Математические структуры и моделирование
2017. № 3(43). С. 150–168

УДК 004.415.2:004.93 +
004.056.52:004.7
DOI: 10.25513/2222-8772.2017.3.150-168

ИДЕНТИФИКАЦИЯ СУИЦИДАЛЬНЫХ ГРУПП И НАРУШИТЕЛЕЙ АВТОРСКИХ ПРАВ В СОЦИАЛЬНЫХ СЕТЯХ

А.В. Костылев
студент, e-mail: kostart_94@mail.ru
Д.Н. Лавров
к.т.н., доцент, e-mail: dmitry.lavrov72@gmail.com
А.К. Гуц
д.ф.-м.н, профессор, e-mail: aguts@mail.ru

Омский государственный университет им. Ф.М. Достоевского

Аннотация. В данной работе поставлена задача разработки программного обеспечения, позволяющего автоматизировать поиск групп и пользователей в социальных сетях, размещающих в сеть суицидальные материалы и материалы, нарушающие авторское право. В качестве социальной сети для исследования возможности подобного поиска была выбрана социальная сеть «ВКонтакте» — крупнейшая социальная сеть СНГ. В основу разрабатываемого программного обеспечения был положен алгоритм латентного семантического анализа. В статье обсуждаются основные варианты алгоритма и подобран наиболее адекватно работающий на реальных данных вариант.

Ключевые слова: классификация текстов, латентно-семантический анализ, авторское право, правонарушения, социальные сети, суицидальные группы.

Введение

Социальные сети стремительно входят в современную социальную реальность. По состоянию на декабрь 2016 года количество пользователей сети «Facebook» превысило 1700 миллионов человек, а сети «ВКонтакте» — 400 миллионов. Согласно опросу исследовательского холдинга «Ромир» более 90% российских пользователей интернета ведут активную социальную жизнь в сети [1].

Из-за стремительного увеличения популярности социальных сетей они стали источником правонарушений. По данным информационно-аналитического центра «Сова», только в 2015 году за посты и репосты в социальной сети «ВКонтакте» к уголовной ответственности привлекли 119 человек [2].

При этом отсутствуют какие-либо данные об автоматизации поиска правонарушений: до сих пор большинство проверок выполняется по жалобам других

пользователей, которые могут быть не всегда заинтересованы в выявлении нарушения. Во многом отсутствие подобных методов можно объяснить новизной способов правонарушений, их непохожестью на прочие способы и относительно недавним осознанием необходимости борьбы с ними.

В данной работе поставлена задача разработки программного обеспечения, позволяющего автоматизировать поиск групп и пользователей в социальных сетях, размещающих суицидальные материалы и материалы, нарушающие авторское право. В дальнейшем по тексту статьи мы будем называть такие материалы запрещёнными.

1. Выбор инструментов и подготовка к созданию приложения

1.1. Выбор инструментов разработки

В качестве социальной сети для исследования возможности подобного поиска была выбрана социальная сеть «ВКонтакте» — крупнейшая социальная сеть СНГ. Как уже было сказано, по состоянию на декабрь 2016 года в ней имеется более 400 миллионов уникальных пользователей, в её базе данных учётных записей представлена огромная доля российского населения. Данная сеть имеет собственное API — VK API, позволяющее осуществлять самые разнообразные запросы.

В качестве языка разработки было принято решение использовать язык программирования Java. Так как в прошлом значительная часть исходного кода уже была написана на языке Python, чтобы избежать его переписывания, был использован Jython — реализация языка Python на языке Java. В качестве сборщика проекта использовался фреймворк Maven. Для обращения к VK API и разбора полученных результатов использовались библиотеки Apache HttpClient и Gson. Графический интерфейс был написан с использованием Swing. Различные полезные функции линейной алгебры были взяты из библиотеки Jama. Для логирования использовались Slf4J и Log4j.

Для взаимодействия с VK API можно использовать библиотеку: vk-java-sdk (`https://github.com/VKCOM/vk-java-sdk`). Однако для достижения большей гибкости запросов и для гарантированной корректной работы API-методов было решено создать свой API, а не пользоваться готовой реализацией.

1.2. Создание standalone-приложения

Все методы VK API можно подразделить на 2 типа: открытые, не требующие ключа доступа, и закрытые, для которых он необходим. К закрытым методам относятся, в частности, методы для работы с аудио- и видеоматериалами.

Для того, чтобы получить возможность обращаться ко всем методам VK API, необходимо зарегистрировать своё standalone-приложение.

Процесс регистрации приложения в социальной сети «ВКонтакте» весьма прост: достаточно указать имя приложения и его тип — «standalone» (рис. 1).

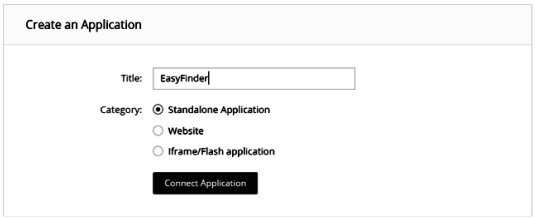

Рис. 1. Создание standalone-приложения

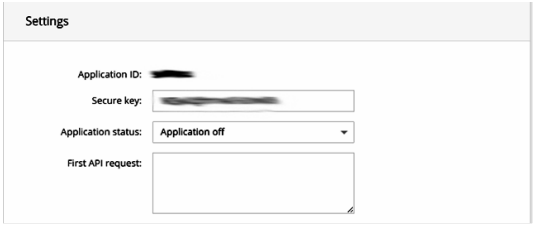

Рис. 2. Полученные Application Id и Secure key

При регистрации приложения в сети «ВКонтакте» ему присваиваются два параметра — Application ID и Secure key (рис. 2). В дальнейшем они будут необходимы при авторизации пользователя «ВКонтакте», от имени которого и будут выполняться запросы к закрытым методам API.

После выполнения этих шагов можно переходить к разработке самого приложения.

1.3. Авторизация

Для того чтобы получить доступ ко всем возможностям VK API, необходимо выполнить вход с одного из аккаунтов социальной сети и дать приложению необходимые права доступа. Для выполнения входа требуется отправить get-запрос по адресу `http://oauth.vk.com/oauth/authorize` с параметрами:

- `client_id = id_нашего_приложения`
- `scope = 'offline, video, audio'` — для получения бессрочного доступа к API видео- и аудиофайлам
- `response_type = 'token'`
- `v = 5.52` — актуальная версия VK API

В случае удачного выполнения запроса вернётся значение `access_token`, необходимое для осуществления запросов к закрытым методам VK API.

1.4. Ограничения VK API

В ходе исследования документации VK API выяснилось, что оно имеет несколько ограничений:

- максимальное количество возвращаемых объектов: 100 для `wall.get`, 200 для `video.get`;
- максимальное количество запросов — 3 запроса в секунду.

Таким образом, при создании функций использовалась следующая логика: API-запросы отправлялись со смещением (offset), каждую треть секунды до тех пор, пока количество полученных очередным API-запросом результатов не становилось равным нулю.

Недостатком данного подхода является относительно длительное время выполнения запроса. Данный недостаток можно устранить, используя API-метод `execute`, позволяющий выполнить произвольный код на языке VKScript

— аналоге языка JavaScript, по уверениям разработчиков, совместимом с ECMAScript. В предыдущей работе было показано, что, несмотря на некоторую ограниченность возможностей данного подхода, с его помощью можно добиться увеличения скорости выполнения запросов к VK API до 6,5 раз.

Также необходимо упомянуть, что 16 декабря 2016 года руководство «ВКонтакте» отключило доступ к публичному музыкальному API сторонним приложениям [3].

2. Разработка части приложения, определяющей предпочтения пользователей и тематику групп по размещённым постам

При решении задачи поиска групп и пользователей социальной сети «ВКонтакте», размещающих в сеть запрещённые материалы, была рассмотрена более общая задача: выявление предпочитаемых тем постов пользователя и определение тематики групп. Решив данную задачу, несложно искать только пользователей и группы с определёнными «интересами». Было принято решение воспользоваться методом латентно-семантического анализа, предназначенного для выявления скрытых связей между документами, о котором будет рассказано ниже.

