1 Руководство пользователя

Программный комплекс состоит из двух приложений, каждое из которых устанавливается и используется в отдельности.

- twnews_consumer — приложение, предназначенное для того, чтобы получать твиты с твиттера и новости с rss каналов.
- twnews — приложение, позволяющее по твитам и новостям, произвести все необходимые для автоматического построения рекомендций преобразования данных, построить рекомендации и оценить полученный результат.

Оба пакета ориентированы на работу в операционных системах семейства linux. Для начала работы необходимо получить содержимое git-репозитория: https://github.com/art-vybor/twnews.git, в котором расположены исходный код обоих приложений. Если установлен пакет git, то получение git-репозитория происходит с помощью следующей команды:

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

Для сборки приложений необходимо иметь установленный менеджер пакетов языка Python — pip. С помощью него необходимо установить Python библиотеку setuptools:

```
$ pip install setuptools
```

Отдельно стоит отметить, что для корректной работы пакетов twnews и twnews_consumer, все директории, указываемые в конфигурации пакетов должны быть созданы заранее.

1.1 Пакет twnews_consumer

Исходный код приложения twnews_consumer расположен в папке consumer в корне репозитория. Приложение позволяет выкачивать и сохранять твиты и новости в формате, удобном для дальнейшей работы пакета twnews.

1.1.1 Конфигурирование

Конфигурирование пакета twnews_consumer производится в файле twnews_consumer/defaults.py. После изменения параметров, необходимо переустановить пакет. Описание задаваемых параметров находится в таблице 1.

1.1.2 Установка

Для установки приложения, необходимо зайти в папку consumer с исходным кодом пакет twnews_consumer, находящуюся в корне репозитория, и выполнить команду:

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/user/twnews_data/'	Путь до директории, в которую будут сохранены данные

RSS_FEEDS	{
	'lenta': {'rss_url':
	'http://lenta.ru/rss'},
	'rt': {'rss_url':
	'https://russian.rt.com/rss'},
	}
		Новостные источники, которые требуется выкачать

TWEETS_LANGUAGES	['ru']	Список языков, твиты с использованием которых выкачиваются из твиттера

```
$ make install
```

Во время установки, с целью распаковки секретного ключа, необходимого для работы с API твиттера, требуется ввести секретный пароль.

1.1.3 Использование

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, численный уникальный идентификатор ,время публикации, информацию о хэштегах и ссылках и является ли он ретвитом.

Приложение обладает интерфейсом командной строки. Для старта скачивания новостей, необходимо запустить команду:

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

Для старта скачивания сообщения твиттера, необходимо запустить команду:

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

Для завершения скачивания, необходимо использовать сочетание клавиш «Ctrl-C». Приложение обработает прерывание и корректно завершится.

Приложение записывает вспомогательную информацию о своём статусе и обработанных ошибках в файл лога. Поэтому из файла лога можно узнать актуальную информацию о работе приложения. 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))
```

Приложение twnews_consumer устойчиво к ошибкам, возникающим при получении новостей и твитов, востановление работы заключается в перезапуске скачивания информации. Ввиду этого приложение реализует скачивание данных согласно семантики *at-most-once* — гарантируется отсутствие дублей, но допускается потеря данных.

1.2 Пакет twnews

Пакет twnews располагается в папке core в корне репозитория. Приложение позволяет обрабатывать данные полученные с помощью консьюмера с целью автоматического установления связей между твитами и новостными статьями и оценки качества полученного решения.

1.2.1 Конфигурирование

Конфигурирование пакета twnews производится в файле twnews/defaults.py. После изменения параметров, необходимо переустановить пакет. Описание задаваемых параметров находится в таблице 2.

1.2.2 Установка

Для установки приложения, необходимо зайти в папку core с исходным кодом пакет twnews, находящуюся в корне репозитория, и выполнить команду:

