Адаптация Google Cloud Speech-to-text API для автоматической транскрибации веб-конференций в реальном времени

А.С. Каменская

ФГБОУ ВПО НГТУ, Новосибирск, Россия

Аннотация. В работе рассмотрена проблема настройки Google Cloud Speech-to-text API для решения задачи транскрибации веб-конференций в реальном времени. Обозначены особенности задачи и возможные трудности, возникающие в ходе работы, а также описаны пути их решения. Также перечислены способы захвата аудиопотоков участников веб-конференции в реальном времени перед потоковым распознаванием речи.

Ключевые слова: WebRTC, распознавание речи, веб-конференция.

ВВЕДЕНИЕ

В настоящее время мультимедиа-конференции широко применяются для коммуникаций в сферах образования, бизнеса и индустрии развлечений. В последние годы особую популярность приобрела технология *WebRTC* (англ. *Web Real-time Communications* – веб-коммуникации в реальном времени). *WebRTC* представляет собой набор стандартов, протоколов и *JavaScript API*, совокупность которых позволяет разрабатывать браузерные решения в области мультимедиа-конференций [1]. Основное достоинство данной технологии с позиции пользователя – возможность участвовать в конференциях без необходимости установки какого-либо дополнительного ПО, кроме веб-браузера.

Транскрибация – это процесс расшифровки речи из аудио- или видеозаписи в текст. Как правило, это выполняется вручную, однако развитие технологий распознавания речи постепенно открывает возможности решения такой задачи с помощью компьютеров. Конференцию можно записать, а затем загрузить на какой-либо сервис, поддерживающий распознавание речи, для генерации субтитров. Так, автоматические субтитры от *Youtube* получаются приемлемыми при должном качестве звука в записи. Однако транскрибация в реальном времени – гораздо более сложная задача. На сегодняшний день отсутствуют как детальные описания её решения, так и какая-либо литература по теме. Задача, тем не менее, является актуальной. Например, в *Skype* такая функция появилась лишь в конце 2018 года [2].

Цель данной работы – исследовать возможность применения одного из самых популярных API распознавания речи – Google *Cloud Speech-to-text API* – для решения задачи транскрибации веб-конференций в реальном времени.

1. ОСОБЕННОСТИ ЗАДАЧИ

Основные сложности транскрибации веб-конференций в реальном времени заключаются в необходимости независимого захвата аудиопотоков каждого участника конференции и выполнении потокового распознавания речи. Аудиопоток каждого участника необходимо транскрибировать отдельно, поскольку смешение аудиопотоков в один сильно скажется на качестве распознавания речи. При этом потенциально возможен некорректный порядок сказанного разными участниками в итоговой расшифровке – проблема будет разобрана в данной работе позднее.

Для начала, необходимо найти способ получить доступ к медиа-данным участников конференции в реальном времени. Возможны следующие подходы:

1. Прямой доступ к медиа-устройствам пользователей из браузера. Такое решение легко реализовать, кроме того, данные скорее всего будут закодированы в формат без потерь (*LINEAR*16). Однако скорее всего потребуется промежуточный сервер, поскольку *JavaScript* совместим не со всеми технологиями потокового распознавания речи. Кроме того, такое решение будет привязано к конкретному веб-приложению для конференций.
2. Реализация приложения-транскрибера как "молчащего участника". Для этого необходимо интегрировать *SIP*-клиент в *WebRTC*-приложение [3]. В конференции при этом должна быть кнопка для приглашения транскрибера. По нажатии происходит звонок транскриберу, он подключается к конференции и получает доступ к аудиоданным участников. Такой подход используется в решении для транскрибации конференций в приложении *Jitsi Meet – Jigasi* [4].
3. Воспользоваться возможностями *WebRTC*-сервера, без которого не обходятся приложения, реализующие групповые звонки. *WebRTC*-сервер необходим для управления движением медиа-потоков между участниками

конференции. Обычно у таких серверов есть возможность настроить дополнительную передачу данных по протоколу RTP (англ. *Real-time Transport Protocol* – протокол передачи данных в реальном времени) [5].

