

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.04.01 «Информатика и вычислительная техника»

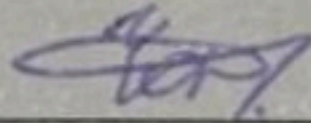
Магистерская программа: «Прикладной искусственный интеллект»

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему

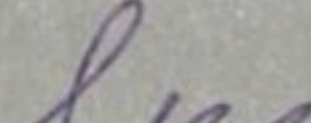
ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ
МУЗЫКАЛЬНЫХ ПРОИЗВЕДЕНИЙ

Студент

 10.06.2026
(подпись, дата)

Чапаев Илья Алексеевич
(ФИО полностью)

Руководитель

 10.06.2026
(подпись, дата)

Митрохина Н.Ю.
(фамилия, инициалы)

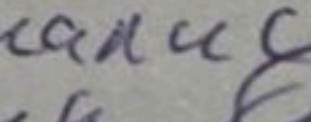
Нормоконтролер

 10.06.2026
(подпись, дата)

Бычков А.С.
(фамилия, инициалы)

Рецензент

главный специалист
до ИИИ Рубин Василий
(должность, место работы)

 10.06.2026
(подпись, дата)

Васильев А.В.
(фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 22.05.2026 № 14)

Заведующий кафедрой

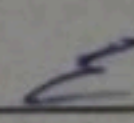

(подпись)

М.А. Митрохин
(фамилия, инициалы)

Работа защищена с отметкой

отлично 23.06.2026
(протокол заседания ГЭК от № 28)

Секретарь ГЭК


(подпись)

Е.И. Гурин
(фамилия, инициалы)

Пенза 2026

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

М.А. Митрохин

« 20 » 09 2024 г.

**ЗАДАНИЕ
НА МАГИСТЕРСКУЮ ДИССЕРТАЦИЮ**

1. Студент Чапаев Илья Алексеевич гр. 24ВВИм1 факультета ВТ

2. Тема работы Интеллектуальный анализ музыкальных произведений

Тема утверждена приказом ПГУ № 920/о от "20" сентября 2024г.

3. Руководитель магистерской диссертации к.т.н., доцент кафедры ВТ ПГУ Митрохина Н.Ю.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

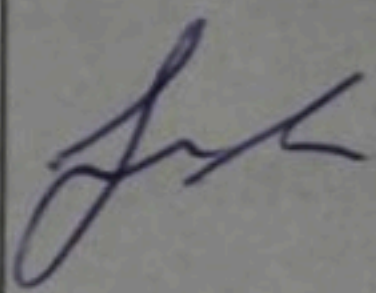
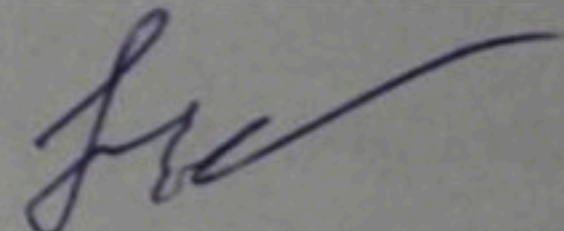
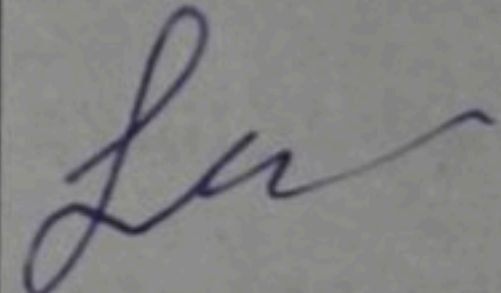
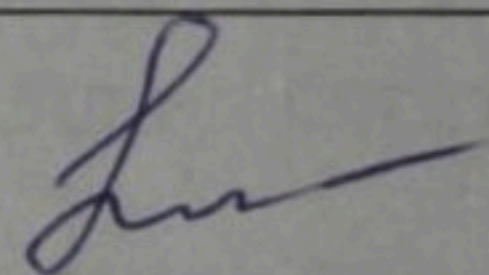
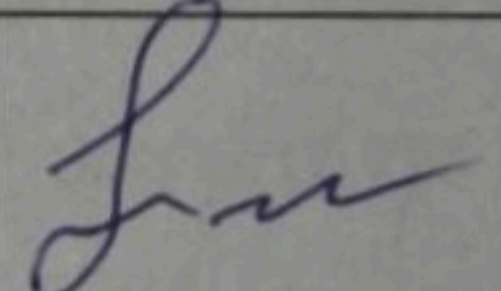
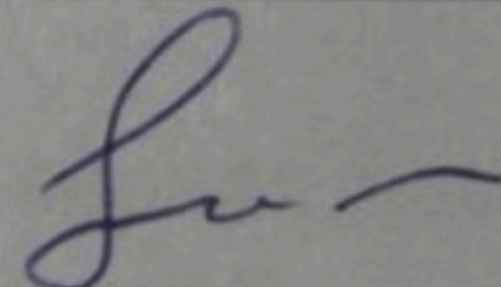
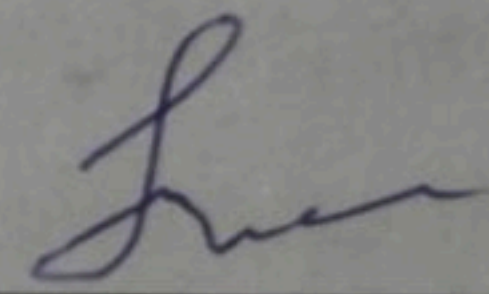
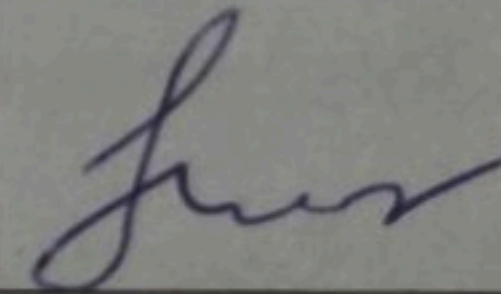
Исследовать современные подходы прикладного искусственного интеллекта к анализу музыкальных произведений (Music Information Retrieval) и на основе обоснованно выбранных предобученных нейросетевых моделей разработать сервис комплексного анализа песни. По загруженной аудиозаписи сервис должен определять тональность, темп, тактовый размер, ритмическую сетку долей, жанр, эмоциональную окраску, последовательность аккордов по долям такта и раскладку слогов вокального текста по ритмической сетке. Исходные данные: открытые предобученные модели MIR (разделение источников, трекинг долей, распознавание тональности, жанра, эмоций, аккордов, распознавание и выравнивание вокального текста), набор тестовых аудиозаписей, вычислительные ресурсы одной видеокарты (целевой пик потребления видеопамати ниже 12 ГБ). Сервис реализовать как клиент-серверное приложение с асинхронной обработкой и контейнерным развёртыванием.