```
$ make install
```

Для повышения производительности рекомендуется вручную собрать пакет numpy с использованием низкоуровневой математической библиотеки OpenBLAS [7].

1.2.3 Использование

Финальным результатом работы приложения является получение рекомендаций для произвольных твитов. Построение рекомендаций происходит в несколько стадий (каждая стадия представляет собой отдельный вызов приложения), количество стадий различается для различных методов рекомендаций.

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/user/twnews_data/'`	Путь до директории, в которую были скачены данные приложением twnews_consumer
DATASET_FRACTION	`1.0`	Часть множества скаченных твитов на основе которых происходит построение набора данных
TMP_FILE_DIRECTORY	`'/tmp/twnews/'`	Путь до директории в которую будут сохранены временные данные
DEFAULT_WTMF_OPTIONS	`{ 'DIM': 90, 'WM': 0.95, 'ITERATIONS': 1, 'LAMBDA': 1.95 }`	Настройки метода WTMF
DEFAULT_WTMFG_OPTIONS	`{ 'DIM': 220, 'WM': 5, 'ITERATIONS': 1, 'DELTA': 0.06, 'LAMBDA': 6 }`	Настройки метода WTMF-G

Приложение обладает интерфейсом командной строки. Для удобства использования функционал приложения разбивается на набор независимых друг от друга точек входа, каждая из которых представляет интерфейс для выполнения одной из стадий работы построения рекомендаций. Список всех точек входа представлен в таблице 3.

Далее приводится более детальное описание каждой точки входа. Для каждой точки входа в качестве примера использования приводится две команды командной строки: результат вызова автоматически порождаемой приложением справочной информации и пример вызова точки входа.

Точка входа tweets_sample позволяет получить случайный набор твитов из собранных данных. Параметризуется двумя необязательными параметрами length — задаёт количество твитов в результате, и tweets — определяет полное имя создаваемого файла, который содержит список твитов. Пример использования:

```
$ twnews tweets_sample -h
usage: twnews tweets_sample [-h] [--length LENGTH] [--tweets TWEETS]
```

Таблица 3 — Описание точек входа пакета twnews

Название точки входа	**Пример команды запуска**	**Описание**
tweets_sample	twnews tweets_sample	Получение случайного набора твитов из собранных данных
resolver	twnews resolver	Расшифровка сокращённых ссылок
build_dataset	twnews build_dataset	Автоматическое построение набора данных
train	twnews train	Построение модели для методов WTMF и WTMF-G
apply	twnews apply	Применение модели для методов WTMF и WTMF-G
tfidf_dataset	twnews tfidf_dataset	Применение метода TF-IDF на набор данных
tfidf_tweets	twnews tfidf_tweets	Применение метода TF-IDF на произвольные твиты
recommend_dataset	twnews recommend_dataset	Построение и оценка качества рекомендаций для набора данных
recommend_tweets	twnews recommend_tweets	Построение рекомендаций для произвольных твитов

```
optional arguments:
  -h, --help            show this help message and exit
  --length LENGTH       num of tweets in sample (default: 1000)
  --tweets TWEETS       tweets sample filepath (default:
                        /home/avybornov/tmp/tweets_sample)
```

```
$ twnews tweets_sample
get sample of random tweets
sample generated and saved at /home/avybornov/tmp/tweets_sample
```

Точка входа build_dataset позволяет автоматически построить набор данных. Параметризуется двумя необязательными параметрами unique_words — задаёт процент уникальных слов в твите для пар твит-новость, и dataset — задаёт полное имя создаваемого файла, который содержит набор данных; Пример использования:

```
$ twnews build_dataset -h
usage: twnews build_dataset [-h] [--unique_words UNIQUE_WORDS]
                            [--dataset DATASET]

optional arguments:
  -h, --help            show this help message and exit
  --unique_words UNIQUE_WORDS
```

```
                        percent of unique words in tweet by corresponding news
                        (default: 0.0)
  --dataset DATASET     dataset filepath (default:
                        /home/avybornov/tmp/dataset)
```

```
$ twnews build_dataset
building automatic dataset
dataset builded and saved at /home/avybornov/tmp/dataset
```

Точка входа resolver позволяет расшифровать все сокращённые ссылки, используемые в скаченном приложением twnews_consumer множестве данных (параметр resolve). Отображение коротких ссылок в расшифрованные хранится отдельным файлом и в дальнешем используется при построении набора данных. Также точка входа resolver позволяет получить статистику по все расшифрованным ссылкам (параметр analyze). Пример использования:

```
$ twnews resolver -h
usage: twnews resolver [-h] (--resolve | --analyze)

optional arguments:
  -h, --help  show this help message and exit
  --resolve   resolve urls from all tweets (default: False)
  --analyze   print stats of resolved urls (default: False)
```

```
$ twnews resolver resolve
```

Точка входа train позволяет построить модели для методов WTMF и WTMF-G. Каждый метод параметризуется тремя необязательными параметрами: dataset — полное имя файла, содержащего набора данных, model_dir — директория в которую будет сохранён файл модели, dataset_applied — полное имя файла, в который будут записаны новости и твиты из набора данных с векторами для сравнения. Пример использования:

