Санкт-Петербургский Государственный Университет
Математико-механический факультет

Кафедра системного программирования

Завадский Илья Олегович

Разработка алгоритма диаризации

Курсовая работа

Научный руководитель:
к. ф.-м. н., ст. преп. Луцив Д. В.

Консультант:
рук. проекта, ООО "Центр Речевых Технологий" Тимченко М. С.

Санкт-Петербург
2020

Оглавление

Введение

Существует достаточно много задач, связанных с информацией, извлекаемой из аудиосигнала. Примерами данных задач являются:

- Распознавание речи: "Что было сказано?"
- Распознавание языка: "На каком языке это было сказано?"
- Распознавание диктора: "Кто говорит?"

Распознавание дикторов (англ. Speaker Recognition) — это задача идентификации того, кто говорит. Задача распознавания дикторов включает в себя множество задач, которые прямо или косвенно связаны между собой. Среди них можно выделить следующие: задача верификации диктора (англ. speaker verification), задача идентификации диктора (англ. speaker identification) и задача классификации диктора (англ. speaker classification). Первая задача старается сравнить диктора с заявленной моделью, поэтому данную задачу можно назвать задачей взаимно-однозначного сравнения. Она используется, чтобы определить, является ли диктор тем, кем он представился. Поэтому задача верификация диктора обычно изучается с точки зрения области применения биометрической идентификации. Цель задачи идентификации диктора — узнать, кто говорит. Так речевой сегмент сравнивается с базой данных моделей дикторов, то есть выполняется сравнение «один ко многим». Задача классификации диктора — задача, в которой определить к какому классу относится диктор. Примерами классификации являются гендерная, возрастная классификации и классификация настроения [3].

Частью каждой из этих задач является подзадача диаризации дикторов (англ. diarization). Диаризация дикторов (или разделение дикторов) — это процесс разделения входного аудиосигнала на сегменты в соответствии с идентификацией личности диктора. Она используется для ответа на вопрос: "Кто говорит и когда?" [9].

В рамках данной курсовой работы рассматривается задача диаризации. Целью является реализация решения данной задачи в определенных условиях эксплуатации, которые будут описаны в дальнейшем.

1. Постановка задачи

1.1. Описание задачи

Два человека, разделенных перегородкой, разговаривают друг с другом. Обоих записывают микрофоны с инфракрасным датчиком, обнаруживающим присутствие и перемещение человека (Рис. 1).

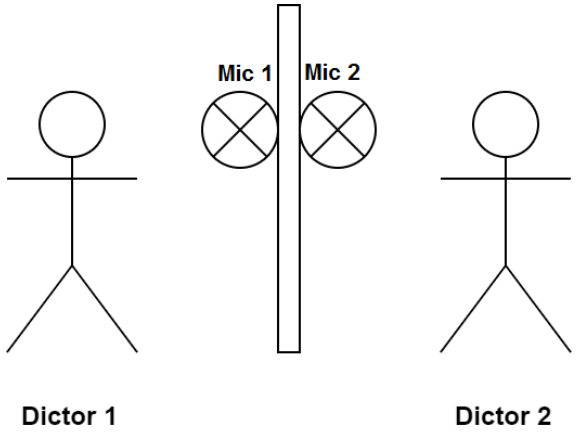

Рис. 1: Расположение дикторов и микрофонов

Каждый такой микрофон подключен к регистратору, который преобразует аналоговый сигнал, поступающий с микрофона, в цифровой. Цифровой сигнал передается на сервер и записывается в формате WAV на жесткий диск (Рис. 2).

