Н И Т У « М И С и С »

Кафедра инженерной кибернетики

ОТЧЕТ

ПО

ЛАБОРАТОРНОЙ РАБОТЕ №1

«Разработка демонстрационного прототипа приложения для решения специализированной задачи интеллектуальной обработки и анализа информации с использованием современных ИИ-сервисов»

учебная дисциплина «Методы искусственного интеллекта»

Группа: БПМ-21-1

Учащийся: Ишмухамедов А.А.

Преподаватель: Хонер П.Д.

Оценка:

Дата защиты:

2024 г.

Оглавление

ВВЕДЕНИЕ

Целью данной лабораторной работы является выработка устойчивых навыков взаимодействия с современными ИИ-сервисами на уровне программного кода для решения задачи интеллектуальной обработки информации, а также оценка качества работы таких сервисов на примере преобразования аудиофайлов в текст.

В рамках данной работы рассматривалась задача преобразования аудиофайлов в текст с использованием современных ИИ-сервисов. Исследуемые API — Whisper и AssemblyAI. Проблемная область относится к обработке текстов на естественном языке (NLP). Цель — определить лучший сервис по качеству распознавания.

1. ОПИСАНИЕ ЗАДАЧИ

1.1. Основное содержание задачи

Изучение возможности преобразования аудиофайлов в текст с использованием современных ИИ-сервисов для обработки естественного языка (NLP). Рассматриваемая задача относится к области интеллектуальной обработки информации.

Задача заключается в преобразовании аудиофайлов, содержащих речевую и музыкальную информацию (голос в сопровождении музыки), в текстовые расшифровки.

1.2. Решение включает

1. Подключение к API ИИ-сервисов (OpenAI Whisper и AssemblyAI).
2. Отправку аудиофайлов для обработки.
3. Получение текстовых результатов.
4. Сравнение качества работы двух сервисов на основании точности расшифровки.
5. Исходные данные для лабораторной работы включают аудиофайлы, созданные лично, а также музыкальные композиции и видеоматериалы, взятые из открытых источников.

Для эксперимента использовались 10 аудиофайлов в формате MP3 с разнообразным содержанием.

Примеры содержаний аудиофайлов:

— Короткая речь (мужской и женский голос).

— Речь с фоновым шумом.

— Детский голос с дефектами речи.

— Музыкальные композиции (рэп, поп).

— Синхронно произнесенные фразы мужским и женским голосом.

Примеры результатов:

Результатом обработки каждого аудиофайла является текстовая расшифровка. Например:

— «Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

— «Мы с тобой шёпотом, шёпотом Спрашивали, что потом, что потом будет Шёпотом, шёпотом Не хочу кричать о том, что друг друга забудем»

1.3. Особенности решения

Использованы два API:

1. OpenAi Whisper API, интегрированный через RapidAPI [1]. Whisper — это система автоматического распознавания речи (ASR), разработанная OpenAI. Эта модель может транскрибировать аудио в текст, а также выполнять перевод между языками и детекцию различных языков. Whisper была обучена на огромном количестве различных аудиоданных с разнообразными языковыми, акцентными и шумовыми условиями, что позволяет ей работать эффективно в разных условиях.

2. AssemblyAI API, использована библиотека assemblyai для языка программирования Python [2]. AssemblyAI — это коммерческая компания, предлагающая мощные API для обработки аудио и видео данных с использованием искусственного интеллекта. Основная цель AssemblyAI — предоставить инструменты для автоматического распознавания речи и анализа контента, обеспечивая высокое качество транскрипций и дополнительных функций для бизнеса. В отличие от Whisper, который является открытым проектом, AssemblyAI ориентирован на коммерческую работу с аудиовизуальными материалами.

Программа предоставляет удобный и простой интерфейс на Python с использованием библиотеки PyQt5. Пользователь может выбирать API, загружать файлы и получать результаты в графическом интерфейсе.

Реализован механизм обработки ошибок и визуализация результата.

Таким образом, задача позволяет оценить качество работы современных ИИ-сервисов для преобразования аудио в текст, а также выявить особенности их применения в различных условиях.

3. ВЫБРАННЫЕ СРЕДСТВА ДЛЯ РАЗРАБОТКИ

Для разработки программного обеспечения были использованы следующие инструменты и технологии:

1. Whisper API — современный программный интерфейс (API), предоставляющий функционал преобразования аудиофайлов в текст. API интегрировался через платформу RapidAPI с использованием стандартных методов HTTP-запросов.

2. AssemblyAI API — аналогичный сервис, обеспечивающий преобразование аудио в текст, однако его использование через платформу RapidAPI вызвало ошибку HTTP 500 (рис. 1). Для работы с данным API применялась официальная библиотека assemblyai [2] для языка программирования Python, что позволило организовать прямое взаимодействие с сервисом.

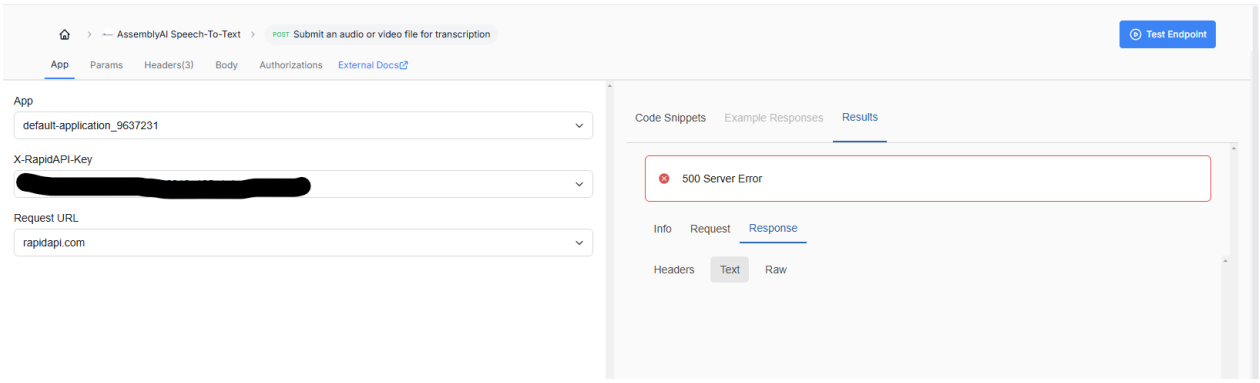

Рис. 1. Ошибка на стороне сервера при подключении к AssemblyAI API через RapidApi

1. Язык программирования Python — основной инструмент разработки, обеспечивающий гибкость интеграции и обработки данных.

2. Среда разработки: Visual Studio Code — кроссплатформенная интегрированная среда разработки, использованная для написания, отладки и тестирования программного кода.

3. Библиотеки:

— requests — для выполнения HTTP-запросов к API.[3]

— dotenv — для безопасного хранения конфиденциальной информации, включая API-ключи, в файле .env.

— PyQt5 — библиотека для создания графического пользовательского интерфейса приложения, предоставляющего пользователю возможность выбора API, загрузки файлов и отображения результатов. [4]

Использование данных инструментов позволило реализовать программное обеспечение, соответствующее требованиям поставленной задачи, а также обеспечить удобство работы пользователя в процессе тестирования.

4. ИСТОЧНИКИ ИСХОДНЫХ ДАННЫХ

Для тестирования были взяты аудиозаписи из свободных источников и записаны собственные аудиофайлы с заранее подготовленным текстом. В итоге среди исходных данных 10 аудиозаписей в формате mp3:

1. Мужской голос, быстрая речь, длительность 9 секунд
2. Мужской голос, медленная речь, длительность 16 секунд
3. Женский голос, обычная речь, длительность 12 секунд
4. Женский голос, быстрая речь, длительность 9 секунд
5. Мужской и женский голос, синхронно, длительность 11 секунд

Общий текст для 5 вышеописанных аудиозаписей: «Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работу и выбрать лучший вариант.»

6. Мужской голос с шумом на фоне, длительность 15 секунд

   Текст: «Привет! Это тестовая запись под шумом. Попытаемся распознать мой голос под шумом. Привет! Это тестовая запись, голосовое сообщение с шумом. Попытаемся распознать мой голос с помощью приложения под шумом.»

7. Отрывок из песни Сергея Лазарева “Шепотом”, длительность 12 секунд

Текст отрывка:

«Мы с тобой шёпотом, шёпотом

Спрашивали, что потом, что потом будет?

Шёпотом, шёпотом

Не хочу кричать о том, что друг друга забудем»

8. Отрывок из песни Скриптонита “Привычка”, длительность 23 секунды

Текст отрывка:

«Можешь говорить о том, что я закрывал за собой двери

Когда шёл сюда - я танцевал в дряни

Много колорита, но Шоу не карнавал в телек

Дикий аппетит — это белая вдова в теле

Утром в голове, под вечер под ножом и вилкой

Эта встреча не случайность и общули не за биткоин

Два года назад я бы назвал это разминкой

Сообщение в вотсап, но я потерял свой пинкод»

9. Отыровк из песни Fike & Jambazi “Minimun”, длительность 20 секунд

Текст отрывка:

«Выниму минимум, мэн, выманю мам, выманю минимум ума

Мэнам феноменам, фона бы нам стена бы фаном менам

С Papalam имени видимо минимум даже самому синему в ум

Не надо муму, сказали кому, не надо по минимуму... Fike!

Fike не на минимум, Papalam не на минимум, фейка на минимум

Нам не на минимум, крышу рвет не на минимум, бас не на минимум

Вы меня видели, выдели, вы меня вынудили да мы выдали вам

Бит не на минимум, бас не на минимум, нам не надо по минимумам»

10. Отрывок с youtube канала “Центр лечения заикания Татьяны Соловьевой”, видео “ Заикание у детей: Милана, 5 лет. Речь до и после курса Татьяны Соловьевой.”, длительность 26 секунд

Текст отрывка:

«Что тебе нравится в детском садике? Например, одушевленное, душа есть, разговаривает, поёт, отвечает на вопрос кто. А неодушевленное молчит, лежит, не лает. Кто?»

