МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Доцент кафедры ВМиИТ
ФГБОУ ВО «ЧелГУ», к.ф.-м.н.

_____ А.Ю. Маковецкий

«___»_____ 2024 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н., профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Реализация веб-приложения для выделения целевого голоса с использованием нейросетевых технологий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.04.04.2024.308-1473.ВКР

Научные руководители:
доцент кафедры СП, к.ф.-м.н.
_____ С.У. Турлакова,

ст. преподаватель кафедры СП
_____ Н.С. Силкина

Автор работы,
студент группы КЭ-228
_____ В.И. Капичай

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы магистранта
студенту группы КЭ-228
Капичаю Владиславу Игоревичу,
обучающемуся по направлению
09.04.04 «Программная инженерия»
(магистерская программа «Искусственный интеллект и инженерия данных»)

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)
Реализация веб-приложения для выделения целевого голоса с использованием нейросетевых технологий.
2. **Срок сдачи студентом законченной работы:** 20.05.2024 г.
3. **Исходные данные к работе**

3.1. Žmolíková K., Delcroix M., Ochiai T., Kinoshita K., Cernocký J., Yu D. Neural Target Speech Extraction: An Overview. // IEEE Signal Processing Magazine, Volume: 60, 2023. – С. 8–29.

3.2. Libri-CSS. [Электронный ресурс] URL: https://github.com/chenzhuo1011/libri_css (дата обращения: 16.04.2024 г.).

3.3. Delcroix M., Žmolíková K., Ochiai T., Kinoshita K., Nakatani T. Speaker activity driven neural speech extraction. // ICASSP 2021 – 2021 IEEE International Conference on Acoustics, Speech and Signal Processing, 2021. – С. 6099–6103.

3.4. Žmolíková K., Delcroix M., Kinoshita K., Ochiai T., Nakatani T., Burget L., Cernocký J. SpeakerBeam: Speaker Aware Neural Network for Target Speaker Extraction in Speech Mixtures. // IEEE Journal of Selected Topics in Signal Processing, Volume: 13, 2019. – С. 800–814.

4. Перечень подлежащих разработке вопросов

4.1. Провести обзор тематической литературы и аналогичных моделей.

4.2. Подготовить обучающий набор данных для референсной модели.

4.3. Разработать нейросетевую модель.

4.4. Провести эксперименты реализованной модели.

4.5. Реализовать веб-приложение для выделения целевого голоса.

5. Дата выдачи задания: 29.01.2024 г.

Научные руководители:

доцент кафедры СП, к.ф.-м.н. С.У. Турлакова

ст. преподаватель кафедры СП Н.С. Силкина

Задание принял к исполнению В.И. Капичай

ГЛОССАРИЙ

1. *TSE (Target Speech Extraction)* – метод выделения речевого сигнала диктора из смеси с несколькими дикторами с шумами и реверберациями или без них, используя подсказки, которые идентифицируют говорящего в смеси [1].

2. *Подсказка* – вспомогательные сигналы – по типу видео с изображением лица и голоса диктора, пространственные подсказки, которые определяют направление говорящего или предварительно записанный экземпляр голоса [1].

3. *Смесь* – аудио поток или аудио файл, на котором присутствуют перекрестные голоса двух и более дикторов [1].

4. *ASR (Automatic Speech Recognition)* – метод технологии машинного обучения или искусственного интеллекта для обработки человеческой речи в читаемый текст [2].

5. *LLM (Large Language Model)* – модель машинного обучения, которая способна понимать и генерировать текст на человеческом языке [3].

6. *Метрика* – количественная оценка эффективности реализованной и обученной модели [4].

7. *Оракул* – воображаемый источник знаний о целевой функции – источник обучающих/тестовых выборок, используемых в каких-то интуитивных доказательствах или мысленных экспериментах. [5].

8. *Эмбеддинг* – относительно низкоразмерное пространство, в которое можно перевести высокоразмерные векторы. Эмбеддинги облегчают машинное обучение на больших входных данных, таких как разреженные векторы, представляющие слова. В идеале они отражают часть семантики входных данных, располагая семантически схожие входные данные близко друг к другу в их пространстве [6].

9. *Диаризация* – процесс разделения звука с учетом принадлежности к диктору [7].

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Разработка нейросетевого веб-приложения для выделения целевого голоса является важной задачей в контексте современных технологий обработки аудиоданных и области искусственного интеллекта, в частности нейросетей. Задача выделения целевого голоса состоит [1] в том, чтобы выделить голос целевого диктора из аудиофайла, либо аудио потока, с помехами в виде речи нескольких дикторов, фонового шума, различных искажений и перекрывающихся аудио источников, наподобие того, как это делают люди, когда слышат несколько человек одновременно в людном месте [8]. Основная цель распознавания диктора – идентифицировать или подтвердить личность говорящего на основе его уникальных голосовых характеристик, которые часто называют «отпечатками голоса» или «биометрическими подписями» [9]. Возможность выделять целевой голос из таких сложных условий имеет огромный потенциал [1] в различных областях, таких как аудио обработка, коммуникации, музыкальная индустрия, аудио аналитика и другие. Корректное решение данной задачи может значительно улучшить качество последующей обработки и анализа аудио.

На данный момент существуют различные подходы для решения задачи выделения целевого голоса в аудио. Некоторые методы [10] основаны на использовании классических сигнальных обработок, таких как фильтрация и преобразование частоты. Однако, такие подходы часто ограничены в своей эффективности и не могут достичь высокой точности и качества в сложных условиях записи. Нейросети же позволяют извлекать абстрактные и сложные признаки из данных и находить зависимости между этими признаками и выделяемым голосом. За последние годы нейросетевые технологии достигли [11] большого успеха в решении данной задачи, но не все реализации качественно справляются ней, а также не все из них имеют удобный и интуитивно понятный интерфейс при взаимодействии с пользователем.

Постановка задачи

Целью выпускной квалификационной работы является исследование возможности выделения целевого голоса на основе нейросетевого подхода и реализация веб-приложения для выделения целевого голоса.

Для достижения поставленной цели необходимо решить следующие задачи:

1) выполнить обзор алгоритмов выделения целевого голоса, метрик и тематической литературы;

2) сформировать обучающий набор данных, осуществить требуемую предобработку данных;

3) реализовать несколько моделей, протестировать их работоспособность;

4) провести эксперименты на реальных данных, сравнить эффективность реализованных моделей с двумя и тремя перекрестными дикторами;

5) реализовать веб-приложение для выделения целевого голоса.

Структура и содержание работы

Работа состоит из введения, трех глав, заключения, списка литературы и приложений. Объем работы составляет 58 страниц, объем списка литературы – 41 источник.

В первой главе описывается обзор научной литературы, обзор алгоритмов выделения целевого голоса, обзор метрик и теоретические основы выделения целевого голоса.

Во второй главе предоставляются обзор актуальных наборов данных и ресурсов в области выделения целевого голоса, реализованные и референсная модели, а также сравнение их эффективности.

В третьей главе описывается реализация веб-приложения, его проектирование и тестирование.

В приложении содержатся листинги кодов реализованной TSE модели и маршрутизаций на страницы веб-приложения.

1. ОБЗОР ЛИТЕРАТУРЫ И АНАЛОГОВ

1.1. Обзор литературы

В начале 2023 года авторы SpeakerBeam [12] и ADEnet [13], известные своими моделями выделения целевого голоса, речь о которых пойдет в следующем подразделе, выпустили статью-обзор о выделении целевого голоса с помощью нейросетей под названием «Neural Target Speech Extraction: An Overview» [1]. В ней рассказывается о последних подходах выделения целевого диктора/речи на основе нейронных сетей, обсуждается проблема коктейльной вечеринки, а сама статья представляет из себя подробный обзор. Работа проводит читателей через различные основные подходы, подчеркиваются сходства между ними и обсуждаются потенциальные будущие направления.

Главными проблемами в достижении выделения целевого голоса, по мнению авторов, являются:

1) речь, записанная с помощью удаленного микрофона;
2) проблема выделения целевого голоса, ее связь со слепым разделением источников и шумоподавлением;
3) исторический контекст.

В статье также обсуждаются различные варианты реализации кодировщика подсказок, подводятся итоги разработки TSE на основе аудио сигнала и представляются некоторые репрезентативные экспериментальные результаты. Показывается, как модели в системах TSE извлекают из движений губ подсказки о состоянии целевой речи, например, говорит ли целевой диктор или молчит, или более точную информацию о произносимой фонеме. Обсуждается, что доступ к многоканальным записям открывает возможность выделения целевых дикторов на основе их местоположения, т. е. с помощью пространственных подсказок, а также, что идеи TSE могут быть применены к другим задачам обработки речи таких как ASR и диаризация.

В сентябре 2023 года студенты из Института инженеров электротехники и электроники (IEEE) выявили влияние вспомогательной информации

на производительность систем извлечения диктора из аудио смесей и предоставили свои результаты в статье «New Insights on Target Speaker Extraction» [14]. В экспериментах сравнивается производительность двух систем извлечения диктора (основанных на аудио и видео) с производительностью систем без использования такой информации. Результаты наглядно показывают, что использование вспомогательной информации не всегда приводит к улучшению производительности по сравнению с методами разделения дикторов без нее. Исследование также выявляет, что способ обучения систем извлечения диктора существенно влияет на способ использования вспомогательной информации. Данное исследование предоставляет ценные инсайты в области развития систем извлечения диктора, а также подчеркивает важность дальнейших исследований для более эффективного использования вспомогательной информации в подобных системах.

1.2. Обзор моделей выделения целевого голоса

SpeakerBeam

В августе 2019 года команда исследователей из Японии и Чехии представила свою работу «SpeakerBeam: Speaker Aware Neural Network for Target Speaker Extraction in Speech Mixtures» [15], в которой был представлен метод выделения целевого голоса. Нейронная сеть SpeakerBeam может быть обучена на различных дикторах и использована для извлечения дикторов, не замеченных во время обучения. Дополнительная информация о дикторе, определяющая целевого диктора, получается из адаптационного высказывания, произнесенного целевым диктором. На практике это адаптационное высказывание может быть получено, например, из части разговора без дублирования или предварительно записано целевым пользователем на его/ее персональное устройство. Команда решала две основные проблемы:

1) как использовать информацию о дикторе для изменения поведения нейронной сети;

2) как извлечь информацию о дикторе из адаптационного высказывания.

В работе проводилась экстракция в области короткопериодного преобразования Фурье (STFT) по формуле (1), где может моделироваться процесс смешивания:

$$Y^{(m)}[t,f] = S_0^{(m)}[t,f] + \sum_{i=1}^{I-1} S_i^{(m)}[t,f] + V^{(m)}[t,f], \quad (1)$$

где $[t,f]$ – индексы, соответствующие временному кадру, а также частотному бину;

Y, S_i, V – STFT-доменные аналоги y, s_i, v, соответственно.

Обозначения Y, S_i, V используются для матриц T × F, включающих все временные и частотные точки $Y[t,f]$, $S_i[t,f]$, $V[t,f]$, соответственно, при этом T – это количество временных кадров, а F – количество частотных бинов в STFT-представлении данного сигнала.

Метод используется для одноканального случая (в этом случае индекс (m) может быть опущен) и выделяет целевого диктора из смеси, используя дополнительную информацию о целевом дикторе в виде адаптированного произнесения.

На рисунке 1 показана общая схема одноканального извлечения для примера с одним мешающим диктором и шумом.

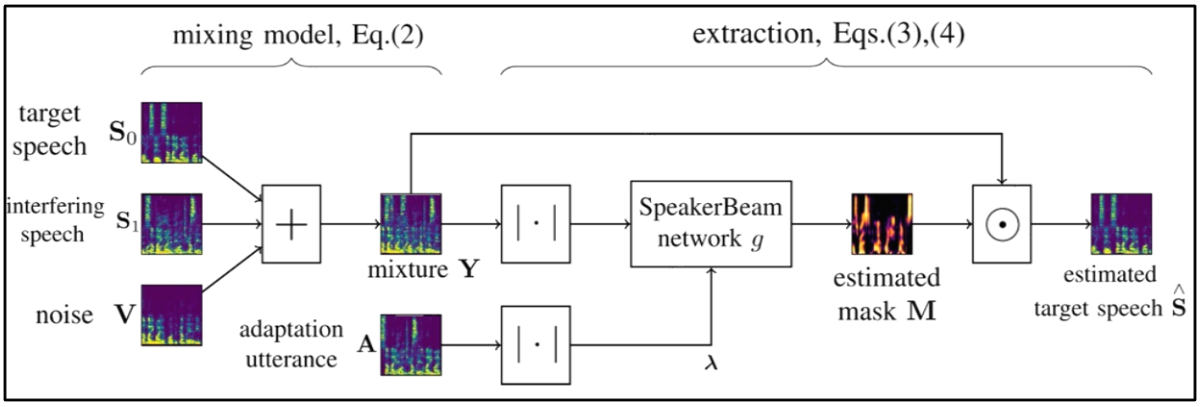

Рисунок 1 – Общая схема одноканального извлечения для примера с одним мешающим диктором и шумом

В результате экспериментов, исследователи пришли к выводу, что SpeakerBeam способна обрабатывать аудио с тремя дикторами и существует возможность расширения метода до многоканальной обработки совместного обучения с системой автоматического распознавания речи.

ADEnet

В феврале 2021 года та же команда представила свою работу «Speaker Activity Driven Neural Speech Extraction» [13]. Исследователи разработали нейронную сеть для извлечения речи на основе активности диктора (ADEnet), которая, по их заверениям, может достичь конкурентоспособного уровня производительности без необходимости предварительной записи. В работе отмечено, что одноканальное разделение речи значительно продвинулось с внедрением глубокого обучения, однако большинство подходов к разделению страдают от двух ограничений: они требуют знания или оценки количества дикторов в смеси и они страдают от проблемы глобальной неоднозначности перестановки, т. е. произвольного отображения между исходными дикторами и выходами. Выделение целевой речи было предложено в качестве альтернативы разделению речи, чтобы смягчить вышеуказанные ограничения. Она сосредоточена на извлечении только речевого сигнала интересующего диктора, используя вспомогательные подсказки о нем. Таким образом, постановка задачи становится независимой от количества дикторов в смеси. Кроме того, благодаря использованию вспомогательных подсказок естественным образом решается глобальная неоднозначность перестановки. Было предложено несколько схем выделения целевой речи, использующих различные типы вспомогательных подсказок, таких как предварительно записанные фразы целевого диктора, информация о направлении, видео целевого диктора или сигналы электроэнцефалограммы (ЭЭГ).

Авторы предложили одноканальную нейронную сеть для извлечения речи на основе активности дикторов (ADEnet) и экспериментально продемонстрировали, что при условии наличия оракула активности диктора

ADEnet может достичь более высокой производительности, чем SpeakerBeam, не требуя предварительно записанной речи.

Исследователи рассмотрели три конфигурации ADEnet. Первая конфигурация использует ту же архитектуру, что и SpeakerBeam, но заменяет произнесенное при записи высказывание входной смесью, а операцию усреднения по времени – взвешенной суммой. Альтернативный вариант состоит в том, чтобы просто соединить активность диктора и речевую смесь на входе сети. Этот подход не вычисляет никакого явного вектора встраивания диктора, а предполагает, что сеть извлечения может научиться отслеживать и идентифицировать целевого диктора внутри себя на основе его активности, следовательно, он не использует никаких вспомогательных сетей. Третья конфигурация представляет собой комбинацию обоих подходов, где используется активность диктора на входе, а также вычисляется вектор встраивания диктора.

В качестве набора данных для всех экспериментов авторы использовали одни и те же обучающие данные. Они состояли из 63 000 смесей реверберирующих голосов двух дикторов с фоновым шумом при SNR (отношение сигнал/шум) от 10 до 20 дБ. Речевые сигналы были взяты из набора данных LibriSpeech. Далее они создали валидационный и тестовый наборы с теми же условиями. Тестовый набор состоял из 1 364 смесей со средним коэффициентом перекрытия 38,5%. Во втором эксперименте использовали корпус LibriCSS, состоящий из 8 дикторских сессий по 10 минут, полученных путем перезаписи произнесений LibriSpeech, воспроизводимых дикторами в конференц-зале. Коэффициент перекрытия варьируется в пределах от 0% до 40%.