В данной работе используется третий подход. Рассмотрим структуру системы с медиа-сервером и внешним транскрибирующим приложением (*Рис*. 1).

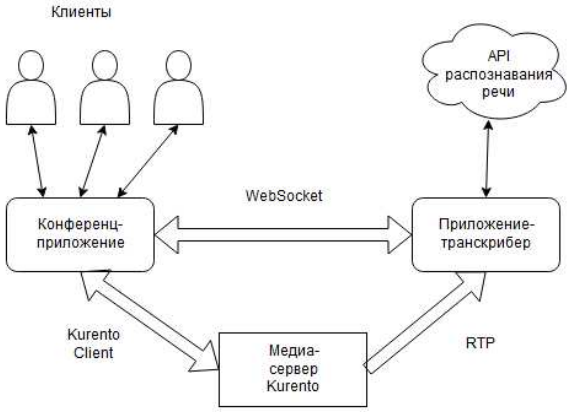

Рис. 1. Схема взаимодействия компонентов решения

В данном решении используется медиа-сервер Kurento [6]. Конференц-приложение управляет им. Исходящие медиа-данные каждого участника конференции дублируются в сторону транскрибирующего приложения по протоколу *RTP*. Приложения общаются между собой с помощью специальных *JSON*-сообщений по протоколу *WebSocket*. Конференц-приложение уведомляет транскрибера о новых и покидающих конференцию участников и отсылает *SDP*-ответы (англ. *Session Description Protocol* – протокол описания сессии [7]) на генерируемые транскрибером *SDP*-предложения, что необходимо для настройки передачи данных по *RTP*. Звук передается в одном из поддерживаемых *WebRTC* форматов (*PCMU* [8]) во избежание транскодинга. Транскрибер извлекает аудиоданные из входящих RTP-пакетов, отправляет *API* распознавания речи, обрабатывает ответы и отсылает готовые части расшифровки конференц-приложению для отображения в реальном времени.

2. ОСОБЕННОСТИ GOOGLE CLOUD SPEECH-TO-TEXT API, ПОСТАНОВКА ПРОБЛЕМ

Google Cloud Speech-to-text API – один из самых популярных API распознавания речи, чему способствуют следующие достоинства:

1. Поддержка более чем сотни языков;
2. Относительно высокая точность распознавания;
3. Поддержка множества форматов аудио;
4. Хорошая устойчивость к шумам в записи;
5. Качественная документация;
6. Быстрое время ответа;
7. Автоматическая пунктуация, специальные речевые модели и некоторые другие продвинутые функции, доступные для некоторых языков, преимущественно, для английского.

Google предоставляет готовые библиотеки клиентов для множества языков программирования, что существенно облегчает настройку данного *API* под конкретные нужды. Кроме того, процесс базовой настройки потокового распознавания речи хорошо изложен в документации данного сервиса, поэтому здесь его описание будет опущено [9]. Стоит, однако, отметить, что в документации нет примеров параллельной транскрибации нескольких аудиопотоков с последующим объединением результатов, и это касается не только *API Google*, но и других популярных сервисов распознавания речи, таких как *IBM Watson* или *Amazon Transcribe*, что, по сути, и стало причиной написания данной статьи.

API Google возвращает результаты транскрибации целыми фразами, ориентируясь на естественные паузы в человеческой речи. Основной недостаток *Google Cloud Speech-to-text API* – ограничение на продолжительность аудио при потоковом распознавании речи, которое составляет 1 *минуту*. После этого распознавание прекращается, и сервис возвращает ошибку.

```
18:14:02.266: Anna: 'да', isfinal=false
18:14:02.267: Anna: 'здравствуйте', isfinal=false
18:14:02.579: Anna: 'здравствуйте', isfinal=false
18:14:02.926: Anna: 'здравствуйте', isfinal=false
18:14:03.346: Anna: 'здравствуйте как', isfinal=false
18:14:03.410: Anna: 'здравствуйте как', isfinal=false
18:14:03.880: Anna: 'здравствуйте как наши', isfinal=false
18:14:04.010: Anna: 'здравствуйте как ваши', isfinal=false
18:14:04.410: Anna: 'здравствуйте как ваши дела', isfinal=false
18:14:48.620: Anna: 'здравствуйте как ваши дела', isfinal=true
```