2.1. Получение постов пользователя/группы

Для того чтобы проанализировать записи пользователя или группы в сети «ВКонтакте», их необходимо получить с помощью VK API. Для этого требуется вызвать только один API-метод: `wall.get`, возвращающий записи пользователя/группы на стене. Данный метод может вернуть до 100 результатов, чтобы получить все результаты, его нужно вызвать несколько раз. Для обращения к VK API была использована библиотека Apache HttpClient, для парсинга получившихся JSON-данных использована библиотека Gson. На основе вышеперечисленных технологий созданы функции `Set<Post> getPostsByUserId(Integer owner_id, boolean isOwnerOnly)` и `Set<Post> getPostsByGrouprId(Integer group_id)`, возвращающие список постов на стене пользователя/группы. Переменная `isOwnerOnly` в случае пользователя определяет, возвращать все посты или только посты непосредственно данного пользователя.

2.2. Обработка полученных данных, латентно-семантический анализ

Из получившихся на прошлом этапе постов выделяется текст, из него отфильтровываются небуквенные символы, остальные символы переводятся в нижний регистр. Из полученных строк строится список слов, присутствующих в постах, а из него составляется матрица A наличий слов в постах. Строки матрицы — все слова, встречающиеся хотя бы в одном из постов, столбцы —

номера постов пользователя/группы, а в пересечениях стоят либо единицы, если данное слово есть в данном посте, либо нули, если нет (рис. 3).

	T_1	T_2	T_3
бензол	1	1	0
углевод	1	0	1
бензин	1	0	1
пластмасса	0	1	0
синтез	0	0	1

Рис. 3. Пример внешнего вида матрицы A

Для классификации текстов может использоваться методы на основе k-средних, хорошо зарекомендовавший себя для классификации научных текстов [7]. Тем не менее, было принято решение применить метод латентно-семантического анализа, который основывается на сингулярном разложении матрицы A с понижением ранга: матрица A раскладывается на 3 матрицы: U, Σ, V^T, где матрицы U и V — это две унитарные матрицы, состоящие из левых и правых сингулярных векторов соответственно, а Σ — диагональная матрица, значения на диагонали которой называются сингулярными значениями (числами) матрицы A. Особенность сингулярного разложения заключается в том, что если в матрице Σ оставить только k наибольших сингулярных значений, то комбинация получившихся матриц $\widehat{A} = U \cdot \Sigma \cdot V^T$ будет наилучшим приближением исходной матрицы A к матрице ранга k.

Ранг полученной матрицы требуется понизить из-за того, что исходная матрица содержит множество «шумов». Однако чрезмерное понижение ранга матрицы ведёт к потере значимой информации. Существует множество мнений по поводу выбора наилучшего значения ранга матрицы k: например, в работе А.Д. Хомоненко и С.А. Краснова показывается, что для решения задачи по разделению двух различных групп наилучшим рангом является $k = 2$ [4].

Получившуюся матрицу $\widehat{A}$ можно рассмотреть как матрицу вектор-значений, соответствующих постам T_i в n-размерном пространстве. Из этой матрицы можно составить матрицу корреляции векторов P, — например, подсчётом косинуса угла для каждой пары вектор-значений из матрицы A.

Также, согласно данным исследований [5, 6], получившуюся матрицу V^T можно рассмотреть в качестве как вектор-значений, соответствующих постам T_i в n-размерном пространстве. Если же взять $k = 2$, то каждому посту T_i будет соответствовать пара значений (x, y), которые очень удобно представить в виде графика.

Существует множество готовых библиотек для Java, позволяющих использовать сингулярное разложение: Jama, ujmp, Commons Math, colt, jblas. В нашей программе была использована Jama (Java Matrix Library) — библиотека функций линейной алгебры, созданная в NIST и являющаяся общественным достоянием.

2.3. Проверка работы латентно-семантического анализа на тестовых данных

Для проверки корректности и демонстрации работы алгоритма сингулярного разложения было написано две функции: первая рассчитывала матрицу

1	0,98	0,97	0,98	1	0,98	0,34	0,37	0,31	0,24	0,41	0,31	0,59	0,63	0,65	0,72	0,65	0,6
0,98	1	1	1	0,99	1	0,17	0,21	0,14	0,07	0,25	0,14	0,44	0,48	0,5	0,59	0,5	0,45
0,97	1	1	1	0,97	1	0,09	0,12	0,05	0	0,16	0,05	0,36	0,41	0,42	0,52	0,42	0,37
0,98	1	1	1	0,98	1	0,14	0,17	0,1	0,03	0,21	0,1	0,41	0,45	0,47	0,56	0,47	0,41
1	0,99	0,97	0,98	1	0,98	0,32	0,35	0,29	0,22	0,39	0,29	0,57	0,61	0,63	0,7	0,63	0,57
0,98	1	1	1	0,98	1	0,15	3,18	0,11	0,04	0,22	3,11	0,42	0,46	0,48	0,57	0.48	0,42
0,34	0,17	0,09	0,14	0,32	0,15	1	1	1	0,99	1	1	0,96	0,95	0,94	0,9	0,94	0,96
0,37	0,21	0,12	0,17	0,35	0,18	1	1	1	0,99	1	1	0,97	0,96	0,95	0,91	0,95	0,97
0,31	0,14	0,05	0,1	0,29	0,11	1	1	1	1	0,99	1	0,95	0,93	0,93	0,88	0,93	0,95
0,24	0,07	0	0,03	0,22	0,04	0,99	0,99	1	1	0,98	1	0,92	0,91	0,9	0,85	0,9	0,92
0,41	0,25	0,16	0,21	0,39	0,22	1	1	0,99	0,98	1	0,99	0,98	0,97	0,96	0,93	0,96	0,98
0,31	0,14	0,05	0,1	0,29	0,11	1	1	1	1	0,99	1	0,95			0,88	0,93	0,95
0,59	0,44	0,36	0,41	0,57	0,42	0,96	0,97	0,95	0,92	0,92	0,95	1	1	1	0,99	1	1
0,63	0,48	0,41	0,45	0,61	0,46	0,95	0,96	0,93	0,91	0,97	0,93	1	1	1	0,99	1	1
0,65	0,5	0,42	0,47	0,63	0,48	0,94	0,95	0,93	0,9	0,96	0,93	1	1	1	0,99	1	1
0,72	0,59	0,52	0,56	0,7	0,57	0,9	0,91	0,88	0,85	0,93	0,88	0,99	0,99	0,99	1	0,99	0,99
0,65	0,5	0,42	0,47	0,63	0,48	0,94	0,95	0,93	0,9	0,96	0,93	1	1	1	0,99	1	1
0,6	0,45	0,37	0,41	0,57	0,42	0,96	0,97	0,95	0,92	0,98	0,95	1	1	1	0,99	1	1

Рис. 5. Матрица корреляций P, $k = 2$. Чёрным цветом отмечены блоки постов на определённую тему, красным — неожиданно высокая корреляция между постами второй и третьей групп

$\widehat{A} = U \cdot \Sigma \cdot V^T$ с заданным рангом k и выводила её на экран, вторая ставила в соответствие списку постов пользователя список пар (x, y) и выводила результаты в графическом виде.

Далее выбрана одна из страниц сети «ВКонтакте», на ней было создано 18 постов со следующей тематикой:

- 6 постов на тему кулинарии;
- 6 постов на тему органической химии;
- 6 постов на тему футбола.

Результат работы функции, ставящей в соответствие списку постов пользователя список пар (x, y), представлен на рисунке 4:

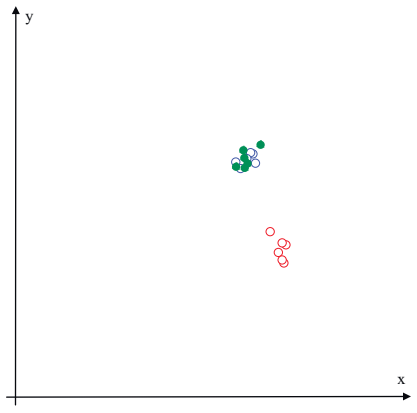

Рис. 4. Вектор-значения матрицы V^T, соответствующие постам. Значения по обеим осям изменяются в промежутке от 0 до 1. Красные круги — посты на тему футбола, зелёные — химии, синии — кулинарии

Из приведённого рисунка можно заметить, что посты на тему футбола формируют отдельную группу, в то время как посты на тему кулинарии и химии сильно слились.

На рисунке 5 приведены эксперименты по определению группы предпочтений по значениям матрицы корреляций.

Можно заметить, что в матрице корреляции наблюдается схожая картина — посты на тему кулинарии и химии сильно слились. Как оказалось, это было вызвано малым выбранным рангом k. Ниже приведены матрицы корреляций для $k = 3$ и $k = 4$.