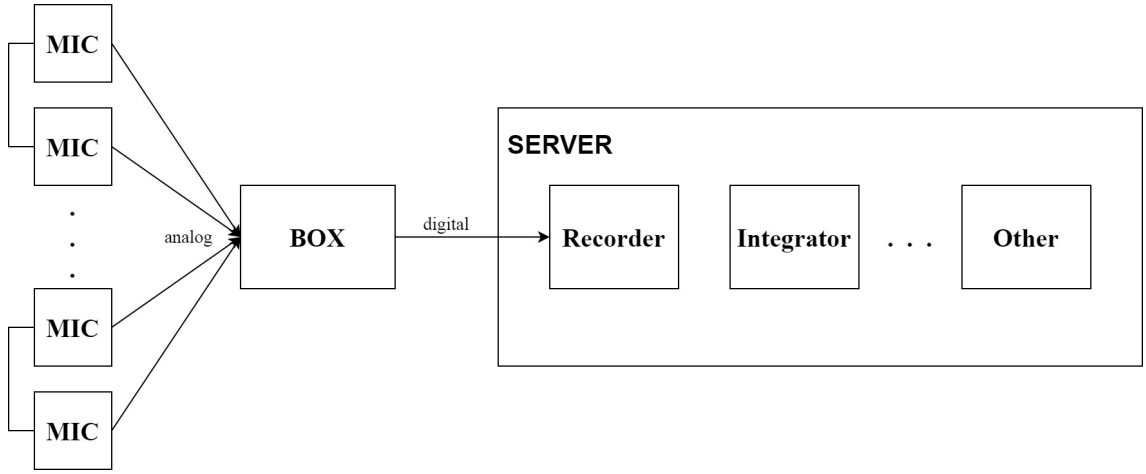

Рис. 2: Схема обработки аудиосигнала

1.2. Постановка задачи

Цель работы — разработать, реализовать и протестировать алгоритм диаризации дикторов, позволяющий уменьшить накладные расходы на вычислительные ресурсы и память в ситуации, описанной выше. Для достижения цели были поставлены следующие задачи:

- Изучить:
    1. Структуру WAV файлов;
    2. Работу текущих алгоритмов диаризации;
- Разработать, реализовать и внедрить алгоритм диаризации;
- Протестировать алгоритм на реальных данных;

Полученный в результате работы алгоритм планируется использовать в будущих продуктах "ЦРТ[1]".

[1]Центр Речевых Технологий

2. Обзор литературы и существующих решений

Эта глава дает некоторые теоретические основы в области распознавания аудиосигнала и рассказывает об уже существующих решениях.

2.1. Диаризация дикторов

Диаризация дикторов (или разделение дикторов) — это процесс разделения входного аудиосигнала на сегменты в соответствии с идентификацией личности диктора. Она используется для ответа на вопрос: "Кто говорит и когда?". Задача разделения дикторов — это комбинация задач сегментации говорящего и кластеризации говорящего. Первая направлена на поиск точек смены динамиков в аудиосигнале. Вторая направлена на группирование речевых сегментов на основе характеристик говорящего [9].

2.1.1. Детектор Речевой Активности

В системах распознавания диктора немаловажную роль играет детектор речевой активности (англ. Voice Activity Detection, VAD) — алгоритм, классифицирующий исходные участки фонограммы как речь или не речь [17]. Среди всех алгоритмов детекции речевой активности наибольшей популярностью из-за своей простоты пользуются алгоритмы, основанные на анализе кратковременной энергии сигнала (англ. short-term energy) и скорости переходов через ноль (англ. zero-crossing rate) [12, 16].

2.1.2. Сегментация

Сегментация — это задача нахождения в аудиосигнале точек изменения, где произошла смена диктора [9]. Методы сегментации обычно делятся на две основные категории:

- Метрическая: Определяет, исходят ли два акустических сегмента из одного динамика, вычисляя расстояние между двумя акустическими сегментами.

- Модельная: Модели с помощью учителя обучаются распознавать дикторов, а затем используются, для оценивания, где есть точки изменения в аудиофайле.

Сегментация на основе метрик является наиболее популярным типом метода, так как никаких предварительных знаний не требуется для ее работы. [13]

2.1.3. Кластеризация

При кластеризации сегментов изучается сходство между ними, и строится иерархия кластеров [9]. Существуют два основных подхода к созданию такой иерархии:

- Аггломеративный: Это подход снизу-вверх (англ. bottom-up). Первоначально для каждого сегмента существует один кластер, и кластеры итеративно объединяются, пока набор кластеров не достигнет оптимального размера.

- Разделяющий: Это подход сверху-вниз (англ. top-down). В начале все сегменты хранятся в одном большом кластере. Новый диктор вводится рекурсивно, и кластер разбивается на все меньшие и меньшие кластеры.

Поскольку агломерационная кластеризация имеет хороший баланс между простотой структуры и производительностью, этот подход наиболее часто используется [5].

2.2. Идентификация диктора

Следующим этапом распознавания диктора является определение того, какие кластеры соотносятся с каким диктором. Данная фаза состоит из двух: фазы обучения и фазы тестирования.