```
$ twnews train -h
usage: twnews train [-h] (--wtmf | --wtmf_g) [--dataset DATASET]
                    [--model_dir MODEL_DIR]
                    [--dataset_applied DATASET_APPLIED]

optional arguments:
  -h, --help            show this help message and exit
  --wtmf                wtmf method (default: False)
  --wtmf_g              wtmf_g method (default: False)
  --dataset DATASET     dataset filepath (default:
                        /home/avybornov/tmp/dataset)
```

```
  --model_dir MODEL_DIR
                        name of directory to save model (default:
                        /home/avybornov/tmp)
  --dataset_applied DATASET_APPLIED
                        dataset_applied filepath (default:
                        /home/avybornov/tmp/dataset_applied)

$ twnews train --wtmf
train model
apply model to dataset
model dumped to /home/avybornov/tmp/WTMF_(dataset_auto_0.0)_90_1_1.95_0.95
applied dataset saved: /home/avybornov/tmp/dataset_applied
```

Точка входа apply позволяет применить ранее построенные модели для методов WTMF и WTMF-G на набор твитов. Каждый метод параметризуется тремя параметрами: model — полное имя файла, содержащего модель, tweets — полное имя файла с множеством твитов, который был построен с использованием точки входа tweets_sample, tweets_applied — полное имя файла, в который будут записаны полученнные твиты с векторами для сравнения. Пример использования:

```
$ twnews apply -h
usage: twnews apply [-h] (--wtmf | --wtmf_g) [--model MODEL] [--tweets TWEETS]
                    [--tweets_applied TWEETS_APPLIED]

optional arguments:
  -h, --help            show this help message and exit
  --wtmf                wtmf method (default: False)
  --wtmf_g              wtmf_g method (default: False)
  --model MODEL         model filepath (default: /home/avybornov/tmp/WTMF_(dat
                        aset_auto_0.0)_90_1_1.95_0.95)
  --tweets TWEETS       tweets sample filepath (default:
                        /home/avybornov/tmp/tweets_sample)
  --tweets_applied TWEETS_APPLIED
                        tweets_applied filepath (default:
                        /home/avybornov/tmp/tweets_applied)

$ twnews apply --wtmf --model /home/avybornov/tmp/WTMF_\(dataset_auto_0.0\)
   _90_1_1.95_0.95
apply model
tweets applied and stored at /home/avybornov/tmp/tweets_applied
```

Точка входа tfidf_dataset позволяет применить метод TFIDF для нахождения векторов сравнения для новостей и твитов из набора данных. Параметризуется двумя параметрами dataset — полное имя файла, содержащего набор данных, dataset_applied — полное имя файла, в который будут записаны новости и твиты из набора данных с векторами для сравнения. Пример использования:

```
twnews tfidf_dataset -h
usage: twnews tfidf_dataset [-h] [--dataset DATASET]
                            [--dataset_applied DATASET_APPLIED]

optional arguments:
  -h, --help            show this help message and exit
  --dataset DATASET     dataset filepath (default:
                        /home/avybornov/tmp/dataset)
  --dataset_applied DATASET_APPLIED
                        dataset_applied filepath (default:
                        /home/avybornov/tmp/dataset_applied)

$ twnews tfidf_dataset
apply tfidf to dataset
dataset applied and stored at /home/avybornov/tmp/dataset_applied
```

Точка входа tfidf_tweets позволяет применить метод TFIDF для нахождения векторов сравнения для набора данных (как в случае с точкой входа tfidf_dataset). Параметризуется четырьмы параметрами dataset — полное имя файла, содержащего набор данных, dataset_applied — полное имя файла, в который будут записаны новости и твиты из набора данных с векторами для сравнения, tweets — полное имя файла с множеством твитов, который был построен с использованием точки входа tweets_sample, tweets_applied — полное имя файла, в который будут записаны полученнные твиты с векторами для сравнения. Пример использования:

```
$ twnews tfidf_tweets -h
usage: twnews tfidf_tweets [-h] [--dataset DATASET]
                           [--dataset_applied DATASET_APPLIED]
                           [--tweets TWEETS] [--tweets_applied TWEETS_APPLIED]

optional arguments:
  -h, --help            show this help message and exit
  --dataset DATASET     dataset filepath (default:
                        /home/avybornov/tmp/dataset)
  --dataset_applied DATASET_APPLIED
```

```
                        dataset_applied filepath (default:
                        /home/avybornov/tmp/dataset_applied)
  --tweets TWEETS       tweets sample filepath (default:
                        /home/avybornov/tmp/tweets_sample)
  --tweets_applied TWEETS_APPLIED
                        tweets_applied filepath (default:
                        /home/avybornov/tmp/tweets_applied)