Из рисунков 6 и 7 видно, что при

1	0,97	0,89	0,73	1	0,97	0,36	0,38	0,33	0,26	0,43	0,33	0,4	0,31	0,42	0,44	0,37	0,41
0,97	1	0,97	0,63	0,97	1	0,26	0,24	0,19	0,17	0,29	0,21	0,17	0,08	0,19	0,22	0,15	0,18
0,89	0,97	1	0,43	0,87	0,97	0,25	0,19	0,15	0,19	0,24	0,19	0	0	0	0	0	0
0,73	0,63	0,43	1	0,77	0,63	0	0,03	0	0	0,06	0	0,61	0,67	0,66	0,72	0,67	0,61
1	0,97	0,87	0,77	1	0,96	0,32	0,35	0,29	0,22	0,39	0,29	0,42	0,34	0,44	0,47	0,4	0,42
0,97	1	0,97	0,63	0,96	1	0,24	0,22	0,17	0,15	0,27	0,19	0,15	0,06	0,17	0,2	0,12	0,15
0,36	0,26	0,25	0	0,32	0,24	1	0,98	0,99	0,99	0,98	1	0,49	0,26	0,43	0,36	0,36	0,5
0,35	0,24	0,19	0,03	0,35	0,22	0,98	1	1	0,96	1	0,99	0,65	0,46	0,6	0,53	0,53	0,65
0,33	0,19	0,15	0	0,29	0,17	0,99	1	1	0,98	0,99	1	0,61	0,42	0,56	0,48	0,49	0,62
0,26	0,17	0,19	0	0,22	0,15	0,99	0,96	0,98	1	0,96	0,99	0,43	0,22	0,37	0,29	0,3	0,44
0,43	0,29	0,24	0,06	0,39	0,27	0,98	1	0,99	0,96	1	0,99	0,65	0,45	0,6	0,53	0,53	0,65
0,33	0,21	0,19	0	0,29	0,19	1	0,99	1	0,99	0,99	1	0,56	0,35	0,5	0,42	0,43	0,56
0,4	0,17	0	0,61	0,42	0,15	0,49	0,65	0,61	0,43	0,65	0,56	1	0,97	1	0,99	0,99	1
0,31	0,08	0	0,67	0,34	0,06	0,28	0,46	0,42	0,22	0,45	0,35	0,97	1	0,98	0,99	1	0,97
0,42	0,19	0	0,66	0,44	0,17	0,43	3,6	0,56	0,37	0,6	0,5	1	0,98	1	1	1	1
0,44	0,22	0	0,72	0,47	0,2	0,36	0,53	0,48	0,29	0,53	0,42	0,99	0,99	1	1	1	0,99
0,37	0,15	0	0,67	0,4	0,12	0,36	0,53	0,49	0,3	0,53	0,43	0,99	1	1	1	1	0,99
0,41	0,18	0	0,61	0,42	0,15	0,5	0,65	0,62	0,44	0,65	0,56	1	0, 97	1	0,99	0,99	1

Рис. 6. Матрица корреляций P, $k = 3$

1	0,81	0,8	0,19	0,97		0,07	0,16	0,17	0	0,2	0,11	0,59	0,52	0,53	0,6	0,55	0,58
0,81	1	0,96	0,52	0,9	0,99	0,24	0,23	0,19	0,16	0,26	0,2	0,15	0,07	0,18	0,19	0,13	0,16
0,8	0,96	1	0,26	0,85	0,97	0,19	0,15	0,13	0,13	0,2	0,15	0,02	0	0,01	0,04	0	0,02
0,19	0,52	0,28	1	0,39	0,45	0,13	0,18	0,06	0,06	0,19	0,09	0,11	0,16	0,29	0,22	0,19	0,14
0,97	0,9	0,85	0,39	1	0,93	0,14	0,22	0,21	0,05	0,27	0,16	0,53	0,46	0,52	0,56	0,5	0,52
0,86	0,99	0,97	0,45	0,93	1	0,18	0,18	0,15	0,1	0,23	0,15	0,18	0,11	0,2	0,23	0,16	0,19
0,07	0,24	0,19	0,14	0,14	0,16	1	0,98	0,96	1	0,98	0,99	0,18	0,01	0,23	0,09	0,1	0,21
0,16	0,23	0,15	0,16	0,22	0,16	0,98	1	0,99	0,96	1	0,99	0,37	0,22	0,43	0,3	0,3	0,4
0,17	0,19	0,13	0,06	0,21	0,15	0,96	0,99	1	0,95	0,99	0,99	0,42	0,26	0,45	0,32	0,33	0,44
0	0,16	0,13	0,06	0,05	0,1	1	0,96	0,95	1	0,96	0,98	0,12	0	0,17	0,03	0,04	0,15
0,2	0,28	0,2	0,19	0,27	0,23	0,98	1	0,99	0,96	1	0,99	0,38	0,23	0,44	0,31	0,31	0,41
0,11	0,2	0,15	0,09	0,16	0,15	0,99	0,99	0,99	0,98	0,99	1	0,3	0,14	0,34	0,21	0,22	0,32
0,59	0,15	0,02	0,11	0,53	0,18	0,18	0,37	0,42	0,12	0,38	0,3	1	0,98	0,98	0,99	0,99	1
0,52	0,07	0	0,16	0,46	0,11	0,01	0,22	0,26	0	0,23	0,14	0,98	1	0,97	0,99	1	0,98
0,53	0,18	0,01	0,29	0,52	0,2	0,23	0,43	0,45	0,17	0,44	0,34	0,98	0,97	1	0,99	0,99	0,99
0,6	0,19	0,04	0,22	0,56	0,23	0,09	0,3	0,32	0,03	0,31	0,21	0,99	0,99	0,99	1	1	0,99
0,55	0,13	0	0,19	0,5	0,16	0,1	0,3	0,33	0,04	0,31	0,22	0,99	1	0,99	1	1	0,99
0,56	0,16	0,02	0,14	0,52	0,19	0,21	0,4	0,44	0,15	0,41	0,32	1	3,98	0,99	0,99	0,99	1

Рис. 7. Матрица корреляций P, $k = 4$. Красным цветом показана слабая корреляция между четвёртым постом первой группы и остальными постами данной группы

увеличении числа k различия групп оказываются существеннее: при $k = 3$ можно разделить посты на группы, предположив, что связь между постами существует при коэффициенте корреляции между соответствующими постам векторами > 0.7, и отсутствует при меньшем.

Однако при дальнейшем увеличении числа k алгоритм может не находить связи постов, что видно на рисунке 7. При ранге $k = 5$, 6 подобных ошибок становится ещё больше. Было решено, что в случае трёх тематик для данных постов наилучшим рангом является $k = 3$.

Далее рассмотрены варианты улучшения существующего алгоритма разбиения постов по тематикам.

2.4. Методы улучшения существующего алгоритма

Методы улучшения алгоритма были подразделены на 3 группы.

- Методы уменьшения размерности матрицы A:
  - стемминг данных;
  - удаление шумовых слов.
- Изменение веса слова в матрице A:
  - подсчёт количества слов в посте;
  - TF-IDF.
- Изменение алгоритма расчёта матрицы корреляций P:
  - использование корреляций Спирмена вместо подсчёта косинуса угла между векторами.

2.5. Методы уменьшения размерности матрицы A

Перед более подробным рассмотрением каждого метода в отдельности стоит заметить, что данная группа методов весьма важна, так как позволяет получить не только более качественные результаты, но и существенно уменьшить время и объём памяти, необходимые для работы алгоритмов сингулярного разложения и построения матрицы корреляций.

2.6. Стемминг

Первая идея, которая возникает после осознания необходимости доработки алгоритма — стемминг строк, — нахождение основы для заданного исходного слова.

Существует немало алгоритмов стемминга:

- алгоритмы поиска — поиск флективной формы слова в таблице поиска;
- алгоритмы усечения окончаний, представляющие собой последовательность шагов по усечению окончания слова;
- алгоритмы усечения окончаний с дополнительными критериями, например, с заменой окончания;
- аффикс-стеммеры, которые также работают не только с суффиксами, но и с префиксами;

- прочие, такие как алгоритмы сопоставления, стохастические алгоритмы, гибридные алгоритмы и т. д.

