004.912, 004.021

Орлов А. С., Севрюков С. Ю.

Реализация алгоритма поиска по субтитрам YouTube

1. Введение. В наши дни платформа хранения, доставки и показа видео YouTube пользуется огромной популярностью. Она обладает большим спектром возможностей, что позволяет упрощать процессы работы с видео. Одним из таких процессов является поиск по тексту видео. YouTube автоматически генерирует субтитры к видео, что предоставляет возможность реализации алгоритма автоматизации этого процесса. Это позволит повысить производительность организаций, активно использующим YouTube в работе.

Примером такой организации может быть центр развития электронных образовательных ресурсов Санкт-Петербургского государственного университета, который занимается созданием и выпуском онлайн-курсов. В процесс создания курсов входит создание видеолекций, которые до выпуска курса просматриваются авторами для идентификации ошибок. В курсах, создаваемых для публикации на платформе Coursera, в среднем 4-6 модулей; для платформы Открытое Образование — 10-12 модулей. Каждый модуль включает в себя около 1,5 часа видео-лекций. На самый маленький курс приходится уже 6 часов видео, однако эта цифра увеличивается в несколько раз на практике. Вдобавок к видео, которые в конечном итоге попадают на платформу, ещё есть черновые видео, которые не проходят проверку и их решают исключить из публикации. Для упрощения ориентации в таком массиве данных было решено разработать описываемый алгоритм.

2. Постановка задачи. В процессе решения задачи и реализации алгоритма были сформулированы следующие начальные условия:

1. Массив видео, опубликованных на YouTube, по которым будет производиться поиск.

Орлов Антон Сергеевич – студент, Санкт-Петербургский государственный университет; e-mail: ion6431@gmail.com, тел.: +7(999)229-50-25

Севрюков Сергей Юрьевич – старший преподаватель, Санкт-Петербургский государственный университет; e-mail: email@email.ru, тел.: +7(000)000-00-00

2. Данные учётной записи Google.
3. Наличие речи в видео. Качество роликов должно быть достаточным, чтобы генерируемые субтитры совпадали с речью людей на видео [2].

Цель заключается в том, чтобы реализовать алгоритм, на вход которому подаётся несколько видео, данные учётной записи Google и поисковый запрос. Результатом работы алгоритма служит список ссылок на видео, отсортированных в порядке уменьшения релевантности.

3. Получение текста видео. Извлечение текста будет производиться с помощью использования YouTube Data API. В него входит метод `captions.download`. Исходя из документации этого метода, он требует авторизации, следовательно её необходимо реализовать.

3.1. Авторизация. YouTube API поддерживает два способа авторизации: OAuth 2.0 и API Key. Второй способ не предусматривает получение прав, поэтому в дальнейшем он рассматриваться не будет. Использование метода запрашивает наличие хотя бы одного из двух разрешений: `youtubepartner` или `youtube.force-ssl`. Они указываются в параметре `scope`. Оставшиеся действия описаны в руководстве API [3]. После прохождения авторизации становится возможным использование метода получения субтитров.

3.2. Получение субтитров. Метод `captions.download` принимает в качестве обязательного параметра `id` дорожки субтитров. Её можно получить с помощью метода `captions.list`. Google накладывает условие, согласно которому субтитры возможно получить только из видео, загруженных пользователем. Данное условие может быть снято, если пользователь разрешает взаимодействие 3-м лицам с его видео. Организации, которым требуется поиск по собственным видео, это условие не создаст препятствий.

4. Индексация. Рассмотрим процесс создания обратного индекса, в котором каждому слову, предварительно приведённому к начальной форме, будет сопоставляться список ID видео, в которых оно встретилось, а каждому видео список чисел, обозначающих секунду, на которых прозвучало это слово.

Полученный текст имеет следующий формат: в первой строке находится указание временного промежутка, во второй — текст, который соответствует этому промежутку, и пустая третья строка. Все

дальнейшие строки подчиняются этому правилу. Это позволяет создать цикл с шагом 3, внутри которого будет происходить наполнение индекса. Первая строка обрабатывается с помощью регулярного выражения: `(^|\n)(\d{1,2}):(\d{2}):(\d{2})\.(\d{3})`. С помощью этого можно получить значение секунд, которое соответствует началу временной промежутка видео с нужным словом. Вторая строка делится пробелами на слова, каждое из которых приводится к нормальной форме, записывается как ключ индекса, а к значению добавляется видео и соответствующие секунды.

5. Реализация метода поиска. Для того, чтобы найти видео, которое удовлетворяет поисковому запросу, необходимо вычислить меру схожести запроса и всех видео, в которых присутствуют слова из запроса. В качестве такой меры можно выбрать cosine similarity. Подробный разбор алгоритма описан в книге Introduction to Information Retrieval [1]. Результатом работы метода является список видео с сопоставленными им cosine similarity. Отсортировав их по убыванию будет получен искомый результат.

6. Использование ElasticSearch. Рассмотрим реализацию поиска с помощью ElasticSearch. Чтобы проиндексировать субтитры, полученные с помощью вышеизложенных методов, необходимо выделить текст без отметок о времени. Они находятся в каждой третьей строке, начиная с первой. Также каждая третья строка является пустой, поэтому их можно игнорировать. Для повышения качества поиска возможно индексировать каждую строку отдельно, чтобы пользователь получал не список видео, а список фрагментов, в которых употребляются слова из его запроса. Это может быть полезно для поиска конкретной темы, которая обсуждалась в видео. ElasticSearch является RESTful сервисом, из-за чего для каждой строки необходимо делать два запроса. Это сильно понижает производительность системы.

7. Тесты. Ниже приведена информация о затратах времени для индексации с помощью рассмотренных методов. Все видео являются лекциями, поэтому количество моментов, в которых нет речи, минимально.

Таблица 1. Сравнение

Длительность видео	Inverted Indexing	ElasticSearch
7:52	656мс.	17,9с.
1:58:35	1358мс.	3м. 6с.
2:55:02	1287мс.	5м. 4с.
35:34	919мс.	2м. 2с.
53:19	989мс.	3м. 46с.

Таким образом, ElasticSearch имеет гораздо более низкую скорость работы, но описываемый алгоритм имеет приложения в таких ситуациях, когда скорость не влияет на продуктивность системы и кампании-заказчику выгоднее работать с готовыми решениями.

8. Вывод. Данный результат может быть интегрирован в информационную систему, в которой присутствуют компоненты взаимодействия с материалами, опубликованными на YouTube, для повышения эффективности этой ИС. Эффективность алгоритма может быть повышена за счёт модификации алгоритма поиска. Возможной модификацией может быть поиск релевантных фрагментов видео. Алгоритм авторизации может быть модифицирован таким образом, чтобы пользователю не нужно было проходить авторизацию самостоятельно.

Литература

1. Manning C. D., Raghavan P., Schutze H. Introduction to Information Retrieval. 2008. 151 p.

2. Как создать субтитры автоматически — Справка — YouTube [Электронный ресурс]: `URL:https://support.google.com/youtube/answer/6373554?hl=ru` (дата обращения: 14.03.18).

3. Implementing OAuth 2.0 Authorization | YouTube API | Google Developers [Электронный ресурс]: `URL:https://developers.google.com/youtube/v3/guides/authentication` (дата обращения 14.03.18)