Разработка программного обеспечения — достаточно трудоемкий и длительный процесс, требующий выполнения большого числа разнообразных операций. Организация и планирование процесса разработки программного продукта или программного комплекса при традиционном методе планирования предусматривает выполнение следующих работ:

- формирование состава выполняемых работ и группировка их по стадиям разработки;
- расчет трудоемкости выполнения работ;
- установление профессионального состава и расчет количества исполнителей;
- определение продолжительности выполнения отдельных этапов разработки;
- построение календарного графика выполнения разработки;
- контроль выполнения календарного графика.

Далее приведен перечень и состав работ при разработке программного средства для автоматического установления связей между сообщениями твиттера и новостными статьями. Отметим, что процесс разработки программного продукта характеризуется совместной работой разработчиков постановки задач и разработчиков программного обеспечения.

Укрупненный состав работ по стадиям разработки программного продукта:

1. Техническое задание:

- Постановка задач, выбор критериев эффективности,
- Разработка технико-экономического обоснования разработки,
- Определение состава пакета прикладных программ, состава и структуры информационной базы,
- Выбор языков программирования,
- Предварительный выбор методов выполнения работы,
- Разработка календарного плана выполнения работ;

2. Эскизный проект:

- Предварительная разработка структуры входных и выходных данных,

- Разработка общего описания алгоритмов реализации решения задач,
- Разработка пояснительной записки,
- Консультации разработчиков постановки задач,
- Согласование и утверждение эскизного проекта;

3. Технический проект:

- Разработка алгоритмов решения задач,
- Разработка пояснительной записки,
- Согласование и утверждение технического проекта,
- Разработка структуры программы,
- Разработка программной документации и передача ее для включения в технический проект,
- Уточнение структуры, анализ и определение формы представления входных и выходных данных,
- Выбор конфигурации технических средств;

4. Рабочий проект:

- Комплексная отладка задач и сдача в опытную эксплуатацию,
- Разработка проектной документации,
- Программирование и отладка программ,
- Описание контрольного примера,
- Разработка программной документации,
- Разработка, согласование программы и методики испытаний,
- Предварительное проведение всех видов испытаний;

5. Внедрение:

- Подготовка и передача программной документации для сопровождения с оформлением соответствующего Акта,
- Передача программной продукции в фонд алгоритмов и программ,
- Проверка алгоритмов и программ решения задач, корректировка документации после опытной эксплуатации программного продукта;

Трудоемкость разработки программной продукции зависит от ряда факторов, основными из которых являются следующие: степень новизны разрабатываемого программного комплекса, сложность алгоритма его функционирования, объем используемой информации, вид ее представления и способ обработки, а также уровень используемого алгоритмического языка программирования. Чем выше уровень языка, тем трудоемкость меньше.

По степени новизны разрабатываемый проект относится к *группе новизны А* – разработка программных комплексов, требующих использования принципиально новых методов их создания, проведения НИР и т.п.

По степени сложности алгоритма функционирования проект относится к *2 группе сложности* - программная продукция, реализующая учетно-статистические алгоритмы.

По виду представления исходной информации и способа ее контроля программный продукт относится к *группе 12* - исходная информация представлена в форме документов, имеющих различный формат и структуру и *группе 22* - требуется печать документов одинаковой формы и содержания, вывод массивов данных на машинные носители.

7.1. Трудоемкость разработки программной продукции

Трудоемкость разработки программной продукции (τ_{PP}) может быть определена как сумма величин трудоемкости выполнения отдельных стадий разработки программного продукта из выражения:

$$\tau_{PP} = \tau_{TZ} + \tau_{EP} + \tau_{TP} + \tau_{RP} + \tau_V,$$

где τ_{TZ} — трудоемкость разработки технического задания на создание программного продукта; τ_{EP} — трудоемкость разработки эскизного проекта программного продукта; τ_{TP} — трудоемкость разработки технического проекта программного продукта; τ_{RP} — трудоемкость разработки рабочего проекта программного продукта; τ_V — трудоемкость внедрения разработанного программного продукта.