Рис. 2. Пример транскрибации с включенной опцией *interim_results*

Исследование работы сервиса для других языков, помимо английского (например, русского и чешского) также выявило ещё одну неприятную особенность: время от времени, несмотря на

молчание говорящего, *API* не возвращает расшифровку уже сказанного в течение длительного времени, пока время сессии не приблизится к лимиту. Это приводит не только к недопустимым для приложения, работающего в реальном времени, задержкам, но и к некорректной последовательности сказанного в итоговой расшифровке, так как время обработки аудиопотоков участников сильно разнится, например:

```
Bob: у меня все хорошо
Anna: Здравствуйте как ваши дела
```

Таким образом, настроек распознавания речи по умолчанию недостаточно для транскрибации веб-конференций в реальном времени, необходима их оптимизация. Также обязательно решение проблемы ограничения на продолжительность аудио.

3. ПОИСК РЕШЕНИЯ ПОСТАВЛЕННЫХ ПРОБЛЕМ

На самом деле API Google отвечает очень быстро, в чём можно убедиться, включив промежуточные результаты при помощи опции *interim_results* (рис. 2). Однако, мы можем видеть, что у промежуточных результатов не устанавливается флаг *is_final*, что означает, что сервис может частично или полностью изменить гипотезу. Также несмотря на то, что пользователь не сказал ничего нового после 18:14:04, сервис не возвращал финальный результат до 18:14:48. Промежуточные результаты предназначены для отображения пользователю, пока идёт распознавание, что может быть удобно в некоторых ситуациях, но для нашей задачи нежелательно захламлять конференцию кучей зачастую бессмысленных данных. Кроме того, промежуточные результаты не получится записывать в файл, поэтому, даже если отображать в браузерах промежуточные результаты, а в файлы писать только финальные, порядок фраз в расшифровке, вероятно, всё равно останется некорректным.

Ответы Google Speech API содержат также поле *confidence* (уверенность движка в корректности расшифровки), значение которого может варьироваться от 0 до 1. Можно было бы использовать промежуточные результаты, уверенность в которых выше некоторого порогового значения, однако данный механизм работает иначе – для всех промежуточных результатов значение *confidence* равняется нулю.

Google Cloud Speech-to-text API может сопровождать ответы временными метками для каждого слова. Значения смещения времени устанавливаются для начала и окончания каждого сказанного слова. Смещения относительны началу аудиозаписи, инкремент составляет 100 *мс*. К сожалению, не получится заставить *API* возвращать данные порциями вместе с временными метками. Данные временные метки полезны для анализа длинных записей, если может возникнуть необходимость найти определённое слово в расшифровке и соответствующий ему момент в аудиозаписи.

Суть решения проблемы лимита на продолжительность аудио сводится к регулярной реинициализации соединения с сервером. Однако если это делать "в лоб", по истечении лимита, то будут теряться большие объемы данных, так как сервер не будет возвращать расшифровку сказанного перед ошибкой превышения лимита. Более продвинутая идея – использовать две перекрывающие друг друга на несколько секунд сессии, однако качество результатов распознавания на границе этих двух сессий оставляет желать лучшего.

В настройках *API* существует опция *single_utterance*, которая может решить проблему задержки ответа и условно решить проблему ограничения. Высказывание (англ. *utterance*) – это минимальная целостная единица речевого общения, речевой отрезок, относительно законченный по интонации и смыслу [10]. При включенной опции, движок распознает единичные высказывания. Если пользователь делает паузу длиною более чем секунда или перестает говорить, *API* возвращает специальное событие, означающее конец высказывание, и прекращает распознавание речи, закрывая соединение наполовину. После этого может быть возвращен результат распознавания. Если пользователь ничего не говорит, "высказывание" длится около 7 *секунд*. После прекращения распознавания можно реинициализировать соединение с сервером. Рассмотрим пример транскрибации с опцией *single_utterance* (*Рис*. 3). Для реинициализации сессии требуется 5-50 *мс*, что, как правило, проходит бесшовно, учитывая естественные паузы в речи. Расшифровка предыдущего высказывания обычно приходит после инициализации следующей сессии, поэтому для всех сессий, открытых для одного и того же пользователя, используется один и тот же обработчик ответов от *API* (англ. *Response Observer*).