5. Перечень подлежащих разработке вопросов

Провести анализ предметной области интеллектуального анализа музыки: рассмотреть задачи MIR, способы представления аудиосигнала, архитектуры нейронных сетей и существующие сервисы-аналоги. Для каждой подзадачи сервиса выполнить сравнительный анализ доступных моделей и обоснованно выбрать решение с учётом качества, вычислительной стоимости, лицензионной чистоты и поддержки русского языка. Формализовать задачу выравнивания слогов вокального текста на ритмическую сетку (Syllable-Rhythm Alignment), предложить метрику оценки её решения и детерминированный алгоритм решения. Спроектировать архитектуру многомодульного конвейера анализа с учётом ограничения вычислительных ресурсов одной видеокарты. Реализовать сервис:

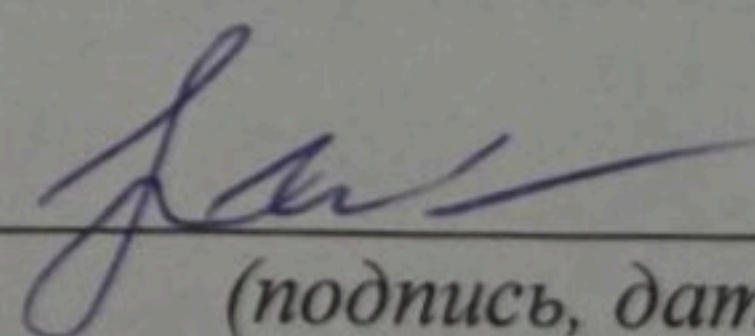
серверную часть с модулями анализа и оркестратором, программный интерфейс, пользовательский интерфейс с редактором ритмической сетки и контейнерное развёртывание. Провести экспериментальную оценку качества выбранных моделей по публичным данным и проверку работоспособности сервиса на наборе реальных композиций.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объём работ ы	Срок выполнени я	Подпись руководителя
1.	Анализ предметной области MIR (задачи, представления звука, архитектуры, сервисы-аналоги	10%	20.09.24– 15.11.2024	
2.	Сравнительный анализ и обоснованный выбор моделей по подзадачам анализа	15%	16.11.2024– 20.12.2024	
3.	Формализация задачи слога-долевого выравнивания и проектирование архитектуры конвейера	20%	21.12.2024– 31.01.2025	
4.	Реализация серверной части: модули анализа, оркестратор, программный интерфейс	25%	01.02.2025– 15.03.2025	
5.	Реализация пользовательского интерфейса и редактора ритмической сетки, развёртывание	10%	16.03.2025– 31.01.2026	
6.	Экспериментальная проверка и функциональное тестирование сервиса	5%	01.02.2026– 28.02.2026	
7.	Анализ полученных результатов и оценка качества	5%	01.03.2026– 30.04.2026	
8	Оформление пояснительной записки	10%	01.05.2026– 20.05.2026	

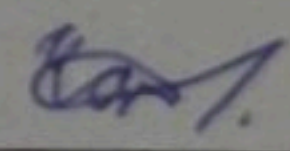
Дата выдачи задания "20" сентября 2024г.

Руководитель магистерской диссертации


(подпись, дата)

Митрохина Н.Ю.

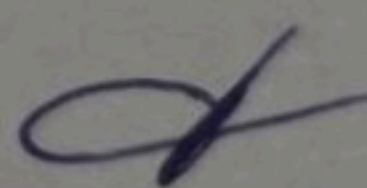
Задание к исполнению принял студент


(подпись, дата)

Чапаев И.А.

Работу к защите допустить

Декан факультета

 18.06.2026

Фионова Л.Р.

РЕФЕРАТ

Пояснительная записка содержит 105 листов, 16 рисунков, 19 таблиц, 49 источников, 3 приложения.