7.1.1. Трудоемкость разработки технического задания

Расчёт трудоёмкости разработки технического задания (τ_{PP}) [чел.-дни] производится по формуле:

$$\tau_{TZ} = T_{RZ}^Z + T_{RP}^Z,$$

где T_{RZ}^Z — затраты времени разработчика постановки задачи на разработку ТЗ, [чел.-дни]; T_{RP}^Z — затраты времени разработчика программного обеспечения на разработку ТЗ, [чел.-дни]. Их значения рассчитываются по формулам:

$$T_{RZ}^Z = t_Z * K_{RZ}^Z,$$

$$T_{RP}^Z = t_Z * K_{RP}^Z,$$

где t_Z — норма времени на разработку ТЗ на программный продукт (зависит от функционального назначения и степени новизны разрабатываемого программного продукта), [чел.-дни]. В нашем случае по таблице получаем значение (группа новизны — А, функциональное назначение — технико-экономическое планирование):

$$t_Z = 79.$$

K_{RZ}^Z — коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки задачи на стадии ТЗ. В нашем случае (совместная разработка с разработчиком ПО):

$$K_{RZ}^Z = 0.65.$$

K_{RP}^Z — коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком программного обеспечения на стадии ТЗ. В нашем случае (совместная разработка с разработчиком постановки задач):

$$K_{RP}^Z = 0.35.$$

Тогда:

$$\tau_{TZ} = 79 * (0.35 + 0.65) = 79.$$

7.1.2. Трудоемкость разработки эскизного проекта

Расчёт трудоёмкости разработки эскизного проекта (τ_{EP}) [чел.-дни] производится по формуле:

$$\tau_{EP} = T_{RZ}^E + T_{RP}^E,$$

где T_{RZ}^E — затраты времени разработчика постановки задачи на разработку эскизного проекта (ЭП), [чел.-дни]; T_{RP}^E — затраты времени разработчика программного обеспечения на разработку ЭП, [чел.-дни]. Их значения рассчитываются по форму-

лам:

$$T_{RZ}^E = t_E * K_{RZ}^E,$$

$$T_{RP}^E = t_E * K_{RP}^E,$$

где t_E – норма времени на разработку ЭП на программный продукт (зависит от функционального назначения и степени новизны разрабатываемого программного продукта), [чел.-дни]. В нашем случае по таблице получаем значение (группа новизны – А, функциональное назначение – технико-экономическое планирование):

$$t_E = 175.$$

K_{RZ}^E – коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки задачи на стадии ЭП. В нашем случае (совместная разработка с разработчиком ПО):

$$K_{RZ}^E = 0.7.$$

K_{RP}^E – коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком программного обеспечения на стадии ТЗ. В нашем случае (совместная разработка с разработчиком постановки задач):

$$K_{RP}^E = 0.3.$$

Тогда:

$$\tau_{EP} = 175 * (0.3 + 0.7) = 175.$$

7.1.3. Трудоемкость разработки технического проекта

Трудоёмкость разработки технического проекта (τ_{TP}) [чел.-дни] зависит от функционального назначения программного продукта, количества разновидностей форм входной и выходной информации и определяется по формуле:

$$\tau_{TP} = (t_{RZ}^T + t_{RP}^T) * K_V * K_R,$$

где t_{RZ}^T – норма времени, затрачиваемого на разработку технического проекта (ТП) разработчиком постановки задач, [чел.-дни]; t_{RP}^T – норма времени, затрачиваемого на разработку ТП разработчиком ПО, [чел.-дни]. По таблице принимаем (функциональное назначение — технико-экономическое планирование, количество разновид-

ностей форм входной информации — 2 (твиты, новости), количество разновидностей форм выходной информации — 2 (набор связей твит-новости, оценка работы рекомендательной системы)):

$$t_{RZ}^T = 52,$$

$$t_{RP}^T = 14.$$

K_R — коэффициент учета режима обработки информации. По таблице принимаем (группа новизны — А, режим обработки информации — реальный масштаб времени):

$$K_R = 1.67.$$

K_V — коэффициент учета вида используемой информации, определяется по формуле:

$$K_V = \frac{K_P * n_P + K_{NS} * n_{NS} + K_B * n_B}{n_P + n_{NS} + n_B},$$

где K_P — коэффициент учета вида используемой информации для переменной информации; K_{NS} — коэффициент учета вида используемой информации для нормативно-справочной информации; K_B — коэффициент учета вида используемой информации для баз данных; n_P — количество наборов данных переменной информации; n_{NS} — количество наборов данных нормативно-справочной информации; n_B — количество баз данных. Коэффициенты находим по таблице (группа новизны - А):