На рисунке 2 показан пример выделенной речи с помощью третьей конфигурации ADEnet.

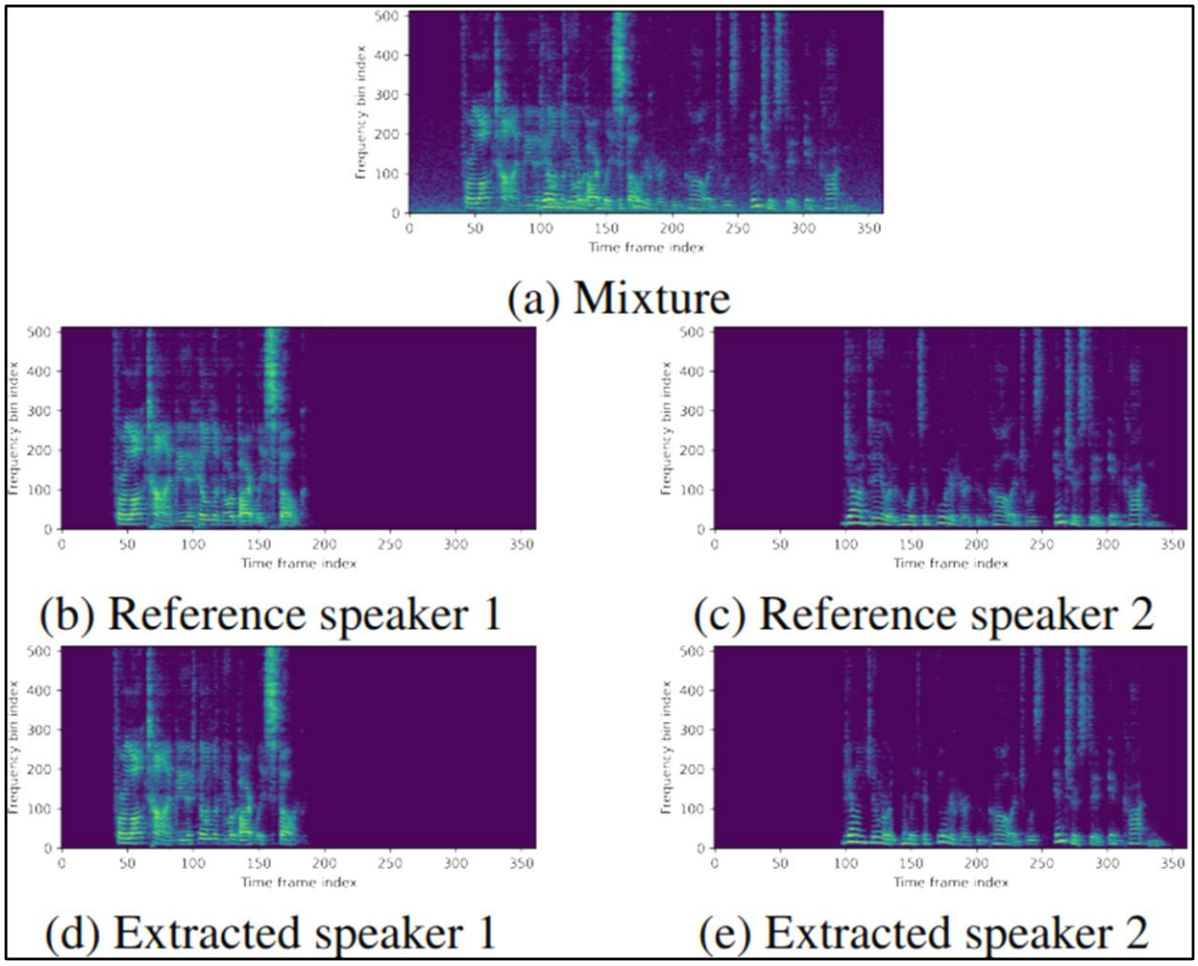

Рисунок 2 – Пример выделенной речи с помощью третьей конфигурации ADEnet

В результате исследования авторы экспериментально показали, что предложенный ADEnet с точной информацией об активности диктора может достичь высоких показателей извлечения одноканальной речи. Кроме того, в сочетании с диаризацией он может значительно улучшить производительность ASR в ситуациях, подобных корпоративной встрече. Диаризация находила речевую активность дикторов в записи. Из результатов диаризации авторы обрабатывали каждый сегмент непрерывной речи отдельно, добавляя по 2 секунды контекста в начале и конце, чтобы создать расширенные сегменты. Они использовали активность диктора, оцененную диаризацией в расширенном сегменте, в качестве сигнала активности диктора. Кроме того, экспериментировали с активностью дикторов без перекрытия,

удаляя из сигнала активности те области, где было обнаружено несколько дикторов. ASR выполнялся отдельно для каждого речевого сегмента, найденного с помощью диаризации, и оценивался с помощью конкатенированного минимально-пермутационного коэффициента ошибок слов (cpWER) [16], который включает ошибки, вызванные диаризацией.

В выводе авторы подтверждают, что ADEnet последовательно улучшает cpWER, особенно для областей с большим количеством перекрытий, с относительным улучшением WER в диапазоне от 6,5 % до 25 %. Результаты работы моделей представлены в таблице 1.

Таблица 1 – Метрика cpWER, полученная для диаризованного вывода с ADEnet и без него

	w/o overlap	**Overlap ratio in %**						
		0L	0S	OV10	OV20	OV30	OV40	Avg
No proc	na	11,2	9,4	16,2	23,1	33,6	41,1	23,9
ADEnet	–	**10,5**	**8,5**	13,8	18,7	26,5	30,7	19,2
ADEnet	+	**10,5**	8,6	**13,6**	**18,3**	**25,8**	**29,8**	**18,8**

LLM-TSE

Выделить целевой голос возможно не только благодаря подсказкам по видео или удаленности диктора от микрофона, но и при внедрении большой языковой модели (LLM) в качестве модуля, которая будет отвечать за определение задачи. Таким образом, возможно задать LLM задачу выделить, к примеру, самого тихого диктора, или диктора женского пола, если в смеси присутствуют разнополые дикторы. Подобное решение было предложено К. Хао, Д. Ву, Д. Ю, Ч. Ксу и К.Ч. Таном из Политехнического Университета Гонконга. В работе от 15 октября 2023 года «Typing to Listen at the Cocktail Party: Text Guided Target Speaker Extraction» [17] авторы предложили модель LLM-TSE, в которой большая языковая модель извлекает полезные семантические подсказки из вводимого пользователем текста. Вероятно, эта работа является первой, в которой применение LLM успешно справляется с

выделением целевого голоса. Сравнение между традиционной системой TSE и моделью LLM-TSE представлено на рисунке 3.

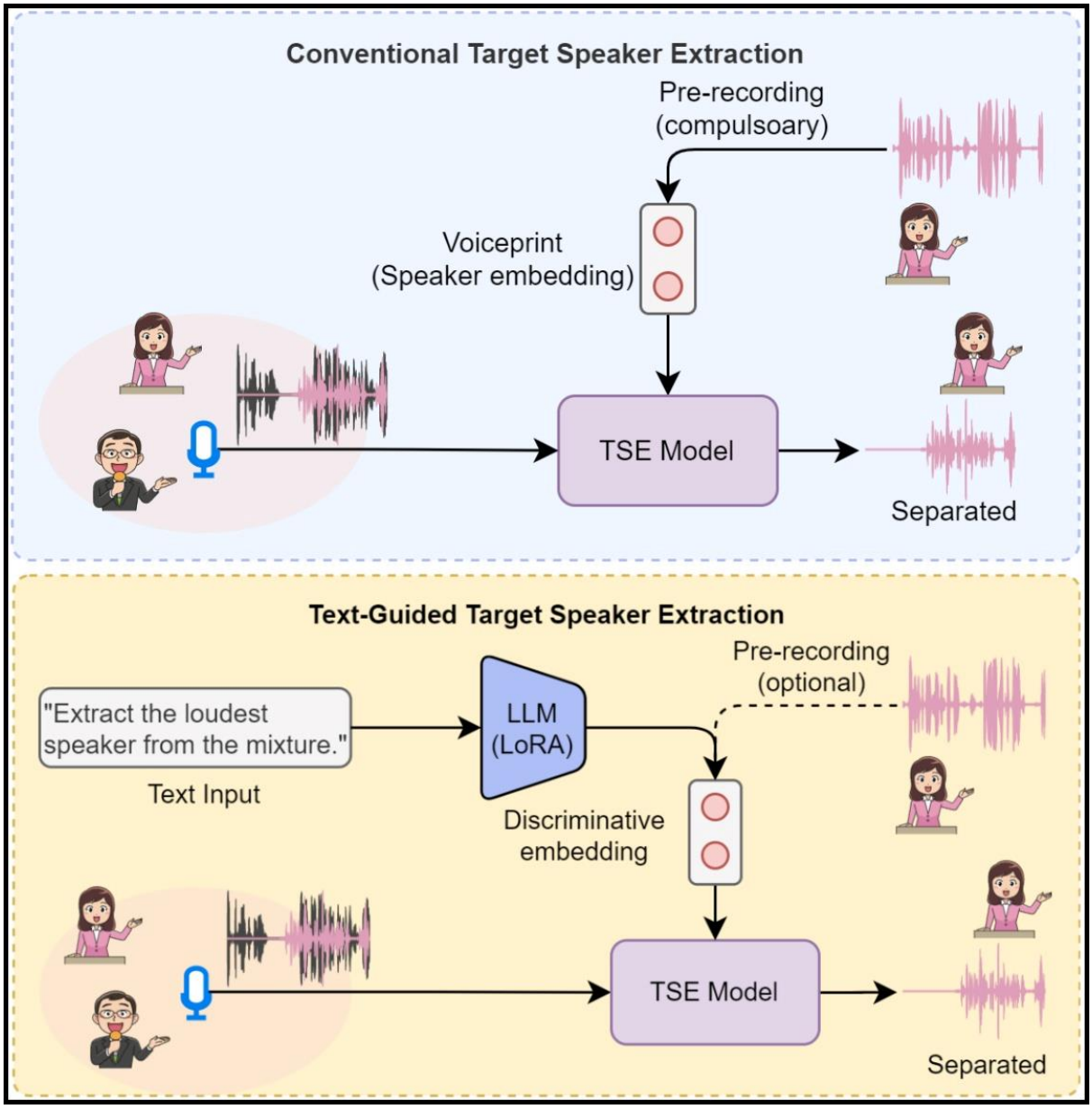

Рисунок 3 – Сравнение между традиционной системой TSE и моделью LLM-TSE

Авторы также отмечают проблему в работах, описанных выше. Используя заранее записанный голос диктора, который необходимо выделить, извлекается только он один и избегается проблема неизвестного количества дикторов, при этом эти записи могут существенно отличаться от базовой аудиозаписи со смесью голосов, из которой нужно выделить целевого диктора. Модель LLM-TSE с текстовым управлением была разработана, чтобы

избежать этих проблем. Модель включает текстовые описания в качестве дополнительных подсказок для улучшения реализуемости, управляемости и производительности существующих моделей TSE. Эти текстовые описания охватывают различные аспекты слухового восприятия человека, включая характеристики диктора, язык, содержание разговора, характеристики помещения и т. д. Благодаря этому производительность моделей TSE значительно повышается в различных сценариях.

Модель LLM-TSE работает по схеме «кодирование – слияние – извлечение – декодирование». На этапе кодирования используются три различных кодера для преобразования предварительно зарегистрированной речи, текстовых подсказок и входной звуковой смеси в соответствующие вкрапления. Используя объединенные вкрапления, представляющие зарегистрированную речь и текстовые подсказки, экстрактор затем выборочно извлекает нужный источник звука из входной звуковой смеси. Наконец, представление признаков в частотной области, полученное с помощью экстрактора, преобразуется обратно во временную область и выдается в виде извлеченной речи. Архитектура модели представлена на рисунке 4, на котором представлены аудио кодировщик, текстовый кодировщик и экстрактор.

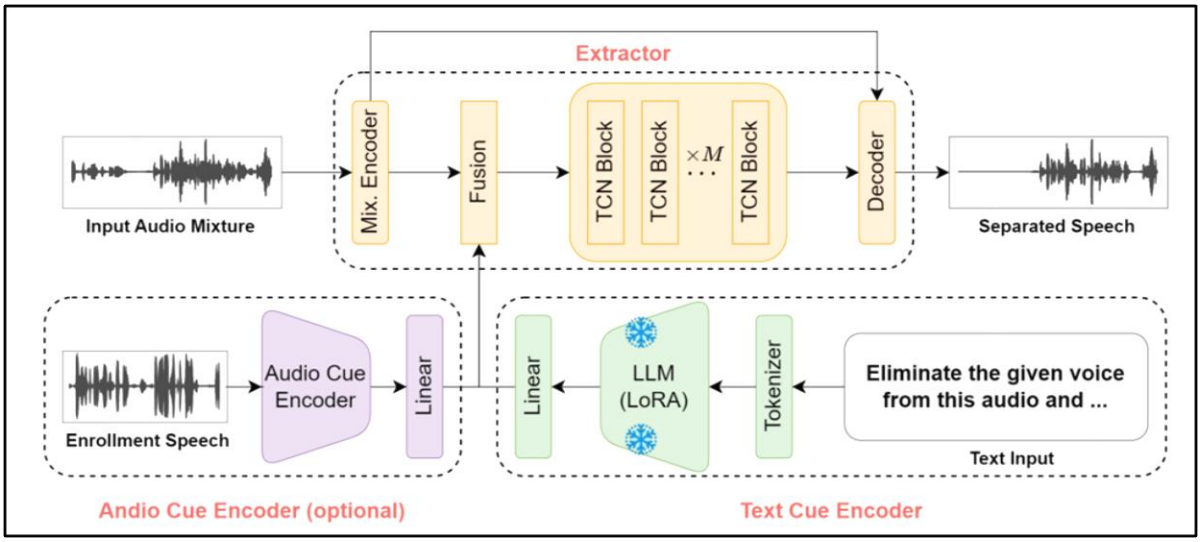

Рисунок 4 – Архитектура модели LLM-TSE

В экспериментах использовались два набора данных: LibriSpeech и Multilingual LibriSpeech (MLS). Авторы выбрали 400 случайных дикторов из MLS содержащих до 20 высказываний в каждом и придерживались стандартного для LibriSpeech разделения обучающих, проверочных и тестовых наборов. Для MLS исследователи случайным образом распределили 5 % дикторов из каждого языка в валидационный и тестовый наборы соответственно, а остальные – в тренировочный. Эксперименты охватывали множество атрибутов, включая фрагменты транскрипции, пол, язык, громкость и удаленность от микрофона.

На рисунке 5 представлены результаты экспериментов и сравнение работы моделей с TD-SpeakerBeam, на котором была построена модель LLM-TSE. В качестве метрики было взято масштабно-инвариантное отношение сигнал/искажение (SI-SDR) [18].

Как показано на рисунке 5, модель LLM-TSE эффективнее справляется с задачей выделения целевого голоса, чем рассмотренная выше SpeakerBeam, а также авторы пришли к выводу, что использование более мощного LLM в качестве кодировщика текстовых подсказок может значительно повысить дискриминационные возможности всей системы.