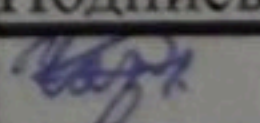
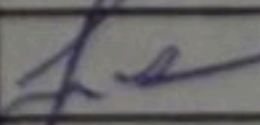
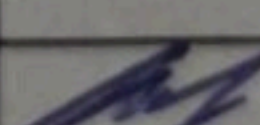
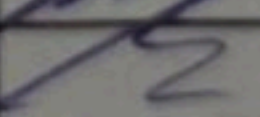
ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ МУЗЫКИ, MUSIC INFORMATION RETRIEVAL, ГЛУБОКОЕ ОБУЧЕНИЕ, РАЗДЕЛЕНИЕ ИСТОЧНИКОВ, ТРЕКИНГ ДОЛЕЙ, ТОНАЛЬНОСТЬ, ЖАНР, ЭМОЦИЯ, РАСПОЗНАВАНИЕ АККОРДОВ, ВЫРАВНИВАНИЕ СЛОГОВ

Целью работы является разработка сервиса комплексного анализа музыкальных произведений на основе методов прикладного искусственного интеллекта.

Проанализированы современные модели и подходы извлечения музыкальной информации (MIR), и по каждой подзадаче анализа обоснован выбор решения. Разработанный сервис по загруженной аудиозаписи определяет тональность, темп, тактовый размер, жанр, эмоциональную окраску, последовательность аккордов по долям такта и раскладку слогов вокального текста по ритмической сетке. Формализована задача слога- долевого выравнивания слогов и предложен детерминированный алгоритм ее решения.

Сервис реализован как клиент-серверное приложение: серверная часть на FastAPI с оркестратором конвейера моделей, пользовательский интерфейс на React с редактором ритмической сетки, контейнерное развертывание. Работоспособность подтверждена прогоном на наборе из пяти композиций.

Область применения – программные средства для музыкантов, педагогов и учеников музыкальных школ при разборе произведений.

					ПГУ 1.090401.27.001 ПЗ				
Изм.	Лист	№ докум	Подпись	Дата	Интеллектуальный анализ музыкальных произведений Пояснительная записка	Лит.	Лист	Листов	
Разраб.		Чапаев И.А.		10.06.26					
Провер.		Митрохина Н.Ю.		10.06.26			2	105	
						ФВТ, каф. ВТ гр. 24ВВИм1			
Н. Контр.		Бычков А.С.		10.06.26					
Утв.		Митрохин М.А.		10.06.26					

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ	5
ВВЕДЕНИЕ.....	7
1 Обзор предметной области интеллектуального анализа музыки	12
1.1 Задачи интеллектуального анализа музыкальных произведений	12
1.2 Представления аудиосигнала.....	14
1.3 Архитектуры нейронных сетей для анализа музыки	17
1.4 Обзор существующих программных средств и сервисов.....	20
1.5 Постановка задачи разрабатываемого сервиса	22
1.6 Выводы по разделу	24
2 Сравнение и выбор моделей по подзадачам анализа музыки	25
2.1 Разделение источников.....	25
2.2 Определение долей, темпа и тактового размера.....	27
2.3 Определение тональности.....	29
2.4 Классификация жанра	30
2.5 Распознавание эмоциональной окраски	31
2.6 Распознавание текста и выравнивание во времени.....	33
2.7 Распознавание аккордов.....	35
2.8 Выводы по разделу	36
3 Проектирование сервиса и формализация задачи выравнивания.....	37
3.1 Функциональные и нефункциональные требования.....	37
3.2 Выбор инструментов разработки	38
3.3 Архитектура многомодульного конвейера	39
3.4 Управление ресурсами и профили моделей.....	41
3.5 Формализация задачи выравнивания слогов	42
3.6 Алгоритм квантизации слогов на ритмическую сетку	44
3.7 Регуляризация ритмической сетки.....	48
3.8 Слоговое деление русского текста.....	49
3.9 Выводы по разделу	50
4 Программная реализация сервиса	51

4.1	Общая структура приложения	51
4.2	Реализация модулей анализа	52
4.3	Оркестратор и формат результата	54
4.4	Программный интерфейс и асинхронная обработка	55
4.5	Пользовательский интерфейс	57
4.6	Редактор ритмической сетки слогов	60
4.7	Экспорт, диагностика и развертывание	62
4.8	Выводы по разделу	63
5	Оценка качества моделей и работоспособности сервиса	65
5.1	Методика оценки	65
5.2	Качество моделей по публичным данным	67
5.3	Функциональное тестирование сервиса	71
5.4	Качественная демонстрация слога-долевого выравнивания	73
5.5	Сравнение движков выравнивания: WhisperX и MFA	75
5.6	Известные ограничения	76
5.7	Выводы по разделу	77
	ЗАКЛЮЧЕНИЕ	79
	СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	83
	ПРИЛОЖЕНИЕ А Иллюстрационные материалы	89
	ПРИЛОЖЕНИЕ Б Листинг программы	91
	ПРИЛОЖЕНИЕ В Презентация	97

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей работе применяют следующие сокращения и обозначения