$$K_P = 1.70,$$

$$K_{NS} = 1.45,$$

$$K_B = 4.37.$$

Количество наборов данных, используемых в рамках задачи:

$$n_P = 3,$$

$$n_{NS} = 0,$$

$$n_B = 1.$$

Находим значение K_V :

$$K_V = \frac{1.70 * 3 + 1.45 * 0 + 4.37 * 1}{3 + 0 + 1} = 2.3675.$$

Тогда:

$$\tau_{TP} = (52 + 14) * 2.3675 * 1.67 = 261.$$

7.1.4. Трудоемкость разработки рабочего проекта

Трудоёмкость разработки рабочего проекта (τ_{RP}) [чел.-дни] зависит от функционального назначения программного продукта, количества разновидностей форм входной и выходной информации, сложности алгоритма функционирования, сложности контроля информации, степени использования готовых программных модулей, уровня алгоритмического языка программирования и определяется по формуле:

$$\tau_{RP} = (t_{RZ}^R + t_{RP}^R) * K_K * K_R * K_Y * K_Z * K_{IA},$$

где t_{RZ}^R — норма времени, затраченного на разработку рабочего проекта на алгоритмическом языке высокого уровня разработчиком постановки задач, [чел.-дни]. t_{RP}^R — норма времени, затраченного на разработку рабочего проекта на алгоритмическом языке высокого уровня разработчиком ПО, [чел.-дни]. По таблице принимаем (функциональное назначение — технико-экономическое планирование, количество разновидностей форм входной информации — 2 (твиты, новости), количество разновидностей форм выходной информации — 2 (набор связей твит-новости, оценка работы рекомендательной системы)):

$$t_{RZ}^R = 15,$$

$$t_{RP}^R = 91.$$

K_K — коэффициент учета сложности контроля информации. По таблице принимаем (степень сложности контроля входной информации — 12, степень сложности контроля выходной информации — 22):

$$K_K = 1.00.$$

K_R — коэффициент учета режима обработки информации. По таблице принимаем (группа новизны — А, режим обработки информации — реальный масштаб времени):

$$K_R = 1.75.$$

K_Y — коэффициент учета уровня используемого алгоритмического языка программирования. По таблице принимаем значение (интерпретаторы, языковые описатели):

$$K_Y = 0.8.$$

K_Z — коэффициент учета степени использования готовых программных модулей. По таблице принимаем (использование готовых программных модулей составляет около 30

$$K_Z = 0.7.$$

K_{IA} — коэффициент учета вида используемой информации и сложности алгоритма программного продукта, его значение определяется по формуле:

$$K_{IA} = \frac{K'_P * n_P + K'_{NS} * n_{NS} + K'_B * n_B}{n_P + n_{NS} + n_B},$$

где K'_P — коэффициент учета сложности алгоритма ПП и вида используемой информации для переменной информации; K'_{NS} — коэффициент учета сложности алгоритма ПП и вида используемой информации для нормативно-справочной информации; K'_B — коэффициент учета сложности алгоритма ПП и вида используемой информации для баз данных. n_P — количество наборов данных переменной информации; n_{NS} — количество наборов данных нормативно-справочной информации; n_B — количество баз данных. Коэффициенты находим по таблице (группа новизны - А):

$$K'_P = 2.02,$$

$$K'_{NS} = 1.21,$$

$$K'_B = 1.05.$$

Количество наборов данных, используемых в рамках задачи:

$$n_P = 3,$$

$$n_{NS} = 0,$$

$$n_B = 1.$$

Находим значение K_{IA} :

$$K_{IA} = \frac{2.02 * 3 + 1.21 * 0 + 1.05 * 1}{3 + 0 + 1} = 1.7775.$$

Тогда:

$$\tau_{RP} = (15 + 91) * 1.00 * 1.75 * 0.8 * .7 * 1.7775 = 185.$$

7.1.5. Трудоемкость выполнения стадии «Внедрение»