5. РЕЗУЛЬТАТЫ

Была разработана программа с простым графическим интерфейсом, с возможностью выбора api и выбора аудиофайла, а также с полем для вывода распознанного текста (рис. 2, рис. 3)

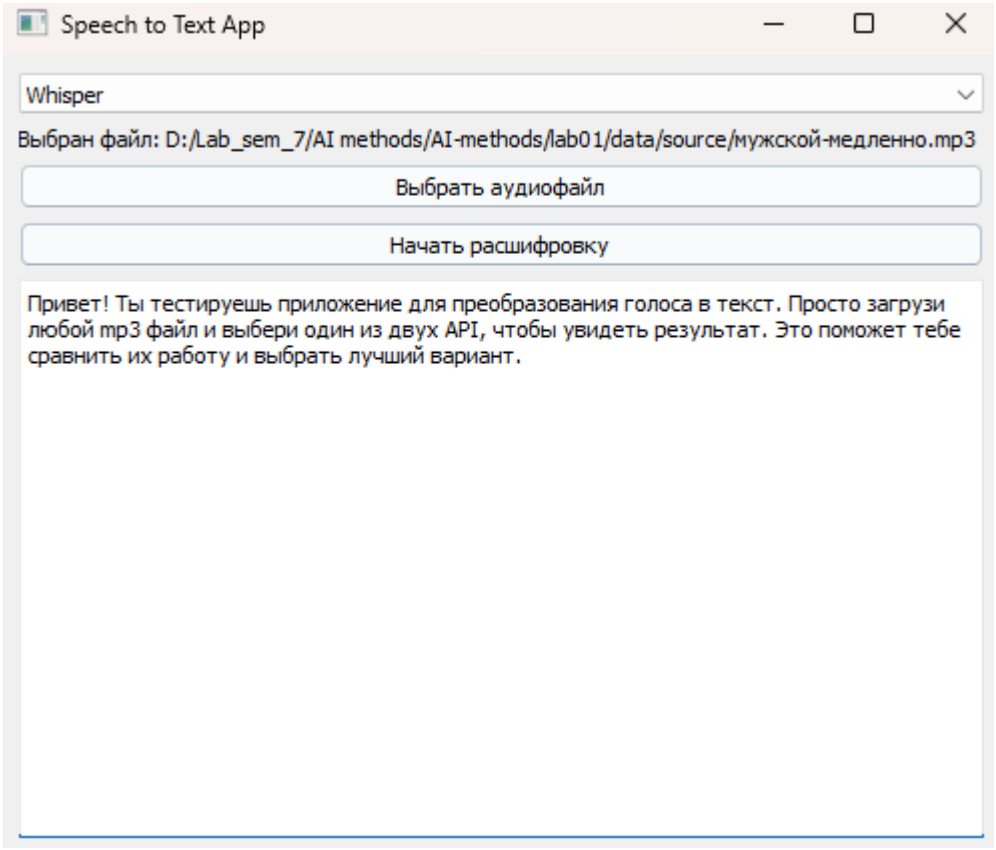

Рис. 2. Графический интерфейс программы

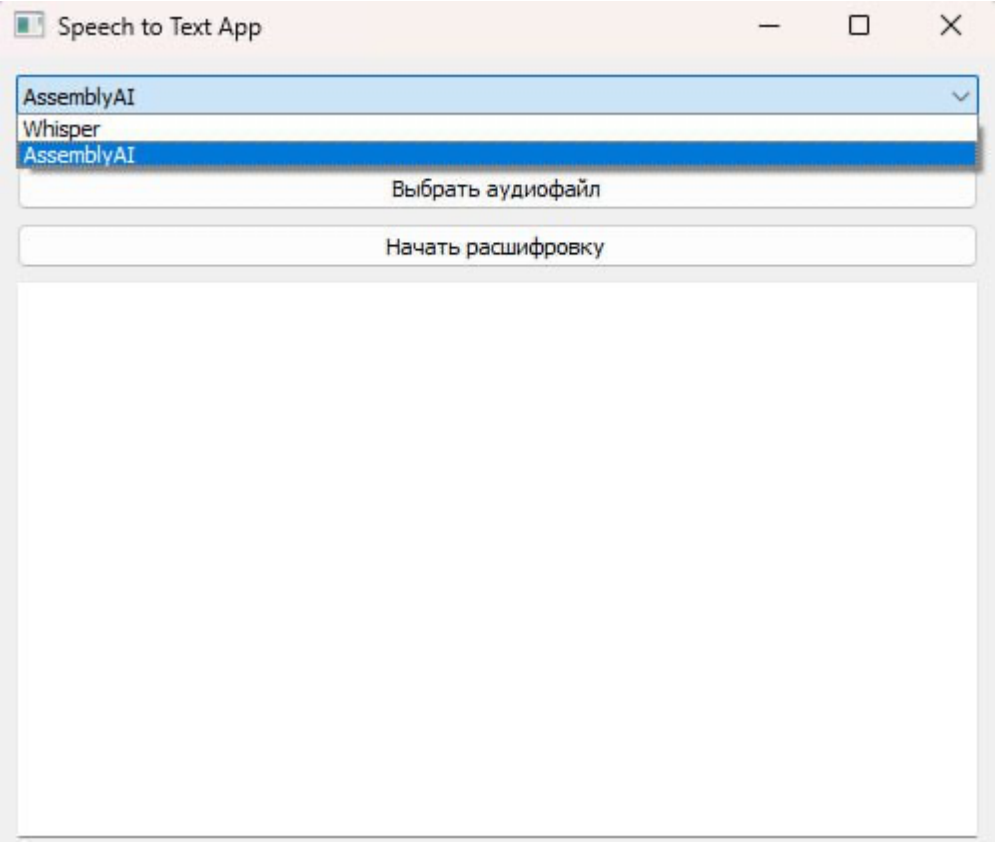

Рис. 3. Выбор api в графическом интерфейсе

Программа была протестирована на указанных файлах. Скриншоты с результатами распознавания приведены ниже. Whisper и AssemblyAI успешно обработали все файлы, предоставив текст с высокой точностью, особенно записанный собственноручно файлы были расшифрованы успешно, точь-в-точь. Не помешали распознаванию и посторонние шумы на записях. Обе модели успешно справились с расшифровкой аудиофайлов в текст, содержащие отрывки из песен, но и Whisper и AssemblyAI плохо справляются с распознаванием сленга и англицизмов. Примером плохого распознавания является результат распознавание текста аудифайла содержащего отрывок из песни Fike & Jambazi “Minimun”, в котором AssemblyAI показала лучше результат. Отрывок речи 5-летнего ребенка с заиканием был распознан лучше чем отрывок содержащий сленг.

AssemblyAI API вызвал ошибку 500, что исключило возможность использовать AssemblyAI API, поэтому использовалась библиотека assemblyai. Исходный код программы можно посмотреть в репозитории github [5] в папке lab01 и в Приложении В.

Результаты работы Whisper соответствующие порядку файлов, описанных в предыдущем пункте, скриншоты с соответствующими результатами представлены в Приложении А:

1. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

2. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работу и выбрать лучший вариант.

3. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

4. Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

5. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

6. Тестовая запись под шумом. Попытаемся распознать мой голос. Это тестовая запись и голосовое сообщение. Попытаемся распознать мой голос с помощью приложения под шумом.

7. Мы с тобой шёпотом, шёпотом Спрашивали, что потом, что потом Будет шёпотом, шёпотом Не хочу кричать о том, что друг друга забудем

8. Можешь говорить о том, что я закрывал за собой двери, когда шел сюда и танцевал, для немного колорита, но шоу не кардинал телек, и кипятит это белая вдова в телек Утром в голове, под вечер спать ножом и вилкой, эту встречу не случайность, я общу линии за биткоин Два года назад я бы назвал это разминкой, сообщение в WhatsApp, но я потерял свой пин-код

9. We menu, ma'am, we're not a minimum, man, Man, fan of men, fan of nam, Senabhaughan men, we've, we've, even for one, even the same one, no, no, said, come, no, no, no, fight, no minimum, f'am'n't, a minimum, Crish, r'rv' not a minimum, vass, not minimum, you've seen, you've, you've, you've, we've, been, we've, , you know, we've, not, minimum,

10. Что тебе нравится в детском садике? Например, зашевелённость, нарушаясь. Он говорит, поёт, лёгко. Третья вопроса – кто? Не зашевелённость, как будто он живёт, молчит, не лает.

Результаты работы AssemblyAI соответствующие порядку файлов, описанных в предыдущем пункте, скриншоты с соответствующими результатами представлены в Приложении Б:

1. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл, выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

2. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работу и выбрать лучший вариант.

3. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

4. Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любое mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

5. Привет! Ты тестируешь приложение для преобразования голоса в текст. Просто загрузи любой mp3 файл и выбери один из двух API, чтобы увидеть результат. Это поможет тебе сравнить их работы и выбрать лучший вариант.

6. Тестовая запись под шумом. Попытаемся распознать мой голос. Это тестовая запись и голосовое сообщение. Попытаемся распознать мой голос помощью приложения.

7. Мы с тобой шёпотом, шёпотом Спрашивали, что потом, что потом будет Шёпотом, шёпотом Не хочу кричать о том, что друг друга забудем

8. Говорить о том, что я закрывал за собой двери Когда шел сюда, я танцевал, да и немного колорита Но шоу не гавновал телек, дикий аппетит Это белая вдова в телек Утром в голове под вечер спать ножом и вилкой Эту встречу не случайность, я общу линии за биткоин Два года назад я бы назвал это разминкой Сообщение в WhatsApp, но я потерял свой пин-код

9. Попала мне на минимум, фейк на минимум, нам не на минимум. Крышу рвет не на минимум, бас не на минимум.

10. Что тебе нравится в детском садике? Дрянь, например, зашевелённый, нарушаясь. Он разговаривает, поёт, вздрёхает. Дрянь на вопрос «Кто?» Он не зашевелённый, просто лежит, молчит, не лает.

5.1. Сравнение API

5.1.1. Качество распознавания

Качество распознавания речи в обоих решениях — **AssemblyAI** и **Whisper** — практически идентично. Обе системы обеспечивают высокую точность распознавания речи даже при наличии посторонних шумов или дефектов записи. Однако **AssemblyAI** показал себя лучше в распознавании сленга и нестандартных выражений, таких как англицизмы и жаргон, что делает его более подходящим для приложений, которые обрабатывают разговорную речь или речи с неформальными элементами.

5.1.2. Доступность и удобство работы с API и библиотеками

Whisper API используется через платформу RapidAPI, что удобно с точки зрения интеграции и простоты работы с API. Однако количество бесплатных запросов мало.

AssemblyAI API на платформе RapidAPI вызывает ошибку 500. Поэтому использована библиотека AssemblyAI [2], библиотека предоставляет приятную и подробную документацию. Количество запросов в бесплатном плане достаточно большое, а платная подписка открывает дополнительные возможности и больше запросов. Также интерфейс личного кабинета отображает количество оставшихся запросов, что упрощает отслеживание и планирование работы.

5.1.3. Интеграция и документация

С точки зрения разработчика, работа с библиотекой AssemblyAI более удобна и приятна, поскольку она предоставляет простой и читаемый код, легко интегрируется в проекты и позволяет тонко настроить параметры распознавания. Документация на AssemblyAI также является приятной для изучения, что облегчает её использование.

В случае с Whisper API через RapidAPI также не возникает серьёзных проблем, но для использования через API необходимо следить за ограничениями по количеству запросов. И корректность передачи параметров и заголовков

Таблица 1. Сравнительная таблица: AssemblyAI библиотека и Whisper API

Параметр	AssemblyAI Библиотека	Whisper API
Качество распознавания речи	Отличное	Отличное, но слабее в распознавании сленга и жаргона
Удобство работы	Легко интегрируется	Удобный интерфейс и доступ через RapidAPI
API доступность	Ошибка 500	Не было проблем с доступом
Количество запросов	Много бесплатных запросов, есть платные планы	Малое количество бесплатных запросов через RapidAPI
Интерфейс отображения запросов	Приятный интерфейс с отображением оставшихся запросов	Нет отображения оставшихся запросов в API интерфейсе
Гибкость настройки	Позволяет тонко настраивать параметры	Менее гибкая настройка, но всё же достаточна для большинства задач
Документация	Подробная, легко читаемая документация	Хорошая документация через RapidAPI

6. ВЫВОДЫ

В ходе выполнения лабораторной работы был подробно изучен процесс взаимодействия с различными ИИ-сервисами для обработки аудиофайлов, в частности, с использованием Whisper и AssemblyAI. Оба сервиса продемонстрировали высокую точность в распознавании речи, несмотря на наличие посторонних шумов, что подчеркивает их способность эффективно работать в условиях неидеальных аудио-сигналов.