MIR (Music Information Retrieval) – извлечение музыкальной информации
ASR (Automatic Speech Recognition) – автоматическое распознавание речи
VAD (Voice Activity Detection) – детекция речевой активности
ACR (Automatic Chord Recognition) – автоматическое распознавание аккордов
SRA (Syllable-Rhythm Alignment) – слога-долевое выравнивание
BPM (Beats Per Minute) – ударов в минуту (темп)
IBI (Inter-Beat Interval) – межбитовый интервал
STFT (Short-Time Fourier Transform) – кратковременное преобразование Фурье
CQT (Constant-Q Transform) – преобразование с постоянной добротностью
MFCC (Mel-Frequency Cepstral Coefficients) – мел-частотные
кепстральные коэффициенты
HPCP (Harmonic Pitch Class Profile) – профиль гармонических классов высоты
LUFS (Loudness Units relative to Full Scale) – единицы громкости относительно
полной шкалы
CNN (Convolutional Neural Network) – сверточная нейронная сеть
RNN (Recurrent Neural Network) – рекуррентная нейронная сеть
LSTM (Long Short-Term Memory) – сеть с долгой краткосрочной памятью
CRNN (Convolutional Recurrent Neural Network) – сверточно-рекуррентная
нейронная сеть
AST (Audio Spectrogram Transformer) – трансформер для спектрограмм
SSL (Self-Supervised Learning) – обучение без учителя (самообучение)
PANNs (Pretrained Audio Neural Networks) – предобученные аудионейросети
HuBERT (Hidden-unit BERT) – самообучаемая модель представлений
MERT (Self-supervised модель) -основание для анализа музыки
DBN (Dynamic Bayesian Network) – динамическая байесовская сеть
MFA (Montreal Forced Aligner) – система форсированного выравнивания
BTC (Bi-directional Transformer for Chord recognition) – двунаправленный
трансформер для распознавания аккордов

SDR (Signal-to-Distortion Ratio) – отношение сигнал/искажение

WER (Word Error Rate) – доля ошибочных слов

CER (Character Error Rate) – доля ошибочных символов

WCSR (Weighted Chord Symbol Recall) – взвешенное совпадение аккордовых меток

RMSE (Root Mean Square Error) – среднеквадратичная ошибка

mAP (mean Average Precision) – средняя точность

F1 (F-мера) – гармоническое среднее точности и полноты

API (Application Programming Interface) – программный интерфейс приложения

REST (Representational State Transfer) – архитектурный стиль программного интерфейса

SSE (Server-Sent Events) – серверные события (поток обновлений)

HTTP (HyperText Transfer Protocol) – протокол передачи гипертекста

JSON (JavaScript Object Notation) – текстовый формат обмена данными

LRC - Формат караоке-субтитров с тайм-кодами

GPU (Graphics Processing Unit) – графический процессор (видеокарта)

VRAM (Video Random Access Memory) – видеопамять

CUDA (Compute Unified Device Architecture) – программно-аппаратная платформа параллельных вычислений

ВВЕДЕНИЕ

Актуальность темы исследования. За последние два десятилетия объем доступной музыки вырос на порядки: стриминговые сервисы (Spotify, Apple Music, «Яндекс Музыка», VK Музыка и другие) предоставляют доступ к десяткам миллионов композиций, а каждый день к ним добавляются десятки тысяч новых записей. Такой масштаб делает ручной музыковедческий разбор каждого произведения невозможным и порождает устойчивый спрос на автоматический анализ музыки – извлечение из звукозаписи содержательных характеристик (темпа, тональности, тактового размера, гармонии, жанра, эмоциональной окраски, вокального текста и его положения относительно ритма). Соответствующая научно-техническая дисциплина – извлечение музыкальной информации (Music Information Retrieval, MIR) – сегодня является активно развивающейся областью на стыке цифровой обработки сигналов, машинного обучения и музыковедения [1, 2].

Развитие методов глубокого обучения качественно изменило MIR. Если еще десять лет назад анализ музыки опирался преимущественно на «ручные» акустические признаки и узкоспециализированные алгоритмы, то появление предобученных нейросетевых моделей и моделей-оснований (foundation models), обучаемых без учителя на больших корпусах, – таких как wav2vec 2.0 для речи и MERT для музыки [3, 4], – сделало высококачественный анализ доступным без обучения собственных моделей с нуля. Сегодня существуют открытые модели промышленного уровня для разделения источников, трекинга долей, распознавания тональности, жанра, эмоций, аккордов и текста, что открывает возможность собрать из них единый сервис комплексного анализа песни.

Вместе с тем существующие сервисы решают эти задачи разрозненно. Одни специализируются на разделении фонограммы на дорожки, другие – на распознавании аккордов, третьи – на синхронизации текста (караоке). При этом для прикладной задачи разбора песни – когда музыканту, педагогу или ученику нужно одновременно видеть тональность, темп, тактовый размер,

последовательность аккордов по долям такта и, что особенно важно, раскладку слогов вокального текста по ритмической сетке – целостного инструмента практически нет. Именно сочетание «аккорды по долям + слоги по ритму такта» наиболее ценно при разучивании и аранжировке произведения, однако ни один из распространенных сервисов его не предоставляет. Это определяет нишу и прикладную актуальность настоящей работы. Дополнительной иллюстрацией практической ценности служит личный опыт автора как гитариста-любителя: при разборе песен «на слух» именно сопоставление слогов с долями такта и аккордами требует наибольших усилий и хуже всего поддерживается доступными средствами.

Таким образом, актуальность работы обусловлена, с одной стороны, зрелостью нейросетевых методов MIR, позволяющих построить качественный многозадачный анализатор музыки, а с другой – отсутствием доступного инструмента, объединяющего музыкально-теоретический анализ песни с наглядным распределением слогов и аккордов по ритмической сетке такта.

Объект исследования – методы и программные средства интеллектуального (нейросетевого) анализа музыкальных произведений.

Предмет исследования – многомодульный конвейер анализа песни на основе предобученных моделей и метод выравнивания слогов вокального текста на ритмическую сетку такта.

Цель работы – исследовать современные подходы прикладного искусственного интеллекта к анализу музыкальных произведений и на основе обоснованно выбранных решений разработать сервис комплексного анализа песни, определяющий тональность, темп, тактовый размер, ритмическую сетку долей, жанр, эмоциональную окраску, последовательность аккордов по долям и раскладку слогов вокального текста по ритмической сетке.