Расчёт трудоёмкости разработки технического проекта (τ_V) [чел.-дни] производится по формуле:

$$\tau_V = (t_{RZ}^V + t_{RP}^V) * K_K * K_R * K_Z,$$

где t_{RZ}^V — норма времени, затрачиваемого разработчиком постановки задач на выполнение процедур внедрения программного продукта, [чел.-дни]; t_{RP}^V — норма времени, затрачиваемого разработчиком программного обеспечения на выполнение процедур внедрения программного продукта, [чел.-дни]. По таблице принимаем (функциональное назначение — технико-экономическое планирование, количество разновидностей форм входной информации — 2 (твиты, новости), количество разновидностей форм выходной информации — 2 (набор связей твит-новости, оценка работы рекомендательной системы)):

$$t_{RZ}^V = 17,$$

$$t_{RP}^V = 19.$$

Коэффициент K_K и K_Z были найдены выше:

$$K_K = 1.00,$$

$$K_Z = 0.7.$$

K_R — коэффициент учета режима обработки информации. По таблице принимаем (группа новизны — А, режим обработки информации — реальный масштаб времени):

$$K_R = 1.60.$$

Тогда:

$$\tau_V = (17 + 19) * 1.00 * 1.60 * 0.7 = 40.$$

Общая трудоёмкость разработки ПП:

$$\tau_{RP} = 79 + 175 + 261 + 185 + 40 = 740.$$

7.2. Расчет количества исполнителей

Средняя численность исполнителей при реализации проекта разработки и внедрения ПО определяется соотношением:

$$N = \frac{t}{F},$$

где t — затраты труда на выполнение проекта (разработка и внедрение ПО); F — фонд рабочего времени. Разработка велась 5 месяцев с 1 января 2016 по 31 мая 2016. Количество рабочих дней по месяцам приведено в таблице 3. Из таблицы получаем, что фонд рабочего времени

$$F = 96.$$

Таблица 3: Количество рабочих дней по месяцам

Номер месяца	Интервал дней	Количество рабочих дней
1	01.01.2016 - 31.01.2016	15
3	01.02.2016 - 29.02.2016	20
4	01.03.2016 - 31.03.2016	21
5	01.04.2016 - 30.04.2016	21
6	01.05.2016 - 31.05.2016	19
Итого		96

Получаем число исполнителей проекта:

$$N = \frac{740}{96} = 8$$

Для реализации проекта потребуются 3 старших инженеров и 5 простых инженеров.

7.3. Ленточный график выполнения работ

На основе рассчитанных в главах 7.1, 7.2 трудоёмкости и фонда рабочего времени найдём количество рабочих дней, требуемых для выполнения каждого этапа разработка. Результаты приведены в таблице 4.

Таблица 4: Трудоёмкость выполнения работы над проектом

Номер стадии	Название стадии	Трудоёмкость [чел.-дни]	Удельный вес [%]	Количество рабочих дней
1	Техническое задание	79	11	10
2	Эскизный проект	175	24	23
3	Технический проект	261	35	34
4	Рабочий проект	185	25	24
5	Внедрение	40	5	5
Итого		740	100	96

Планирование и контроль хода выполнения разработки проводится по ленточному графику выполнения работ. По данным в таблице 4 в ленточный график (таблица 5), в ячейки столбца “продолжительности рабочих дней” заносятся времена, которые требуются на выполнение соответствующего этапа. Все исполнители работают одновременно.

Таблица 5: Ленточный график выполнения работ

Номер стадии		Продолжительность [раб.-дни]	Календарные дни																							
			Количество рабочих дней																							
			01.01.2016 - 03.01.2016	04.01.2016 - 10.01.2016	11.01.2016 - 17.01.2016	18.01.2016 - 24.01.2016	25.01.2016 - 31.01.2016	01.02.2016 - 07.02.2016	08.02.2016 - 14.02.2016	15.02.2016 - 21.02.2016	22.02.2016 - 28.02.2016	29.02.2016 - 06.03.2016	07.03.2016 - 13.03.2016	14.03.2016 - 20.03.2016	21.03.2016 - 27.03.2016	28.03.2016 - 03.04.2016	04.04.2016 - 10.04.2016	11.04.2016 - 17.04.2016	18.04.2016 - 24.04.2016	25.04.2016 - 01.05.2016	02.05.2016 - 08.05.2016	08.05.2016 - 15.05.2016	16.05.2016 - 22.05.2016	23.05.2016 - 29.05.2016	30.05.2016 - 31.05.2016	
1	10			5	5		5		5	6	3	5	3	5	5	5	5	5	5	5	3	4	5	5	2	
2	23					5	5	5	6	2																
3	34									1	5	3	5	5	5	5	5									
4	24																	5	5	3	4	5	2			
5	5																						3	2		

7.4. Определение себестоимости программной продукции

Затраты, образующие себестоимость продукции (работ, услуг), состоят из затрат на заработную плату исполнителям, затрат на закупку или аренду оборудования, затрат на организацию рабочих мест, и затрат на накладные расходы.