Однако, при использовании AssemblyAI API возникла ошибка 500, что привело к переходу на использование локальной библиотеки assemblyai. Несмотря на технические проблемы с доступом к API, использование библиотеки AssemblyAI в значительной степени компенсировало этот момент, так как она продемонстрировала превосходные результаты в распознавании не только стандартной речи, но и сложных, менее формальных выражений, таких как сленг и англицизмы.

При сравнении двух систем можно сделать вывод, что Whisper и AssemblyAI обе превосходно справляются с расшифровкой речи даже при наличии постороннего шума или дефектов записи. Однако обе модели показывают определенные трудности при обработке специфичных выражений, таких как англицизмы и специализированные термины, что может быть ограничением в приложениях, требующих точности в этих областях.

Вывод: Обе модели, несмотря на их высокую общую эффективность, имеют слабые места при распознавании специфичных выражений и терминов, в частности англицизмов и жаргонизмов. Тем не менее, AssemblyAI показал себя более устойчивым в распознавании таких сложных элементов речи, что может быть полезным для приложений, требующих точности в этих аспектах.

ИСТОЧНИКИ

1. OpenAI Whisper Speech-to-Text API [Электронный ресурс]. URL: https://rapidapi.com/data-elysium-software-incorporation-data-elysium-software-incorporation-default/api/openai-whisper-speech-to-text-api (дата обращения: 15.10.2025).

2. AssemblyAI Documentation [Электронный ресурс]. URL: https://www.assemblyai.com/docs (дата обращения: 15.10.2025).

3. Requests: HTTP for Humans [Электронный ресурс]. URL: https://requests.readthedocs.io/en/latest/ (дата обращения: 15.10.2025

4. PyQt5 Documentation [Электронный ресурс]. URL: https://www.riverbankcomputing.com/static/Docs/PyQt5/ (дата обращения: 15.10.2025).

5. AI Methods GitHub Repository [Электронный ресурс]. URL: https://github.com/fesevu/AI-methods (дата обращения: 23.10.2025).

ПРИЛОЖЕНИЕ А. Скриншоты с результатами Whisper

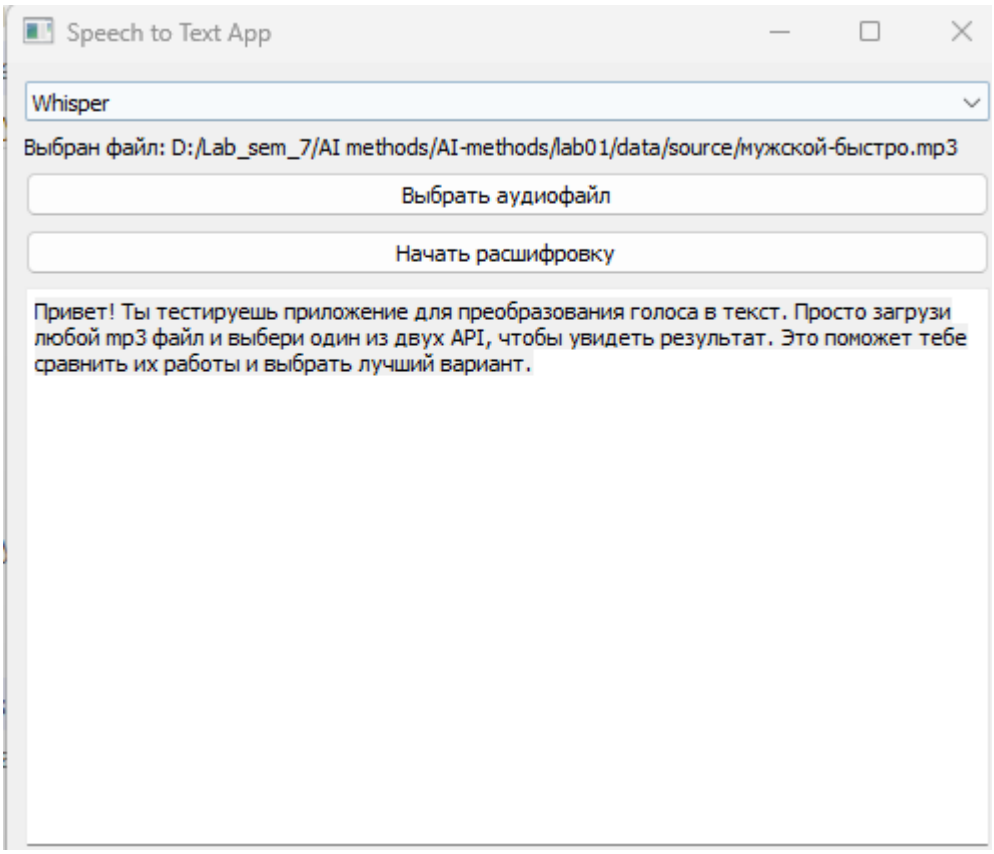

Рис. 4. Результат Whisper для файла с мужской быстрой речью.

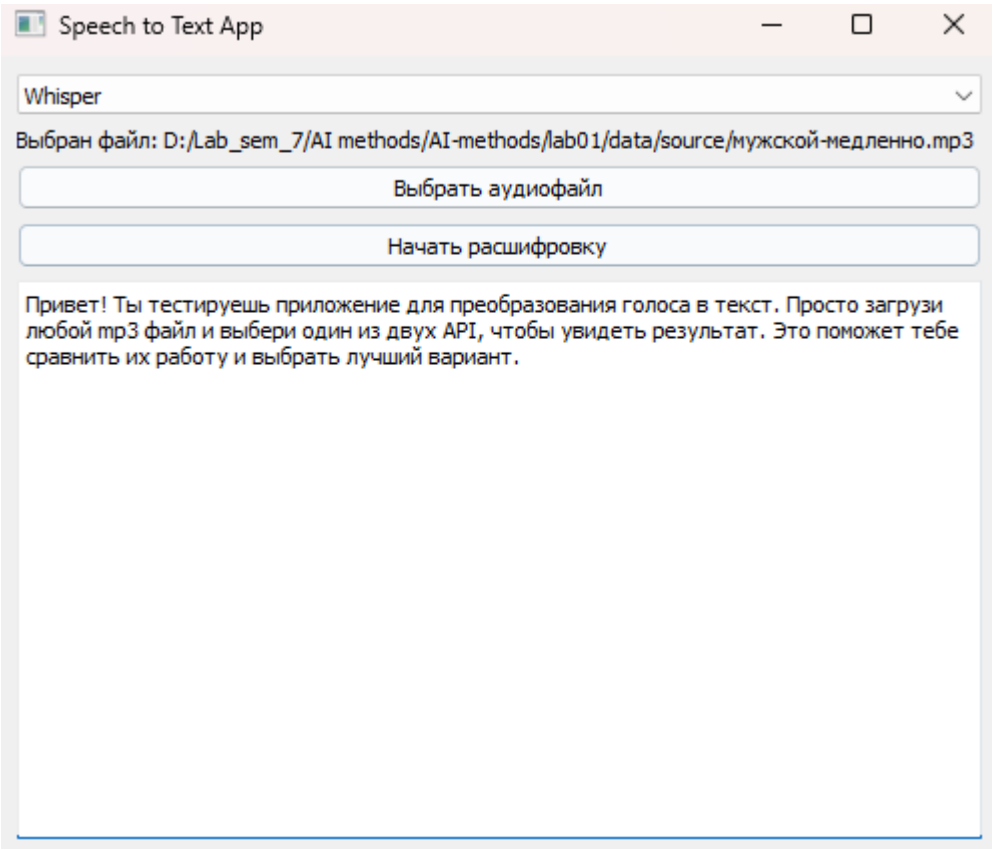

Рис. 5. Результат Whisper для файла с мужской медленной речью.

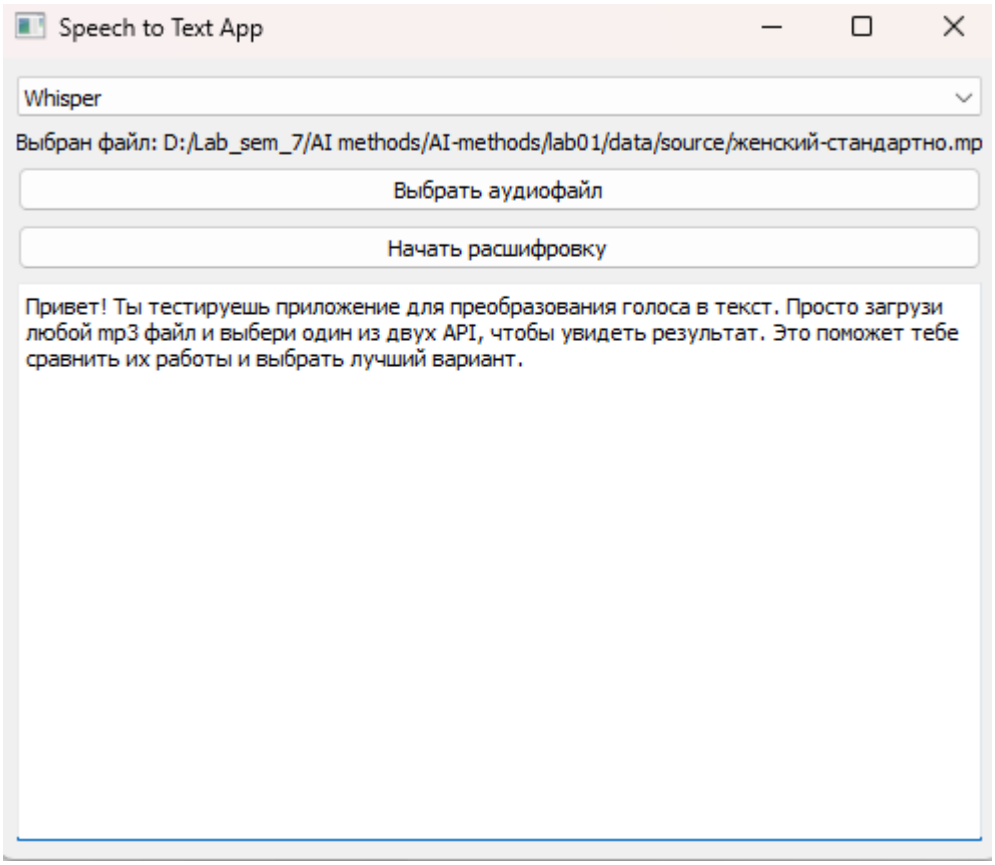

Рис. 6. Результат Whisper для файла с женской стандартной речью.