Фаза обучения включает в себя подготовку модели для каждого диктора. Основная идея обучения состоит в том, чтобы иметь набор параметров, которые могут представлять характеристики речи оратора. Этап обучения может быть выполнен независимо от системы распознавания. Если набор моделей дикторов обучен и создана база данных этих моделей, база данных может быть включена в существующие данные в систему распознавания.

Фаза тестирования берет данные от тестового диктора и сравнивает эти данные с каждой моделью в базе данных. Таким образом можно идентифицировать личность говорящего в каждом кластере [3].

2.3. Формат аудиофайла

При записи аудиосигнала используется формат WAV, который был создан в 1991 году компаниями IBM и Microsoft [7]. Файл WAV является сферой приложения формата RIFF[2] для хранения аудио в «цепочках» (англ. chunk). Формат RIFF действует как «обертка» для различных форматов кодирования звука. Зачастую формат WAV используется для хранения несжатого звука в формате линейной импульсной кодовой модуляции (англ. Linear Pulse Code Modulation, LPCM). Файл WAV состоит из заголовка и блока данных.

2.3.1. Заголовок

Заголовок — начало WAV файла. Он используется для предоставления информации о размере файла, количестве каналов, частоте дискретизации (англ. sample rate), количестве бит в сэмпле.

Заголовок WAV файла имеет длину 44 байта. Следующая таблица наглядно показывает структуру заголовка [4]:

[2]Resource Interchange File Format

Позиция	Значение	Описание
0-3	”RIFF”	Маркер начала RIFF-цепочки.
4-7	File size	Размер файла минус 8.
8-11	”WAVE”	Тип файла.
12-15	”fmt ”	Маркер формата подцепочки.
16-19	16	Оставшийся размер подцепочки.
20-21	1	Аудио формат.
22-23	2	Количество каналов.
24-27	16000	Частота дискретизации.
28-31	64000	Количество байт, переданных за секунду.
32-33	4	Количество байт для одного сэмпла.
34-35	16	Количество бит в сэмпле.
36-39	”data”	Маркер подцепочки данных.
40-43	”data” size	Количество байт в области данных.

2.3.2. Блок Данных

Сразу же после заголовка начинается блок, в котором хранятся данные.

В зависимости от количества каналов меняется то, как располагаются значения амплтитуд канала/каналов. Например, если количество каналов равно единице, то значения амплитуды расположены последовательно. В случае, если каналов больше одного, сначала идет значение амплитуды для первого канала, затем для второго и так далее [7].

2.4. Существующие решения

Существует несколько свободно распространяемых продуктов и библиотек, которые могут реализовывать данную функциональность.

2.4.1. LIUM_SpkDiarization

LIUM_SpkDiarization — это программное обеспечение, предназначенное для диаризации дикторов. Оно написано на Java и включает в себя самые последние разработки в этой области [15].

Данное программное обеспечение было разработано для французской оценочной кампании ESTER2, где оно получило наилучшие результаты в задаче диаризации дикторов трансляции новостей в 2008 году. Этот набор инструментов оптимизирован для радио или телешоу, поэтому уровен производительности на телефонных разговорах и встречах может быть ниже.

LIUM_SpkDiarization содержит полный набор инструментов для создания системы для диаризации дикторов, которые включают в себя MFCC[3], обнаружение речи / неречевой речи и способы диаризации динамика.

Данное ПО расспространяется в качесте JAR-архива, который содержит полную, готовую к использованию скомпилированную версию. Также можно воспользоваться и исхониками данного ПО, которое можно загрузить с официального сайта университета Ле-Мана [10].

2.4.2. AudioSeg

AudioSeg — инструментарий для сегментации звука и классификации аудиопотоков, который написан на языке C и распространяется в виде исходных кодов [8].

AudioSeg представляет собой набор алгоритмов, помогающих создавать прототипы и разрабатывать приложения, использующие следующие возможности:

- обнаружение тишины / звуковой активности;
- слепая сегментация с помощью BIC[4];
- кластеризация сегментов;
- классификация сегментов с использованием GMM[5];
- совместная сегментация и классификация с использованием HMM[6];

[3]Mel-Frequency Cepstral Coefficients
[4]Bayesian information Criterion
[5]Gaussian Mixture Models
[6]Hidden Markov Models

2.4.3. ALIZE

ALIZÉ — это платформа с открытым исходным кодом для распознавания говорящего, написанной на языке C++. [1]

Целью этого проекта является предоставление набора низкоуровневых и высокоуровневых структур, которые позволят любому разрабатывать приложения, выполняющие различные задачи в области распознавания диктора.