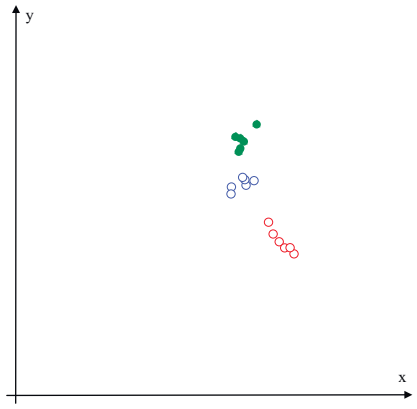

Рис. 8. Вектор-значения матрицы V^T при применении стемминга Портера

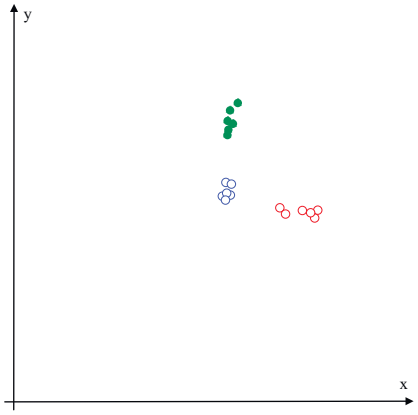

Рис. 9. Вектор-значения матрицы V^T при применении алгоритмов стемминга и удаления шумовых слов

Среди всего многообразия алгоритмов было решено остановиться на классическом Портеровском алгоритме усечения окончаний. Основная его идея заключается в том, что существует конечное количество словообразующих суффиксов. На каждом шаге алгоритма отсекается суффикс и оставшаяся часть проверяется на соответствие правилам языка. Если полученное слово удовлетворяет правилам — переход на следующий шаг, если нет — алгоритм выбирает другой суффикс для отсечения. Сначала отсекается максимальный формообразующий суффикс, затем — буква «и», потом словообразующие суффиксы, суффиксы превосходных форм, «ь» и одна из двух «н».

Среди плюсов алгоритма Портера — простота реализации, высокая скорость работы, отсутствие необходимости хранения каких-либо вспомогательных данных. К минусам же его работы можно отнести затруднённое получение правильной основы слова, отсутствие проверок на выпадающие и беглые гласные, отсутствие отсечений префиксов.

После применения алгоритма стемминга были получены следующие результаты, представленные на рисунке 8.

Из рисунка 8 видно, что посты формируют гораздо более чёткие группы, можно выделить не только группу постов про футбол, но и про кулинарию и органическую химию. Однако всё ещё может быть затруднительно чётко обозначить границы отдельных групп лишь по значениям матрицы V^T.

Стоит отметить, что при использовании стемминга Портера (рис. 10) группы постов хорошо различимы по матрице корреляций уже при ранге $k = 2$, однако, вновь наилучший результат достигается при использовании ранга $k = 3$.

2.7. Удаление шумовых слов

Следующая идея — удаление шумовых слов.

1	0,92	0,93	0,95	0,99	0,92	0,24	0,26	0,35	0,11	0,27	0,2	0,58	0,5	0,57	0,58	0,55	0,57
0,92	1	1	0,99	0,95	1	0,24	0,2	0,28	0,12	0,22	0,17	0,22	0,13	0,22	0,22	0,19	0,21
0,93	1	1	1	0,95	1	0,19	0,16	0,24	0,08	0,18	0,12	0,23	0,14	0,22	0,23	0,2	0,22
0,95	0,99	1	1	0,97	0,99	0,18	0,16	0,25	0,07	0,18	0,12	0,3	0,21	0,29	0,3	0,28	0,29
0,99	0,95	0,95	0,97	1	0,94	0,15	0,16	0,25	0,02	0,17	0,11	0,5	0,42	0,49	0,51	0,49	0,5
0,92	1	1	0,99	0,94	1	0,25	0,21	0,29	0,13	0,23	0,18	0,21	0,12	0,21	0,21	0,18	0,21
0,24	0,24	0,19	0,i8	0,15	0,25	1	0,99	0,98	0,99	0,99	0,99	0,17	0,13	0,22	0,13	0,07	0,19
0,26	0,2	0,16	0,16	0,16	0,21	0,99	1	1	0,98	1	1	0,3	0,26	0,34	0,26	0,2	0,31
0,35	0,26	0,24	0,25	0,25	0,29	0,98	1	1	0,95	1	0,99	0,36	0,32	0,41	0,33	0,26	0,37
0,11	0,12	0,08	0,07	0,02	0,13	0,99	0,98	0,95	1	0,98	0,99	0,09	0,06	0,14	0,05	0	0,1
0,27	0,22	0,18	0,18	0,17	0,23	0,99	1	1	0,98	1	1	0,28	0,24	0,33	0,24	0,18	0,29
0,2	0,17	0,12	0,12	0,11	0,18	0,99	1	0,99	0,99	1	1	0,24	0,21	0,29	0,2	0,14	0,25
0,58	0,22	0,23	0,3	0,5	0,21	0,17	0,3	0,36	0,09	0,28	0,24	1	1	1	1	0,99	1
0,5	0,13	0,14	0,21	0,42	0,12	0,13	0,26	0,32	0,06	0,24	0,21	1	1	0,99	1	1	1
0,57	0,22	0,22	0,29	0,49	0,21	0,22	0,34	0,41	0,14	0,33	0,29	1	0,99	1	1	0,99	1
0,55	0,22	0,23	0,3	0,51	0,21	0,13	0,26	0,33	0,05	0,24	0,2	1	1	1	1	1	1
0,55	0,19	0,2	0,28	0,49	0,18	0,07	0,2	0,26	0	0,18	0,14	0,99	1	0,99	1	1	0,99
0,57	0,21	0,22	0,29	0,5	0,21	0,19	0,31	0,37	0,1	0,29	0,25	1	1	1	1	0,99	1

Рис. 10. Матрица корреляции P при использовании стемминга Портера, $k = 3$

1	0,9	0,91	0,92	0,98	0,87	0	0,05	0,1	0	0,03	0,03	0,5	0,38	0,44	0,45	0,5	0,46
0,9	1	1	1	0,97	1	0	0,02	0,09	0	0	0,02	0,09	0	0,02	0,02	0,08	0,03
0,91	1	1	1	0,97	0,99	0	0	0,07	0	1	0	0,11	0	0,04	0,05	0,1	0,06
0,92	1	1	1	0,97	0,99	0,02	0,04	0,1	0,01	0,01	0,04	0,13	0	0,06	0,06	0,12	0,07
0,98	0,97	0,97	0,97	1	0,94	0	0	0,05	0	0	0	0,33	0,2	0,27	0,26	0,33	0,29
0,87	1	0,99	0,99	0,94	1	0,04	0,04	0,11	0,03	0,01	0,05	0,01	0	0	0	0	0
0	0	0	0,02	0	0,04	1	0,99	0,99	1	0,98	1	0,03	0,02	0,05	0	0	0
0,05	0,02	0	0,04	0	0,04	0,99	1	1	0,99	1	1	0,18	0,17	0,19	0,06	0,04	0,04
0,1	0,09	0,07	0,1	0,05	0,11	0,99	1	1	0,99	0,99	1	0,16	0,14	0,17	0,04	0,03	0,02
0	0	0	0,01	0	0,03	1	0,99	0,99	1	0,99	1	0,06	0,06	0,08	0	0	0
0,03	0	0	0,01	0	0,01	0,98	1	0,99	0,99	1	1	0,2	0,19	0,22	0,09	0,06	0,06
0,03	0,02	0	0,04	0	0,05	1	1	1	1	1	1	0,12	0,11	0,13	0	0	0
8,5	0,09	0,11	0,13	0,33	0,01	0,03	0,18	0,16	0,06	0,2	0,12	1	0,99	1	0,99	0,99	0,99
0,38	0	0	0	0,2	0	0,02	0,17	0,14	0,06	0,19	0,11	0,99	1	1	0,99	0,98	0,99
0,44	0,02	0,04	0,06	0,27	0	0,05	0,19	0,17	0,08	0,22	0,13	1	1	1	0,99	0,99	0,99
0,45	0,02	0,05	0,06	0,28	0	0	0,06	0,04	0	0,09	0	0,99	0,99	0,99	1	1	1
0,5	0,08	0,1	0,12	0,33	0	0	0,04	0,03	0	0,06	0	0,99	0,98	0,99	1	1	1
0,46	0,03	0,06	0,07	0,29	0	0	0,04	0,02	0	0,06	0	0 99	0, 99	0,99	1	1	1

Рис. 11. Матрица корреляции P при использовании алгоритмов стемминга и удаления шумовых слов, $k = 3$

Стоп-слова (шумовые слова) — слова, самостоятельно не несущие никакой смысловой нагрузки. Их можно разбить на две группы: общие и зависимые. К общим можно отнести предлоги, союзы, междометия, частицы и прочие служебные части речи. Зависимые стоп-слова зависят от контекста и достаточно трудны для учёта.