Entry	Inputs		Transcription Snippet			Gender	Language	Far-near	Loudness
	Audio	Text	50%	80%	100%				
Unproc.	-			-0.02		-0.02	-0.03	-0.01	-0.10
TD-SpeakerBeam	✓	✗		7.21		10.15	8.38	9.38	7.57
LLM-TSE (LoRA Adapters, LLaMA-2 7B Chat)	✓	✗		7.30		10.17	8.87	9.77	7.75
	✗	✓	2.70	3.97	7.48	10.40	9.38	10.57	8.89
	✓	✓	7.96	9.81	10.05	10.87	9.72	10.66	9.41
No LoRA Adapters (only Linear Projection)	✗	✓	1.66	3.38	5.38	8.76	7.38	8.45	5.46
	✓	✓	4.85	7.60	7.98	9.02	7.97	8.67	7.11
Use Vicuna-7b-v1.3 (Zheng et al. (2023))	✗	✓	2.23	3.31	8.79	9.44	8.29	9.27	5.75
	✓	✓	7.41	9.05	9.35	10.15	9.01	9.94	6.47

Рисунок 5 – Оценка метрики SI-SDR (дБ ↑) различными методами

1.3. Обзор метрик

Превалирующее число TSE систем оцениваются с помощью таких показателей как SNR (отношение сигнал/шум) или SDR (отношение сигнал/искажение) [12, 15, 17]. Несмотря на то, что эти показатели указывают на эффективность извлечения, существуют две проблемы при их использовании. Первая проблема заключается в том, что эти показатели не всегда коррелируют с восприятием и разборчивостью человеком. Вторая проблема – система TSE должна идентифицировать целевую речь, подразумевая другие источники ошибок. Неспособность идентифицировать целевую речь может привести к неправильной оценке голоса диктора с помехами или неточному выводу смеси. Эти ошибки напрямую влияют на показатель SDR. Было бы полезно согласовать показатели оценки, которые разделяют производительность извлечения и идентификации, чтобы лучше выявить поведение систем TSE. Показатели уровня сигнала могут плохо отражать производительность извлечения для случаев с неактивными дикторами.

В ходе поиска оптимальных метрик было выявлено, что специалисты используют SI-SDR (масштабно-инвариантное отношение сигнал/искажение) для выявления качества речи и ESTOI (расширенный показатель кратковременной объективной разборчивости) для определения разборчивости [19]. ESTOI вычисляет корреляцию между чистыми и искаженными спектрами, чтобы можно было обнаружить «проблески чистой речи».

Перцептивная оценка качества речи (PESQ) также является одной из метрик для оценки качества речи. Она сравнивает исходный сигнал с полученной для него оценкой. Описание вычисления PESQ приводится в работе [20]. На практике величина PESQ находится в диапазоне от – 0,5 до 4,5, более высокие значения указывают на лучшее качество.

1.4. TSE модель

1.4.1. Подсказки для TSE моделей

Для качественного извлечения голоса целевого диктора из смеси, в моделях выделения целевого голоса применяются подсказки. Подсказками являются вспомогательные сигналы – по типу видео с изображением лица и голоса диктора, пространственные подсказки, которые определяют направление говорящего или предварительно записанный экземпляр голоса. TSE связано с другими задачами обработки речи и звука, такими как шумоподавление и разделение источников звука вслепую (BSS), которые не используют подсказки о целевом дикторе.

Звуковая подсказка состоит из записи речевого сигнала целевого говорящего. Такая подсказка может быть полезна, например, в случае использования персональных устройств, где пользователь может предварительно записать пример своего голоса. В качестве альтернативы, для длинных записей, таких как совещания, подсказки могут быть получены непосредственно из части записи. Интерес к звуковым подсказкам резко возрос в последнее время с использованием нейронных моделей для TSE.

Звуковые подсказки, пожалуй, самые универсальные, поскольку они не требуют использования каких-либо дополнительных устройств, таких как несколько микрофонов или камера. Однако производительность может быть ограниченной по сравнению с другими подсказками, поскольку распознавание говорящих только по их голосовым характеристикам подвержено ошибкам из-за вариабельности между говорящими и внутри группы. Например, голосовые характеристики разных говорящих, таких как члены семьи, часто очень похожи друг на друга. С другой стороны, характеристики голоса одного говорящего могут меняться в зависимости от таких факторов, как эмоции, состояние здоровья или возраст.

Визуальная подсказка состоит из видеозаписи разговора целевого диктора. Этот тип часто ограничивается лицом говорящего, иногда только областью губ. В отличие от звуковых подсказок, визуальные подсказки

обычно синхронизируются со звуковыми сигналами, которые обрабатываются, т.е. не записываются предварительно. В некоторых работах также исследовалось простое использование фотографии говорящего [21]. Визуальные подсказки использовались для определения характера активности и местоположения целевого диктора или для совместного моделирования аудио и визуальных сигналов. В недавних работах [22, 23] обычно используются визуальные подсказки, чтобы направлять дискриминационные модели к выделению целевого диктора. Визуальные подсказки особенно полезны, когда у говорящих в записи похожие голоса. Однако они могут быть чувствительны к физическим препятствиям со стороны диктора на видео.

Пространственная подсказка относится к местоположению целевого диктора, например, угол, под которым расположены записывающие устройства. На практике местоположение можно определить по видеозаписи помещения или записи выступающего в том же положении. Извлечение голоса дикторов на основе их местоположения исследовалось с середины 1980-х годов с использованием методов формирования луча, которые стали пионерами в этой области [24]. Наконец, несколько работ показали, что нейронные сети, информированные о местоположении, также могут достигать многообещающих результатов [25]. Пространственные подсказки по своей сути применимы только тогда, когда доступна запись с нескольких микрофонов. Однако они могут довольно надежно идентифицировать целевого диктора в смеси, особенно когда они неподвижны.

1.4.2. Общая структура TSE модели

Модель TSE состоит из нейронной сети, которая оценивает целевую речь с подсказкой. На рисунке 6 представлена принципиальная схема универсальной нейронной системы TSE, которая состоит из двух основных модулей: кодера подсказки и модуля извлечения речи.

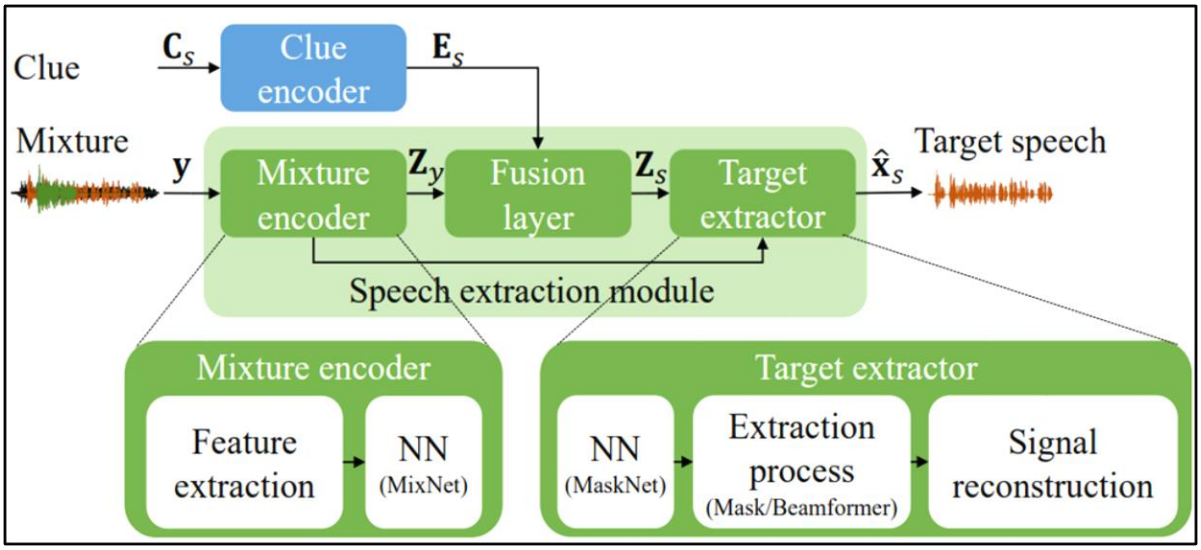

Рисунок 6 – Общая структура нейросетевой модели TSE

Кодер подсказки, представленный в формуле (2), извлекает информацию, которая позволяет модулю извлечения речи идентифицировать и извлечь целевую речь из смеси. Обработку можно выразить как:

$$E_s = ClueEncoder(C_s; \theta^{Clue}), \quad (2)$$

где $ClueEncoder(C_s; \theta^{Clue})$ – кодер подсказки, который может быть нейронной сетью с обучаемыми параметрами θ^{Clue};

E_s – эмбеддинг подсказки.

Модуль извлечения речи оценивает целевую речь из смеси, учитывая эмбеддинги целевых дикторов. Можно использовать одну и ту же конфигурацию независимо от типа подсказки. Процесс можно разделить на три основные части: кодировщик смеси (формула 3), слой слияния (формула 4) и целевой экстрактор (формула 5) с параметрами θ^{Mix}, θ^{Fusion}, $\theta^{TgtExtractor}$:

$$Z_y = MixEncoder\left(y; \theta^{Mix}\right), \quad (3)$$

$$Z_s = Fusion\left(Z_y, E_s; \theta^{Fusion}\right), \quad (4)$$

$$\hat{x}_s = TgtExtractor(Z_s, y; \theta^{TgtExtractor}), \quad (5)$$

где Z_y, Z_s – внутренние представления смеси до и после настройки на эмбеддинг E_s.

Средство извлечения признаков вычисляет признаки на основе наблюдаемого сигнала смешивания по формуле (6). Это могут быть такие спектральные характеристики, как коэффициенты спектра амплитуды, полученные из кратковременного преобразования Фурье (STFT) входной смеси, либо процесс выделения признаков, реализованный с помощью нейронной сети, такого как одномерный сверточный слой, который работает непосредственно с необработанной входной формой сигнала микрофона. Затем объекты обрабатываются с помощью нейронной сети $MixNet$, которая выполняет нелинейное преобразование и фиксирует временной контекст, т.е. несколько прошлых и будущих кадров сигнала. Результирующее представление смеси Z_y (формула 7) не зависит от цели.

Кодировщик смеси выполняет следующие действия:

$$Y = FE(y; \theta^{FE}), \tag{6}$$

$$Z_y = MixNet\left(Y; \theta^{MixNet}\right), \tag{7}$$

где FE – средство извлечения признака с параметром θ^{FE};

$MixNet$ – средство извлечения признака с параметром θ^{MixNet}.

Слой слияния, иногда обозначаемый как слой адаптации, является ключевым компонентом системы TSE и позволяет привязать процесс к подсказке. Слой сочетает в себе Z_y с эмбеддингами подсказок E_s.

Последней частью модуля извлечения речи является целевой экстрактор, который оценивает целевой сигнал. Энергия речевого сигнала сосредоточена в нескольких частотно-временных диапазонах речевого спектра. Соответственно, речевые сигналы разных носителей редко перекрываются в частотно-временной области в речевой смеси. Таким образом, можно извлечь целевую речь, применив частотно-временную маску к наблюдаемой речевой смеси, где маска указывает на частотно-временные интервалы, в которых целевая речь доминирует над другими сигналами. На рисунке 7 показан пример идеальной бинарной маски для извлечения целевой речи из

смеси двух дикторов. Такая идеальная бинарная маска предполагает, что вся мощность в каждом TF-бине принадлежит одному диктору.

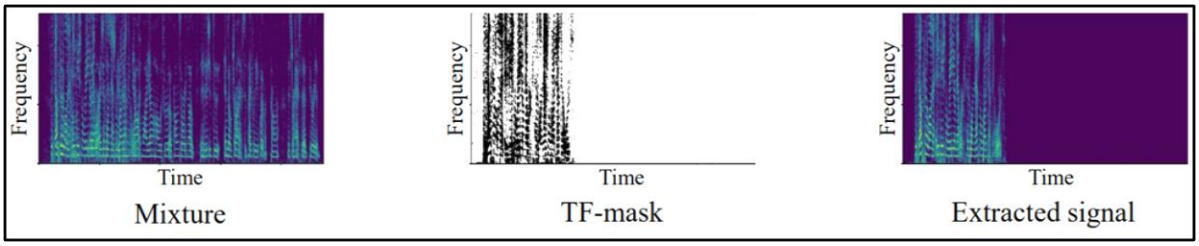

Рисунок 7 – Смесь, идеальная маска и выделенный сигнал

Процесс работы экстрактора (формулы 8–10) на основе маскировки можно суммировать следующим образом:

$$M_s = MaskNet(Z_s; \theta^{Mask}), \quad (8)$$

$$\hat{X}_s = M_s \odot Y, \quad (9)$$

$$\hat{x}_s = Reconstruct\left(\hat{X}_s; \theta^{Reconst}\right), \quad (10)$$

где $MaskNet$ – нейронная сеть, которая оценивает частотно-временную маску для целевой речи;

θ^{Mask} – параметры сети;

$\odot$ – поэлементное умножение Адамара;

Y – смесь;

$\hat{X}$ – оцененные целевые речевые сигналы в области признаков;

$Reconstruct$ – операция восстановления сигнала во временной области путем выполнения операции, обратной извлечению признака кодера смеси.

Также существуют и другие возможности для выполнения процесса извлечения. Например, мы можем модифицировать $MaskNet$, чтобы напрямую определять целевой речевой сигнал в области признаков.

1.4.3. Обучение TSE модели

Прежде чем использовать модель TSE, необходимо узнать ее параметры: $\theta^{TSE} = \theta^{Mix}, \theta^{Clue}, \theta^{Fusion}, \theta^{TgtExtractor}$. В большинстве существующих исследований используется полностью контролируемое обучение,

которое требует большого количества обучающих данных, состоящих из триплетов речевой смеси y, целевого речевого сигнала x_s и соответствующей подсказки C_s для изучения параметров θ^{TSE}. Поскольку для этого требуется доступ к чистому целевому речевому сигналу, такие обучающие данные обычно моделируются путем искусственного смешивания чистого речевого сигнала и шума в соответствии с сигнальной моделью уравнения (11):

$$y^m = x_s^m + \underbrace{\sum_{k \neq s} x_k^m + v^m}_{\triangleq i^m}, \tag{11}$$

где y^m – сигнал смеси во временной области;

$x_s^m \in R^T$ – целевая речь;

$x_k^m \in R^T$ – речь с помехами;

$v^m \in R^T$ – зашумленный сигнал.

Переменная T представляет собой длительность (количество выборок) сигналов, m – индекс микрофона в массиве микрофонов, s – индекс целевого диктора и k – индекс других источников речи. Если используется только один микрофон, то индекс m можно опустить. В задаче TSE важно восстановить только целевую речь диктора, а все остальные источники рассматривать как нежелательные сигналы, которые необходимо подавить. Таким образом, мы можем определить сигнал помехи как $i^m \in R^T$.

На рисунке 8 показан процесс создания данных с использованием многоязычного аудиовизуального речевого набора данных, содержащего несколько видео для каждого диктора. В этом примере предполагается наличие видео, чтобы можно было генерировать аудио и визуальные подсказки. Для TSE на основе аудио подсказок видео не требуется. Для TSE на основе визуальных подсказок нам не обязательно требуется несколько видео с одним и тем же диктором.