ALIZÉ была разработана с многоуровневой архитектурой.

Базовым уровнем является ALIZE-Core — низкоуровневая библиотека, которая включает в себя все функции, необходимые для использования гауссовых смесей, а также функции ввода-вывода для различных форматов файлов.

Поверх этого ядра был построен LIA_RAL — инструментарий, предлагающий функциональность более высокого уровня, который состоит из следующих компонентов:

- LIA_SpkDet — набор инструментов для обучения моделей, нормализации функций, нормализации оценок и т.д.;
- LIA_SpkSeg — инструменты для диаризации дикторов;
- LIA_Utils — утилиты для управления различными; форматами данных, используемыми в ALIZÉ;
- LIA_SpkTools — библиотека, которая обеспечивает высокоуровневые функции поверх ядра ALIZE;

Параллельно LIA_RAL также включает в себя библиотеку SimpleSpkDetSy которая предлагает простой высокоуровневый API для разработчиков, которые хотят легко встроить проверку или идентификацию динамика в свои приложения. Существует также Java-версия этого API, предназначенная для разработки приложений Android [2].

2.4.4. DiarTk

DiarTk - это набор инструментов с открытым исходным кодом на C++ для диаризации дикторов, который распространяется по лицензии GPL. Он распространяется в виде архива с исходным кодом, который можно скачаить с официального сайта разработчиков и который необходимо скомпилировать на вашем компьютере. При работе с аудиофайлами система использует акустические признаки, MFCC[7], а также может дополнительно использовать такие признаки как TDOA[8] и FDLP[9] / MS[10] [14].

2.4.5. Bob

Bob - это бесплатный набор инструментов для обработки сигналов и машинного обучения, первоначально разработанный группой Biometrics в Научно-исследовательском институте Idiap, Швейцария.

Инструментарий написан на смеси Python и C ++ и предназначен как для эффективной работы, так и для сокращения времени разработки. Он состоит из достаточно большого количества пакетов, в которых реализованы инструменты для обработки изображений, аудио и видео, машинного обучения и распознавания образов [6].

[7]Mel-Frequency Cepstral Coefficients
[8]Time Delay of Arrivals
[9]Frequency Domain Linear Prediction
[10]Modulation Spectrum

3. Описание предложенного решения

На сервере одновременно запущены несколько микро-сервисов (Рис. 3). В течение рабочего дня работает сервис Recorder, который отвечает за запись аудио-потока, поступающего из BOX.

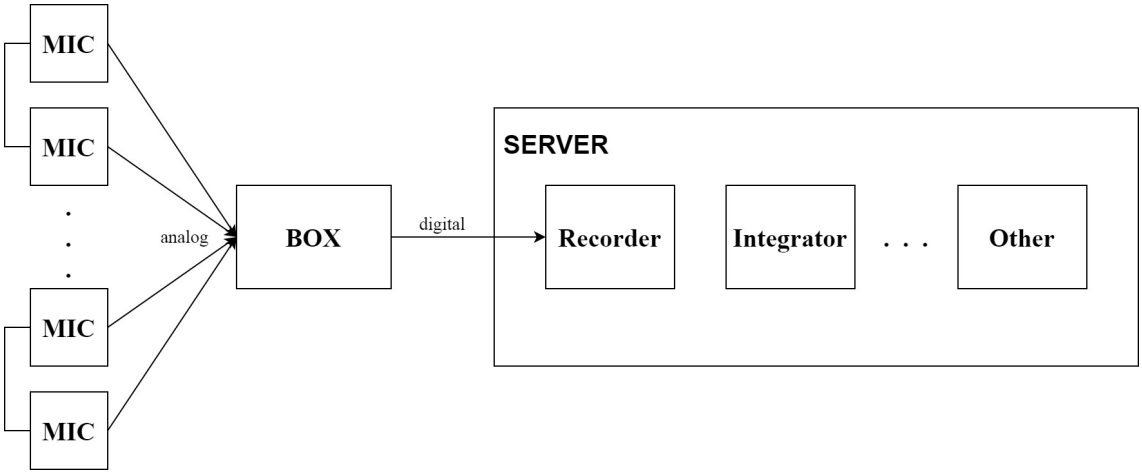

Рис. 3: Схема обработки аудиосигнала

После окончания рабочего дня запускается микро-сервис Integrator, который в свою очередь группирует, обрабатывает, анализирует и подготавливает файлы, записанные за весь день.