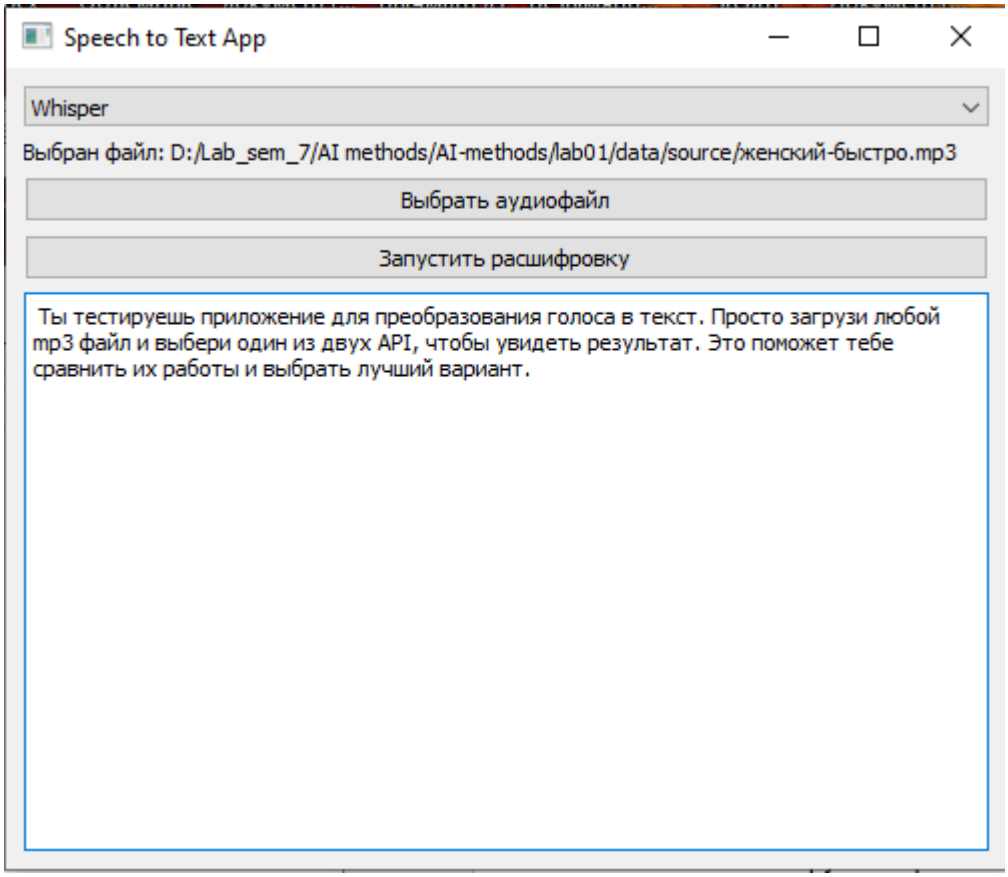

Рис. 7. Результат Whisper для файла с женской быстрой речью.

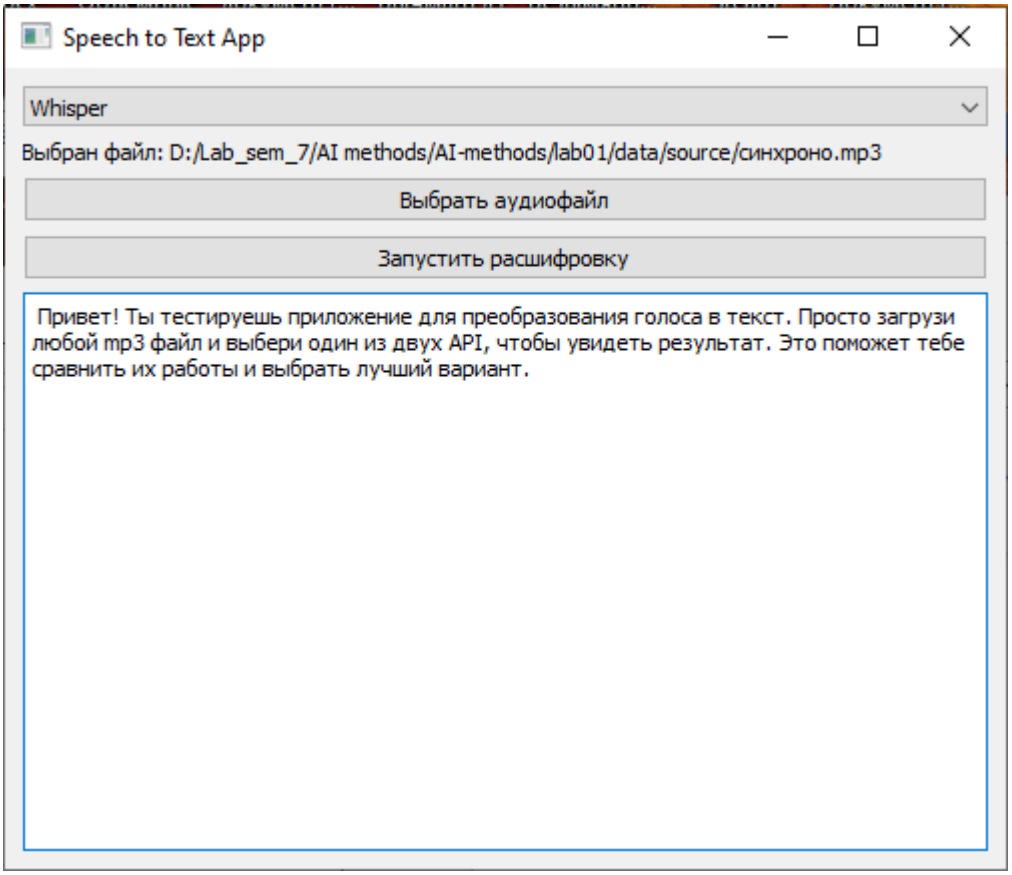

Рис. 8. Результат Whisper для файла с синхронной мужской и женской речью

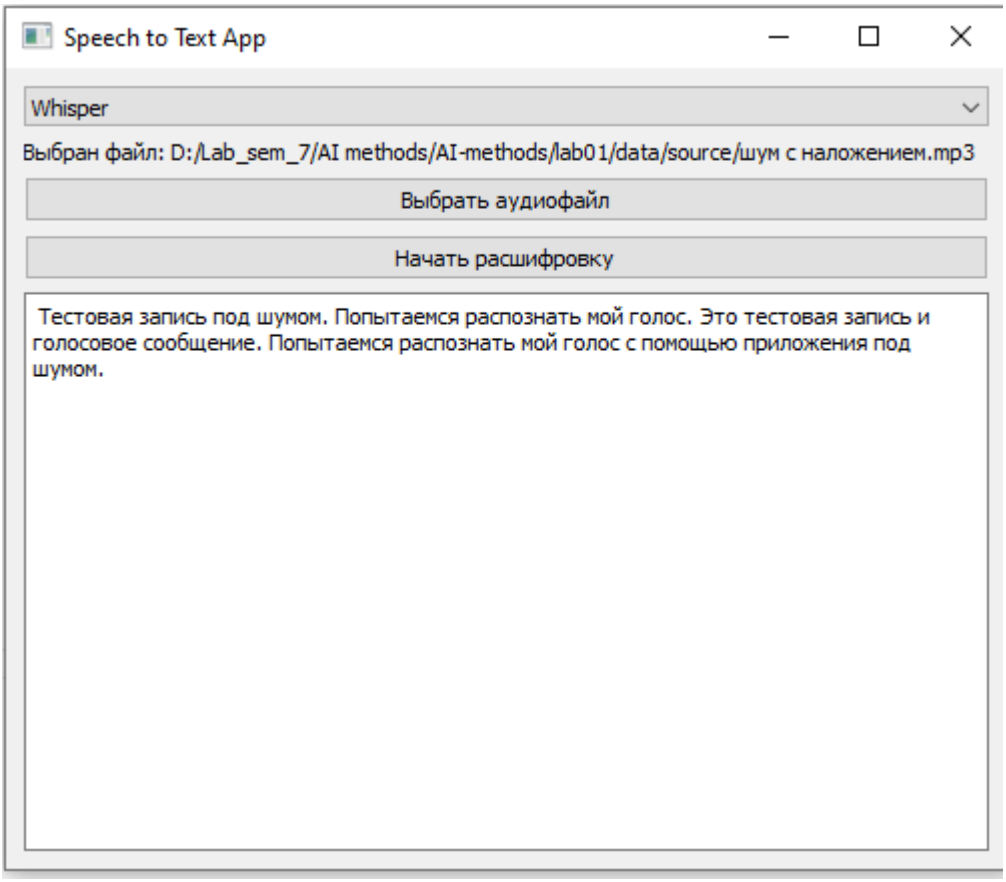

Рис. 9. Результат Whisper для файла с мужской речью с посторонними шумами.

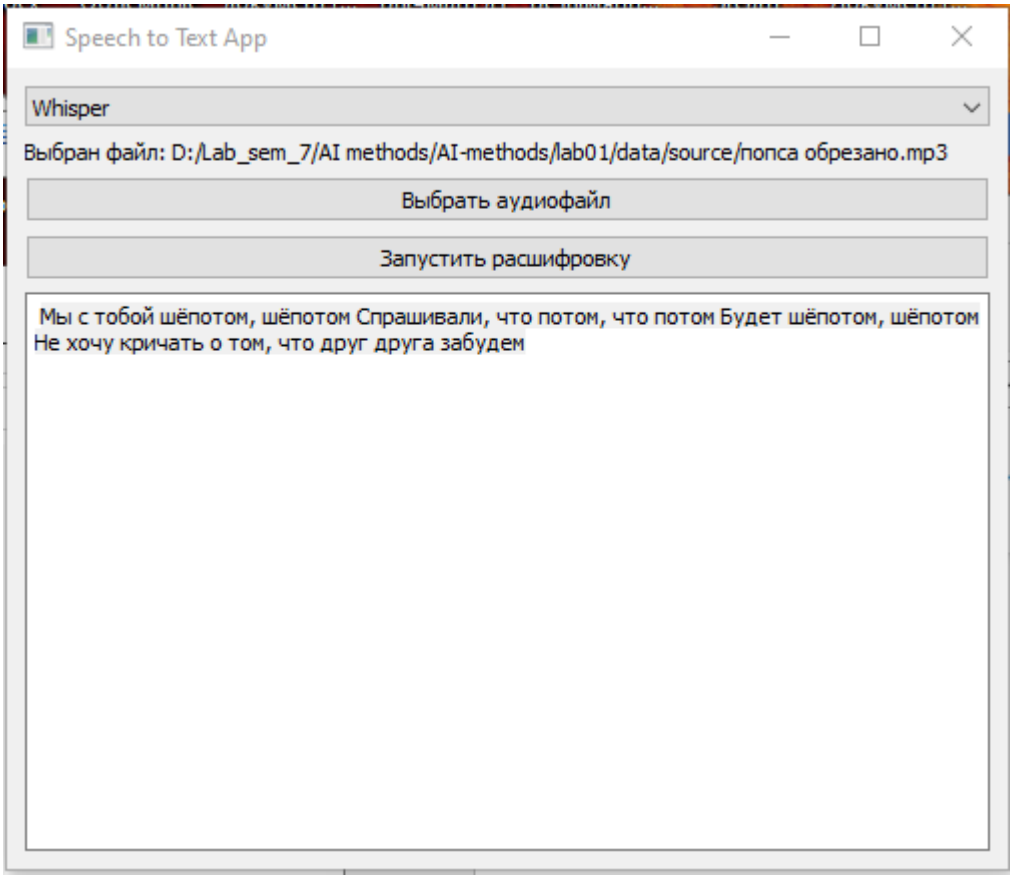

Рис. 10. Результат Whisper для файла с отрывком из песни Сергея Лазарева “Шепотом”.