$ twnews tfidf_tweets
apply tfidf to dataset
dataset applied and stored at /home/avybornov/tmp/dataset_applied
apply tfidf to tweets
tweets applied and stored at /home/avybornov/tmp/tweets_applied
```

Точка входа recommend_dataset позволяет построить рекомендации по набору новостей и твитов из набора данных с векторами для сравнения и получить метрики качества построенных рекомендаций. Параметризуется двумя параметрами: dataset_applied — название используемого набора данных с векторами для сравнения; dump — имя файла в который будут сохранены полученные рекомендации. Пример использования:

```
$ twnews recommend_dataset -h
usage: twnews recommend_dataset [-h] [--dataset_applied DATASET_APPLIED]
                                [--dump DUMP]

optional arguments:
  -h, --help            show this help message and exit
  --dataset_applied DATASET_APPLIED
                        dataset_applied filepath (default:
                        /home/avybornov/tmp/dataset_applied)
  --dump DUMP           recommendation dump filepath (default:
                        /home/avybornov/tmp/reccommendation_dump)
$ twnews recommend_dataset
build recommendation
recommendation result evaluation
RR = 0.908454481596
TOP1 = 0.869796484736
TOP3 = 0.940564292322
recommendation dumped to /home/avybornov/tmp/reccommendation_dump
```

Точка входа recommend_tweets позволяет построить рекомендации для произвольных твитов с векторами для сравнения. Параметризуется тремя параметрами dataset_applied —

название используемого набора данных с векторами для сравнения; tweets_applied — название файла с набором твитов с векторами для сравнения; dump — имя файла в который будут сохранены полученные рекомендации. Пример использования:

```
$ twnews recommend_tweets -h
usage: twnews recommend_tweets [-h] [--dataset_applied DATASET_APPLIED]
                               [--tweets_applied TWEETS_APPLIED] [--dump DUMP]

optional arguments:
  -h, --help            show this help message and exit
  --dataset_applied DATASET_APPLIED
                        dataset_applied filepath (default:
                        /home/avybornov/tmp/dataset_applied)
  --tweets_applied TWEETS_APPLIED
                        tweets_applied filepath (default:
                        /home/avybornov/tmp/tweets_applied)
  --dump DUMP           recommendation dump filepath (default:
                        /home/avybornov/tmp/reccommendation_dump)
$ twnews recommend_tweets
recommendation dumped to /home/avybornov/tmp/reccommendation_dump
```

Приложение записывает вспомогательную информацию о своём статусе и обработанных ошибках в файл лога. Поэтому из файла лога можно узнать актуальную информацию о работе приложения. 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
```

2 Эксперименты

Главной 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. Также, ввиду того, что алгоритмы WTMF и WTMF-G используют случайным образом инициализированные матрицы, каждый запуск этих алгоритмов производился трёхкратно, в результатах приведено усреднённое значение.

2.1 Методы оценки качества

Для оценки качества рассматриваются метрики применимые для решения задач информационного поиска. Твит рассматривается как запрос, а список новостей как ответ. Для каждого твита, получаемый список новостей ранжирован по мере убывания их схожести. В работе использованы две метрики: MRR и TOP_I, их описание дано ниже.

2.1.1 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:

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

2.1.2 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 получить по формуле:

$$TOP_I = \frac{1}{n}\sum_{i=1}^{n}Q_I(i). \tag{2}$$

В дальшейшем будут рассматриваться две метрики из группы метрик TOP_I: TOP_1, TOP_3.

2.2 Оптимизация качества WTMF, путём варьирования параметров

Оптимизация параметров модели для метода WTMF производится на наборе данных cutted с использованием метрики 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 [1], 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 4.

Таблица 4 — 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 4 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 5.

Таблица 5 — 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

Из таблицы 5 получаем оптимальные значения коэффициентов $\lambda = 0.95$ и $w_m = 1.95$.

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 6.

Таблица 6 — 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 6 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 7.

Таблица 7 — Качество работы алгоритма WMTF для различных значений K при фиксированных значениях $I = 1$, $\lambda = 0.95$, $w_m = 1.95$

K	**Значение метрики MRR**
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

Из таблицы 7 получаем оптимальное значение коэффициента $K = 90$.

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

2.3 Оптимизация качества WTMF-G, путём варьирования параметров

Оптимизация параметров модели для метода WTMF-G производится на наборе данных cutted с использованием метрики MRR. Модель WTMF-G зависит от пяти параметров: K, I, λ, δ, 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 параметров взяты оптимальные параметры для метода WTMF, а именно $K = 90$, $I = 1$, $w_m = 1.95$,$\lambda = 0.95$. В качестве начального приближения параметра δ берем значение 0.1.