3.1. Алгоритм VAD

В нашей задаче используются микрофоны с инфракрасным датчиком, поэтому в первую очередь нам нужно выделить участки, когда перед микрофоном есть диктор. Для решения данной проблемы был реализован детектор речевой активности, основанный на расчете кратковременной энергии [11], работающий в сервисе Recorder (Рис. 4).

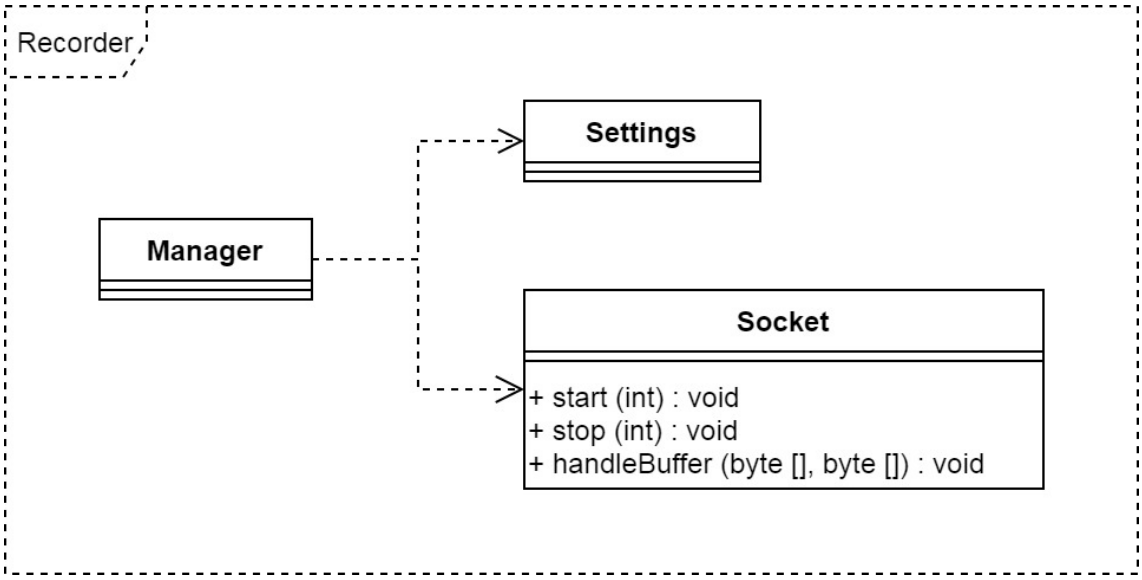

Рис. 4: Диаграмма Recorder

Расчет кратковременной энергии - это параметр, используемый при классификации сегментов. Если энергия входящего кадра высокая, кадр классифицируется на вокализованный кадр, а если энергия входящего кадра низкая, он классифицируется на невокализованный кадр.

Кратковременная энергия кадра E_n определяется согласно формуле

$$\sum_{m=1}^{\infty} |x(m)| \cdot h(m,n)$$

Где

$$h(m,n) = \begin{cases} 1, & \text{if n} \cdot \text{N} \leq \text{m} \leq (\text{n+1}) \cdot \text{N} \\ 0, & \text{Otherwise} \end{cases}$$

В этом методе используется прямоугольное окно.

Поток данных для классификации входного сигнала на вокализованные или невокализованные сегменты выполняется, как показано на блок-схеме.

Способ начинается с накопления достаточного размера сегмента для записи. После этого на данном сегменте вычисляется кратковременная энергия.