Для достижения поставленной цели решаются следующие задачи:

- провести анализ предметной области MIR: рассмотреть задачи анализа музыки, способы представления аудиосигнала и архитектуры

нейронных сетей, применяемые для их решения, а также существующие сервисы-аналоги,

- для каждой подзадачи сервиса выполнить сравнительный анализ доступных моделей и методов и обоснованно выбрать решение с учетом качества, вычислительной стоимости, лицензионной чистоты и поддержки русского языка,

- формализовать задачу выравнивания слогов с долями такта (Syllable-Rhythm Alignment, SRA), предложить метрику оценки ее решения и детерминированный алгоритм решения,

- спроектировать архитектуру многомодульного конвейера анализа и сервиса в целом с учетом ограничения вычислительных ресурсов одной видеокарты,

- реализовать сервис: серверную часть с модулями анализа и оркестратором, программный интерфейс, пользовательский интерфейс с редактором ритмической сетки и контейнерное развертывание,

- провести экспериментальную оценку качества выбранных моделей по публичным данным и апробацию работоспособности сервиса на наборе реальных композиций.

Методы исследования. В работе использованы методы анализа научной литературы и сравнительного анализа моделей по опубликованным результатам на общепринятых наборах данных, методы цифровой обработки звуковых сигналов (спектральный анализ, хроматические представления) и глубокого обучения, принципы объектно-ориентированного и модульного проектирования программных систем, методы экспериментального тестирования программного обеспечения. Для постановки задачи выравнивания слогов использованы средства дискретной математики, а для ее решения – детерминированные алгоритмы округления времени слогов к ритмической сетке и разрешения их наложений.

Научная новизна работы заключается в следующем:

- предложена формализация задачи слога-долевого выравнивания (SRA): для каждого спетого слога определяются его время, доля такта, позиция внутри доли и признак сильной доли, а качество решения оценивается введенной метрикой SRA-Accuracy(τ),
- разработан детерминированный и объяснимый метод раскладки слогов по ритмической сетке (с точностью до шестнадцатой доли): с автоматическим выбором дробности округления, разведением слогов, попавших в одну клетку, по соседним клеткам с наименьшим суммарным сдвигом и исправлением ритмической сетки, что обеспечивает корректную работу в том числе на составном размере 6/8, а также детерминированной привязкой «аккорд – доля – слог»,
- предложена архитектура многомодульного MIR-конвейера, рассчитанная на бюджет одной видеокарты, с ленивой загрузкой и выгрузкой моделей, честной деградацией при сбоях отдельных модулей и сменными движками выравнивания текста (речевое форсированное выравнивание WhisperX с защитным согласованием с альтернативным движком Montreal Forced Aligner).

Теоретическая значимость работы состоит в формализации задачи слога-долевого выравнивания и в предложении метрики ее оценки с описанными свойствами, что создает основу для последующих количественных исследований. Практическая значимость определяется тем, что разработан работающий сервис комплексного анализа песни, пригодный для музыкантов, педагогов и учеников музыкальных школ при разучивании и разборе произведений, а реализованный редактор ритмической сетки может использоваться как инструмент ручной коррекции и подготовки эталонной разметки для дальнейших исследований.

Апробация результатов. Основные результаты работы отражены в обзорной статье автора «Современные методы представления звука и архитектуры нейронных сетей для интеллектуального анализа музыкальных произведений», опубликованной в журнале «Вестник науки» (2025) [5].

Структура и объем работы. Диссертация состоит из введения, пяти разделов, заключения, списка использованных источников и приложений. В первой разделе проводится анализ предметной области MIR: рассматриваются задачи анализа музыки, способы представления аудиосигнала, архитектуры нейронных сетей и существующие сервисы-аналоги, формулируется решаемая задача и ее отличие от смежных. Во второй разделе для каждой подзадачи сервиса выполняется сравнительный анализ моделей и обосновывается выбор. В третьей разделе проектируется архитектура многомодульного конвейера и формализуется задача слога-долевого выравнивания: вводятся определения, метрика SRA-Accuracy(τ) и ее свойства, описываются детерминированный алгоритм квантизации, регуляризация сетки и слогаделение. В четвертой разделе описывается программная реализация сервиса: структура приложения, модули анализа, оркестратор и формат результата, программный и пользовательский интерфейсы, редактор ритмической сетки, экспорт и развертывание. В пятой разделе приводится экспериментальная оценка: качество моделей по публичным бенчмаркам, функциональное тестирование сервиса на наборе композиций, качественная демонстрация слога-долевого выравнивания, сравнение движков выравнивания и анализ ограничений. В заключении сформулированы основные результаты, научная новизна и направления дальнейшей работы.

1 Обзор предметной области интеллектуального анализа музыки

Интеллектуальный анализ музыкальных произведений сформировался на стыке музыковедения, цифровой обработки сигналов, машинного обучения и анализа данных. Если традиционный музыкальный анализ опирается на слух и знания эксперта и требует значительных затрат времени, то развитие вычислительных методов позволило автоматизировать извлечение из звукозаписи широкого набора характеристик – от темпа и тональности до жанра, эмоциональной окраски и расположения слов вокальной партии относительно ритмической сетки [1, 5]. Самостоятельная научно-техническая дисциплина, занимающаяся такими задачами, в англоязычной литературе носит название Music Information Retrieval (MIR) – извлечение музыкальной информации [2].