В таблице 6 приведены затраты на заработную плату и отчисления на социальное страхование в пенсионный фонд, фонд занятости и фонд обязательного медицинского страхования (30.5 %). Для старшего инженера предполагается оклад в размере 120000 рублей в месяц, для инженера предполагается оклад в размере 100000 рублей в месяц.

Таблица 6: Затраты на зарплату и отчисления на социальное страхование

Должность	Зарплата в месяц	Рабочих месяцев	Суммарная зарплата	Затраты на социальные нужды
Старший инженер	120000	5	600000	183000
Старший инженер	120000	5	600000	183000
Старший инженер	120000	5	600000	183000
Инженер	100000	5	500000	152500
Инженер	100000	5	500000	152500
Инженер	100000	5	500000	152500
Инженер	100000	5	500000	152500
Инженер	100000	5	500000	152500
Суммарные затраты			5611500	

Расходы на материалы, необходимые для разработки программной продукции, указаны в таблице 7.

Таблица 7: Затраты на материалы

Наименование материала	Единица измерения	Кол-во	Цена за единицу, руб.	Сумма, руб.
Бумага А4	Пачка 400 л.	2	200	400
Картридж для принтера HP P10025	Шт.	3	450	1350
Суммарные затраты				1750

В работе над проектом используется специальное оборудование — персональные электронно-вычислительные машины (ПЭВМ) в количестве 8 шт. Стоимость одной ПЭВМ составляет 90000 рублей. Месячная норма амортизации $K = 2,7\%$. Тогда за 4 месяцев работы расходы на амортизацию составят $P = 90000 * 8 * 0.027 * 4 = 77760$ рублей.

Общие затраты на разработку программного продукта (ПП) составят
 $5611500 + 1750 + 77760 = 5691010$ рублей.

7.5. Определение стоимости программной продукции

Для определения стоимости работ необходимо на основании плановых сроков выполнения работ и численности исполнителей рассчитать общую сумму затрат на разработку программного продукта. Если ПП рассматривается и создается как продукция производственно-технического назначения, допускающая многократное тиражирование и отчуждение от непосредственных разработчиков, то ее цена P определяется по формуле:

$$P = K * C + Pr,$$

где C — затраты на разработку ПП (сметная себестоимость); K — коэффициент учёта затрат на изготовление опытного образца ПП как продукции производственно-технического назначения ($K = 1.1$); Pr — нормативная прибыль, рассчитываемая по формуле:

$$Pr = \frac{C * \rho_N}{100},$$

где ρ_N — норматив рентабельности, $\rho_N = 30\%$;

Получаем стоимость программного продукта:

$$P = 1.1 * 5691010 + 5691010 * 0.3 = 7967414 \text{ рублей.}$$

7.6. Расчет экономической эффективности

Основными показателями экономической эффективности является чистый дисконтированный доход (NPV) и срок окупаемости вложенных средств. Чистый дисконтированный доход определяется по формуле:

$$NPV = \sum_{t=0}^T (R_t - Z_t) * \frac{1}{(1 + E)^t},$$

где T — горизонт расчета по месяцам; t — период расчета; R_t — результат, достигнутый на t шаге (стоимость); Z_t — текущие затраты (на шаге t); E — приемлемая для инвестора норма прибыли на вложенный капитал.

На момент начала 2016 года, ставка рефинансирования 11% годовых (ЦБ РФ), что эквивалентно 0.87% в месяц. В виду особенности разрабатываемого продукта он

может быть продан лишь однократно. Отсюда получаем

$$E = 0.0087.$$

В таблице 8 находится расчёт чистого дисконтированного дохода. График его изменения приведён на рисунке 2.