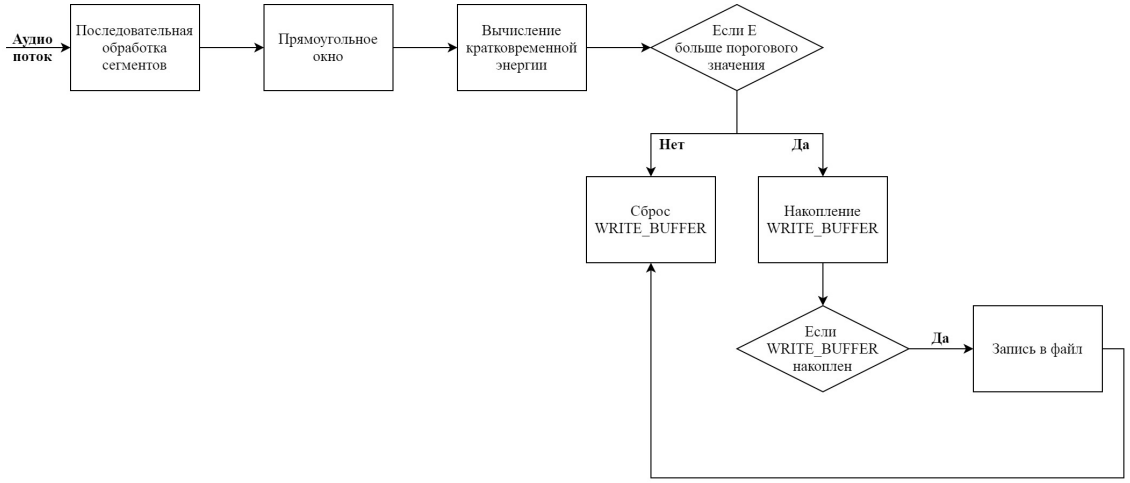

Рис. 5: Блок диаграмма работы VAD-алгоритма

В покадровом кадре речевой сигнал делится на неперекрывающиеся кадры из 64 выборок с частотой дискретизации 16 кГц, которая эквивалентна продолжительности времени 128 мс. Измерения кратковременной энергии этих кадров сравниваются с их пороговым значением, равным десятой части Short.MAX_VALUE. Кадры классифицируются как вокализованные сегменты. Иначе, кадры классифицируются как невокализованные сегменты.

3.2. Алгоритм диаризации

При решении задачи мы предполагаем, что в каждый микрофон говорит только один человек. При решении задачи мы знаем, как микрофоны расположены относительно каждого из дикторов, поэтому мы можем определить, на каком из микрофонов амплитуда диктора будет больше или меньше.

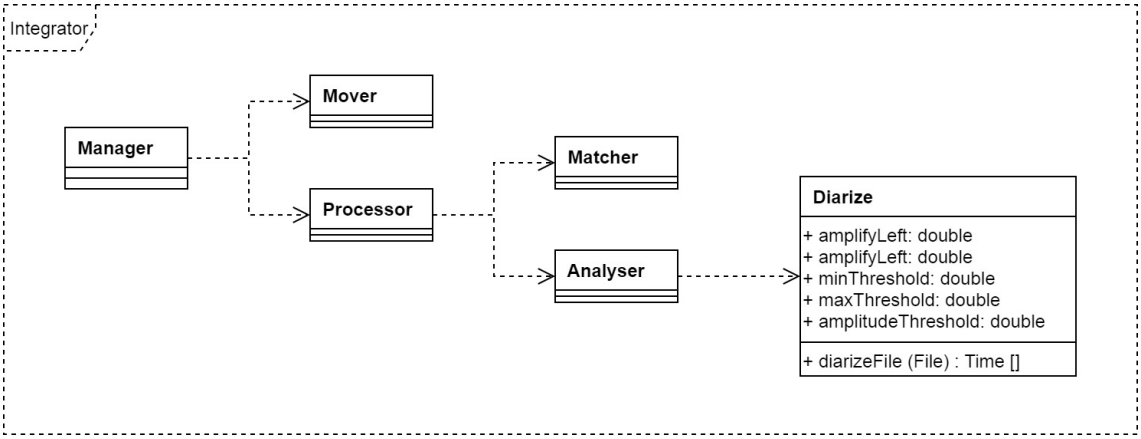

Рис. 6: Диаграмма Integrator

Исходя из этого предположения и условий эксплуатации, задачу кластеризации можно свести к задаче определения есть человек перед микрофоном или нет в VAD-алгоритме. Поэтому для данной задачи был использован модифицированный детектор речевой активности, который используется в сервисе Integrator (Рис. 6).