Зачастую шумовые слова можно выявить за счёт их небольшого размера: обычно их длина составляет не более 3-4 символов. Чтобы случайно не отсечь значимое слово, было решено ограничиться длиной слова в 2 символа после проведения стемминга. Однако есть и более длинные смысловые слова, например «почему», «который». Помимо проверки слова на длину было решено также фильтровать слова по словарю стоп-слов.

Видно, что на рисунке 9 посты формируют отчётливые группы, можно наблюдать чёткие границы между ними.

1	1	1	1	1	1	0	0	0,03	1	0	0,01	0,03	0	0,01	0,01	0	0
1	1	1	1	1	1	0,01	0,01	0,04	0	0,01	0,02	0,05	0,01	0,04	0,03	0,01	0,02
1	1	1	1	1	1	0	0,01	0,03	0	0	0,01	0,05	0,01	0,04	0,03	0,01	0,02
1	1	1	1	1	1	0	0	0,03	0	0	0,01	0,12	0,08	0,1	0,1	0,07	0,08
1	1	1	1	1	1	0	0	0,02	0	1	0	0,09	0,05	0,08	0,07	0,05	0,06
1	1	1	1	1	1	0,02	0,02	11	0,02	0	0,03	0,05	0,01	0,03	0,03	0	0,01
0	0,01	1	0	0	0,02	1	1	1	1	1	1	0,03	0,06	0,06	0	0	0
0	0,01	0,01	0	0	0,02	1	1	1	1	1	1	0,07	0,1	0,1	0,04	0,02	0,02
0,03	0,04	0,03	0,03	0,02	0,05	1	1	1	1	1	1	0,05	0,08	0,08	0,02	0	0
0	0	0	0	0	0,02	1	1	1	1	1	1	0,04	0,06	0,07	0,01	0	0
0	0,01	0	0	0	0,02	1	1	1	1	1	1	0,06	0,09	0,09	0,03	0,01	0,01
0,01	0,02	0,01	0,01	0	0,03	1	1	1	1	1	1	0,06	0,08	0,09	0,03	0,01	0,01
0,03	0,05	0,05	0,12	0,09	0,05	0,03	0,07	0,05	0,04	0,06	0,06	1	1	1	1	1	1
0	0,01	0,01	0,08	0,05	0,01	0,06	0,1	0,08	0,06	0,09	0,08	1	1	1	1	1	1
0,01	0,04	0,04	0,1	0,08	0,03	0,06	0,1	0,08	0,07	0,09	0,09	1	1	1	1	1	1
0,01	0,03	0,03	0,1	0,07	0,03	0	0,04	0,02	0,01	0,03	0,03	1	1	1	1	1	1
0	0,01	0,01	0,07	0,05	0	0	0,02	0	0	0,01	0,01	1	1	1	1	1	1
0	0,02	0,02	0,08	0,06	0,01	0	0,02	0	0	0,01	0,01	1	1	1	1	1	1

Рис. 13. Матрица корреляции P при подсчёте частоты слова в посте, $k = 3$

При сравнении рисунков 10 и 11 можно заметить улучшение в матрице, вносимое данным методом.

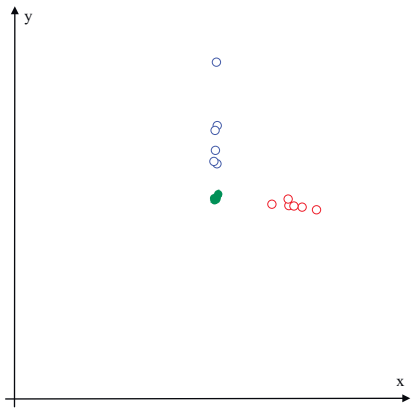

Рис. 12. Вектор-значения матрицы V^T при подсчёте частоты слова в посте, использовании алгоритмов стемминга и удалении шумовых слов

2.8. Методы изменения веса слова в матрице A

2.8.1. Подсчёт частоты слова в посте

При формировании матрицы A, отражающей присутствие слов в посте пользователя, может возникнуть вопрос, почему данная матрица состоит лишь из нулей и единиц. Возможно, качество алгоритма возрастёт, если использовать некую меру. Для начала было решено попробовать вместо единиц и нулей подсчитывать частоту слова в посте.

Видно (рисунки 12 и 13), что использование частоты слова в посте в качестве коэффициентов матрицы A приводит к существенному улучшению работы алгоритма — теперь корреляция для постов на одинаковую тему достигает единицы, для постов на разную — не превышает одной десятой.

2.8.2. TF-IDF

В качестве ещё одной меры было решено попробовать TF-IDF — статистическую меру, используемую для оценки важности слова в контексте документа, являющегося частью коллекции документов. Вес некоторого слова пропорционален количеству его употребления в документе, и обратно пропорционален

1	1	1	1	1	1	0	0,08	0,07	0	0,12	0,02	0,08	0,08	0,14	0,09	0	0
1	1	1	1	1	1	0	0,07	0,06	0	0,11	0,02	0,1	0,09	0,15	0,11	0	0
1	1	1	1	1	1	0	0,07	0,06	0	0,11	0,01	0,09	0,08	0,14	0,1	0	0
1	1	1	1	1	1	0	0,08	0,07	0	0,12	0,03	0,15	0,14	0,2	0,16	0,06	0
1	1	1	1	1	1	0	0,07	0,05	0	0,11	0,01	0,1	0,1	0,16	0,12	0,01	0,02
1	1	1	1	1	1	0	0,08	0,07	0	0,12	0,03	0,09	0,08	0,15	0,1	0	0
0	0	0	0	0	0	1	0,99	0,99	1	0,99	1	0,05	0,14	0,1	0,01	0	0
0,08	0,07	0,07	0,08	0,07	0,08	0,99	1	1	0,99	1	1	0,08	0,17	0,13	0,04	0	0,01
0,07	0,06	0,06	0,07	0,05	0,07	0,99	1	1	0,99	1	1	0,08	0,16	0,12	0,03	0	0
0	0	0	0	0	0	1	0,99	0,99	1	0,99	1	0,06	0,14	0,1	0,01	0	0
0,12	0,11	0,11	0,12	0,11	0,12	0,99	1	1	0,99	1	1	0,07	0,16	0,13	0,03	0	0,01
0,02	0,02	0,01	0,03	0,01	0,03	1	1	1	1	1	1	0,07	0,16	0,12	0,03	0	0
0,08	0,1	0,09	0,15	0,1	0,09	0,05	0,08	0,08	0,06	0,07	0,07	1	0,94	0,89	0,99	0,99	0.16
0,08	0,09	0,06	0,14	0,1	0,08	0,14	0,17	0,16	0,14	0,16	0,16	0,94	1	0,99	0,98	0,95	0,47
0,14	0,15	0,14	0,2	0,16	0,15	0,1	0,13	0,12	0,1	0,13	0,12	0,89	0,99	1	0,95	0,9	0,58
0,09	0,11	0,1	0,16	0,12	0,1	0,01	0,04	0,03	0,01	0,03	0,03	0,99	0,98	0,95	1	0,99	0,31
0	0	0	0,06	0,01	0	0	0	0	0	0	0	0,99	0,95	0,9	0,99	1	0,22
0	0	0	0	0,02	0	0	0,01	0	0	0,01	0	0,16	0,47	0.58	0,31	0,22	1

Рис. 14. Матрица корреляции P при использовании TF-IDF, $k = 4$

частоте употребления слова в других документах коллекции. Большой вес в TF-IDF получат слова с высокой частотой в пределах конкретного документа и с низкой частотой использования в других.

Меру TF-IDF можно представить в виде произведения двух сомножителей: $tf(t, d)$ и $idf(t, d)$, где

$$\mathrm{tf}(t, d) = \frac{n_t}{\sum_k n_k},$$

$$\mathrm{idf}(t, D) = \log \frac{|D|}{|\{\, d_i \in D \mid t \in d_i \,\}|},$$

t — слово, d — документ, D — совокупность документов. Во второй формуле не имеет значения выбор основания логарифма, так как его изменение приводит к изменению веса каждого слова на постоянный множитель.