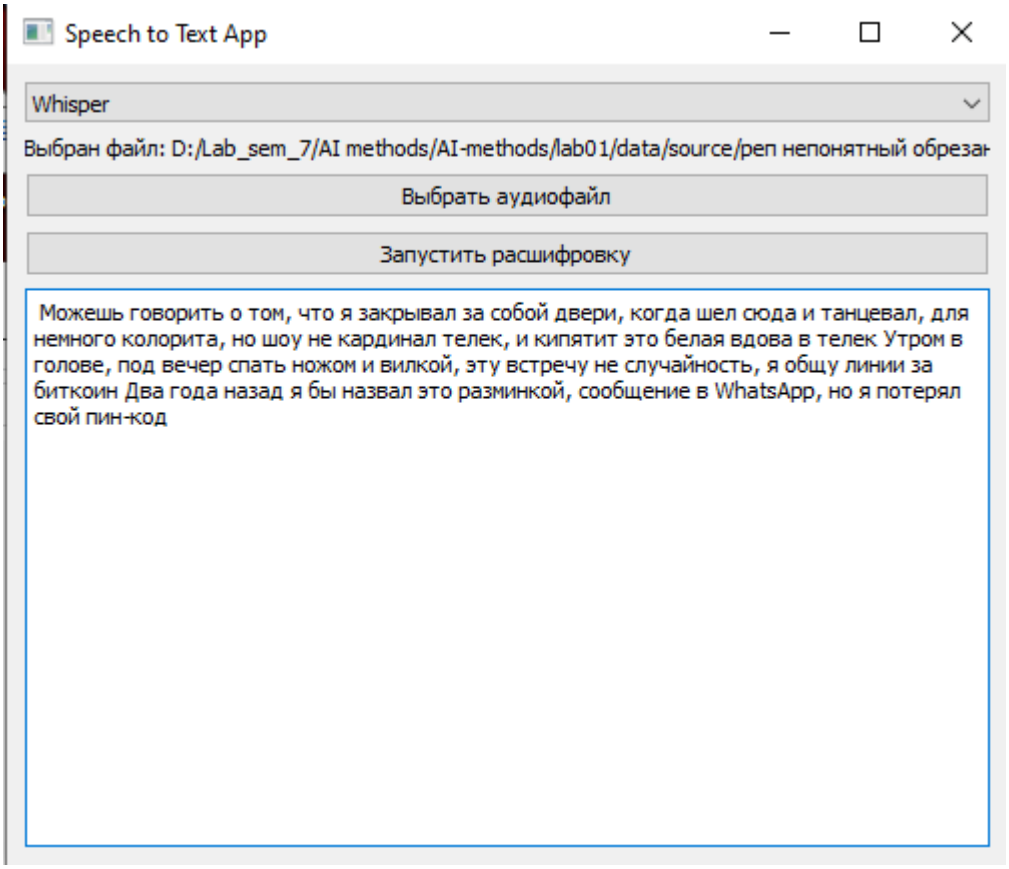

Рис. 11. Результат Whisper для файла с отрывком из песни Скриптонита “Привычка”.

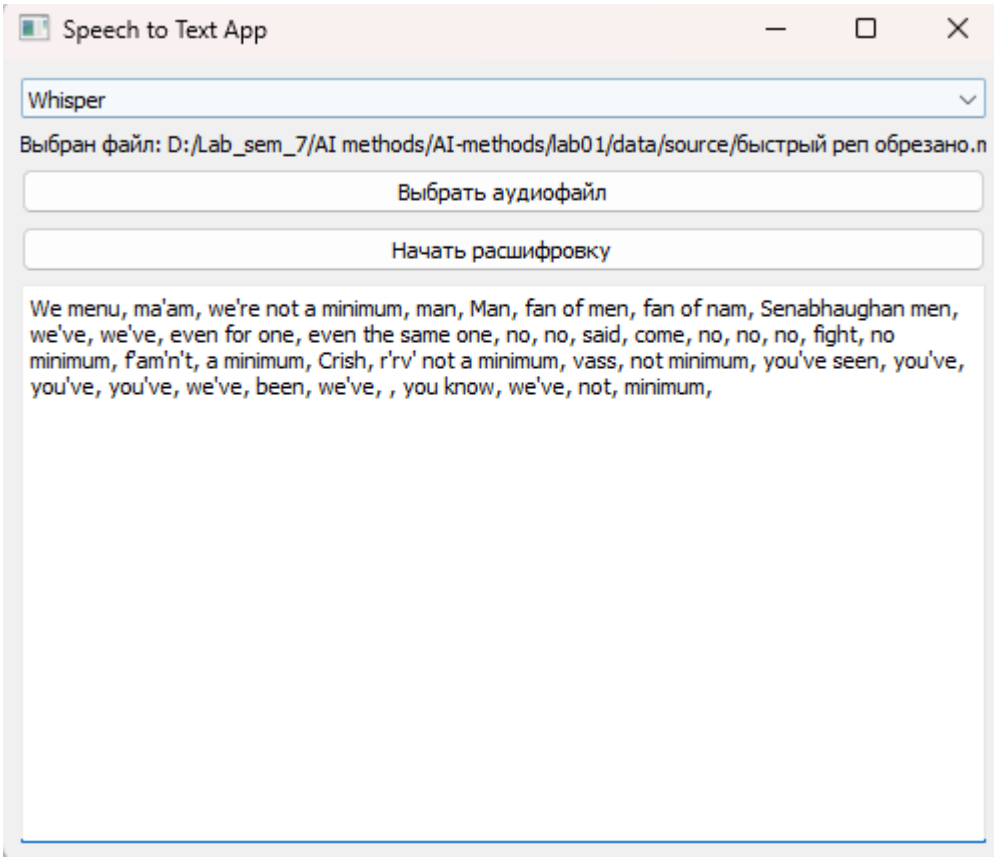

Рис. 12. Результат Whisper для файла с отрывком из песни Fike & Jambazi “Minimun”.

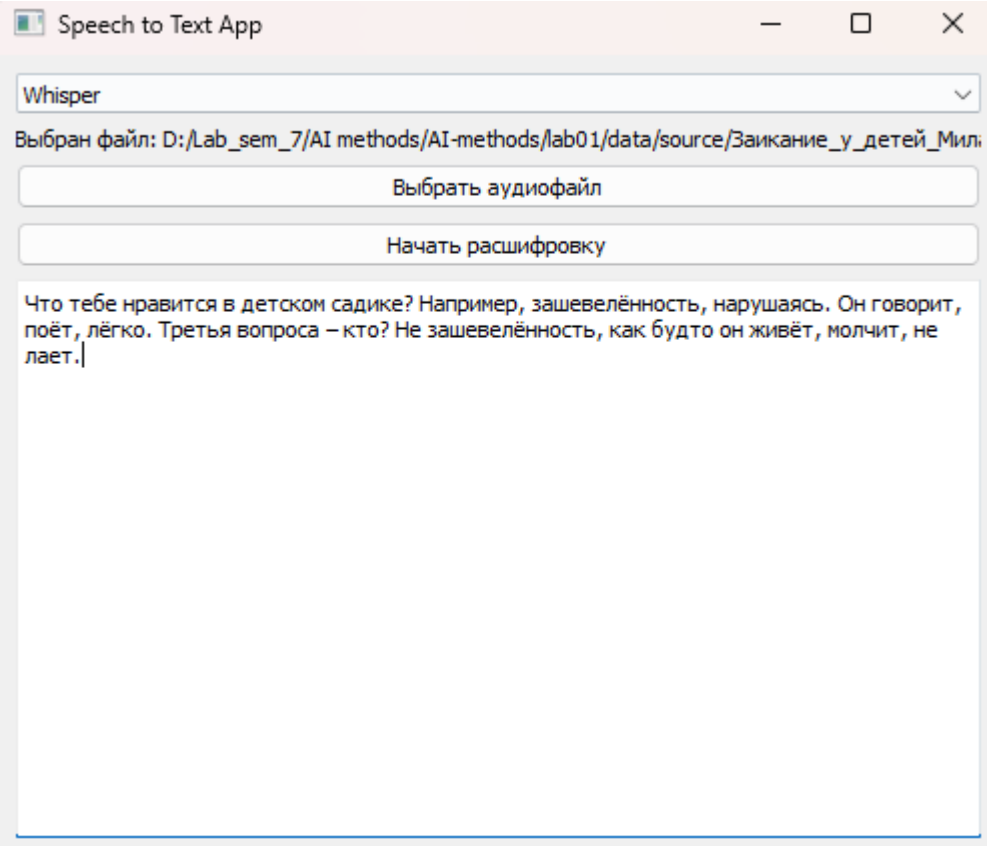

Рис. 13. Результат Whisper для файла с речью 5-летнего ребёнка с заиканием.

ПРИЛОЖЕНИЕ Б. Скриншоты с результатами AssemblyAI

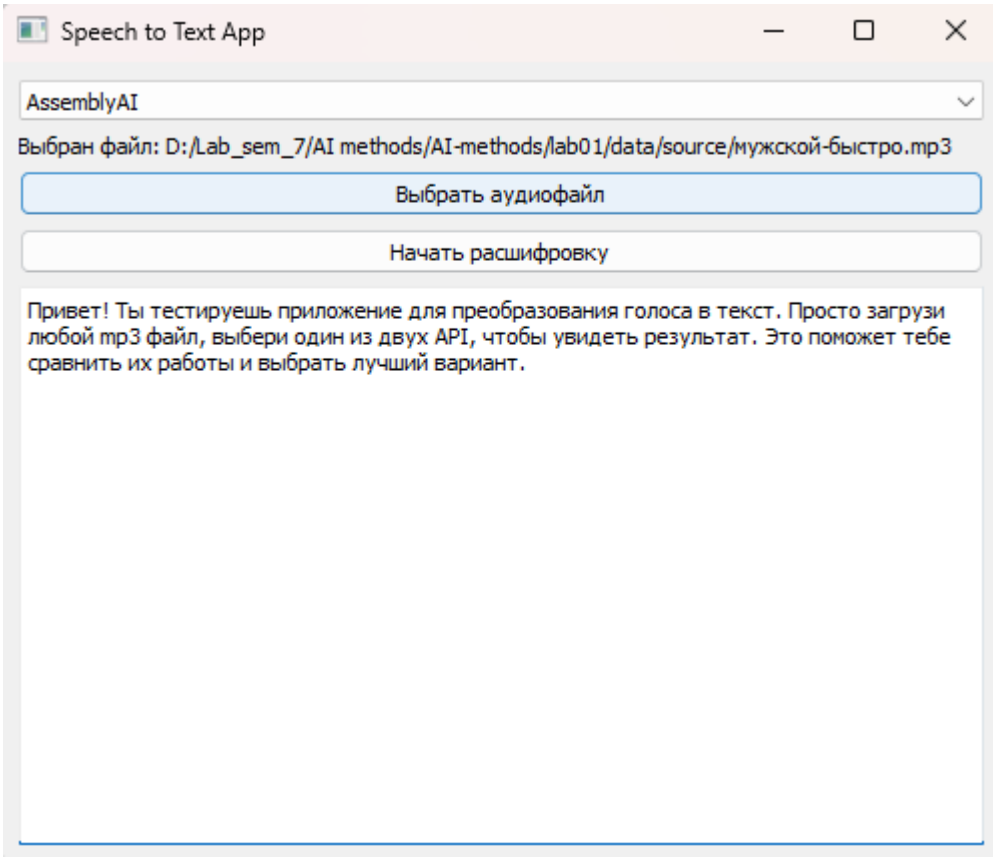

Рис. 14. Результат AssemblyAI для файла с мужской быстрой речью.