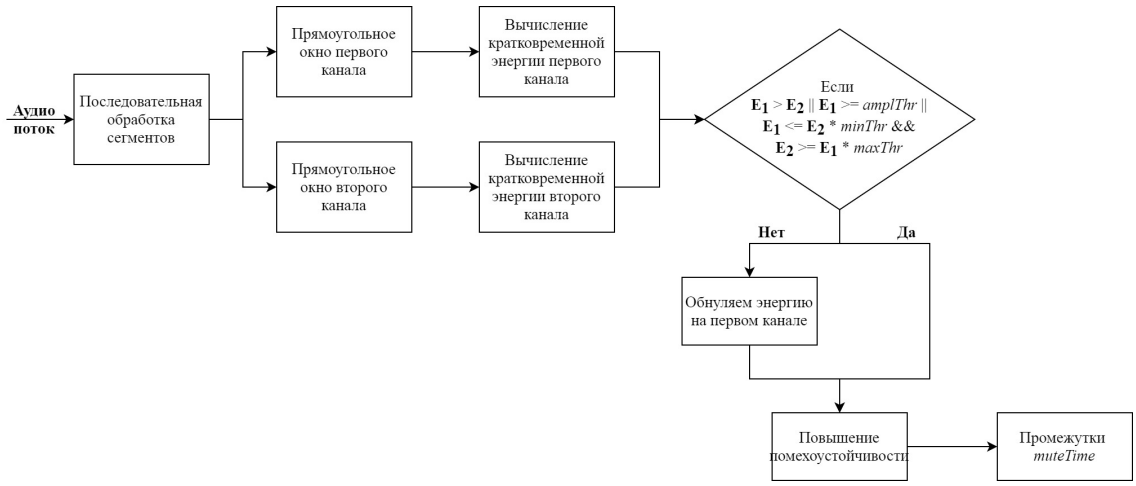

Рис. 7: Блок диаграмма работы алгоритма диаризации

Поток данных для классификации входного сигнала на вокализованные или невокализованные сегменты выполняется, как показано на блок-схеме.

В покадровом кадре речевой сигнал делится на неперекрывающиеся кадры из 64 выборок с частотой дискретизации 16 кГц, которая эквивалентна продолжительности времени 128 мс. На каждом из сегментов вычисляется кратковременная энергия с учетом усиления, которое передано программе.

Измерения кратковременной энергии E_{1_n} каждого кадра первого канала вычисляется по следующей формуле:

$$E_{1_n} = \sum_{m=1}^{\infty} |x(m)| \cdot h(m,n)$$

Где

$$h(m,n) = \begin{cases} \text{amplifyLeft}, & \text{if n} \cdot \text{N} \leq \text{m} \leq (\text{n}+1) \cdot \text{N} \\ 0, & \text{Otherwise} \end{cases}$$

Измерения кратковременной энергии E_{2_n} этих кадров второго канала вычисляются по следующей формуле:

$$E_{2_n} = \sum_{m=1}^{\infty} |x(m)| \cdot h(m,n)$$

Где

$$h(m,n) = \begin{cases} \text{amplifyRight}, & \text{if n} \cdot \text{N} \leq \text{m} \leq (\text{n}+1) \cdot \text{N} \\ 0, & \text{Otherwise} \end{cases}$$

Кадры первого канала классифицируются как речь, если выполняется одно из следующих условий:

- $E_{1_n} > E_{2_n}$
- $E_{1_n} > \text{amplitudeThreshold}$
- $E_{1_n} \leq E_{2_n} \cdot \text{minThreshold} \;\&\&\; E_{2_n} \geq E_{1_n} \cdot \text{maxThreshold}$

Если не выполняется ни одно условие, то в кадрах первого канала нет речи.

После классификации для повышения помехоустойчивости мы делаем следующую проверку:

- Если между двумя кадрами, в которых нет речи, лежит кадр, классифицированный как речь, мы меняем его классификацию и считаем, что в данном кадре нет речи.

После окончания обработки файла. На выходе мы выдаем промежутки времени, в течение которых на первом канале молчали.

4. Результаты тестирования

В ходе работы над данным проектом было проведено тестирование алгоритма на данных, взятых у заказчика.

Тестирование проводилось на компьютере с 64-разрядным четырёхъядерным процессом Intel® Core™ i5-3450 с базовой частотой 3.10 ГГц, вместимостью ОЗУ[11] 16 ГБ и SSD на 512 ГБ.