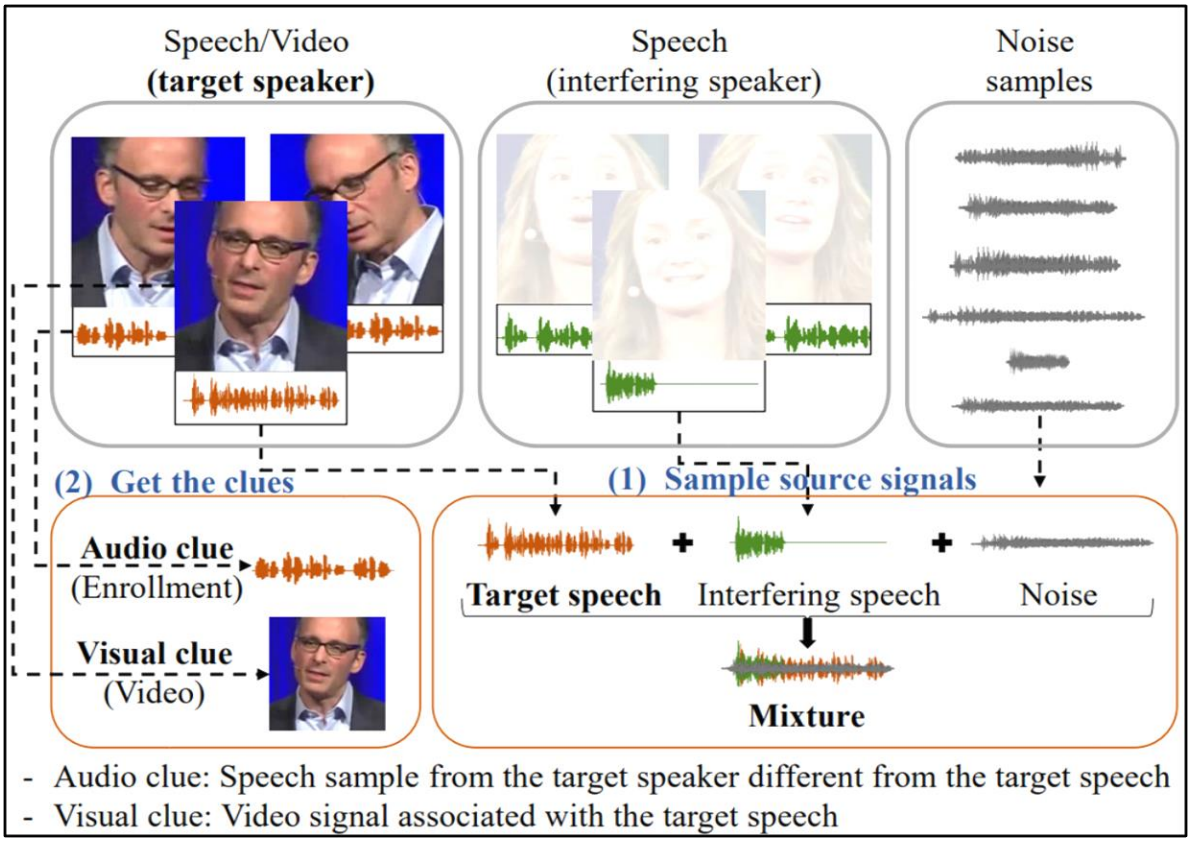

Рисунок 8 – Пример генерирования данных симуляции
для обучения или тестирования

Сначала мы генерируем смесь, используя случайно выбранные речевые сигналы целевого диктора, диктора-помехи и фоновый шум. Мы получаем звуковую подсказку, выбирая другой речевой сигнал от целевого диктора, а также визуальную подсказку из видеосигнала, связанного с целевой речью.

Обучение системы TSE происходит по схеме обучения нейронной сети с обратным распространением ошибки (формула 12). Параметры оцениваются путем минимизации функции потерь при обучении:

$$\theta^{TSE} = argmin_{\theta} L(x_s, \hat{x}_s), \quad (12)$$

где L – потери при обучении, которые измеряют, как близко оцениваемая целевая речь $\hat{x}_s = TSE(y, C_s, \theta)$ к целевому исходному сигналу x_s.

Существует несколько вариантов потерь, действующих в различных доменах, например, перекрестная энтропия между оракулом и оцененными

временно-частотными масками и средняя квадратичная ошибка (MSE) между спектрами магнитуды исходной и оцениваемой целевой речи. В последнее время широко используется отрицательное отношение сигнал/шум (SNR) (формула 13), измеренное во временной области [26]:

$$L^{SNR}(x_s, \hat{x}_s) = -10log_{10}\left(\frac{\|x_s\|^2}{\|x_s - \hat{x}_s\|^2}\right) \quad (13)$$

Потеря SNR вычисляется непосредственно во временной области, что заставляет систему TSE учиться правильно оценивать величину и фазу целевого речевого сигнала. Эта потеря улучшила производительность извлечения. Во многих работах также используются версии потерь, инвариантные к произвольному масштабированию, т.е. инвариантный по масштабу SNR (SI-SNR) [27] или линейная фильтрация оцененного сигнала, часто называемая отношением сигнал-искажение (SDR) [28].

Кодировщик подсказок может быть нейронной сетью, обученной совместно с модулем извлечения речи или предварительно обученной для другой задачи, например, идентификации диктора на основе аудио подсказок или чтения по губам для TSE на основе визуальных подсказок. Использование предварительно обученного кодировщика подсказок позволяет использовать большие объемы данных для обучения надежных и высокодискриминативных эмбеддингов. С другой стороны, совместная оптимизация кодировщика подсказок позволяет оптимизировать обучение эмбеддингов именно для TSE.

Выводы по первой главе

Выполнен обзор литературы и аналогов нейросетевых моделей для выделения целевого голоса, таких как SpeakerBeam, ADEnet, LLM-TSE. Описаны принципы работы, архитектуры и особенности моделей, а также метрики для оценки эффективности. Описаны подсказки, их разновидности и влияние в выделении целевого голоса, а также общая структура модели и схема обучения.

2. ВЫДЕЛЕНИЕ ЦЕЛЕВОГО ГОЛОСА

Для реализации нейросетевой модели выделения целевого голоса использовались следующие технологии:

1) язык программирования Python 3.11.7;

2) библиотека обработки аудио Asteroid на основе PyTorch для разделения голосов в смеси;

3) библиотека обработки аудио librosa;

4) нейросетевая библиотека SpeechBrain на основе PyTorch для распознавания диктора.

2.1. Изучение актуальных наборов данных и ресурсов

В качестве одного из наборов данных подходит LibriCSS [29]. LibriCSS состоит из удаленных микрофонных записей объединенных речевых высказываний LibriSpeech, воспроизводимых дикторами в офисном помещении, что позволяет оценить алгоритмы разделения речи, которые обрабатывают аудио в длинной форме. Коэффициент перекрытия варьируется от 0% до 40%.

Для некоторых задач разделения и распознавания речи требуется регистрация диктора. В LibriCSS есть список зарегистрированных дикторов для каждой мини-сессии. В libricss_speaker_info.json каждый элемент соответствует одному мини-сеансу, содержащему три ключа: «dataid», «used_uttid» и «unused_uttid», где «dataid» – это имя сеанса, в то время как «used_uttid» и «unused_uttid» содержат используемые и неиспользуемые идентификаторы высказываний, соответственно, для каждого диктора в десятиминутной записи мини-сеанса. Аудио могут быть загружены по ссылке [30] и имеют объем 6 Гб.

Авторы публикуют примеры выходных сигналов алгоритма CSS, а также скрипты, которые выполняют оценку ASR и WER для этих сигналов. Те, кто сосредоточен на интерфейсных алгоритмах разделения речи, смогут

протестировать свои собственные алгоритмы, заменив примеры сигналов своими собственными.

Второй набор данных, который можно использовать для экспериментов – LibriSpeech [31]. LibriSpeech является коллекцией из примерно 1 000 часов аудиокниг, которые являются частью проекта LibriVox. Большинство аудиокниг взято из проекта Project Gutenberg. Обучающие данные разделены на 3 части: наборы по 100 часов, 360 часов и 500 часов, а тестовые данные разделены на «чистые» и «другие» категории, соответственно, в зависимости от того, насколько хорошо или сложно будет работать системам автоматического распознавания речи. Каждый из тестовых и экспериментальных наборов имеет длительность около 5 часов. В этом наборе также представлены языковые модели n-грамм и соответствующие тексты, взятые из книг Project Gutenberg, которые содержат 803 миллиона лексем и 977 тысяч уникальных слов.

Набор данных в исходном виде не представляет из себя ресурс, который можно использовать для выделения целевого голоса, но при модернизации данных возможно накладывать друг на друга аудиодорожки, и таким образом получать аудио, в котором один диктор говорит поверх другого, после чего использовать данные для экспериментов по выделению целевого голоса.

Третьим набором данных является LibriMix [32]. LibriMix – это набор данных с открытым исходным кодом для разделения источников в шумных средах. Он основан на сигналах LibriSpeech и шуме WHAM [33]. Он предлагает бесплатную альтернативу набору данных WHAM и дополняет его. Это также позволит проводить эксперименты с перекрестными наборами данных. Набор данных используется для разделения речи, однако его можно использовать и для выделения целевого голоса, т.к. в данных происходит перекрытие голосов и шумы WHAM.

По умолчанию LibriMix будет сгенерирован для 2 и 3 дикторов, как на частоте дискретизации 16 кГц, так и на частоте 8 кГц, для режимов minmax,

и все типы микширования (mix_clean, mix_both и mix_single). Это составляет около 430 Гб данных для Libri2Mix и 332 Гб для Libri3Mix. Также потребуется сохранить LibriSpeech и wham_noise_augmented во время генерации для дополнительных 30 Гб и 50 Гб.

В LibriMix можно выбрать:

1) количество источников в смесях;

2) частоту дискретизации набора данных от 16 кГц до любой частоты ниже;

3) режим смесей: минимальный (смесь заканчивается, когда заканчивается самый короткий источник) или максимальный (смеси заканчиваются с самым длинным источником);

4) тип смеси: mix_clean (только высказывания) mix_both (высказывания + шум) mix_single (1 высказывание + шум).

Данный набор данных используется в модели SpeakerBeam и является наиболее подходящим для обучения этой модели выделения целевого голоса.

Существует возможность сгенерировать данные, а не брать уже созданные. Для этого может подойти WSJ0Mix generation scripts [34]. Скрипт создан из набора данных WSJ0 [35], который содержит аудио сигналы для 2 и 3 дикторов. Он является стандартным набором данных для задачи разделения дикторов, но при этом есть возможность создать аудио смесь с 4 и 5 дикторами. Несмотря на то, что набор разрабатывался для решения другой задачи, он может подойти для выделения целевого голоса. Аудио сигналы смешиваются с различными отношениями сигнал-помеха. Содержит только аудио данные, без соответствующих видеозаписей дикторов.

2.2. Референсная модель

Прежде чем приступать к разработке нейросетевой модели выделения целевого голоса, необходимо определиться с референсной моделью, с которой будут сравниваться результаты модели разработанной. В качестве такой

референсной модели была выбрана SpeakerBeam. У данной нейросетевой TSE модели есть ряд преимуществ:

1) простота запуска и обучения;
2) открытый и доступный код;
3) наличие предобученной модели;
4) хорошие результаты выделения целевого голоса.

Обучение модели нейронной сети, использующуюся для референса, проводящееся с использованием Jupyter Notebook, происходило на компьютере со следующими характеристиками:

1) GPU: GeForce RTX 3060Ti 8Gb;
2) CPU: 12th Gen Intel Core i5-12400F @ 6x2.5GHz;
3) общий объем оперативного запоминающего устройства: 32Gb;
4) операционная система: Ubuntu 22.04.4 LTS.

В качестве набора данных для обучения был выбран LibriMix для двух дикторов, размер набора данных составил 430 Гб. Было выявлено, что стандартные параметры не подходят для используемой аппаратной конфигурации, поэтому некоторые из них были изменены. Размер батча уменьшен с 6 до 2, количество эпох с 200 до 20, остальные параметры были не тронуты. В результате тринадцатичасового обучения, на выходе была получена модель, которая удовлетворительно выполняла свою задачу, но присутствовали артефакты и помехи, что не является хорошим примером для референса. В итоге было принято решение использовать предобученную стандартную модель для сравнения результатов, которая обладает высоким качеством выделения целевого голоса.

2.3. TSE модели

Первая TSE модель

Первая модель предназначена для выделения целевого голоса из смеси с двумя дикторами. Модель состоит из двух модулей:

1) модуль разделения голосов из смеси;

2) модуль распознавания голоса.

Модель реализована как Python файл process_audio.py. Полный код модели представлен в листинге 1 приложения.

Модуль диаризации

Перед подачей смеси первому модулю необходимо провести обработку аудиофайла. Библиотека Asteroid [36] выполняет разделение голосов в аудиофайлах только с форматом .wav и .mp3, и только в моно. Пользователь не должен самостоятельно менять формат и канальный формат файла, поэтому была реализована функция первоначальной обработки файла, функция представлена в листинге 1.

Листинг 1 – Функция преобразования кодировки и канального формата

```
def ctts(wav_file):
    mix, ctts_sr = sf.read(mixwav, dtype="float32", always_2d=True)
    if ctts_sr >= 16000:
        ctts_sr = 16000
        ctts_model_sr = "high"
    else:
        ctts_sr = 8000
        ctts_model_sr = "low"

    # checking the file for stereo or mono
    if len(mix.shape) == 2:
        # stereo to mono
        y, ctts_sr = librosa.load(mixwav, sr=ctts_sr)
        mix = librosa.to_mono(y)

    return sf.write(os.path.join(input_audio_dir, 'mixwav.wav'), data=mix,
samplerate=ctts_sr), ctts_sr, ctts_model_sr
```

В результате обработки файла вышеописанной функцией аудиозапись приводится к стандартной частоте дискретизации 8 000 Гц или 16 000 Гц, при этом назначая переменной `ctts_model_sr` значение «high» или «low», для выбора предобученной модели под конкретную частоту дискретизации, а переменной `ctts_sr` присваивается выбранная частота дискретизации для последующего корректного воспроизведения аудиозаписей, смесь приводится к канальному формату моно, и в конце файл сохраняется как новый, обработанный и приведенный к стандарту файл смеси.

После обработки смеси и приведения ее к стандарту идет автоматический выбор предобученной модели в зависимости от частоты дискретизации

оригинальной смеси. Если частота дискретизации оригинальной смеси больше или равна 16 000 Гц, то после обработки будет выбрана модель «JorisCos/ConvTasNet_Libri2Mix_sepclean_16k», иначе частота дискретизации изменяется до 8 000 Гц и будет выбрана модель «JorisCos/ConvTasNet_Libri2Mix_sepclean_8k».

Библиотека Asteroid может принимать NumPy массив или путь до файла с форматами .wav и .mp3. В первом случае, при помощи библиотеки soundfile смесь прочитывается и возвращается как shape (time, channels), в то время как Asteroid для разделения требует (batch, channels, time). Прочитанная смесь транспонируется и реформируется, после чего происходит разделение смеси на голоса. Данная реализация представлена на листинге 2.

Листинг 2 – Выбор предобученной модели и разделение смеси

```
if model_sr == "high":
    model = BaseModel.from_pretrained("JorisCos/Con-
vTasNet_Libri2Mix_sepclean_16k")
else:
    model = BaseModel.from_pretrained("JorisCos/Con-
vTasNet_Libri2Mix_sepclean_8k")
mixture, _ = sf.read(mixwav, dtype="float32", always_2d=True)
mixture = mixture.transpose()
mixture = mixture.reshape(1, mixture.shape[0], mixture.shape[1])
out_wavs = model.separate(mixture)
```

После разделения смеси на голоса с помощью IPython появляется возможность прослушать результаты диаризации и при помощи функции, представленной в листинге 3, посмотреть на спектрограммы разделенных голосов. Пример спектрограмм голосов представлен на рисунке 9.

Листинг 3 – Функция для отображения спектрограмм разделенных голосов

```
def show_magspec(waveform, **kw):
    return librosa.display.specshow(
        librosa.amplitude_to_db(np.abs(librosa.stft(waveform))),
        y_axis="log", x_axis="time",
        **kw
    )
fig, ax = plt.subplots(1, 2, figsize=(15, 5))
show_magspec(out_wavs[0][0], sr=sr, ax=ax[0])
show_magspec(out_wavs[0][1], sr=sr, ax=ax[1])
```

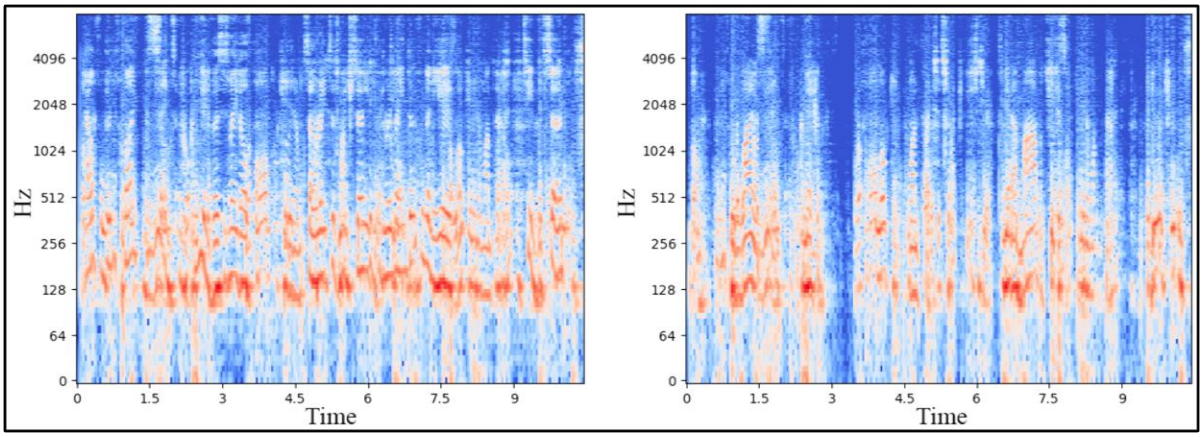

Рисунок 9 – Спектрограммы разделенных голосов

В конце первого модуля разделенные голоса записываются как отдельные аудио с установленной ранее частотой дискретизации и в формате моно.