Сначала оптимизируем параметры, влияющие на регуляризующий член: λ и δ, Для этого фиксируем остальные параметры: $I = 1$, $K = 90$, $w_m = 1.95$. Сначала найдём оптимальный порядок значений начального приближения. Результаты занесены в таблицу 8. Как видно из

Таблица 8 — Качество работы алгоритма WMTF-G для различных значений λ и δ при фиксированных значениях $I = 1$, $K = 90$, $w_m = 1.95$

$\lambda\backslash\delta$	**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

таблицы 8 порядок параметра δ оказывает влияние на качество, но достаточно слабое, порядок параметра λ, напротив очень сильно влияет на получаемое качество. Максимальное значение метрики достигнуто при $\lambda = 10$ и $\delta = 0.1$.

Для уточнения значения коэффициентов, производится исследование качества работы алгоритма в окрестностях максимального значения метрики. Результаты приведены в таблице 9. Результаты, описанные в таблице 9 достаточно однородны. Возьмём в качестве оптимального значения коэффициент полученную точку максимум: $\lambda = 6$, $\delta = 0.06$.

Рассмотрим влияние параметра w_m и найдём его оптимальное значение. Сначала рассмотрим качество алгоритма для различных порядков w_m. Результаты занесены в таблицу 10 Как видно из таблицы 10 порядок параметра w_m оказывает заметное влияние на качество. При значительном увеличении до 10^2 качество начинает резко падать. Максимальное значение метрики достигнуто при $w_m = 5$, уточним полученное значение.

Таблица 9 — Качество работы алгоритма WMTF-G для различных значений λ и δ при фиксированных значениях $I = 1$, $K = 90$, $w_m = 1.95$

$\lambda\backslash\delta$	**0.06**	**0.08**	**0.1**	**0.12**	**0.14**
4	0.8501	0.8512	0.8489	0.8530	0.8476
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

Таблица 10 — Качество работы алгоритма WMTF-G для различных значений w_m при фиксированных значениях $I = 1$, $K = 90$, $\lambda = 6$, $\delta = 0.6$

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

Для уточнения значения коэффициента w_m, производится исследование качества работы алгоритма в окрестностях максимального значения метрики. Результаты приведены в таблице 11. Из таблицы 11 получаем оптимальное значение параметра $w_m = 5$.

Таблица 11 — Качество работы алгоритма WMTF-G для различных значений w_m при фиксированных значениях $I = 1$, $K = 90$, $\lambda = 6$, $\delta = 0.6$

w_m	3.0	3.5	4.0	4.5	5.0	5.5	6.0	6.5	7.5
MRR	0.8592	0.8585	0.8594	0.8603	**0.8639**	0.8591	0.8586	0.8574	0.8536

Теперь рассмотрим оставшиеся два параметра K и I. Качество работы алгоритма WTMF-G для различных значений K и I приведено в таблице 12. Как показано в таблице 12, WTMF-G показывает аналогично методу WTMF поведение при изменении параметров K и I, а именно, в среднем с увеличением количества итераций, качество работы алгоритма уменьшается. Максимум был достигнут при $K = 220$, $I = 1$.

В итоге оптимизации качества рекомендаций на основе алгоритма WMTF-G были получены оптимальные параметры: $K = 220$, $I = 1$, $\delta = 0.6$, $\lambda = 6$, $w_m = 5$.

Таблица 12 — Качество работы алгоритма WMTF-G для различных значений K и I при фиксированных значениях $w\_m = 5$, $\lambda = 6$, $\delta = 0.06$

$K \backslash I$	1	2	3	4	5
30	0.7529	0.7927	0.7577	0.6736	0.5794
40	0.7992	0.8194	0.7695	0.6813	0.5828
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
130	0.8725	0.8588	0.8160	0.7264	0.6206
140	0.8740	0.8597	0.8157	0.7248	0.6241
150	0.8763	0.8620	0.8171	0.7263	0.6200
160	0.8740	0.8596	-	-	-
170	0.8768	0.8606	-	-	-
180	0.8785	0.8613	-	-	-
190	0.8767	0.8616	-	-	-
200	0.8769	0.8613	-	-	-
210	0.8786	0.8613	-	-	-
220	**0.8816**	0.8632	-	-	-
230	0.8814	0.8646	-	-	-
240	0.8758	0.8632	-	-	-

2.4 Сравнительные результаты

Для выявления влияния добавления связей текст-текст на результаты работы метода WMTF-G производится сравнительное тестирование алгоритма WTMF и WTMF-G. Тестирование производится для различных наборов данных. Результаты тестирования приведены в таблице 13.