Цель настоящего раздела – рассмотреть предметную область: систематизировать задачи MIR, способы представления аудиосигнала и применяемые архитектуры нейронных сетей, проанализировать существующие программные средства и сервисы, выявить незанятую нишу и сформулировать перечень задач, решаемых разрабатываемым сервисом. Раздел опирается в том числе на опубликованный автором обзор предметной области [5].

1.1 Задачи интеллектуального анализа музыкальных произведений

Музыкальное произведение представляет собой сложный многомерный объект, отдельные стороны которого поддаются формальному анализу. В музыковедении принято выделять следующие его элементы [6]:

- мелодия – последовательность звуков различной высоты и длительности, образующая узнаваемую музыкальную мысль,
- гармония – совокупность одновременно звучащих тонов (аккордов) и закономерности их следования,
- ритм и метр – временная организация звуков: чередование длительностей, доли, такты и тактовый размер,

- тембр – окраска звука, определяемая спектральным составом и позволяющая различать инструменты и голоса,
- жанр – стилевая категория, обобщающая характерные сочетания вышеперечисленных элементов,
- эмоциональная окраска (настроение) – субъективно воспринимаемое аффективное содержание музыки.

Каждый из этих элементов порождает соответствующую задачу автоматического анализа. По уровню извлекаемой информации задачи MIR удобно разделить на несколько групп.

Низкоуровневые задачи связаны с преобразованием сигнала и извлечением акустических признаков (спектральных, кепстральных, хроматических), они служат входом для всех последующих этапов и подробно рассмотрены в подразделе 1.2.

Среднеуровневые задачи извлекают музыкально интерпретируемые характеристики, непосредственно связанные с нотным текстом: определение темпа и трекинг долей (beat tracking), нахождение сильных долей и тактового размера, оценку тональности и лада, распознавание аккордов (Automatic Chord Recognition), а также разделение фонограммы на отдельные источники – вокал и аккомпанемент (music source separation).

Высокоуровневые (семантические) задачи оперируют целостным восприятием произведения: классификация жанра, распознавание эмоциональной окраски, автоматическое тегирование, анализ структуры (выделение куплета, припева и т. д.).

Задачи на стыке аудио и текста занимают особое положение: к ним относятся распознавание вокального текста (Automatic Speech Recognition, ASR), его выравнивание во времени (forced alignment), а также транскрипция и синхронизация лирики и нот. Именно к этой группе примыкает центральная задача настоящей работы – выравнивание слогов вокального текста на ритмическую сетку, подробно формализуемая в разделе 3.

Практическая значимость интеллектуального анализа музыки определяется широтой его применений [1, 5, 7]: автоматизация музыковедческих исследований и каталогизация фонотек, рекомендательные системы стриминговых сервисов, подбирающие композиции по акустическому сходству, генерация и аранжировка музыки с помощью искусственного интеллекта [7], образовательные приложения, помогающие разучивать произведения. Результаты исследований в этой области публикуются в первую очередь на профильной конференции ISMIR (International Society for Music Information Retrieval), а также в журналах IEEE/ACM Transactions on Audio, Speech and Language Processing и на конференции ICASSP [2].

Таким образом, предметная область объединяет разнородные задачи – от обработки сигнала до семантической классификации и синхронизации текста с музыкой. Разрабатываемый сервис носит комплексный характер и затрагивает все перечисленные группы, что определяет необходимость рассмотреть как способы представления звука (подраздел 1.2), так и нейросетевые архитектуры, применяемые для отдельных задач (подраздел 1.3).

1.2 Представления аудиосигнала

Цифровая звукозапись представляет собой дискретизированный во времени сигнал – последовательность отсчетов амплитуды, полученных с частотой дискретизации f_s . Согласно теореме Котельникова, для безыскаженного представления сигнала с максимальной частотой f_{max} необходимо $f_s \geq 2f_{max}$, для музыки, верхняя граница слышимого диапазона которой близка к 20 кГц, стандартными являются частоты 44 100 и 48 000 Гц [5, 8]. Непосредственно временное представление (осциллограмма) несет полную информацию о сигнале, но малоприспособно для анализа музыкального содержания, поскольку не выявляет частотную структуру. Поэтому ключевую роль играют частотно-временные и производные представления.

Спектрограмма на основе кратковременного преобразования Фурье (STFT) получается разбиением сигнала на перекрывающиеся окна и применением к каждому окну дискретного преобразования Фурье. Результат – матрица «время $\times$ частота», отражающая изменение спектра во времени. STFT-спектрограмма содержит наиболее полную частотную информацию, однако обладает фиксированным линейным частотным разрешением и высокой размерностью, что не всегда удобно для машинного обучения [5, 8].

Мел-спектрограмма строится пропусканием спектра мощности через банк треугольных фильтров, расположенных по мел-шкале, которая аппроксимирует нелинейное восприятие высоты звука слуховой системой человека. Типовая конфигурация включает 128 мел-фильтров в диапазоне 0–22 050 Гц при $f_s = 44\,100$ Гц [5]. Мел-спектрограмма компактнее STFT-спектрограммы, согласована с особенностями слуха и является де-факто стандартным входом для сверточных и трансформерных моделей анализа музыки [5, 8].

Мел-частотные кепстральные коэффициенты (MFCC) получаются применением дискретного косинусного преобразования к логарифму мел-спектра. Обычно используются 13–20 коэффициентов, компактно описывающих огибающую спектра и, следовательно, тембр. MFCC хорошо зарекомендовали себя в распознавании речи и тембровом анализе, но теряют информацию о точной высоте звуков и гармонии [5, 8].