Модуль верификации

Второй модуль использует библиотеку SpeechBrain [37], основанной на PyTorch, и позволяет распознать целевого диктора на основе аудио подсказки. На листинге 4 представлен модуль распознавания голоса, и ключевая функция `verify_files`, которая использует предобученную модель «spkrec-ecapa-voxceleb» и принимает на вход предполагаемый голос и аудио подсказку, которые сравнивает между собой. На выходе модель выдает оценку True/False и уверенность модели со значением от 0 до 1. В зависимости от результата верификации модель возвращает путь к файлу с выделенным голосом.

Листинг 4 – Модуль распознавания голоса

```
from speechbrain.inference.speaker import SpeakerRecognition
verification = SpeakerRecognition.from_hparams(source="speechbrain/spkrec-ecapa-voxceleb", savedir="pretrained_models/spkrec-ecapa-voxceleb")
score, prediction = verification.verify_files(voice1, sample)
```

Вторая TSE модель

Структурно вторая модель представляет из себя то же самое, что и первая модель. Она также использует двухмодульный принцип. Разница заключается лишь в выборе предобученной модели в первом модуле. Вместо моделей `ConvTasNet_Libri2Mix_sepclean_8k` и `ConvTasNet_Libri-`

2Mix_sepclean_16k используются ConvTasNetLibri3Mix-sepclean_8k и ConvTasNet_Libri3Mix_sepclean_16k. Результаты диаризации этих моделей оставляют желать лучшего: качество смеси ухудшается, а разделения голосов как такового не происходит, модели не справляются с задачей диаризации, поэтому было принято решение не включать модель для трех дикторов в конечное веб-приложение.

2.4. Сравнение реализованной модели с референсом

Для сравнения была выбрана смесь с двумя спикерами разных полов, частотой дискретизации 8 000 Гц, длиной в 5 секунд и шумом в виде музыки на заднем фоне.

Для демонстрации спектрограмм и качественных метрик был написан Python файл comparison.py (листинг 5), в котором с помощью библиотеки matplotlib выводятся спектрограммы, а метрики сохраняются в html файл при помощи библиотеки pandas. Для отображения используются смесь, выделенный голос и оригинальная запись голоса, на основе которой создана смесь.

Листинг 5 – Файл comparison.py

```
import matplotlib.pyplot as plt
import soundfile as sf
from asteroid.metrics import get_metrics
import pandas as pd

mix_sb,fs = sf.read('C:\\Users\\BASE\\PycharmProjects\\TSE\\test_au-
dio\\mixture.wav')
source_sb,fs = sf.read('C:\\Users\\BASE\\PycharmProjects\\TSE\\test_au-
dio\\source.wav')
est_output_sb,fs = sf.read('C:\\Users\\BASE\\PycharmProjects\\TSE\\test_au-
dio\\est_output_TSE.wav')

fig, ax = plt.subplots(3, figsize=(8, 6))
ax[0].specgram(mix_sb)
ax[0].set_title('Mixture')
ax[1].specgram(est_output_sb)
ax[1].set_title('Estimated target speaker')
ax[2].specgram(source_sb)
ax[2].set_title('Reference target speaker')
plt.tight_layout()
plt.show()

all_results = get_metrics(mix_sb, source_sb, est_output_sb, sam-
ple_rate=8000)
```

```
metrics = [m for m in all_results if not m.startswith('input') and 'sir'
not in m]
data = pd.DataFrame([[m, all_results[f'input_{m}'], all_results[m]] for m
in metrics],
                    columns=['metric', 'input', 'output'])

data_html = data.to_html(index=False)
with open('data.html', 'w') as f:
    f.write(data_html)
```

Результаты референсной модели

На слух результаты SpeakerBeam следующие: выделенный голос четкий, громче чем в смеси, фоновая музыка очень тихо слышна. На рисунке 10 представлены спектрограммы смеси, выделенного моделью голоса и оригинальная запись этого голоса. Качественные метрики аудиозаписи отображены на рисунке 11.

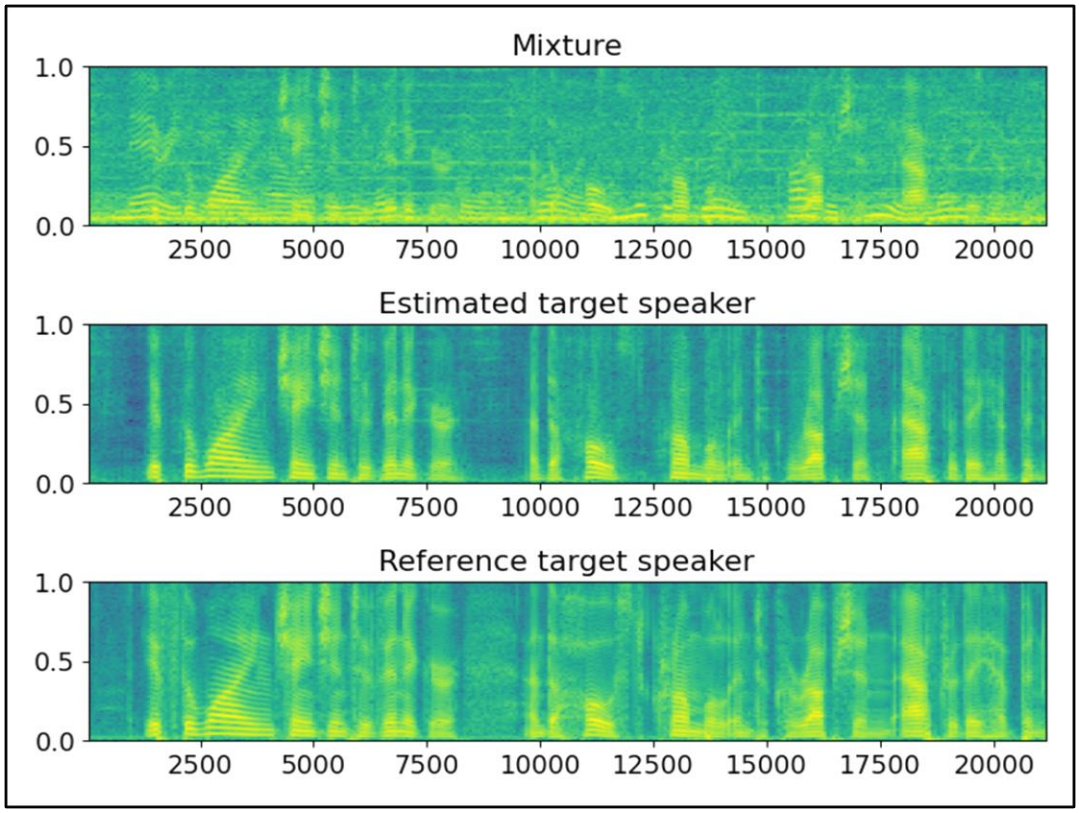

Рисунок 10 – Спектрограммы смеси, выделенного референсной моделью голоса и оригинального голоса

metric	input	output
si_sdr	-3.809231	12.140856
sdr	-3.572491	12.507056
sar	-3.572491	12.507056
stoi	0.658744	0.921440
pesq	1.562157	2.706475

Рисунок 11 – Метрики выделенного референсной моделью голоса

Результаты реализованной модели

На слух результаты реализованной модели следующие: выделенный голос четкий, с незначительными для понимания дефектами, громкостью как в смеси, фоновая музыка тише чем в смеси, но громче чем в референсной модели. На рисунке 12 представлены спектрограммы смеси, выделенного моделью голоса и оригинальная запись этого голоса. Качественные метрики аудиозаписи отображены на рисунке 13.

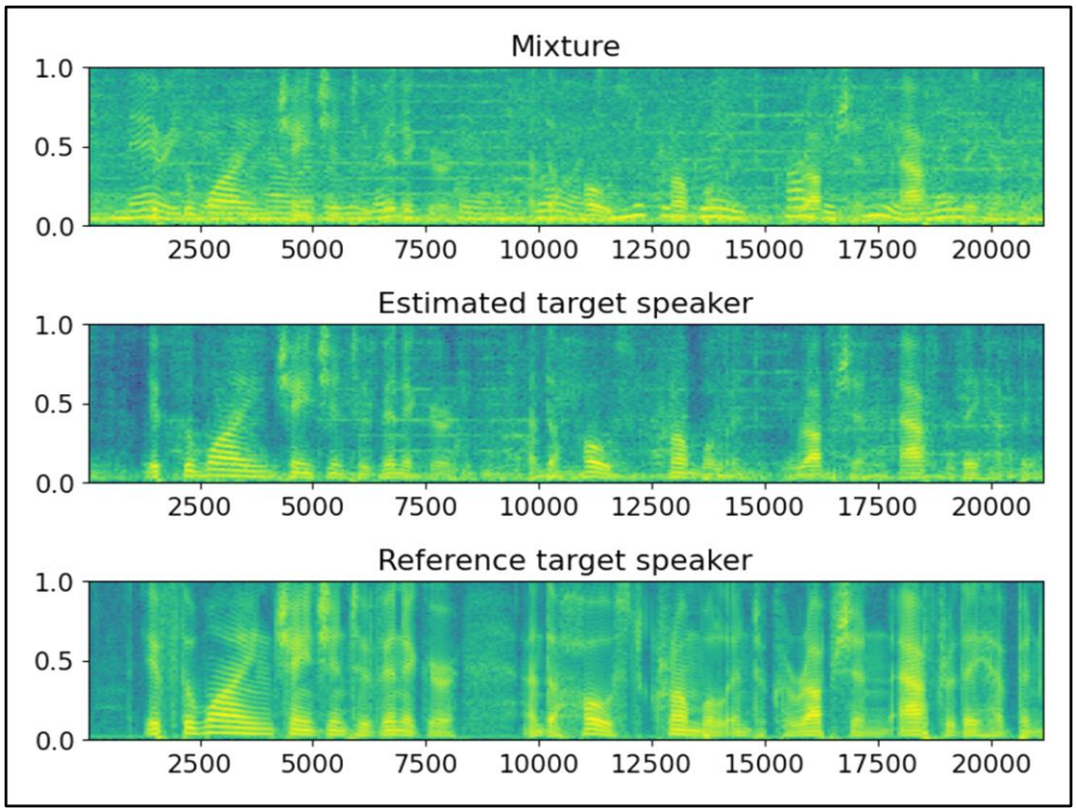

Рисунок 12 – Спектрограммы смеси, выделенного реализованной моделью голоса и оригинального голоса

metric	input	output
si_sdr	-3.809231	6.294573
sdr	-3.572491	6.499285
sar	-3.572491	6.499285
stoi	0.658744	0.854958
pesq	1.562157	1.985491

Рисунок 13 – Метрики выделенного реализованной моделью голоса

Как видно из сравнения, результаты SpeakerBeam превосходят результаты реализованной модели, но при этом качество выделения реализованной модели остается на хорошем уровне, что позволяет использовать ее в качестве основной в будущем веб-приложении.

Выводы по второй главе

Был выполнен обзор актуальных наборов данных в том числе LibriCSS, LibriSpeech, WSJ0Mix и LibriMix, показаны их содержание, особенности и параметры. Последний из перечисленных наборов данных был использован в качестве основного при обучении референсной модели, т.к. имеет наибольшее количество данных. Было представлено описание референсной модели, с которой будет сравниваться модель реализованная. Описана реализация TSE модели для двух и трех дикторов в смеси, при этом по результатам работы модели для трех дикторов принято решение не использовать ее при реализации веб-приложения. Выполнено сравнение модели для двух дикторов с референсной моделью и представлены их результаты.

3. РЕАЛИЗАЦИЯ ВЕБ-ПРИЛОЖЕНИЯ

3.1. Проектирование

3.1.1. Функциональные и нефункциональные требования

Функциональные требования представляют собой основные функциональные возможности или особенности, которыми должна обладать система для достижения цели. Требования определяют, что должна делать система. Они описывают взаимодействие между программным обеспечением и его пользователями, а также поведение программного обеспечения в различных условиях [38].

Разрабатываемое приложение должно удовлетворять следующим функциональным требованиям:

1) приложение должно позволять пользователям загружать аудиофайлы только с форматами .wav, .mp3, .aac и .flac;

2) приложение должно автоматически переконвертировать загруженные аудиофайлы в формат .wav;

3) приложение должно проводить обработку загруженных аудиофайлов с целью извлечения голоса целевого диктора;

4) приложение должно предоставлять выделенный голос целевого диктора для сохранения и прослушивания;

5) приложение должно хранить данные не более суток с момента загрузки их пользователем;

6) приложение должно содержать информацию о требованиях системы, таких как поддерживаемые форматы и поддерживаемый язык, на котором общаются дикторы.

Нефункциональные требования представляют собой то, как система должна выполнять определенные функции. Они определяют качества, характеристики и ограничения системы, а не ее конкретные особенности. Нефункциональные требования устанавливают стандарты производительности, безопасности и удобства использования системы [38].

Разрабатываемое приложение должно удовлетворять следующим нефункциональным требованиям:

1) приложение должно быть реализовано с помощью фреймворка Flask или Django;

2) приложение должно быть доступно из браузера;

3) приложение должно иметь интуитивно понятный, минималистичный и удобный интерфейс;

4) приложение должно быть написано на языке Python.

3.1.2. Диаграмма вариантов использования

На рисунке 14 представлена диаграмма вариантов использования веб-приложения. Главным актером системы является пользователь. Находясь на главной странице, пользователь может нажать на кнопку «Select audio files» для перехода на страницу с загрузкой смеси и подсказки, либо нажать на кнопку «About», находящуюся в хедере.

При нажатии на кнопку «About» пользователь переходит на страницу с описанием приложения и требований к формату файлов. Пользователь загружает собственные смесь и подсказку, для каждого файла присутствует собственное поле. После отправки данных система начинает обработку загруженных файлов. На время обработки файлов на странице вместо кнопки появляется анимация загрузки. По завершению обработки, пользователя автоматически переносит на страницу с результатами, где пользователь может прослушать результат выделения целевого голоса через встроенный аудиоплеер и выполнить сохранение файла с выделенным голосом на свое устройство. При возникновении ошибок в работе приложения, пользователю будет выведена соответствующая информация об ошибке.