Таблица 13 — 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	0.7854	0.6187	0.6756	0.8193	0.8775
auto	0.8640	0.8685	0.8096	0.8149	0.9148	0.9195
total	0.8196	0.8487	0.7452	0.7828	0.8851	0.9049
cutted	0.8630	0.8816	0.8045	0.8204	0.9118	0.9279
manual_nt	0.6194	0.7000	0.4733	0.5502	0.7223	0.8217
auto_nt	0.5297	0.5695	0.4436	0.4798	0.5750	0.6099
total_nt	0.5729	0.6341	0.4587	0.5143	0.6448	0.7296
cutted_nt	0.6495	0.7039	0.5371	0.5974	0.7372	0.7840

Как видно из таблицы 13 алгоритм WMTF-G показывает стабильно более высокий результат чем алгоритм WMTF, из этого можно сделать вывод, что добавление связей текст-текст позволяет построить более точные рекомендации.

Сравним два метода рекомендаций: TF-IDF и WTMF-G. Сначала посмотрим на результаты полученные на базовых эталонных наборах данных, то есть тех, которые наряду с нетривиальными содержат большое количество тривиальных связей. Результаты тестирования приведены в таблице 14.

Таблица 14 — Сравнительное тестирование алгоритмов TF-IDF и WTMF-G на базовых эталонных наборах данных

Набор данных	**Метрика MRR**		**Метрика** TOP_1		**Метрика** TOP_3	
	TF-IDF	**WTMF-G**	**TF-IDF**	**WTMF-G**	**TF-IDF**	**WTMF-G**
manual	0.8310	0.7854	0.7337	0.6756	0.9137	0.8775
auto	0.8817	0.8685	0.8344	0.8149	0.9299	0.9195
total	0.8610	0.8487	0.8044	0.7828	0.9249	0.9049
cutted	0.9075	0.8816	0.8499	0.8204	0.9453	0.9279

Как видно из таблицы 14 метод TF-IDF показывает заметно более лучший результат на всех наборах данных. В целом для метода TF-IDF получены неожиданно высокие результаты, качество полученное для метода TF-IDF авторами метода WTMF-G при связывании твитов и новостей почти в два раза меньше (качество полученное авторами метода WTMF-G приведено

в разделе **??**). Настолько высокие результаты метода TF-IDF получены по следующим причинам: во-первых, в русском твиттере очень много тривиальных связей твит-новость, во-вторых, ввиду специфики русского сегмента твиттер, в заголовках новостей и твитах их описывающих оказалось большое количество общих слов.

С целью нивелирования влияния тривиальных связей было проведено тестирование на наборах данных, которые содержат исключительно нетривиальные связи. Результаты экспериментов приведены в таблице 15.

Таблица 15 — Сравнительное тестирование алгоритмов TF-IDF и WTMF-G на наборах данных с нетривиальными твитами

Набор данных	**Метрика MRR**		**Метрика** TOP_1		**Метрика** TOP_3	
	TF-IDF	**WTMF-G**	**TF-IDF**	**WTMF-G**	**TF-IDF**	**WTMF-G**
manual_nt	0.7565	0.7000	0.6250	0.5502	0.8688	0.8217
auto_nt	0.6035	0.5695	0.5254	0.4798	0.6461	0.6099
total_nt	0.6914	0.6341	0.5833	0.5143	0.8688	0.7296
cutted_nt	0.7485	0.7039	0.6442	0.5974	0.8303	0.7840

Как видно из таблицы 15 на наборах данных с нетривиальными твитами наблюдается такой же результат, как и для наборов данных с полным набором твитов. То есть метод TF-IDF во всех рассматриваемых случаях показывает более высокое качество, чем метод WTMF-G. Из этого следует, что рассматриваемые нетривиальные твиты, оказались «большими» текстами, очень похожими на заголовки новостей — это вызвано влиянием специфики русского сегмента твиттера.

Также из таблиц 14 и 15 можно оценить влияние различных наборов данных. Автоматически собранный датасет для полного набора твитов даёт более лучшее качество, чем в ручную собранный, это вызвано влиянием большого числа тривиальных связей. Как и ожидается результаты на наборах данных с нетривиальными твитами показывают обратную картину.

На наборе данных cutted для полного набора данных разница качества между TF-IDF и WTMF-G наименьшая - это вызвано тем, что оптимизация метода WTMF-G производилась на этом наборе данных.