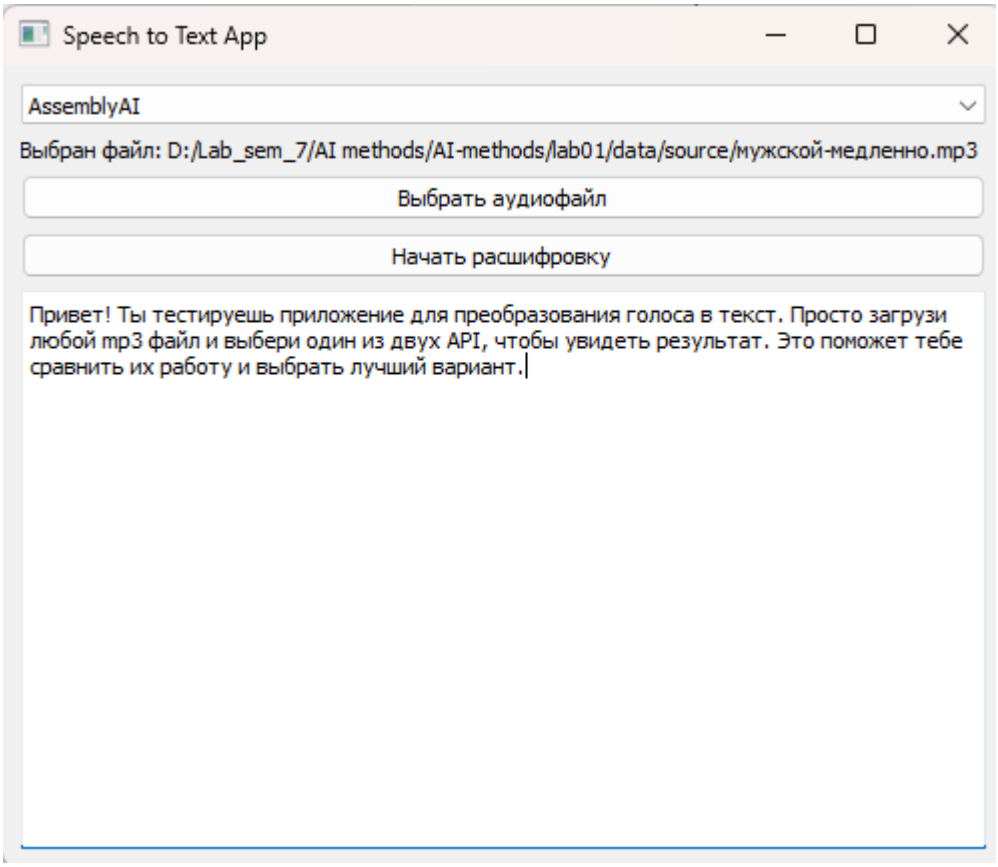

Рис. 15. Результат AssemblyAI для файла с мужской медленной речью.

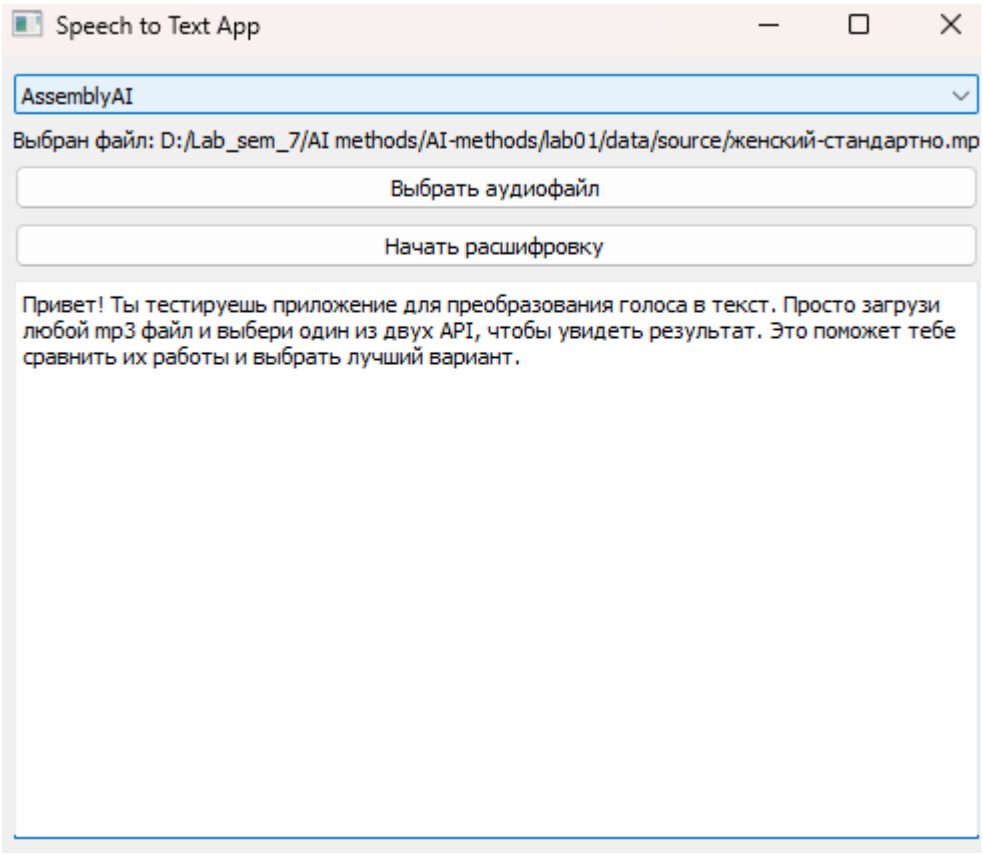

Рис. 16. Результат AssemblyAI для файла с женской стандартной речью.

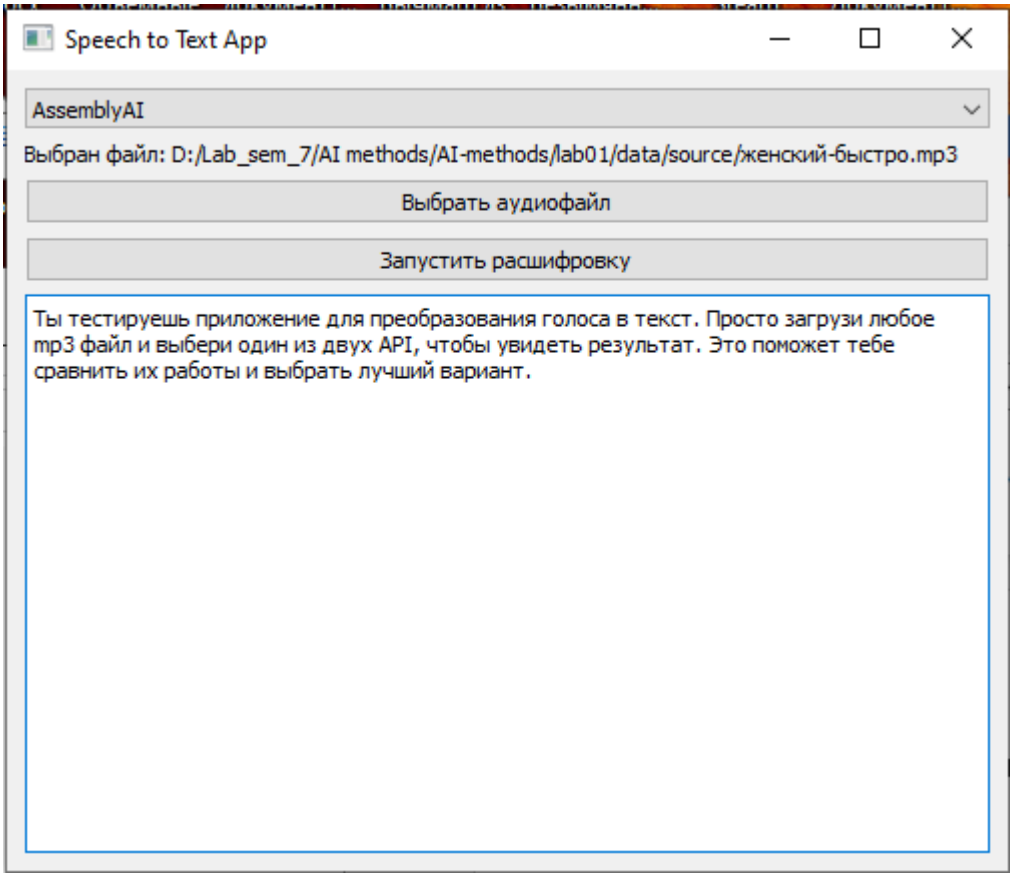

Рис. 17. Результат AssemblyAI для файла с женской быстрой речью.