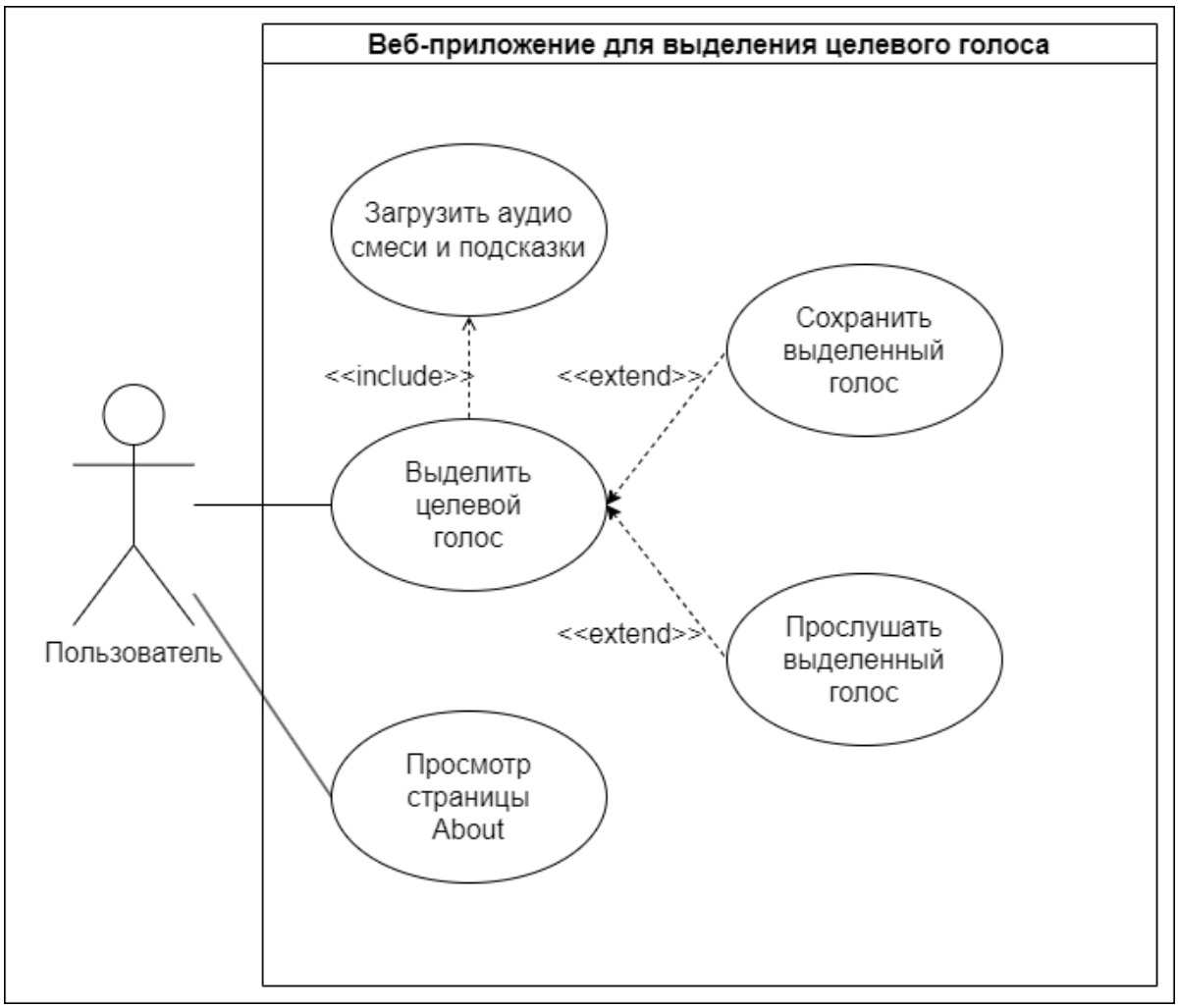

Рисунок 14 – Диаграмма вариантов использования приложения

3.2. Реализация

Программные средства реализации

Для реализации веб-приложения для выделения целевого голоса был использован язык программирования Python (реализация моделей, фронтэнд и бэкэнд), язык разметки HTML и каскадные таблицы стилей CSS. Для написания кода использовался IDE PyCharm Community Edition 2022.2.

Реализация сервера и приложения

При реализации сервера и приложения были использованы следующие инструменты.

1. Flask – легкий и мощный фреймворк для создания веб-приложений на языке Python [39]. Фреймворк использует приложение для обработки

шаблонов Jinja2 и Werkzeug – инструмент для работы со стандартом взаимодействия между программой, написанной на Python. Также во Flask используется модуль Virtualenv для создания изолированной среды в Python;

2. Bootstrap – свободный набор инструментов для создания сайтов и веб-приложений [40]. Фреймворк включает в себя различные HTML/CSS шаблоны оформления типографики, элементы по типу кнопок, форм, блоков навигации, JavaScript расширения и прочие.

3.2.1. Реализация серверной части

Серверная часть была разработана на языке Python с помощью фреймворка Flask. Основой фреймфорка является маршрутизация запросов и генерация ответов. Маршрутизация в Flask отвечает за соединение функций с конкретными URL адресами. Пример маршрутизаций на страницы представлен на листинге 2 приложения.

Логика работы сервера разделена между файлами app.py и process_audio.py. Файл app.py отвечает за маршрутизацию, создание сессии, обязательные настройки работы сервера и подключает маршруты к ресурсам. Функция `upload_file`, в частности, отвечает за загрузку смеси и подсказки, проводит предварительную обработку файла – преобразование к формату .wav. Далее файлы отправляются на обработку в process_audio.py (работа которого описана в главе 2), после чего, результат ее работы сохраняется в переменную `voice_path`, которая далее используется функцией download_file для возможности сохранения выделенного голоса пользователем.

Для каждой загрузки файлов пользователями определяется сессия. Данная процедура позволяет загружать пользовательские данные на сервер в папки сессий, сохраняя уникальность каждой итерации обработки смесей. Сессии работают по принципу, похожему на cookie и являются способом хранить данные конкретных пользователей между запросами к серверу. Каждый раз, при запросе на обработку данных, создаются две папки с уникальным названием, основанным на секретном ключе Flask и обработанным

алгоритмом UUID [41]. В качестве уникального ключа выступает набор сгенерированных символов длинной в 42 единицы. Папки находятся в основной папке audio, которая находится в папке static, и разделены между другими двумя папками: input и output для загруженных и обработанных данных соответственно. При загрузке смеси и подсказки пользователем, те, после предобработки, помещаются в папку project_dir/static/audio/input/{unique_folder}. После того, как process_audio.py выполнит диаризацию, разделенные голоса записываются в папку project_dir/static/audio/output/{unique_folder}. Пример дерева папки audio представлен на рисунке 15.

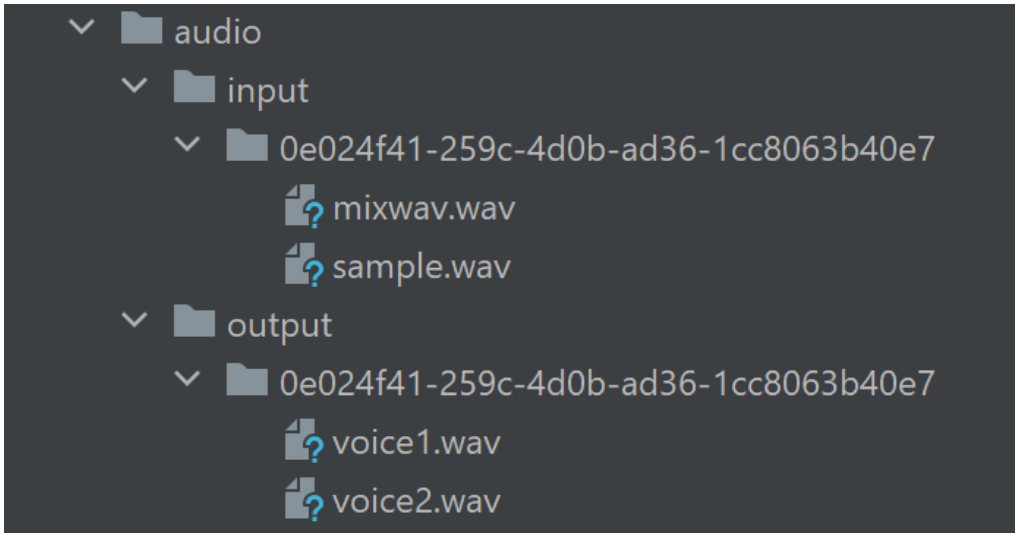

Рисунок 15 – Пример дерева папки audio

Все уникальные папки с данными пользователей удаляются, если с момента их создания прошло 24 часа или больше, когда кто-либо заходит на главную страницу сервиса. Это позволяет сохранить место на сервере и не хранить возможно конфиденциальную информацию долгое время во избежание утечек. За удаление данных отвечает функция `delete_old_sessions`, представленная на листинге 6. Алгоритм функции следующий: для каждой папки в input и output проверяется, является ли текущий элемент папкой, а не файлом, далее алгоритм получает время создания папки и проверяет, превышает ли возраст сессии `max_age_seconds`, если да, то сессия считается устаревшей и удаляется с помощью `shutil.rmtree`.

Листинг 6 – Функция `delete_old_sessions`

```
def delete_old_sessions(input_dir, output_dir, max_age_seconds):
    current_time = time.time()
    for session_folder in os.listdir(input_dir):
        session_path = os.path.join(input_dir, session_folder)
        if os.path.isdir(session_path):
            session_creation_time = os.path.getctime(session_path)
            if current_time - session_creation_time > max_age_seconds:
                shutil.rmtree(session_path)
    for session_folder in os.listdir(output_dir):
        session_path = os.path.join(output_dir, session_folder)
        if os.path.isdir(session_path):
            session_creation_time = os.path.getctime(session_path)
            if current_time - session_creation_time > max_age_seconds:
                shutil.rmtree(session_path)

input_dir = os.path.join(project_dir, 'static', 'audio', 'input')
output_dir = os.path.join(project_dir, 'static', 'audio', 'output')
max_age_seconds = 86400
delete_old_sessions(input_dir, output_dir, max_age_seconds)
```

3.2.2. Реализация клиентской части

Веб-клиент, как и серверная часть, был реализован на языке Python с помощью фреймворка Flask. Внешне клиент выполнен в минималистичном дизайне с красным акцентным цветом на бело-сером фоне. Пользователю для просмотра и взаимодействия доступно три страницы: главная страница (рисунок 16), страница загрузки смеси и подсказки (рисунок 17), а также страница с информацией о проекте.

На главной странице присутствует хедер, футер, и контейнер, в котором находится краткое описание функционала, кнопка для перехода к загрузке файлов и иконки поддерживаемых форматов. На странице с загрузкой данных пользователю представлено две формы – для загрузки смеси и для загрузки подсказки, иконки, сигнализирующие о том, что файл был или не был выбран, а также кнопка «Upload», при нажатии на которую, вместо нее появляется лоадер. Во время обработки файлов пользователь не должен покидать страницу, иначе обработка прервется.

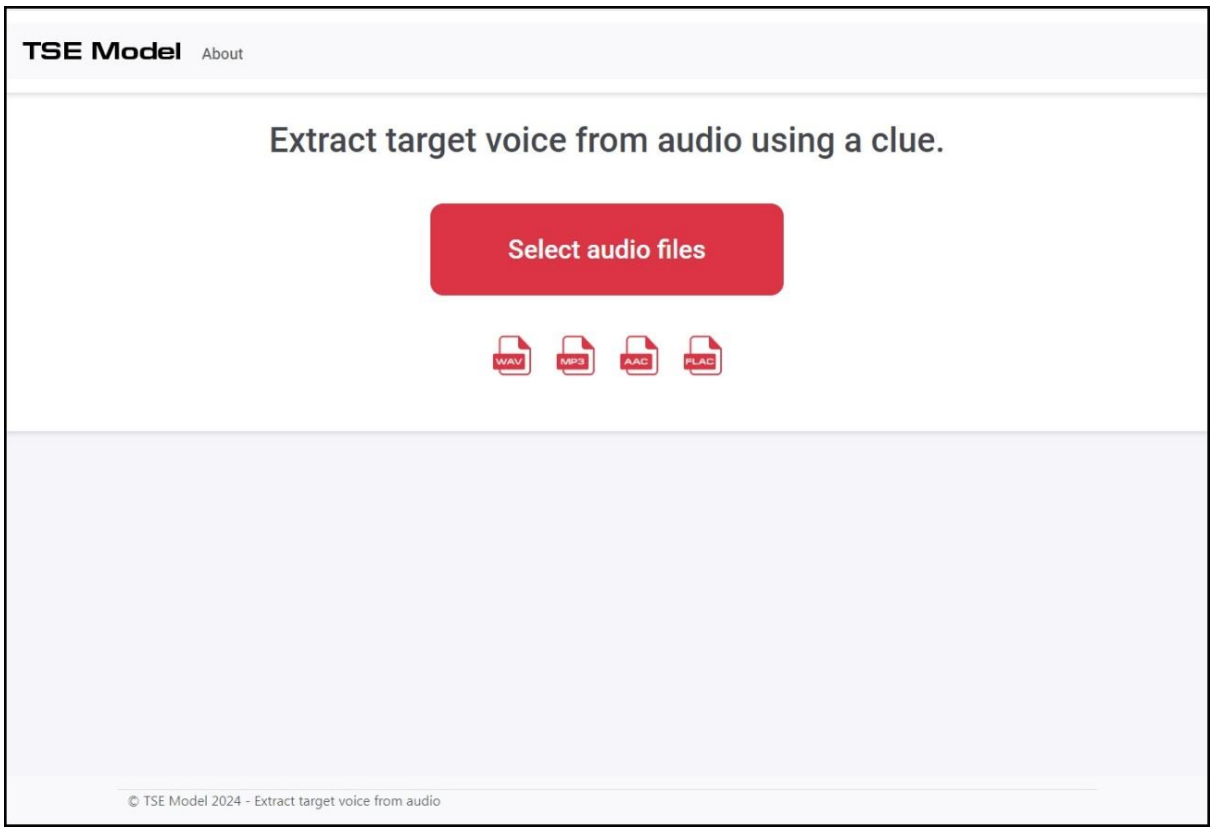

Рисунок 16 – Главная страница сервиса

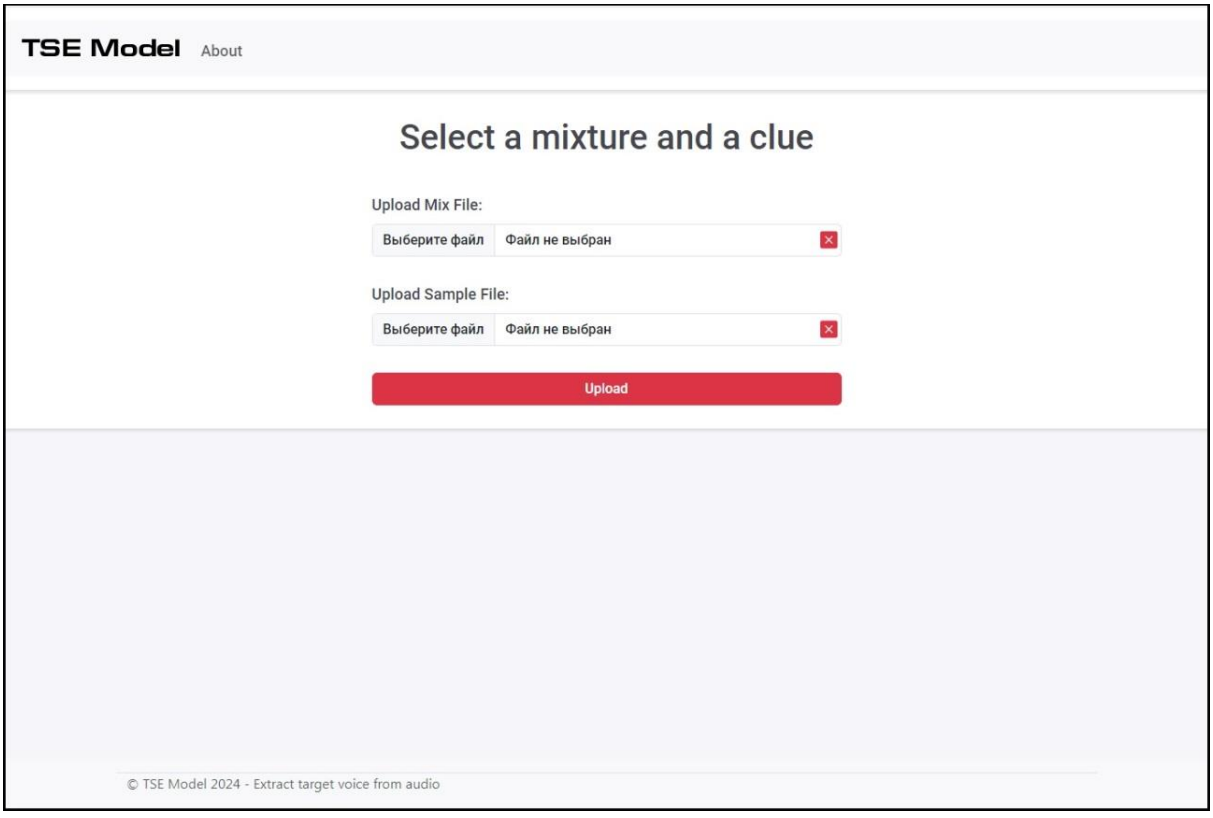

Рисунок 17 – Страница загрузки смеси и подсказки

В хедере, который присутствует на всех страницах, при нажатии на логотип «TSE Model» пользователя отправляет на главную страницу, а при нажатии на кнопку навигации «About» пользователь пересылается на страницу с информацией о проекте (рисунок 18). На странице с информацией о проекте содержится текст, описывающий проблему, которую решает сервис. Также описан функционал проекта, поддерживаемый язык и форматы аудиофайлов, а также используемые технологии.