При подсчёте матрицы корреляции выяснялось, что лучшие результаты при использовании меры TF-IDF получаются при $k = 4$.

Выяснилось, что получившиеся результаты всё же оказались хуже, чем при подсчёте частоты слова в посте, однако, существенно лучше, чем при простом учёте наличия слова в посте (рис. 14).

2.9. Использование корреляций Спирмена

Вместо подсчёта косинуса угла между двумя векторами для составления матрицы смежности было решено попробовать воспользоваться ранговыми корреляциями Спирмена.

Коэффициент корреляции Спирмена между двумя векторами вычисляется по формуле

$$\hat{\rho} = 1 - \frac{6 \sum (R_i - S_i)_t^2}{n(n^2 - 1)},$$

где R_i – ранг наблюдения x_i в векторе x, S_i – ранг наблюдения y_i в векторе y.

1	0,58	0,66	0,6	0,97	0,56	0	0	0	0	0	0	0,02	0	0	0,09	0,23	0,15
0,58	1	0,95	0,99	0,67	0,99	0,17	0	0	0	0	0	0	0	0	0	0	0
0,66	0,95	1	0,94	0,75	0,92	0	0	0	0	0	0	0	0	0	0	0	0
0,6	0,99	0,94	1	0,68	0,98	0,16	0	0	0	0	0	0	0	0	0	0	0
0,97	0,67	0,75	0,68	1	0,65	0	0	0	0	0	0	0	0	0	0	0,14	0,06
0,56	0,99	0,92	0,98	0,65	1	0,21	0	0,01	0,02	0	0	0	0	0	0	0	0
0	0,17	0	0,16	0	0,21	1	0,6	0,75	0,96	0,51	0,83	0	0	0	0	0	0
0	0	0	0	0	0	0,6	1	0,79	0,74	0,96	0,87	0,1	0,21	0,17	0	0	0
0	0	0	0	0	0,01	0,75	0,79	1	0,79	0,66	0,9	0	0	0	0	0	0
0	0	0	0	0	0,02	0,96	0,74	0,79	1	0,65	0,93	0	0	0	0	0	0
0	0	0	0	0	0	0,51	0,96	0,68	0,65	1	0,78	0,23	0,35	0,31	0,12	0	0,05
0	0	0	0	0	0	0,83	0,87	0,9	0,93	0,78	1	0	0	0	0	0	0
0,02	0	0	0	0	0	0	0,1	0	0	0,23	0	1	0,97	0,99	0,96	0,93	0,95
0	0	0	0	0	0	0	0,21	0	0	0,35	0	0,97	1	0,99	0,94	0,89	0,92
0	0	0	0	0	0	0	0,17	0	0	0,31	0	0,99	0,99	1	0,95	0,9	0,93
0,09	0	0	0	0	0	0	0	0	0	0,12	0	0,96	0,94	0,95	1	0,98	0,99
0,23	0	0	0	0,14	0	0	0	0	0	0	0	0,93	0,89	0,9	0,98	1	0,99
0,15	0	0	0	0,06	0	0	0	0	0	0,05	0	0,95	0,92	0,93	0,99	0,99	1

Рис. 15. Матрица корреляций P, составленная для матрицы наличия слова в посте при использовании корреляций Спирмена, $k = 3$

Матрица корреляций с использованием корреляций Спирмена была составлена для трёх матриц: матрицы из 1 и 0, матрицы встречаемости слова в посте и матрицы из мер TF-IDF.

По получившимся матрицам корреляций нельзя однозначно сказать, являются ли корреляции Спирмена лучшим инструментом, чем подсчёт косинуса угла между векторам: они хорошо фильтруют «мусорные» связи, но также могут уменьшать и реальные связи (рисунки 15, 16 и 17).

2.10. Определение наличия у пользователя заданного предпочтения

Видно, что программа умеет разбивать предпочтения пользователей на группы, но как понять, что конкретная группа относится к данному предпочтению? Самый очевидный вариант — хранить некоторый объём текстовых данных, точно относящийся к этой теме.

Для проверки работоспособности предложенной теории было взято 5 текстов на футбольную тему, отличных от тех, что размещались в постах пользователя. Тексты были «подмешаны» в результат выполнения API-запроса `wall.get`.

Можно заметить (рисунки 19 и 18), что подмешанные тексты теперь составляют одну группу с постами пользователя на данную тему.

При написании функции проверки наличия у пользователя заданного предпочтения рассматривались только первые n-строк матрицы корреляции, где n — количество подмешанных текстов. Если более половины значений столбца, соответствующего посту T, были больше значения, говорящего о существовании связи между ним и проверочным словом (по умолчанию 0,7), то пост относился к данной тематике. Если доля постов пользователя/группы, относящихся к данной тематике, превышала заданное значение (по умолчанию 0,25), то принималось, что пользователь интересуется этой тематикой.

1	0,92	0,95	0,77	0,78	0,91	0	0	0	0	0	0	0	0	0	0	0	0
0,92	1	0,92	0,76	0,77	0,96	0	0	0	0	0	0	0	0	0	0	0	0
0,95	0,92	1	0,87	0,88	0,85	0	0	0	0	0	0	0	0	0	0	0	0
0,77	0,76	0,87	1	0,99	0,69	0	0	0	0	0	0	0	0	0	0	0	0,03
0,78	0,77	0,66	0,99	1	0,69	0	0	0	0	0	0	0	0	0	0	0	0,01
0,91	0,96	0,85	0,69	0,69	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0,95	0,87	0,99	0,96	0,94	0	0,2	0,15	0	0	0
0	0	0	0	0	0	0,95	1	0,86	0,96	0,99	0,97	0,13	0,38	0,34	0	0	0
0	0	0	0	0	0	0,87	0,86	1	0,85	0,84	0,93	0	0,04	0,06	0	0	0
0	0	0	0	0	0	0,99	0,96	0,85	1	0,96	0,95	0	0,26	0,2	0	0	0
0	0	0	0	0	0	0,96	0,99	0,84	0,98	1	0,97	0,1	0,36	0,31	0	0	0
0	0	0	0	0	0	0,94	0,97	0,93	0,95	0,97	1	0,05	0,25	0,24	0	0	0
0	0	0	0	0	0	0	0,13	0	0	0,1	0,05	1	0,9	0,95	0,91	0,8	0,82
0	0	0	0	0	0	0,2	0,36	0,04	0,26	0,36	0,25	0,9	1	0,97	0,83	0,77	0,75
0	0	0	0	0	0	0,15	0,34	0,06	0,2	0,31	0,24	0,95	0,97	1	0,85	0,75	0,75
0	0	0	0	0	0	0	0	0	0	0	0	0,91	0,83	0,85	1	0,95	0,96
0	0	0	0	0	0	0	0	0	0	0	0	0,8	0,77	0,75	0,95	1	0,99
0	0	0	0,03	0,01	0	0	0	0	0	0	0	0,82	0,75	0,75	0,96	0,99	1

Рис. 16. Матрица корреляций P, составленная для матрицы встречаемостей слов в постах при использовании корреляций Спирмена, $k = 3$

1	0,99	1	0,97	0,98	1	0	0	0	0	0	0	0	0	0	0,11	0	0
0,99	1	1	0,98	0,99	0,99	0	0	0	0	0	0	0	0	0	0,15	0	0
1	1	1	0,97	0,99	1	0	0	0	0	0	0	0	0	0	0,15	0	0
0,97	0,98	0,97	1	0,97	0,97	0	0	0	0	0	0	0	0	0	0,22	0	0
0,98	0,99	0,99	0,97	1	0,98	0	0	0	0	0	0	0	0	0	0,2	0	0
1	0,99	1	0,97	0,98	1	0	0	0	0	0	0	0	0	0	0,11	0	0
0	0	0	0	0	0	1	0,58	0,62	1	0,52	0,79	0,03	0,28	0	0	0,16	0,09
0	0	0	0	0	0	0,58	1	0,98	0,58	0,97	0,89	0,07	0,19	0,05	0	0	0
0	0	0	0	0	0	0,62	0,98	1	0,61	0,94	0,93	0,14	0,19	0,04	0	0	0
0	0	0	0	0	0	1	0,58	0,61	1	0,51	0,79	0,02	0,32	0,01	0	0,15	0,11
0	0	0	0	0	0	0,52	0,97	0,94	0,51	1	0,81	0	0,04	0	0	0	0
0	0	0	0	0	0	0,79	0,89	0,93	0,79	0,81	1	0,25	0,35	0,13	0	0	0
0	0	0	0	0	0	0,03	0,07	0,14	0,02	0	0,25	1	0,72	0,8	0,68	0,42	0,05
0	0	0	0	0	0	0,28	0,19	0,19	0,32	0,04	0,35	0,72	1	0,93	0,55	0,3	0,25
0	0	0	0	0	0	0	0,05	0,04	0,01	0	0,13	0,8	0,93	1	0,75	0,22	0,28
0,11	0,15	0,15	0,22	0,2	0,11	0	0	0	0	0	0	0,68	0,55	0,75	1	0,46	0,3
0	0	0	0	0	0	0,16	0	0	0,15	0	0	0,42	0,3	0,22	0,46	1	0,19
0	0	0	0	0	0	0,09	0	0	0,11	0	0	0,05	0,25	0,28	0,3	0,19	1