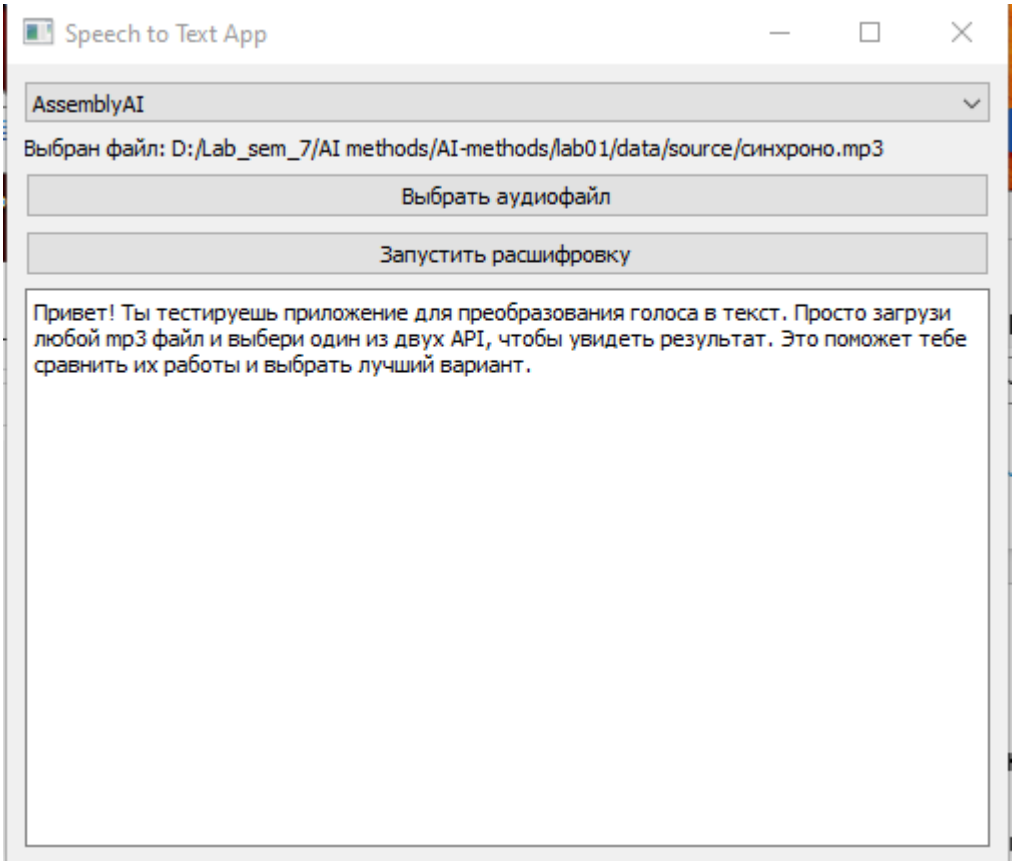

Рис. 18. Результат AssemblyAI для файла с синхронной мужской и женской речью

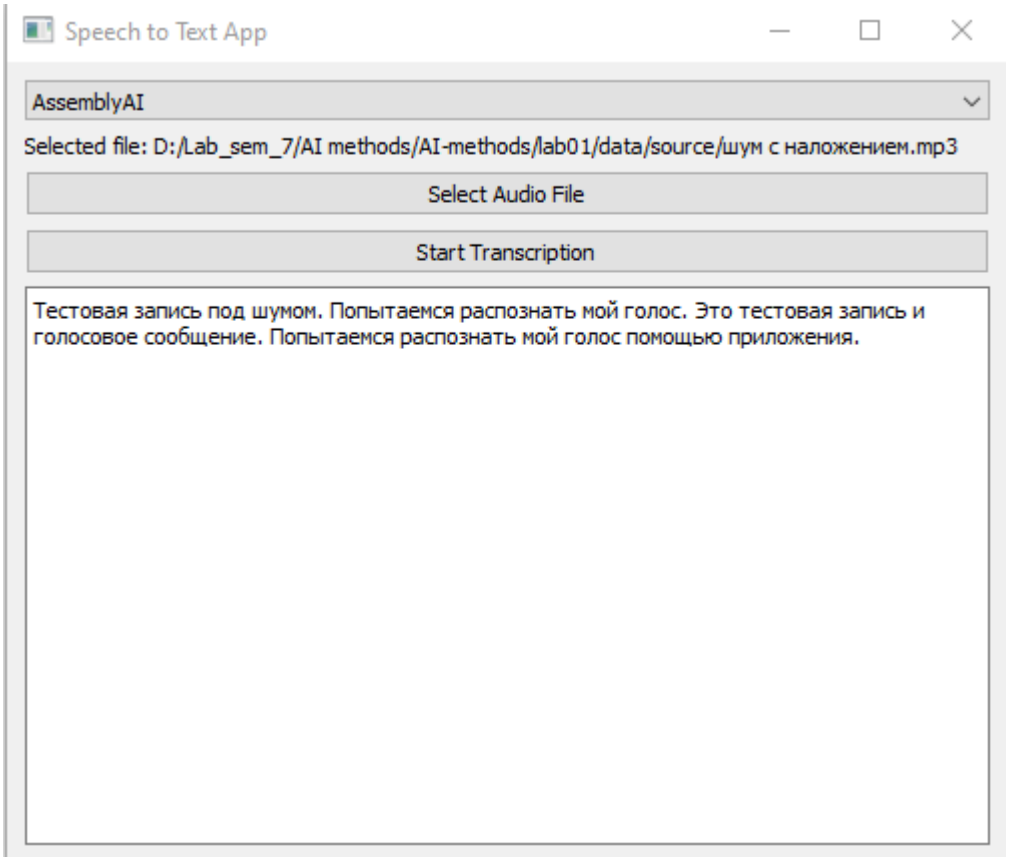

Рис. 19. Результат AssemblyAI для файла с мужской речью с посторонними шумами.

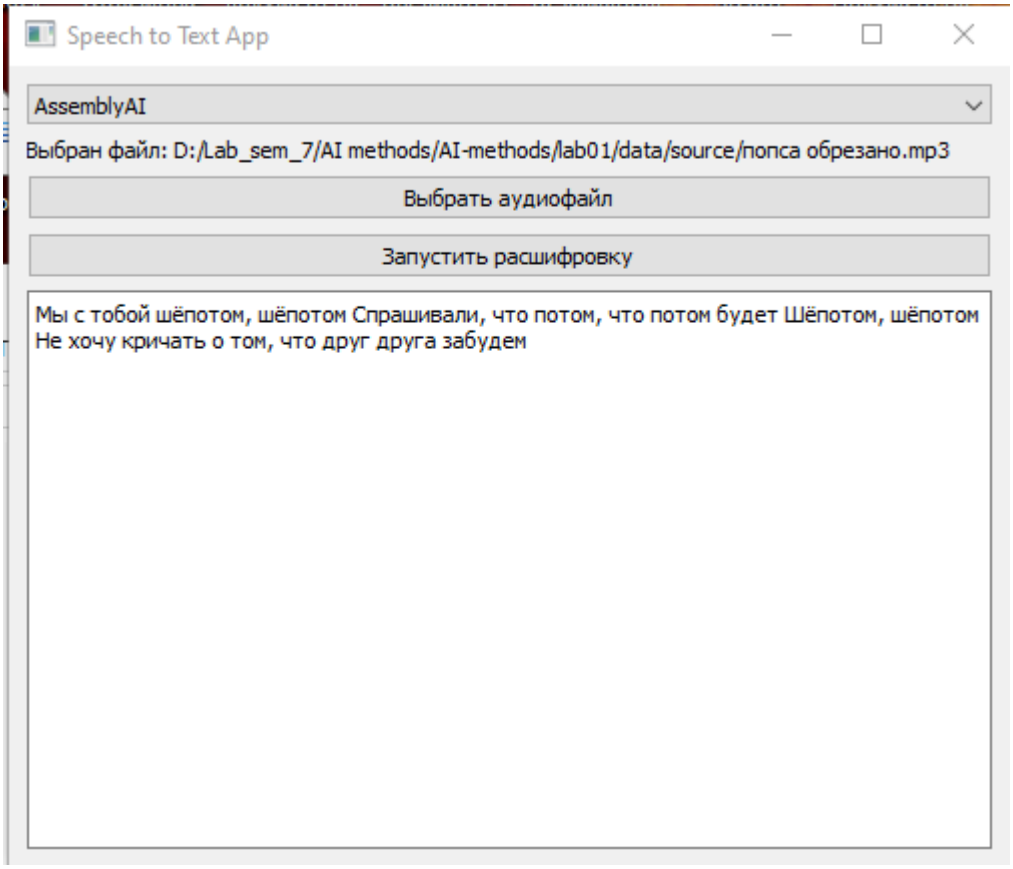

Рис. 20. Результат AssemblyAI для файла с отрывком из песни Сергея Лазарева “Шепотом”.

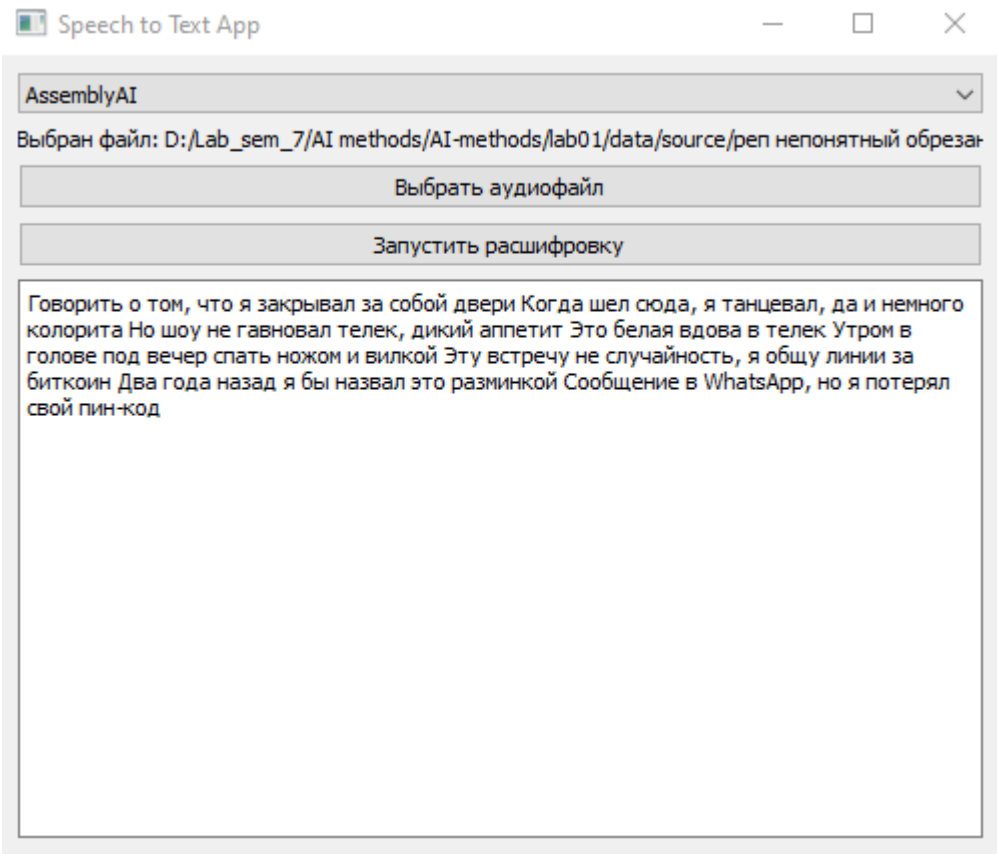

Рис. 21. Результат AssemblyAI для файла с отрывком из песни Скриптонита “Привычка”.