Хромаграмма сворачивает спектр в 12-мерный вектор, соответствующий двенадцати ступеням равномерно темперированного строя, суммируя энергию всех октав каждой ступени. Благодаря инвариантности к октаве хромаграмма отражает гармоническое содержание и служит основным признаком для определения тональности и распознавания аккордов [8].

Постоянное-Q преобразование (CQT) обеспечивает логарифмическое распределение частотных бинов с постоянной добротностью, при котором каждый бин соответствует определенной ноте. Такое представление

естественно для музыки и применяется, в частности, как вход в моделях распознавания аккордов и оценки высоты тона [8].

Сравнительная характеристика рассмотренных представлений приведена в таблице 1.

Примеры представления аудио представлены на рисунке 1.

Таблица 1 – Способы представления аудиосигнала

Представление	Что отражает	Преимущества	Ограничения	Типовое применение
Осциллограмма (PCM)	Амплитуда во времени	Полнота информации	Нет явной частотной структуры	Сегментация, end-to-end модели
STFT-спектрограмма	Спектр во времени	Полная частотная информация	Высокая размерность, линейная шкала	Разделение источников, общий анализ
Мел-спектрограмма	Спектр по мел-шкале (128 полос)	Компактность, согласие со слухом	Потеря тонкой структуры	Вход CNN/Transformer (стандарт)
MFCC (13–20)	Огибающая спектра (тембр)	Очень компактны	Потеря гармонии и высоты	Распознавание речи, тембр
Хромограмма (12)	Распределение по ступеням	Инвариантность к октаве	Не различает тембр	Тональность, аккорды
CQT	Лог-частотный спектр (по нотам)	Естественность для музыки	Выше вычислительная стоимость	Аккорды, оценка высоты тона

В итоге выбор представления определяется задачей: для гармонических задач (тональность, аккорды) используются хромограмма и CQT, для тембровых и классификационных – мел-спектрограмма и MFCC, для разделения источников – полная STFT-спектрограмма. Мел-спектрограмма выступает универсальным входом для большинства современных нейросетевых моделей, именно на ней строится основной массив рассматриваемых далее архитектур.

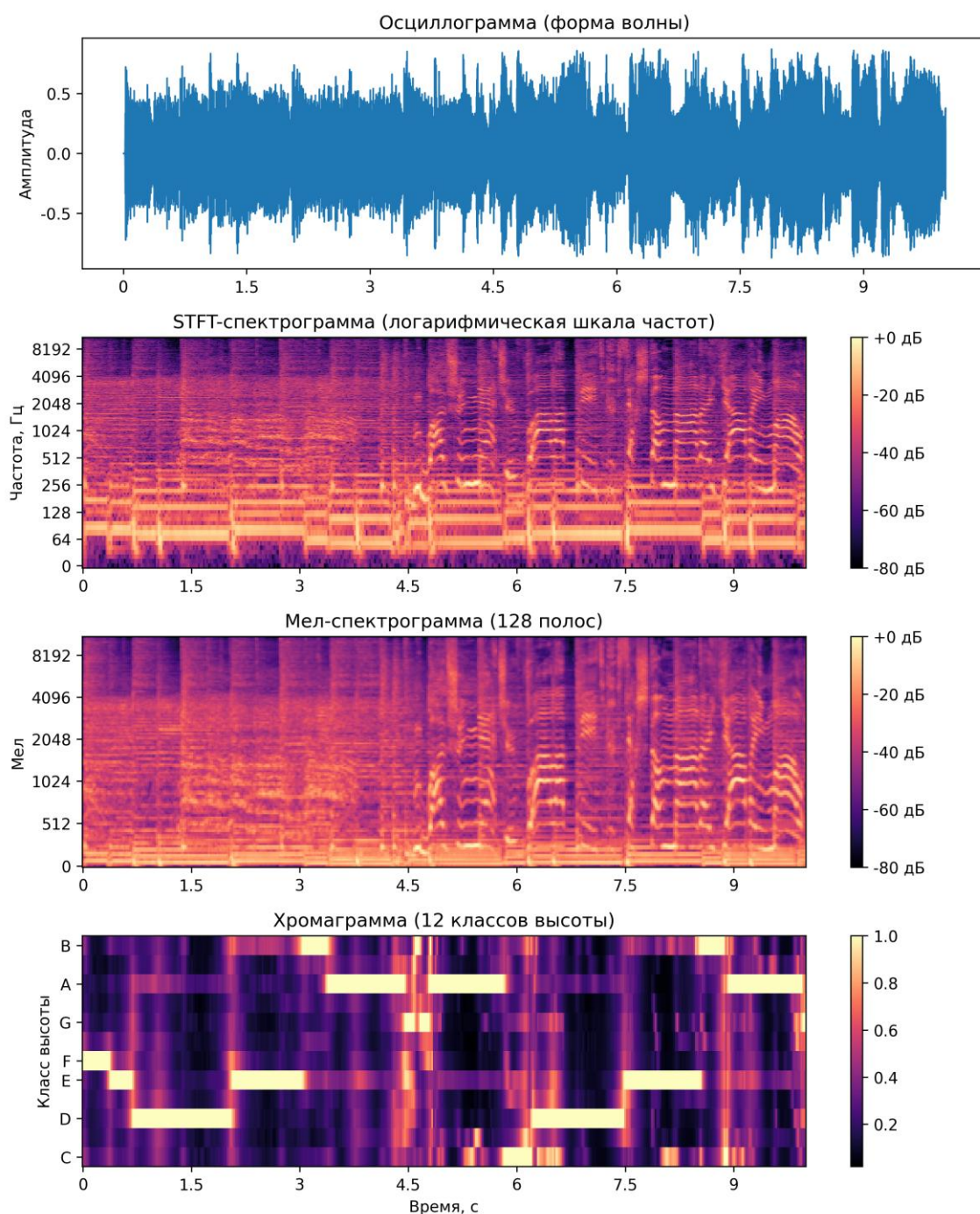


Рисунок 1 – Примеры представлений одного фрагмента: осциллограмма, STFT-спектрограмма, мел-спектрограмма и хромограмма