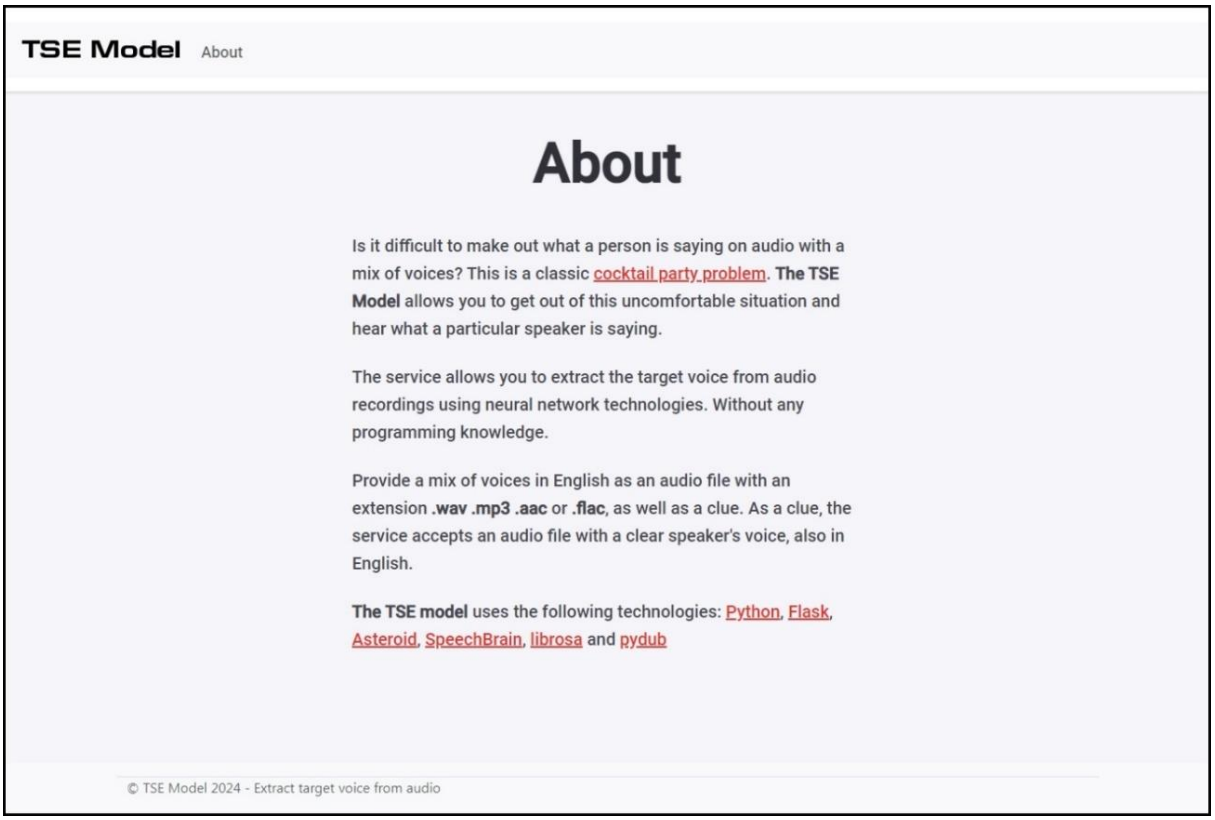

Рисунок 18 – Страница About

Если пользователь ошибочно или намеренно попытается загрузить файл с расширением, которое не обрабатывается системой, то будет показано окно с ошибкой (рисунок 19) и напоминанием о поддерживаемых типах файлов. Окно с ошибкой будет объявляться пользователю до тех пор, пока не будут поданы обрабатываемые системой форматы аудиофайлов.

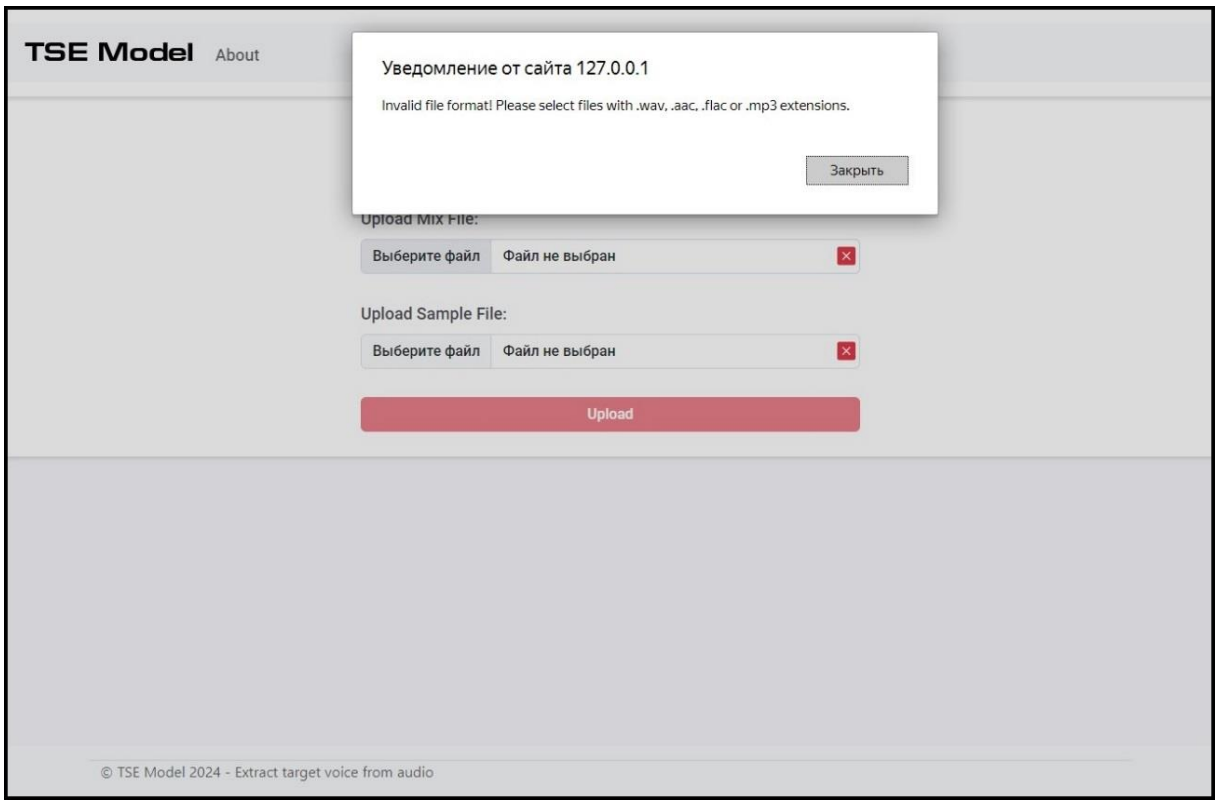

Рисунок 19 – Окно с ошибкой

На листинге 7 показана реализация проверки файлов на поддерживаемый формат, а на листинге 8 показана функция `checkFileType` на языке JavaScript, которая выводит окно с ошибкой.

Листинг 7 – Проверка файлов на поддерживаемый формат

```
<form id="uploadForm" action="/upload" method="post" enctype="multi-
part/form-data" onsubmit="startLoader()">
    <div class="mb-3">
        <label for="mix" class="form-label">Upload Mix File:</label>
        <input type="file" class="form-control" id="mix" name="mix" ac-
cept=".wav, .aac, .flac, .mp3" onchange="checkFileType()">
        <div id="fileStatusMix" class="file-icon1">
            <img src="static/img/cross.png" alt="Cross Icon" id="mixIcon"
style="width: 20px; height: 20px;">
        </div>
    </div>
    <div class="mb-3">
        <label for="sample" class="form-label">Upload Sample File:</label>
        <input type="file" class="form-control" id="sample" name="sample"
accept=".wav, .aac, .flac, .mp3" onchange="checkFileType()">
        <div id="fileStatusSample" class="file-icon2">
            <img src="static/img/cross.png" alt="Cross Icon" id="sam-
pleIcon" style="width: 20px; height: 20px;">
        </div>
    </div>
    <div class="d-grid gap-2" style="margin-top: 0px;">
        <button id="uploadButton" type="submit" class="btn btn-danger">Up-
load</button>
    </div>
</form>
```

Листинг 8 – Функция checkFileType

```
function checkFileType() {
    var mixFileInput = document.getElementById('mix');
    var sampleFileInput = document.getElementById('sample');
    var mixFilePath = mixFileInput.value;
    var sampleFilePath = sampleFileInput.value;
    var allowedExtensions = /(\.wav|\.aac|\.flac|\.mp3)$/i;

    var mixValid = allowedExtensions.test(mixFilePath) || mixFilePath ===
'';
    var sampleValid = allowedExtensions.test(sampleFilePath) ||
sampleFilePath === '';

    if (!mixValid || !sampleValid) {
        alert('Invalid file format! Please select files
with .wav, .aac, .flac or .mp3 extensions.');
        mixFileInput.value = '';
        sampleFileInput.value = '';
        document.getElementById('uploadButton').disabled = true;
    } else {
        document.getElementById('uploadButton').disabled = false;
    }
}
```

После обработки файлов, пользователя перенаправляет на страницу «/download» (рисунок 20), где он может прослушать файл или сохранить его на свое устройство.

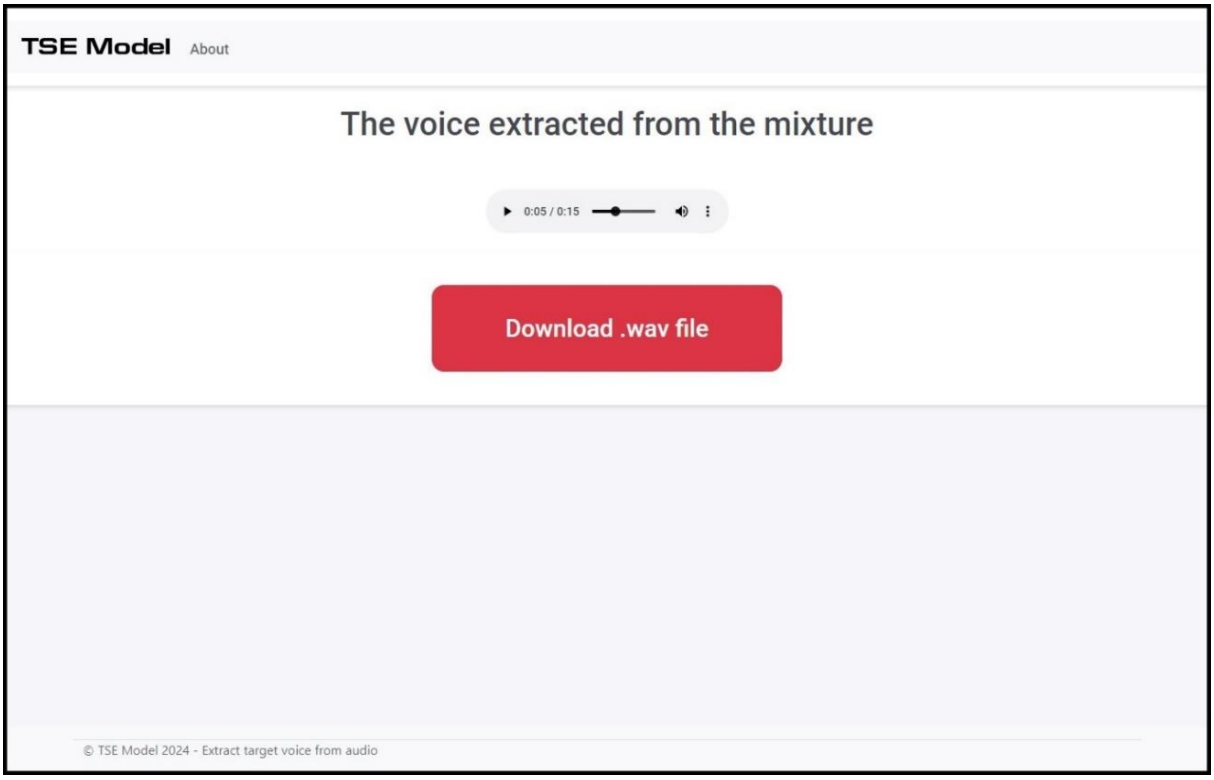

Рисунок 20 – Страница Download

3.3. Тестирование

Тестирование модели выделения целевого голоса

Тестирование модели выделения целевого голоса проводилось в целях выявления неисправностей и уязвимостей реализации. В протоколе (таблица 2) приведены результаты тестирования.

Таблица 2 – Тестирование реализованной модели

№	Тест	Ожидаемый результат	Тест пройден?
1	Модель сохраняет файлы в папки сессий для дальнейшей обработки	Модель корректно сохраняет загруженные пользователем файлы в папки сессий	Да
2	Модель приводит файлы смеси и подсказки к единому стандарту по частоте дискретизации, формату, и конвертирует файлы в моно	Модель, при необходимости, корректно изменяет частоту дискретизации, формат аудиофайла и конвертирует в моно	Да
3	Модель автоматически выбирает предобученную модель для диаризации аудиофайла	Модель корректно выбирает предобученную модель для диаризации в зависимости от приведения к стандарту	Да
4	Модель диаризирует смесь	Модель проводит диаризацию корректно и в рамках хорошего на слух качества	Да
5	Модель распознает целевого диктора	Модель верно распознает целевого диктора при загруженной пользователем корректной подсказке	Да
6	На выходе модель предоставляет выделенный голос целевого диктора	На выходе модель предоставляет путь до файла с выделенным голосом целевого диктора	Да

Тестирование веб-приложения

Тестирование веб-приложения проводилось в целях выявления соответствия функциональным требованиям. Как видно из протокола (таблица 3), все тесты успешно пройдены.

Таблица 3 – Протокол функционального тестирования

№	Функциональное требование	Ожидаемый результат	Тест пройден?
1	Приложение должно позволять пользователям загружать аудиофайлы только с форматами .wav, .mp3, .aac и .flac	Приложение не позволяет загрузить файлы с форматами, отличные от перечисленных в требованиях	Да

Окончание таблицы 3

№	Функциональное требование	Ожидаемый результат	Тест пройден?
2	Приложение должно автоматически переконвертировать загруженные аудиофайлы в формат .wav	Приложение приводит аудиофайлы к стандарту	Да
3	Приложение должно проводить обработку загруженных аудиофайлов с целью извлечения голоса целевого диктора	Приложение проводит диаризацию и распознает целевой голос из разделенных аудиофайлов	Да
4	Приложение должно предоставлять выделенный голос целевого диктора для сохранения и прослушивания	Приложение, после обработки файлов, предоставляет пользователю возможность сохранить и прослушать выделенный голос	Да
5	Приложение должно хранить данные не более суток с момента загрузки их пользователем	Приложение удаляет сессии с файлами после суток с момента загрузки их пользователем	Да
6	Приложение должно содержать информацию о требованиях системы, таких как поддерживаемые форматы и поддерживаемый язык, на котором общаются дикторы	Информация о требованиях приложения присутствует на странице About	Да

Выводы по третьей главе

В данной главе были показаны функциональные и нефункциональные требования к разрабатываемому веб-приложению выделения целевого голоса. На основании требований была разработана диаграмма вариантов использования приложения. Описаны средства реализации, которые были использованы при создании веб-приложения для выделения целевого голоса. Показан интерфейс клиентского приложения, методы его взаимодействия с серверной частью, а также логика работы серверной части. Проведено тестирование реализованной модели и веб-приложения, предоставлены результаты.