Так как длина каждой сессии различна, и так как нас интересует как измненились накладные расходы на вычислительные ресурсы и память, то нас интересуют следующие параметры:

- Память;
- Отношение времени диаризации аудиофайла до и после внедрения детектора речевой активности;
- Отношение времени полной обработки аудиофайла до и после внедрения детектора речевой активности и алгоритма диаризации;
- Отношение времени полной работы сервиса Integrator до и после внедрения детектора речевой активности и алгоритма диаризации;

Таблица 1: Результаты тестирования

Память	2.57 ГБ / 1.44 ГБ
Отношение времени диаризации	1
Отношение времени полной обработки	0.80
Отношение времени полной работы сервиса Integrator	0.80

[11] Оперативного Запоминающего Устройства

5. Заключение

В рамках работы были выполнены следующие задачи:

- Рассмотреть существующие решения
- Изучить формат хранения данных.
- Разработать алгоритм диаризации дикторов.
- Внедрить алгоритм в один из текущих проектов.
- Протестировать на реальных данных.

Список литературы

[1] ALIZÉ // ALIZÉ wiki. — 2020. — Access mode: https://alize.univ-avignon.fr/mediawiki/index.php/Main_Page (online; accessed: 18.05.2020).

[2] ALIZÉ // ALIZÉ opensource speaker recognition. — 2020. — Access mode: https://alize.univ-avignon.fr/ (online; accessed: 18.05.2020).

[3] Beigi H. Fundamentals of Speaker Recognition. — Springer Publishing Company, Incorporated, 2011. — ISBN: 0387775919, 9780387775913.

[4] Coding Audio. Структура WAV файла. — 2008. — Режим доступа: https://audiocoding.ru/articles/2008-05-22-wav-file-structure/ (дата обращения: 11.12.2019).

[5] A Comparative Study of Bottom-Up and Top-Down Approaches to Speaker Diarization / N. Evans, S. Bozonnet, D. Wang et al. // IEEE Transactions on Audio, Speech, and Language Processing. — 2012. — Feb. — Vol. 20, no. 2. — P. 382–392.

[6] Continuously Reproducing Toolchains in Pattern Recognition and Machine Learning Experiments / A. Anjos, M. Günther, T. de Freitas Pereira et al. // International Conference on Machine Learning (ICML). — 2017. — 08. — Access mode: http://publications.idiap.ch/downloads/papers/2017/Anjos_ICML2017-2_2017.pdf.

[7] IBM, Microsoft. Multimedia Programming Interface and Data Specifications 1.0. — 1991. — Aug.

[8] InriaForge // AudioSeg: Project Home. — 2010. — Access mode: https://gforge.inria.fr/projects/audioseg (online; accessed: 18.05.2020).

[9] Kotti M., Moschou V., Kotropoulos C. Speaker segmentation and clustering // Signal Processing. — 2008. — P. 1091–1124.

[10] LIUM_SpkDiarization. — 2020. — Access mode: https://projets-lium.univ-lemans.fr/spkdiarization/ (online; accessed: 18.05.2020).

[11] Meduri Sameeraj, Ananth Rufus. A Survey and Evaluation of Voice Activity Detection Algorithms: Speech Processing Module. — Koln, DEU : LAP Lambert Academic Publishing, 2012. — ISBN: 3659172049.

[12] Moattar M., Homayoonpoor M. A simple but efficient real-time voice activity detection algorithm // European Signal Processing Conference. — 2010. — 12.

[13] Speaker Diarization: A Review of Recent Research / X. Anguera, S. Bozonnet, N. Evans et al. // IEEE Transactions on Audio, Speech, and Language Processing. — 2012. — Feb. — Vol. 20, no. 2. — P. 356–370.

[14] Vijayasenan Deepu, Valente Fabio. DiarTk : An Open Source Toolkit for Research in Multistream Speaker Diarization and its Application to Meetings Recordings. — Vol. 3. — 2012. — 01.

[15] An open-source state-of-the-art toolbox for broadcast news diarization / Mickael Rouvier, Grégor Dupuy, Paul Gay et al. — 2013.

[16] Будько М.Б. Алгоритм определения речевой активности и генератор комфортного шума высокого быстродействия. — 2006. — Режим доступа: https://cyberleninka.ru/article/n/algoritm-opredeleniya-rechevoy-aktivnosti-i-generator-komfortnog

[17] Симончик К.К., Галинина О.С., Капустин А.И. Алгоритм обнаружения речевой активности на основе статистик основного тона в задаче распознавания диктора. — 2010. —

Режим доступа: https://cyberleninka.ru/article/n/algoritm-obnaruzheniya-rechevoy-aktivnosti-na-osnove-statistik-