Таблица 8: Расчёт чистого дисконтированного дохода

Месяц	Текущие затраты, руб.	Затраты с начала года, руб.	Текущий доход, руб.	ЧДД, руб.
Январь	1201810	1201810	0	-1201810
Февраль	1122300	2324110	0	-2314430
Март	1122300	3446410	0	-3417454
Апрель	1122300	4568710	0	-4510964
Мая	1122300	5700730	7967414	2101032

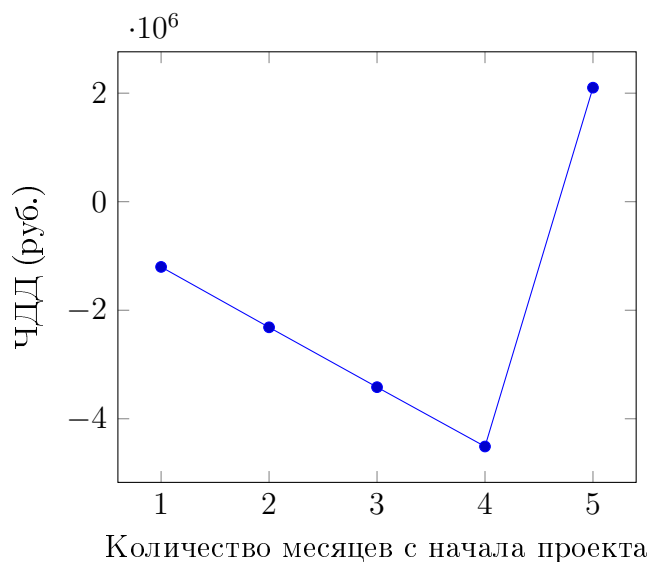


Рисунок 2 — График изменения чистого дисконтированного дохода

Согласно проведенным расчетам, проект является рентабельным. Разрабатываемый проект позволит превысить показатели качества существующих систем и сможет их заменить. Итоговый ЧДД составил: 2101032 рублей.

7.7. Результаты

В рамках организационно-экономической части был спланирован календарный график проведения работ по созданию подсистемы поддержки проведения диа-

гностики промышленных, а также были проведены расчеты по трудозатратам. Были исследованы и рассчитаны следующие статьи затрат: материальные затраты; заработная плата исполнителей; отчисления на социальное страхование; накладные расходы.

В результате расчетов было получено общее время выполнения проекта, которое составило 96 рабочих дней, получены данные по суммарным затратам на создание системы для автоматического сопоставления твитов и новостных статей, которые составили 5700730 рублей. Согласно проведенным расчетам, проект является рентабельным. Цена данного программного проекта составила 7967414 рублей, итоговый ЧДД составил 2101032 рублей.

8. Заключение

Что получилось. Должно отображать задачи описанные во введении

Список литературы

- [1] W. Guo, H. Li, H. Ji, and M. T. Diab. Linking tweets to news: A framework to enrich short text data in social media. - ACL, pages 239–249, 2013.
- [2] Manos Tsagkias, Maarten de Rijke, Wouter Weerkamp. Linking Online News and Social Media. - ISLA, University of Amsterdam.
- [3] T. Hoang-Vu, A. Bessa, L. Barbosa and J. Freire. Bridging Vocabularies to Link Tweets and News. - International Workshop on the Web and Databases (WebDB 2014), Snowbird, Utah, US, 2014.
- [4] J. Sankaranarayanan, H. Samet, B. Teitler, M. Lieberman, J. Sperling. TwitterStand: news in tweets. - 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2009, Seattle, Washington.
- [5] Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. 2011. Transferring topical knowledge from auxiliary long texts for short text clustering. In Proceedings of the 20th ACM international conference on Information and knowledge management.
- [6] Weiwei Guo and Mona Diab. 2012a. Modeling sentences in the latent space. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics.
- [7] Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
- [8] Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In Proceedings of the Twentieth International Conference on Machine Learning.
- [9] Eric Huns. Hunseblog on Wordpress: URL: <https://hunseblog.wordpress.com/2014/09/15/installing-numpy-and-openblas/>.
- [10] Арсеньев В.В., Сажин Ю.Б. Методические указания к выполнению организационно-экономической части дипломных проектов по созданию программной продукции. М.: изд. МГТУ им. Баумана, 1994. 52 с. 2.
- [11] Под ред. Смирнова С.В. Организационно-экономическая часть дипломных проектов исследовательского профиля. М.: изд. МГТУ им. Баумана, 1995. 100 с.
- [12] ГОСТ 34.601 "АС. Стадии создания".