3 Технико-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 [9]:

- 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 [10] [8]:

а) 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;

б) 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;

в) 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;

г) 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;

д) 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.

3.1 Трудоемкость разработки программной продукции

Трудоемкость разработки программной продукции (τ_{PP}) может быть определена как сумма величин трудоемкости выполнения отдельных стадий разработки программного продук-

та из выражения:

$$\tau_{PP} = \tau_{TZ} + \tau_{EP} + \tau_{TP} + \tau_{RP} + \tau_V, \tag{3}$$

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.

3.1.1 Трудоемкость разработки технического задания

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

$$\tau_{TZ} = T^Z_{RZ} + T^Z_{RP}, \tag{4}$$

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

$$T^Z_{RZ} = t_Z \cdot K^Z_{RZ}, \tag{5}$$

$$T^Z_{RP} = t_Z \cdot K^Z_{RP}, \tag{6}$$

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

$$K^Z_{RZ} = 0.65.$$

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

$$K^Z_{RP} = 0.35.$$

Тогда:

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

3.1.2 Трудоемкость разработки эскизного проекта

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

$$\tau_{EP} = T^E_{RZ} + T^E_{RP}, \tag{7}$$

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

$$T^E_{RZ} = t_E \cdot K^E_{RZ}, \tag{8}$$

$$T^E_{RP} = t_E \cdot K^E_{RP}, \tag{9}$$

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

$$K^E_{RZ} = 0.7.$$

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

$$K^E_{RP} = 0.3.$$

Тогда:

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

3.1.3 Трудоемкость разработки технического проекта

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

ходной информации и определяется по формуле:

$$\tau_{TP} = (t^T_{RZ} + t^T_{RP}) \cdot K_V \cdot K_R, \tag{10}$$

где t^T_{RZ} — норма времени, затрачиваемого на разработку технического проекта (ТП) разработчиком постановки задач, [чел.-дни]; t^T_{RP} — норма времени, затрачиваемого на разработку ТП 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)):

$$t^T_{RZ} = 52,$$

$$t^T_{RP} = 14.$$

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:

$$K_V = \frac{K_P \cdot n_P + K_{NS} \cdot n_{NS} + K_B \cdot n_B}{n_P + n_{NS} + n_B}, \tag{11}$$

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:

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

Тогда:

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

3.1.4 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:

$$\tau_{RP} = (t^R_{RZ} + t^R_{RP}) \cdot K_K \cdot K_R \cdot K_Y \cdot K_Z \cdot K_{IA}, \tag{12}$$

где t^R_{RZ} — норма времени, затраченного на разработку рабочего проекта на алгоритмическом языке высокого уровня разработчиком постановки задач, [чел.-дни]. t^R_{RP} — 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)):

$$t^R_{RZ} = 15,$$

$$t^R_{RP} = 91.$$

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:

$$K_I A = \frac{K'_P \cdot n_P + K'_{NS} \cdot n_{NS} + K'_B \cdot n_B}{n_P + n_{NS} + n_B}, \tag{13}$$

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}:

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

Тогда:

$$\tau_{RP} = (15 + 91) \cdot 1.00 \cdot 1.75 \cdot 0.8 \cdot 0.7 \cdot 1.7775 = 185.$$

3.1.5 Трудоемкость выполнения стадии «Внедрение»

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

$$\tau_V = (t^V_{RZ} + t^V_{RP}) \cdot K_K \cdot K_R \cdot K_Z, \tag{14}$$

где t^V_{RZ} — норма времени, затрачиваемого разработчиком постановки задач на выполнение процедур внедрения программного продукта, [чел.-дни]; t^V_{RP} — 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)):

$$t^V_{RZ} = 17,$$

$$t^V_{RP} = 19.$$

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:

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

Общая трудоёмкость разработки ПП:

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

3.2 Расчет количества исполнителей

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

$$N = \frac{t}{F}, \tag{15}$$

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 16. Из таблицы получаем, что фонд рабочего времени

$$F = 96.$$

Таблица 16 — 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.

3.3 Ленточный график выполнения работ

На основе рассчитанных в главах 3.1, 3.2 трудоёмкости и фонда рабочего времени найдём количество рабочих дней, требуемых для выполнения каждого этапа разработка. Результаты приведены в таблице 17.

Таблица 17 — 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 17 в ленточный график (таблица 18), в ячейки столбца “продолжительности рабочих дней” заносятся времена, которые требуются на выполнение соответствующего этапа. Все исполнители работают одновременно.