Рис. 17. Матрица корреляций P для матрицы из мер TF-IDF при использовании корреляций Спирмена, $k = 4$

1	1	1	0,99	1	1	1	1	1	1	1	0	0,01	0,03	0	0	0,01	0,06	0,03	0,04	0,05	0,02	0,04
1	1	1	0,99	1	1	1	1	1	1	1	0	0	0,03	0	0	0,01	0,03	0,01	0,03	0,03	0	0,02
1	1	1	0,99	1	1	1	1	1	1	1	0	0,01	0,04	0	0	0,02	0,05	0,02	0,03	0,03	0,01	0,02
0,99	0,99	0,99	1	0,98	0,99	0,99	0,99	1	0,99	0,99	0,12	0,14	0,16	0,12	0,12	0,14	0,11	0,08	0,09	0,09	0,06	0,08
1	1	1	0,98	1	1	1	1	0,99	1	1	0	0	0	0	0	0	0,02	0	0	0	0	0
1	1	1	0,99	1	1	1	1	0,99	1	1	0,01	0,02	0,04	0	0	0,02	0,03	0	0,01	0,01	0	0
1	1	1	0,99	1	1	1	1	1	1	1	0	0,02	0,04	0	0	0,02	0,05	0,02	0,03	0,03	0	0,02
1	1	1	0,99	1	1	1	1	1	1	1	0	0,01	0,03	0	0	0,01	0,05	0,02	0,03	0,03	0	0,02
1	1	1	1	0,99	0,99	1	1	1	1	1	0,03	0,04	0,07	0,02	0,03	0,05	0,13	0,1	0,11	0,11	0,08	0,1
1	1	1	0,99	1	1	1	1	1	1	1	0,01	0,02	0,05	0,01	0,01	0,03	0,19	0,06	0,07	0,07	0,05	0,06
1	1	1	0,99	1	2	1	1	1	1	1	0,01	0,02	0,05	0,01	0,01	0,03	0,04	0,01	0,02	0,03	0	0,01
0	0	0	0	0	0,01	0	0	0,03	0,01	0,01	1	1	1	1	1	1	0,03	0,05	0,06	0	0	0
0,01	0	0,01	0,14	0	0,02	0,02	0,01	0,04	0,02	0,02	1	1	1	1	1	1	0,07	0,09	0,1	0,04	0,01	0,02
0,03	0,03	0,04	0,16	0	0,04	0,04	0,03	0,07	0,05	0,05	1	1	1	1	1	1	0,05	0,07	0,08	0,02	0	0
0	0	0	0,12	0	0	0	0	0,02	0,01	0,01	1	1	1	1	1	1	0,03	0,06	0,07	0,01	0	0
0	0	0	0,12	0	0	0	0	0,03	0,01	0,01	1	1	1	1	1	1	0,06	0,05	0,09	0,03	0,01	0,01
0,01	0,01	0,02	0,14	0	0,02	0,02	0,01	0,05	0,03	0,03	1	1	1	1	1	1	0,06	0,08	0,09	0,03	0	0,01
0,06	0,05	0,05	0,11	0,02	0,03	0,05	0,05	0,13	0,09	0,04	0,03	0,01	0,05	0,03	0,06	0,01	1	1	1	1	1	1
0,03	0,01	0,02	0,08	0	0	0,02	0,02	0,1	0,06	0,01	0,05	0,09	0,07	0,06	0,08	0,08	1	1	1	1	1	1
0,04	0,01	0,03	0,09	0	0,01	0,03	0,03	0,11	0,07	0,02	0,06	0,1	0,08	0,07	0,09	0,09	1	1	1	1	1	1
0,05	0,03	0,03	0,09	0	0,01	0,03	0,03	0,11	0,07	0,03	0	0,04	0,02	0,01	0,03	0,03	1	1	1	1	1	1
0,02	0	0,01	0,06	0	0	0	0	0,08	0,05	0	0	0,01	0	0	0,01	0	1	1	1	1	1	1
0,01	0,02	0	0,08	0	0	0,02	0,02	0,1	0,06	0,01	0	0,02	0	0	0,01	0,01	1	1	1	1	1	1

Рис. 18. Матрица корреляций при использовании матрицы частоты слова в посте. Первые пять столбцов и строк соответствуют подмешанным текстам

Из приведённого примера видно, что, храня некий набор данных, гарантированно относящийся к некоей тематике, можно проверять существование у пользователя постов на данную тематику и выявлять его заинтересованность ею.

2.11. Выбор ранга k для групп с примерно равным количеством постов

Одним из недостатков метода латентно-семантического анализа является отсутствие четкого правила выбора ранга k. У различных авторов на этот счёт разное мнение: так, ранее упоминалось, что, например, в работе А. Д. Хомоненко и С.А. Краснова показывается, что для разделения двух групп наилучшим рангом является $k = 2$ [4]. Было решено подобрать коэффициент эмпирически.

В ходе экспериментов обнаружено, что данный коэффициент зависит от количества различных тематик в постах пользователя/группы: так, для двух различных тематик наилучший $k = 2$, для трёх — 3, для четырёх — 4. Сложно сказать, можно ли экстраполировать полученные результаты на большее число тематик, но можно предположить, что при их малом количестве и при примерно одинаковом количестве постов в каждой группе, наилучший ранг k равен количеству различных тематик.

В случае же разного количества постов в различных тематиках было замечено, что тематики, для которых количество текстов невелико, оказывают значительно меньшее влияние на выбор k.

2.12. Проверка работы программы на реальных данных

Было решено посмотреть, как поведёт себя алгоритм поиска не на синтезированных данных, а на реальных. Для эксперимента выбрана группа, занимающаяся выкладыванием новостей на тему дорожно-транспортных происшествий,

откуда было взято 50 постов, и несколько прочих групп, откуда было взято ещё 50. Результаты запросов к VK API смешивались, и программе предстояло отличить посты, связанные с ДТП, от прочих.

Для выполнения этого задания в запрос было также «подмешано» 5 проверочных текстов на тему ДТП.

На данном этапе лучший результат показала матрица, состоящая из TF-IDF мер. Так, для вышеприведённого случая она дала только 2 ложноположительных результата и ни одного ложноотрицательного. Результаты работы программы можно ещё улучшить, увеличив количество «подмешанных» текстов. Однако на данном этапе вновь возникла проблема выбора ранга k. Был найден наилучший k для 15, 25, 35, 45, 55, 75 и 105 запросов (где в каждом случае 5 запросов являлись «примесью», а остальные — в равных частях посты на тему ДТП и др. посты) и составлен график

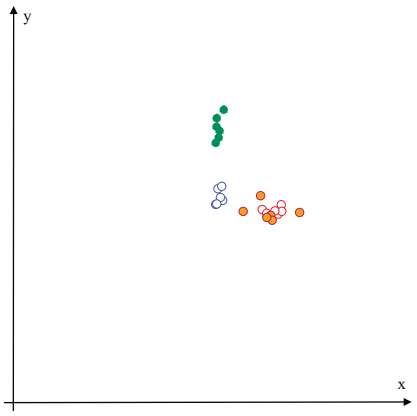

Рис. 19. Поиск постов с заданной тематикой с использованием матрицы из 0 и 1

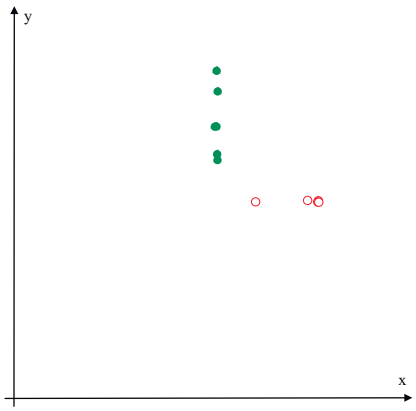

Рис. 20. Поиск по видеофайлам

По полученным результатам было замечено, что наименьшая ошибка аппроксимации возникает, если предположить, что зависимость наилучшего k от количества постов — логарифмическая (рисунок 21).