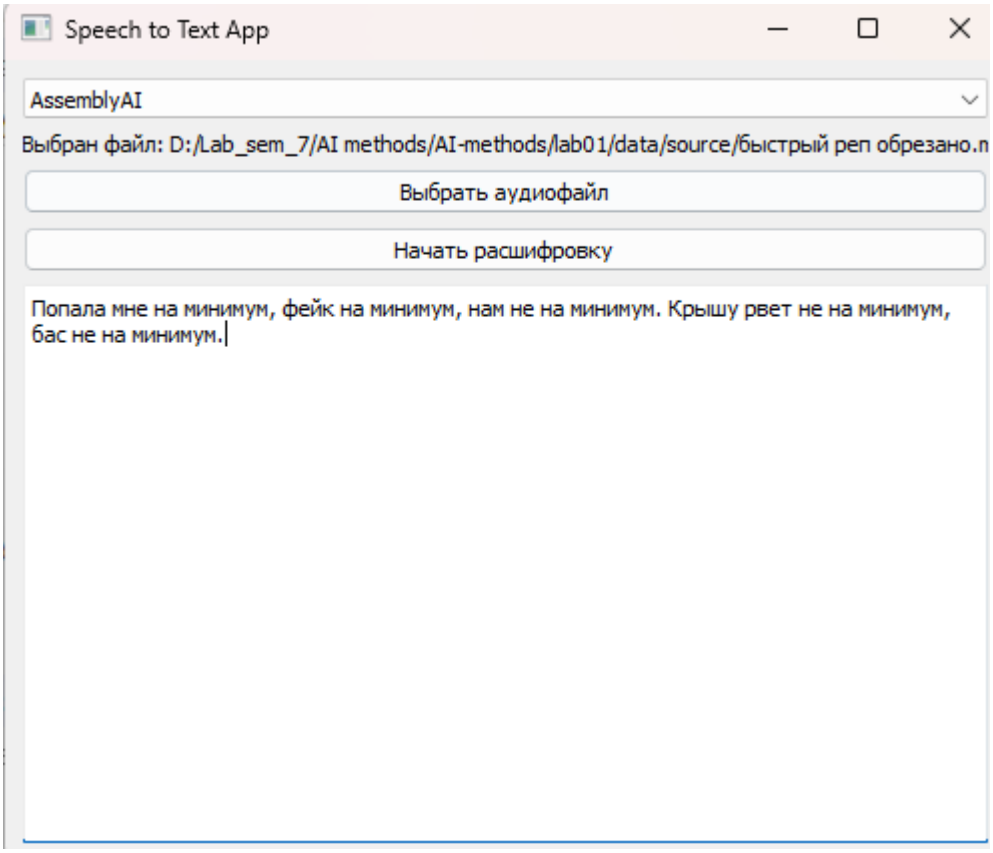

Рис. 22. Результат AssemblyAI для файла с отрывком из песни Fike & Jambazi “Minimun”.

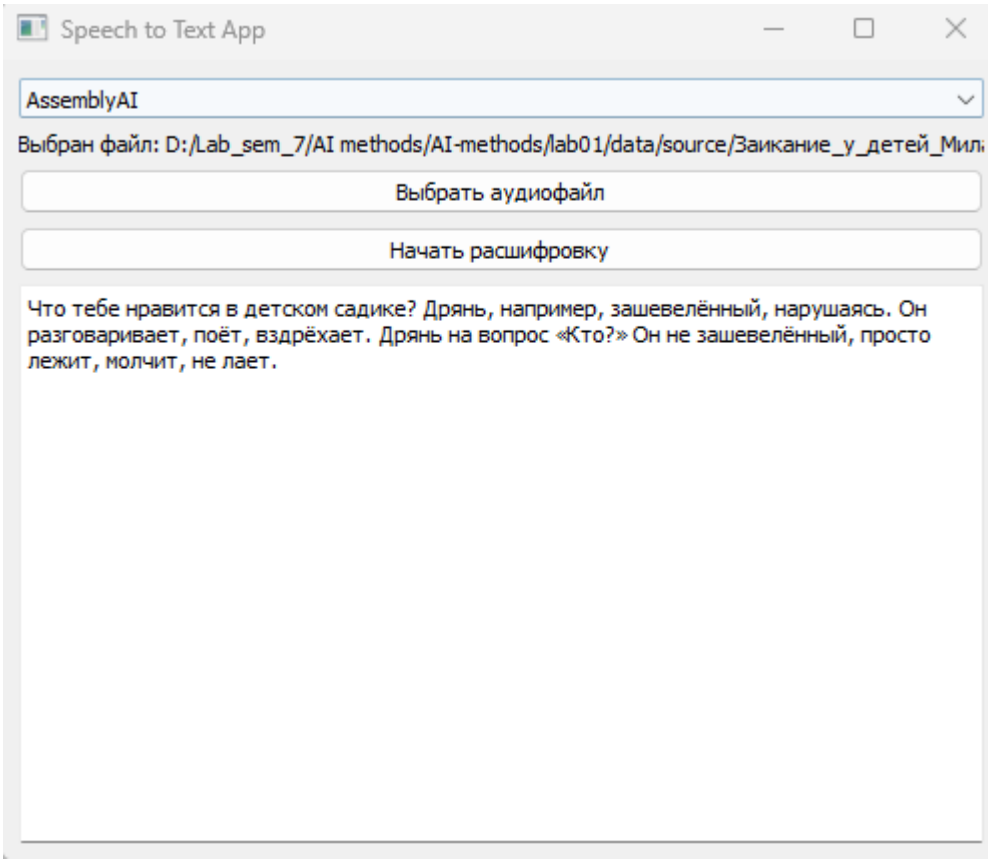

Рис. 23. Результат AssemblyAI для файла с речью 5-летнего ребёнка с заиканием.

ПРИЛОЖЕНИЕ В. Исходный код программы

Содержимое файла main.py

```
import sys
from PyQt5.QtWidgets import QApplication
from interface import SpeechToTextApp

# Запуск приложения


def main():
    print("Запуск приложения...")
    app = QApplication(sys.argv)
    window = SpeechToTextApp()
    window.show()
    sys.exit(app.exec_())


if __name__ == "__main__":
    main()
```

Содержимое файла interface.py

```
from PyQt5.QtWidgets import QMainWindow, QVBoxLayout, QLabel, QComboBox, QPushButton,
QTextEdit, QFileDialog
from PyQt5.QtCore import QThread, pyqtSignal
from integration import transcribe_with_whisper, transcribe_with_assembly


class TranscribeThread(QThread):
    result_ready = pyqtSignal(str)

    def __init__(self, file_name, api):
        super().__init__()
        self.file_name = file_name
        self.api = api

    def run(self):
        if self.api == "Whisper":
            result = transcribe_with_whisper(self.file_name)
        elif self.api == "AssemblyAI":
            result = transcribe_with_assembly(self.file_name)
        else:
            result = "Неверный выбор API."
        self.result_ready.emit(result)


class SpeechToTextApp(QMainWindow):
    def __init__(self) -> None:
        super().__init__()

        print("Инициализация SpeechToTextApp...")
```

```
        self.setWindowTitle("Speech to Text App")
        self.setGeometry(200, 200, 500, 400)

        # Компоненты интерфейса
        layout = QVBoxLayout()

        self.api_selector = QComboBox()
        self.api_selector.addItems(["Whisper", "AssemblyAI"])
        layout.addWidget(self.api_selector)

        self.file_label = QLabel("Файл не выбран")
        layout.addWidget(self.file_label)

        self.select_file_btn = QPushButton("Выбрать аудиофайл")
        self.select_file_btn.clicked.connect(self.select_file)
        layout.addWidget(self.select_file_btn)

        self.transcribe_btn = QPushButton("Начать расшифровку")
        self.transcribe_btn.clicked.connect(self.transcribe_audio)
        layout.addWidget(self.transcribe_btn)

        self.result_area = QTextEdit()
        layout.addWidget(self.result_area)

        self.central_widget = QLabel()
        self.central_widget.setLayout(layout)
        self.setCentralWidget(self.central_widget)

        self.selected_file: str = ""

    def select_file(self) -> None:
        """
        Функция для открытия диалогового окна и выбора аудиофайла.
        Устанавливает выбранный путь к файлу в self.selected_file.
        """
        print("Открытие диалогового окна для выбора аудиофайла...")
        options = QFileDialog.Options()
        file_name, _ = QFileDialog.getOpenFileName(
            self, "Выбрать файл", "", "Аудиофайлы (*.mp3 *.wav)", options=options)
        if file_name:
            self.selected_file = file_name
            self.file_label.setText(f"Выбран файл: {file_name}")
            print(f"Выбран файл: {file_name}")
        else:
            print("Файл не выбран.")

    def transcribe_audio(self) -> None:
        """
        Функция для расшифровки выбранного аудиофайла с использованием выбранного API
(Whisper или AssemblyAI).
        Отображает расшифрованный текст в текстовом поле результата.
        """
        selected_api = self.api_selector.currentText()
        print(f"Выбранный API для расшифровки: {selected_api}")

        if self.selected_file:
```

```
            print(f"Начало расшифровки для файла: {self.selected_file}")
            self.thread = TranscribeThread(self.selected_file, selected_api)
            self.thread.result_ready.connect(self.display_result)
            self.thread.start()
        else:
            self.result_area.setText("Пожалуйста, выберите файл.")
            print("Файл для расшифровки не выбран.")

    def display_result(self, result: str) -> None:
        """
        Функция для отображения результата расшифровки.
        """
        self.result_area.setText(result.strip())
        print(f"Результат расшифровки: {result}")
```

Содержимое файла integration.py

```
import os
import requests
import assemblyai as aai
from dotenv import load_dotenv

# Загрузка переменных окружения
load_dotenv()

# Установка API ключей из переменных окружения
aai.settings.api_key = os.getenv("ASSEMBLYAI_API_KEY")
RAPIDAPI_KEY = os.getenv("RAPIDAPI_KEY")

# Расшифровка аудио с использованием Whisper API


def transcribe_with_whisper(file_name: str) -> str:
    """
    Функция для расшифровки аудио с использованием Whisper API.
    Аргументы:
        file_name (str): Путь к аудиофайлу для расшифровки.
    Возвращает:
        str: Расшифрованный текст из аудиофайла или сообщение об ошибке.
    """
    print(f"Начало расшифровки Whisper для файла: {file_name}")
    url = "https://openai-whisper-speech-to-text-api.p.rapidapi.com/transcribe"

    with open(file_name, 'rb') as audio_file:
        files = {
            'file': ('audio.mp3', audio_file),
            'type': (None, 'RAPID'),
            'response_format': (None, 'JSON')
        }

        headers = {
            "x-rapidapi-key": RAPIDAPI_KEY,
            "x-rapidapi-host": "openai-whisper-speech-to-text-api.p.rapidapi.com"
        }
```

```
        try:
            response = requests.post(
                url, files=files, headers=headers, timeout=60)
            print(
                f"Получен ответ от Whisper API с кодом статуса: "
                f"{response.status_code}"
            )

            if response.status_code == 200:
                result = response.json()
                print(f"Результат расшифровки: {result}")
                return result['response'].get('text', 'Текст не найден в ответе.')
            else:
                print(f"Ошибка при расшифровке Whisper: {response.text}")
                return f"Ошибка: {response.status_code}, {response.text}"
        except requests.exceptions.RequestException as e:
            print(f"Ошибка при выполнении запроса Whisper: {e}")
            return f"Ошибка при выполнении запроса: {e}"

# Расшифровка аудио с использованием AssemblyAI API


def transcribe_with_assembly(file_name: str) -> str:
    """
    Функция для расшифровки аудио с использованием AssemblyAI API.
    Аргументы:
        file_name (str): Путь к аудиофайлу для расшифровки.
    Возвращает:
        str: Расшифрованный текст из аудиофайла или сообщение об ошибке.
    """
    print(f"Начало расшифровки AssemblyAI для файла: {file_name}")
    transcriber = aai.Transcriber()
    config = aai.TranscriptionConfig(language_code="ru")
    transcript = transcriber.transcribe(file_name, config)
    print(f"Статус расшифровки AssemblyAI: {transcript.status}")
    if transcript.status == aai.TranscriptStatus.error:
        print(f"Ошибка при расшифровке AssemblyAI: {transcript.error}")
        return transcript.error
    else:
        print(f"Результат расшифровки: {transcript.text}")
        return transcript.text
```