ЗАКЛЮЧЕНИЕ

В рамках данной работы была разработана двухмодульная модель выделения целевого голоса на основе нейросетевых технологий и реализовано веб-приложение на основе разработанной модели. В ходе выполнения данной работы были решены следующие задачи.

1. Проведен обзор литературы и аналогов. В рамках главы были описаны аналогичные модели выделения целевого голоса и показана общая структура нейросетевых TSE моделей.

2. Реализована TSE модель. В рамках главы выполнен обзор наборов данных для референсной модели, реализована TSE модель для двух дикторов, проведена работа с референсной моделью и предоставлено их сравнение.

3. Разработано веб-приложение. В рамках главы было разработано веб-приложение для выделения целевого голоса из смеси с двумя дикторами и представлены результаты тестирования.

Для улучшения веб-приложения необходимо добавить модель для выделения целевого голоса из смесей с тремя и более дикторами, добавить регистрацию и кабинет пользователя с историей и результатами обработки данных. Также можно добавить результаты выделения целевого голоса в виде спектрограмм и качественных метрик для опытных пользователей.

ЛИТЕРАТУРА

1. Žmolíková K., Delcroix M., Ochiai T., Kinoshita K., Cernocký J.,Yu D. Neural Target Speech Extraction: An Overview. // IEEE Signal Processing Magazine, Volume: 60, 2023. – С. 8–29.

2. O'Shaughnessy D. Trends and developments in automatic speech recognition research. // Computer Speech & Language. Volume 83, 2024. – 75 с.

3. Naveed H., Khan A.U., Qiu S. Saqib M., Anwar S., Usman M., Akhtar N., Barnes N., Mian A. A Comprehensive Overview of Large Language Models. [Электронный ресурс] // arXiv.org. 2024. Дата обновления: 09.04.2024 г. URL: https://arxiv.org/abs/2307.06435 (дата обращения: 14.05.2024 г.).

4. Munoz C., da Costa K., Modenesi B., Koshiyama A. Local and Global Explainability Metrics for Machine Learning Predictions. [Электронный ресурс] // arXiv.org. 2024. Дата обновления: 29.01.2024 г. URL: https://arxiv.org/abs/2302.12094 (дата обращения: 14.05.2024 г.).

5. Oracle. [Электронный ресурс] URL: https://stackoverflow.com/questions/19604450/what-does-oracle-means-in-machine-learning (дата обращения: 07.03.2024 г.).

6. Embeddings. [Электронный ресурс] URL: https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture (дата обращения: 07.03.2024 г.).

7. Нутфуллин Б.М., Ильюшин Е.А. Применение синусоидального моделирования речи к задаче диаризации звука. // International Journal of Open Information Technologies. Vol. 9, no. 7, 2021. – С. 14–18.

8. McDermott J.H. The cocktail party problem. // Current Biology, Volume: 19, 2009. – 4 с.

9. Cheng S., Shen Y., Wang D. Target Speaker Extraction by Fusing Voiceprint Features. // Applied Sciences, 2022. – 15 с.

10. Столбов М.Б., Татарникова М.Ю. Разделение речи целевого и сторонних дикторов с использованием двухмикрофонной системы. // Известия Высших Учебных Заведений «Приборостроение», 2014. – С. 53–57.

11. Mosseri I., Rubinstein M. Audiovisual Speech Enhancement in YouTube Stories. [Электронный ресурс] URL: https://blog.research.google/2020/10/audiovisual-speech-enhancement-in.html (дата обращения: 14.01.2024 г.).

12. Žmolíková K. SpeakerBeam for neural target speech extraction. [Электронный ресурс] URL: https://github.com/BUTSpeechFIT/speakerbeam (дата обращения: 10.04.2024 г.).

13. Delcroix M., Žmolíková K., Ochiai T., Kinoshita K., Nakatani T. Speaker activity driven neural speech extraction. // IEEE International Conference on Acoustics, Speech and Signal Processing, 2021. – С. 6099–6103.

14. Elminshawi M., Mack W., Chetupalli S.R., Chakrabarty S., Habets E. New Insights on Target Speaker Extraction. [Электронный ресурс] // arXiv.org. 2024. Дата обновления: 15.09.2023 г. URL: https://arxiv.org/abs/2202.00733 (дата обращения: 14.05.2024 г.).

15. Žmolíková K., Delcroix M., Kinoshita K., Ochiai T., Nakatani T., Burget L., Cernocký J. SpeakerBeam: Speaker Aware Neural Network for Target Speaker Extraction in Speech Mixtures. // IEEE Journal of Selected Topics in Signal Processing, Volume: 13, 2019. – С. 800–814.

16. Watanabe S., Mandel M., Barker J., Vincent E., Arora A., Chang X., Khudanpur S., Manohar V., Povey D., Raj D., Snyder D., Shanmugam Subramanian A., Trmal J., Yair B.B., Boeddeker C., Ni Z., Fujita Y., Horiguchi S., Kanda N., Yoshioka T., Ryant N. CHiME-6 Challenge: Tackling Multispeaker Speech Recognition for Unsegmented Recordings. // 6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020), 2020. – 7 с.

17. Hao X., Wu J., Yu J., Xu C., Chen Tan K. Typing to Listen at the Cocktail Party: Textguided Target Speaker Extraction. [Электронный ресурс] // arXiv.org. 2024. Дата обновления: 15.10.2023 г. URL: https://arxiv.org/abs/2310.07284 (дата обращения: 14.05.2024 г.).

18. Roux J.L., Wisdom S., Erdogan H., Hershey J.R. SDR – Half-Baked or Well Done? // IEEE International Conference on Acoustics, Speech, and Signal Processing, 2019. – 5 с.

19. Kuyk S.V., Kleijn W., Hendriks R.C. An Evaluation of Intrusive Instrumental Intelligibility Metrics. // IEEE/ACM Transactions on Audio Speech and Language Processing abbreviation, 2017. – 14 с.

20. Rix A., Beerends J.G., Hollier M.P., Hekstra A.P. Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs. // 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume: 2, 2001. – 4 с.

21. Chung S. W., Choe S., Chung J.S., Kang H. –G. FaceFilter: Audio-visual speech separation using still images. // Interspeech, 2020. – 5 с.

22. Afouras T., Chung J.S., Zisserman A. The Conversation: Deep Audio-Visual Speech Enhancement. // Interspeech, 2018. – С. 3244–3248.

23. Owens A., Efros A.A. Audio-visual scene analysis with selfsupervised multisensory features. // Proceedings of the European Conference on Computer Vision (ECCV), 2018. – С. 631–648.

24. Flanagan J.L., Johnston J.D., Zahn R., Elko G.W. Computer-steered microphone arrays for sound transduction in large rooms. // The Journal of the Acoustical Society of America, vol. 78, no. 5, 1985. – С. 1508–1518.

25. Gu. R., Chen L., Zhang S. –X., Zheng J., Xu Y., Yu M., Su D., Zou Y., Yu D. Neural spatial filter: Target speaker speech separation assisted with directional information. // Interspeech, 2019. – С. 4290–4294.

26. Xu C., Rao W., Chng E.S., Li H. Time-domain speaker extraction network. // Automatic Speech Recognition & Understanding. 2019. – С. 327–334.

27. Luo Y., Mesgarani N. TasNet: Surpassing ideal time-frequency masking for speech separation. // ICASSP, 2019. – 12 с.

28. Vincent E., Gribonval R., Fevotte C. Performance measurement in blind audio source separation. // IEEE transactions on audio, speech, and language processing. Vol. 14, no. 4, 2016. – С. 1462–1496.

29. Yoshioka T., alphgoo, Zhuo, Lu L. Libri-CSS: dataset and evaluation pipeline. [Электронный ресурс] URL: https://github.com/chenzhuo1011/libri_css (дата обращения: 27.03.2024 г.).

30. Yoshioka T., alphgoo, Zhuo, Lu L. Libri-CSS: dataset and evaluation pipeline. Data. [Электронный ресурс] URL: https://drive.google.com/file/d/1Piioxd5G_85K9Bhcr8ebdhXx0CnaHy7l/view (дата обращения: 27.03.2024 г.).

31. Panayotov V. LibriSpeech. [Электронный ресурс] URL: https://paperswithcode.com/dataset/librispeech (дата обращения: 28.03.2024 г.).

32. Cos J., Haag J., Manuel P. LibriMix. [Электронный ресурс] URL: https://github.com/JorisCos/LibriMix (дата обращения: 28.03.2024 г.).

33. WHAM! and WHAMR! [Электронный ресурс] URL: http://wham.whisper.ai/ (дата обращения: 04.02.2024 г.).

34. Manuel P., Roux J.L. wsj0-mix generation scripts, in Python. [Электронный ресурс] URL: https://github.com/mpariente/pywsj0-mix (дата обращения: 05.02.2024 г.).

35. Hershey J.R., Roux J.L., Watanabe S., Harsham B., Isik Y., Chen Z. Single-Channel Multi-Speaker Separation using Deep Clustering. // Interspeech 2016, 2016. – 5 с.

36. Asteroid. [Электронный ресурс] URL: https://github.com/asteroid-team/asteroid (дата обращения: 16.04.2024 г.).

37. SpeechBrain. [Электронный ресурс] URL: https://github.com/speechbrain/speechbrain (дата обращения: 16.04.2024 г.).

38. Функциональные и нефункциональные требования. [Электронный ресурс] URL: https://visuresolutions.com/ru/requirements-management-traceability-guide/functional-vs-non-functional-requirements/ (дата обращения: 25.04.2024 г.).

39. Flask. [Электронный ресурс] URL: https://flask.palletsprojects.com/en/3.0.x/ (дата обращения: 25.04.2024 г.).

40. Bootstrap 5. [Электронный ресурс] URL: https://getbootstrap.com/ (дата обращения: 25.04.2024 г.).

41. Algorithm for Creating a Name-Based UUID. [Электронный ресурс] URL: https://datatracker.ietf.org/doc/html/rfc4122#section-4.4 (дата обращения: 27.04.2024 г.).

ПРИЛОЖЕНИЕ. Код модели и веб-приложения

Листинг 1 – Реализация модели TSE для двух дикторов в смеси

```
import os
import librosa
import librosa.display
import soundfile as sf
from asteroid.models import BaseModel
from speechbrain.inference.speaker import SpeakerRecognition


def process_audio(session_id):
    project_dir = os.path.dirname(os.path.abspath(__file__))
    input_audio_dir = os.path.join(str(project_dir) + '\\audio\\input',
session_id)
    output_audio_dir = os.path.join(str(project_dir) + '\\audio\\output',
session_id)
    os.makedirs(input_audio_dir, exist_ok=True)
    os.makedirs(output_audio_dir, exist_ok=True)
    mixwav = os.path.join(input_audio_dir, 'mixwav.wav')
    print(input_audio_dir)

    def ctts(wav_file):
        mix, ctts_sr = sf.read(mixwav, dtype="float32", always_2d=True)
        if ctts_sr >= 16000:
            ctts_sr = 16000
            ctts_model_sr = "high"
        else:
            ctts_sr = 8000
            ctts_model_sr = "low"

        if len(mix.shape) == 2:
            # stereo to mono
            y, ctts_sr = librosa.load(mixwav, sr=ctts_sr)
            mix = librosa.to_mono(y)

        return sf.write(os.path.join(input_audio_dir, 'mixwav.wav'),
data=mix, samplerate=ctts_sr), ctts_sr, ctts_model_sr

    _, sr, model_sr = ctts(mixwav)
    print("---")
    print(sr)
    print(model_sr)
    if model_sr == "high":
        model = BaseModel.from_pretrained("JorisCos/Con-
vTasNet_Libri2Mix_sepclean_16k")
    else:
        model = BaseModel.from_pretrained("JorisCos/Con-
vTasNet_Libri2Mix_sepclean_8k")

    mixture, _ = sf.read(mixwav, dtype="float32", always_2d=True)
    # Soundfile returns the mixture as shape (time, channels), and Asteroid
expects (batch, channels, time)
    mixture = mixture.transpose()
    mixture = mixture.reshape(1, mixture.shape[0], mixture.shape[1])
    # Separated speakers as out_wavs[0][0] for the first speaker and
out_wavs[0][1] for the second
    out_wavs = model.separate(mixture)

    sf.write(os.path.join(output_audio_dir, "voice1.wav"),
data=out_wavs[0][0], samplerate=sr)
    sf.write(os.path.join(output_audio_dir, "voice2.wav"),
```

```
data=out_wavs[0][1], samplerate=sr)

    voice1_path = os.path.join(output_audio_dir, 'voice1.wav')
    voice2_path = os.path.join(output_audio_dir, 'voice2.wav')
    sample_path = os.path.join(input_audio_dir, 'sample.wav')
    print(voice1_path, voice2_path, sample_path)
    verification = SpeakerRecogni-
tion.from_hparams(source="speechbrain/spkrec-ecapa-voxceleb",
                                                    savedir="pretrained_mod-
els/spkrec-ecapa-voxceleb")
    score, prediction = verification.verify_files(voice2_path, sample_path)

    print(prediction, score)

    if str(prediction) == "tensor([True])":
        return voice2_path
    else:
        return voice1_path
```

Листинг 2 – Маршрутизация на страницы

```
from flask import Flask, render_template, request, send_file, url_for,
redirect, session
import os
import uuid
import time
import shutil
from pydub import AudioSegment
import process_audio
app = Flask(__name__, template_folder="templates")
app.secret_key = '3d6f45a5fc1242225dac2f59c3b6c7cb1/ild2asex'


@app.route('/', methods=['GET'])
def index():
    project_dir = os.path.dirname(os.path.abspath(__file__))

    def delete_old_sessions(input_dir, output_dir, max_age_seconds):
        current_time = time.time()
        for session_folder in os.listdir(input_dir):
            session_path = os.path.join(input_dir, session_folder)
            if os.path.isdir(session_path):
                session_creation_time = os.path.getctime(session_path)
                if current_time - session_creation_time > max_age_seconds:
                    shutil.rmtree(session_path)
        for session_folder in os.listdir(output_dir):
            session_path = os.path.join(output_dir, session_folder)
            if os.path.isdir(session_path):
                session_creation_time = os.path.getctime(session_path)
                if current_time - session_creation_time > max_age_seconds:
                    shutil.rmtree(session_path)

    input_dir = os.path.join(project_dir, 'static', 'audio', 'input')
    output_dir = os.path.join(project_dir, 'static', 'audio', 'output')

    max_age_seconds = 86400
    delete_old_sessions(input_dir, output_dir, max_age_seconds)
    return render_template('index.html')
```

```
@app.route('/about', methods=['GET'])
def about():
    return render_template('about.html')


@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    session_id = str(uuid.uuid4())
    session['session_id'] = session_id
    if request.method == 'GET':
        return render_template('upload.html')
    elif request.method == 'POST':
        project_dir = os.path.dirname(os.path.abspath(__file__))
        audio_dir = os.path.join(project_dir, 'static', 'audio', 'input',
session_id)

        os.makedirs(audio_dir, exist_ok=True)

        if 'mix' in request.files:
            mix_file = request.files['mix']
            sound = AudioSegment.from_file(mix_file)
            sound.export(os.path.join(audio_dir, 'mixwav.wav'),
format="wav")

        if 'sample' in request.files:
            sample_file = request.files['sample']
            sound = AudioSegment.from_file(sample_file)
            sound.export(os.path.join(audio_dir, 'sample.wav'),
format="wav")

        voice_path = process_audio.process_audio(session_id)
        session['voice_path'] = voice_path
        session['voice_filename'] = os.path.basename(voice_path)

        return render_template('result.html', session_id=session_id,
voice_path=voice_path)



@app.route("/result", methods=['GET'])
def result():

    return render_template('result.html')



@app.route('/download')
def download_file():
    voice_path = session.get('voice_path')
    if voice_path:
        print(voice_path)
        return send_file(voice_path, as_attachment=True)
    else:
        return "File not found", 404

if __name__ == '__main__':
    app.run(debug=True)
```