Полученное решение не совершенно, поскольку практические эксперименты показали, что иногда части расшифровки теряются. Кроме того, данное решение не гарантирует синхронизацию частей расшифровки по времени, полагаться можно только на скорость ответа сервиса, чего, впрочем, обычно достаточно.

```
13:18:10.839 Got start message from the server
13:18:14.298 End of single utterance
13:18:14.305 Got start message from the server
13:18:14.370 Anna: 'здравствуйте как ваши дела'
13:18:19.521 End of single utterance
13:18:19.531 Got start message from the server
13:18:20.356 Anna: 'у меня все хорошо'
```

Рис. 3. Пример транскрибации с включенной опцией *single_utterance*

4. АНАЛИЗ РЕЗУЛЬТАТОВ

Пример транскрибации фрагмента конференции с двумя участниками после проведенных настроек приведен на *Рис*. 4. Как видно, порядок высказываний корректен. Система также демонстрировала высокое быстродействие. В ходе оптимизации настроек была также подобран оптимальная длительность фрагмента аудио, отправляемого одним запросом – 200 *мс*.

Следует отметить, что на качество работы *API* распознавания речи влияет множество факторов, а именно: используемое оборудование (микрофон), фоновый шум, дикция и акцент.

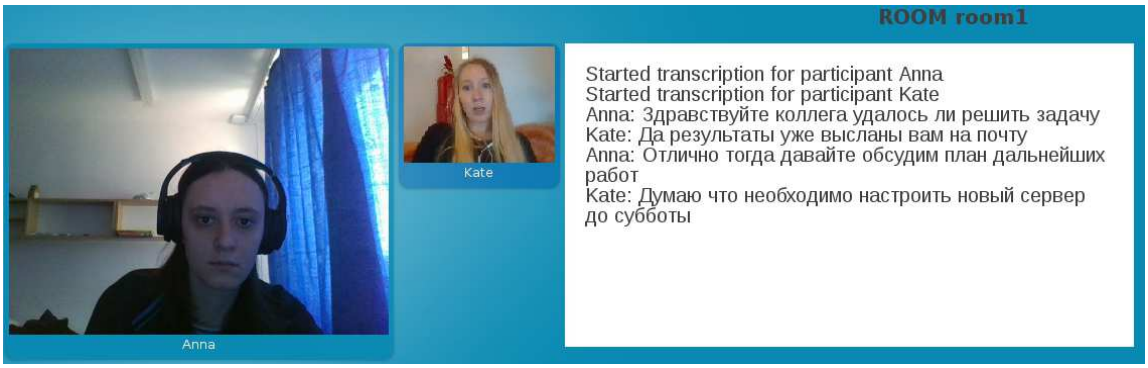

Рис. 4. Пример транскрибации веб-конференции с двумя участниками

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы были получены следующие результаты:

- Рассмотрены способы извлечения аудиопотоков участников конференции в реальном времени;
- Проведён анализ работы Google Cloud Speech-to-text API;
- Перечислены возможные проблемы при использовании данного сервиса для транскрибации веб-конференций и предложены способы их решения.

В результате проделанной работы можно сделать вывод, что данный сервис подходит для распознавания речи в веб-конференциях в реальном времени.

ЛИТЕРАТУРА

[1] Grigorik, I. High-performance browser networking. Sebastopol, CA: O'Reilly, 2013. ISBN 1449344763.

[2] Introducing live captions subtitles in Skype [Электронный ресурс] // Skype Team, 2018. URL: https://blogs.skype.com/news/2018/12/03/introducing-live-captions-and-subtitles-in-skype/ (дата обращения 03.27.2018).

[3] Segeč, P.; Palúch, P.; Papán, J.; Kubina, M. The integration of WebRTC and SIP: Way of enhancing real-time, interactive multimedia communication. 2014. Режим доступа: DOI: 10.1109/IGETA.2014.7107624.