Данную зависимость можно выразить уравнением: $y = 7.06535 \ln(0.10906x)$, или приближённо: $y = 7 \ln(0.1x)$.

3. Разработка части приложения, определяющей предпочтения пользователей по размещённым аудио- и видеоматериалам

При разработке программы было выдвинуто предположение, что определять предпочтения пользователя можно не только по постам, но также по видео- и аудиоматериалам. В ходе разработки данной части программы использовались следующие API методы:

- `audio.get` — возвращает список аудиозаписей пользователя;
- `video.get` — возвращает информацию о видеозаписях.

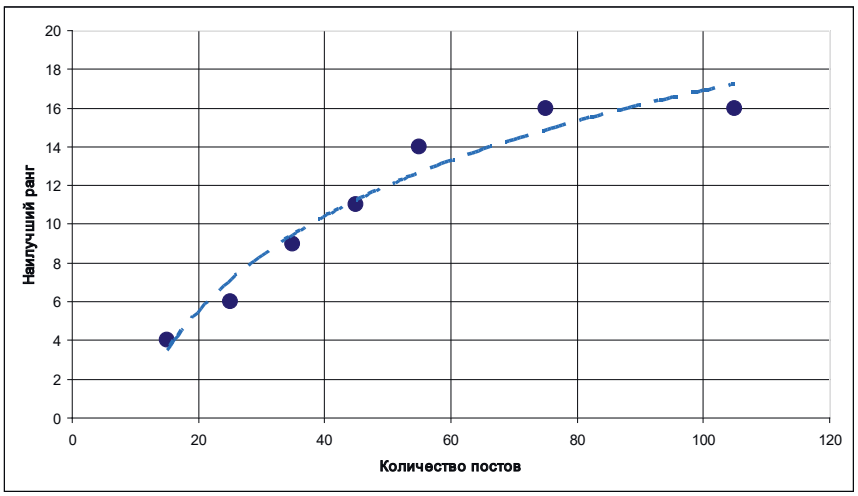

Рис. 21. Зависимость наилучшего ранга k от количества постов. Пунктирной линией показана предполагаемая логарифмическая зависимость

Полученные результаты приводились к соответствующим классам Video и Audio, откуда затем извлекались строки title — название и description — описание.

На основе вышеперечисленных API-методов и приведённой логики были созданы следующие функции:

- `Set<Video> getVideoByUserId(Integer owner_id)` — выполняет поиск видеозаписей пользователя или группы;
- `Set<Audio> getAudioByUserId(Integer owner_id)` — выполняет поиск аудиозаписей пользователя или группы.

Далее проводилось сингулярное разложение, стемминг и удаление шумовых слов по тем же алгоритмам, что и для записей на стене пользователя.

Работа функции была протестирована на следующем наборе данных:

- 5 видео на тему кулинарии;
- 5 видео на тему футбола.

Как видно из рисунка 20, результат выполнения функции чётко разбивается на 2 группы.

Стоит отметить, что данный метод поиска предпочтений пользователя не является эффективным из-за коротких названий видео- и аудиофайлов и очень редкого наличия описания к ним. Возможно, в данном случае намного эффективнее применять поиск по ключевым интересующим словам.

Также стоит обратить внимание на то, что с 16 декабря 2016 года руководство «ВКонтакте» отключило доступ к публичному музыкальному API сторонним приложениям, поэтому поиск аудиопредпочтений конкретного пользователя данным, ровно как и другими методами, становится затруднительным, а то и вовсе невозможным.

Заключение

В ходе данной работы были показаны этапы разработки приложения, осуществляющего поиск групп и пользователей в социальной сети «ВКонтакте», размещающих запрещённые материалы, а также проанализированы различные алгоритмы такого поиска, их эффективность. Наилучшим методом поиска был признан метод латентно-семантического анализа с использованием алгоритмов стемминга, фильтрации стоп-слов, TF-IDF мер и корреляций Спирмена, при выборе ранга матрицы Σ в сингулярном разложении $k = 7\ln(0.1x)$, где x — количество постов.

Также хотелось бы отметить, что работа данной программы в социальной сети «ВКонтакте» часто бывает затруднена: нередко у пользователей отсутствуют посты или присутствуют в небольшом количестве, посты многих пользователей состоят из фотографий с краткой подписью или видеозаписей, из которых сложно извлечь сколь угодно значимую информацию, часто текст накладывают поверх картинки в графическом редакторе и т. п. Особый интерес для проведения поисков запрещённых материалов представляет социальная сеть «Твиттер», так как её особенности располагают к получению хороших результатов работы программы: твит ограничен 140 символами, что ограничивает количество слов, а, следовательно, и размер матрицы A; большинство сообщений содержат текст, а не ссылки или изображения.

Настройка программы на поиск суидцидальных групп «Синий кит» осуществляется добавлением в классификацию текстов заданий этих групп, находящихся в открытом доступе на новостных ресурсах[1] [2].

Благодарности

Выражаем искреннюю признательность Надежде Фёдоровне Богаченко и Ольге Анатольевне Вишняковой за ценные советы, позволившие улучшить изложение результатов.

Литература

1. Социально-сетевая жизнь [Электронный ресурс]. 2015. URL: `http://romir.ru/studies/670_1432155600/` (дата обращения: 21.01.2017).
2. Антиэкстремизм в виртуальной России в 2014–2015 годы [Электронный ресурс]. 2016. URL: `http://www.sova-center.ru/racism-xenophobia/publications/2016/06/d34913/` (дата обращения: 02.02.2017).

[1] Игра синий кит все задания с 1 по 50 — полная информация про синего кита (подробный материал) [Электронный ресурс] // POLIKSAL. URL: `http://poliksal.ru/novosti-v-mire/180727-igra-siniy-kit-vse-zadaniya-s-1-po-50-polnaya-informaciya-pro-sinego-kita-podrobnyy-material.html` (дата обращения: 28.08.2017)

[2] Появился список заданий из суицидальных игр для подростков [Электронный ресурс] // СУПЕРОМСК. URL: `http://superomsk.ru/news/44483-poyavilsya_spisok_zadaniy_iz_suitsidalnx_igr_dlya_/` (дата обращения: 28.08.2017)

3. Отключение публичного API для аудио [Электронный ресурс]. 2017. URL: `https://vk.com/dev/audio_api/` (дата обращения: 22.01.2017).
4. Хомоненко А.Д., Краснов С.А. Применение метода латентно-семантического анализа для автоматической рубрикации документов // Известия Петербургского университета путей сообщения. 2012. № 2(31). С. 124–132.
5. Латентно-семантический анализ [Электронный ресурс]. 2010. URL: `https://habrahabr.ru/post/110078/` (дата обращения: 22.01.2017).
6. Алгоритм LSA для поиска похожих документов [Электронный ресурс]. 2014. URL: `http://netpeak.net/ru/blog/algoritm-lsa-dlya-poiska-pohozhih-dokumentov/` (дата обращения: 31.01.2017).
7. Осипова Ю.А., Лавров Д.Н. Применение кластерного анализа методом k-средних для классификации текстов научной направленности // Математические структуры и моделирование. № 3(43). 2017.

IDENTIFICATION OF SUICIDAL GROUPS AND VIOLATORS OF COPYRIGHT IN SOCIAL NETWORKS

A.V. Kostylev
Student, e-mail: kostart_94@mail.ru
D.N. Lavrov
Ph.D. (Eng.), Associate Professor, e-mail: dmitry.lavrov72@gmail.com
A.K. Guts
Dr.Sc. (Phys.-Math.), Professor, e-mail: aguts@mail.ru

Dostoevsky Omsk State University

Abstract. In this paper, the task is to develop software that allows you to automate the search for groups and users in social networks that host suicidal materials and materials that violate copyright in the network. As a social network for exploring the possibility of such a search, the social network “VKontakte” — the largest social network of the CIS was chosen. The latent semantic analysis algorithm was based on the developed software. The main variants of the algorithm and the most adequately working on real data version is selected.

Keywords: Classification of texts, latent-semantic analysis, copyright, offenses, social networks, suicidal groups.

Дата поступления в редакцию: 29.06.2017