Таблица 18 — Ленточный график выполнения работ

Номер стадии	Продолжительность [раб.-дни]	Календарные дни																							
		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
		Количество рабочих дней																							
		0	0	5	5	5	5	5	6	3	5	3	5	5	5	5	5	5	5	3	4	5	5	2	
1	10			5	5																				
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	

3.4 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 19 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 19 — 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 20.

Таблица 20 — 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 К = 2,7%. Тогда за 4 месяцев работы расходы на амортизацию составят $P = 90000 \cdot 8 \cdot 0.027 \cdot 4 = 77760$ рублей.

Общие затраты на разработку программного продукта (ПП) составят

$$5611500 + 1750 + 77760 = 5691010 \text{ рублей.}$$

3.5 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:

$$P = K \cdot C + Pr, \tag{16}$$

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:

$$Pr = \frac{C \cdot \rho_N}{100}, \tag{17}$$

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

Получаем стоимость программного продукта:

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

3.6 Расчет экономической эффективности

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

$$NPV = \sum_{t=0}^{T} (R_t - Z_t) \cdot \frac{1}{(1+E)^t}, \tag{18}$$

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 21 находится расчёт чистого дисконтированного дохода. График его изменения приведён на рисунке 1.

Согласно проведенным расчетам, проект является рентабельным. Разрабатываемый проект позволит превысить показатели качества существующих систем и сможет их заменить.

Таблица 21 — 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

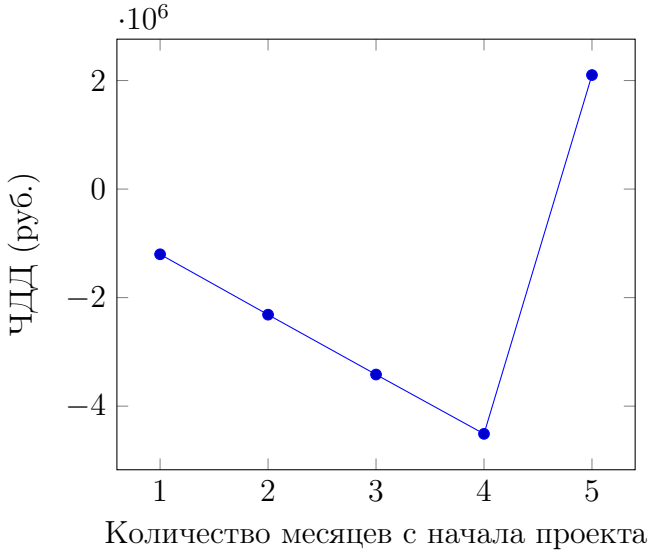

Рисунок 1 — График изменения чистого дисконтированного дохода

Итоговый ЧДД составил: 2101032 рублей.

3.7 Результаты

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

ЗАКЛЮЧЕНИЕ

В рамках дипломной работы было проведено исследование методов установления связей между твитами и новостными статьями. Было собрано несколько наборов данных и проанализированы различные подходы к их разметке. Реализован программный комплекс, позволяющий устанавить связи между твитами и новостными статьями в формате рекомендаций на основе различных подходов: методов WTMF, WTMF-G и метода, основанного на частотности слов (TF-IDF). На основе написанного программного комплекса произведено сравнение различных подходов. Как результат сравнения получено, что в для русского сегмента твиттера наиболее оптимальным подходом является метод, основанный на частотности слов (TF-IDF).

К сожалению, исследование показало, что наиболее простой метод, основанный на частотности слов, проявил наилучшее качество. Высокое качество метода TF-IDF является серьёзной проблемой, так как из этого следует, что рассматриваемые твиты, оказались «большими» текстами, очень похожими на заголовки новостей, то есть рассматриваемые твиты не репрезентативны. Основными причинами вызвавшими это являются как специфика собранных наборов данных, так и особенности русскоязычного сегмента твиттер.

Построение набора данных, состоящего из большего числа твитов, сильно отличающихся от заголовков новостей, позволит получить более реалистичную оценку качества. Что в свою очередь приведёт к возможности оценить преимущество методов, использующих дополнительные признаки, предоставляемые предметной областью.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

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. 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.

5. 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.

6. Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In Proceedings of the Twentieth International Conference on Machine Learning.

7. Eric Huns. Hunseblog on Wordpress: URL: https://hunseblog.wordpress.com/2014/09/15/installing-numpy-and-openblas/.

8. 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.

9. Под ред. Смирнова С.В. Организационно-экономическая часть дипломных проектов исследовательского профиля. М.: изд. МГТУ им. Баумана, 1995. 100 с.

10. ГОСТ 34.601 "АС. Стадии создания".

11. ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.