[4] A speech-to-text prototype in Jitsi Meet! [Электронный ресурс] // Vaessen, N., 2017. URL: https://nikvaessen.github.io/jekyll/update/2017/08/01/speech-to-text-prototype-in-Jitsi_Meet.html (дата обращения 03.06.2019).

[5] Real-time Transport Protocol [Электронный ресурс] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Real-time_Transport_Protocol (дата обращения 01.04.2019).

[6] About Kurento and WebRTC [Электронный ресурс] // URL: https://doc-kurento.readthedocs.io/en/stable/user/about.html (дата обращения 03.06.2019).

[7] Session Description Protocol [Электронный ресурс] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Session_Description (дата обращения 03.06.2019).

[8] G.711 [Электронный ресурс] // Wikipedia. URL: https://ru.wikipedia.org/wiki/G.711 (дата обращения 03.06.2019).

[9] Transcribing audio from streaming input [Электронный ресурс] // Google. URL: https://cloud.google.com/speech-to-text/docs/streaming-recognize (дата обращения 15.05.2019)

[10] Высказывание (лингвистика) [Электронный ресурс] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Высказывание_(лингвистика) (дата обращения: 01.06.2019).

Анна Сергеевна Каменская – магистрантка кафедры автоматики. Область научных интересов: высоконагруженные и распределённые веб-системы, автоматизация процессов тестирования и доставки кода.
Email: ladymacbeth94@yandex.ru
630073, Новосибирск,
просп. К.Маркса, д. 20

Статья поступила 21.05.2019.

Adaptation of Google Cloud Speech-to-text API for automatic real-time transcription of web conference

A. S. Kamenskaia

Novosibirsk State Technical University, Novosibirsk, Russia

Abstract: The paper deals with the problem of setting up Google Cloud Speech-to-text API for solving the task of automatic real-time transcription of a web conference. The features of the task and possible difficulties arising in the course of the work are indicated, and ways to solve them are described. It also lists ways to capture live stream audio of conference participants before streaming speech recognition.

Keywords: WebRTC, speech recognition, web conference.

REFERENCES

[11] Grigorik, I. High-performance browser networking. Sebastopol, CA: O'Reilly, 2013. ISBN 1449344763.

[12] Introducing live captions subtitles in Skype [Electronic resource] // Skype Team, 2018. URL: https://blogs.skype.com/news/2018/12/03/introducing-live-captions-and-subtitles-in-skype/ (date of access 03.27.2018).

[13] Segeč, P.; Palúch, P.; Papán, J.; Kubina, M. The integration of WebRTC and SIP: Way of enhancing real-time, interactive multimedia communication. 2014. Режим доступа: DOI: 10.1109/IGETA.2014.7107624.

[14] A speech-to-text prototype in Jitsi Meet! [Electronic resource] // Vaessen, N., 2017. URL: https://nikvaessen.github.io/jekyll/update/2017/08/01/speech-to-text-prototype-in-Jitsi_Meet.html (date of access).

[15] Real-time Transport Protocol [Electronic resource] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Real-time_Transport_Protocol (date of access 01.04.2019).

[16] About Kurento and WebRTC [Electronic resource] // URL: https://doc-kurento.readthedocs.io/en/stable/user/about.html (date of access 03.06.2019).

[17] Session Description Protocol [Electronic resource] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Session_Description (date of access 03.06.2019).

[18] G.711 [Electronic resource] // Wikipedia. URL: https://ru.wikipedia.org/wiki/G.711 (date of access 03.06.2019).

[19] Transcribing audio from streaming input [Electronic resource] // Google. URL: https://cloud.google.com/speech-to-text/docs/streaming-recognize (date of access 15.05.2019)

[20] Высказывание (лингвистика) [Electronic resource] // Wikipedia. URL: https://ru.wikipedia.org/wiki/Высказывание_(лингвистика) (date of access 01.06.2019).

Anna Kamenskaya - master student of the department of automation. Research interests: high-load and distributed web systems, automation of testing processes and code delivery.

Email: ladymacbeth94@yandex.ru

630073, Novosibirsk,
str. Prosp. K. Marksa, h. 20

The paper has been received on 21.05.2019.