1.3 Архитектуры нейронных сетей для анализа музыки

Развитие методов интеллектуального анализа музыки прошло несколько этапов [5, 9]. На раннем этапе (приблизительно до 2014 г.) преобладал подход «ручные признаки + классический алгоритм машинного обучения»: из сигнала извлекались признаки (MFCC, хромограмма, спектральный центроид и др.), которые подавались на вход методам опорных векторов, случайному лесу или

гауссовым смесям. Затем доминирующим стало глубокое обучение, последовательно прошедшее через сверточные, рекуррентные и гибридные модели, а с 2020 г. – через трансформеры и модели, обученные без учителя (self-supervised) на больших корпусах [5].

Сверточные нейронные сети (CNN) обрабатывают спектрограмму как изображение, выявляя локальные частотно-временные паттерны. Характерным представителем является семейство PANNs (Pretrained Audio Neural Networks) [10], в частности модель CNN14, содержащая 14 сверточных слоев и предобученная на крупном корпусе AudioSet [11] (около двух миллионов размеченных десятисекундных фрагментов, 527 классов аудиособытий). Предобучение с последующим дообучением под конкретную задачу (transfer learning) позволяет достигать высокого качества классификации даже на небольших наборах данных [5, 10].

Рекуррентные сети (RNN, LSTM, GRU) моделируют временные зависимости и применяются там, где важна последовательная динамика, – например, при оценке темпа и трекинге долей. Сверточно-рекуррентные сети (CRNN) объединяют сверточный экстрактор признаков с рекуррентным слоем и долгое время являлись сильным решением задач автоматического тегирования музыки [12].

Трансформеры, основанные на механизме внимания [13], вытеснили рекуррентные архитектуры в большинстве задач благодаря способности моделировать дальние зависимости и эффективному распараллеливанию. Модель Audio Spectrogram Transformer (AST) первой применила полностью основанную на внимании, свободную от сверток архитектуру к классификации звука: спектрограмма разбивается на перекрывающиеся участки 16×16 , которые линейно проецируются и обрабатываются трансформером-энкодером [14].

Модели, обученные без учителя (self-supervised, foundation models), представляют современный этап. Они обучаются на больших неразмеченных корпусах решать вспомогательную задачу (например, восстановление

замаскированных фрагментов), формируя универсальные представления, которые затем используются для множества прикладных задач. В обработке речи такими моделями стали wav2vec 2.0 [3] и HuBERT [15], для музыки разработана модель MERT, построенная по принципу HuBERT и обученная предсказывать акустические и музыкальные «токены», в конфигурациях с 95 и 330 млн параметров она показала результаты уровня лучших по широкому набору задач MIR [4]. Теоретические основы перечисленных архитектур изложены в [9].

Характерные результаты применения различных архитектур по данным литературы сведены в таблице 2. Приведенные значения отражают типовые показатели, опубликованные авторами соответствующих моделей и в обзоре [5], и служат ориентиром, а не результатом собственных измерений.

Эволюция архитектур анализа музыки представлена на рисунке 2.

Таблица 2 – Архитектуры нейронных сетей и характерные результаты по литературным данным

Архитектура	Типовая задача	Характерный результат (по литературе)	Источник
CNN (CNN14 / PANNs)	Классификация тегов и жанра	mAP 0,431 на AudioSet, после дообучения высокая точность на жанровых наборах	[10]
CRNN	Автоматическое тегирование музыки	Конкурентное качество автоматического тегирования музыки	[12]
AST (Transformer)	Классификация звука	mAP 0,485 на AudioSet, 95,6 % на ESC-50, 98,1 % на Speech Commands V2	[14]
wav2vec 2.0 (SSL)	Распознавание речи	WER 1,8/3,3 % на LibriSpeech (clean/other)	[3]
MERT (SSL, музыка)	Универсальные задачи MIR	Сильные результаты на 14 задачах MIR	[4]

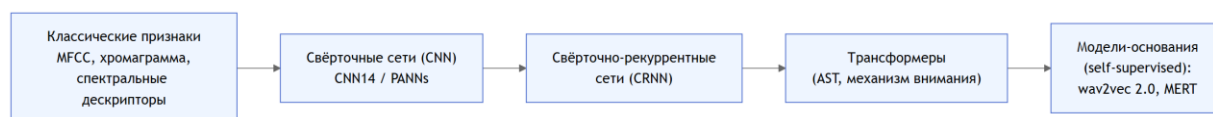


Рисунок 2 – Эволюция архитектур анализа музыки: классические признаки CNN – CRNN –Transformer – self-supervised foundation models

Таким образом, современная парадигма анализа музыки опирается на трансформерные и self-supervised-модели, предобученные на больших корпусах, с последующим дообучением или использованием их

представлений в прикладных задачах. Это означает, что разработчику прикладного сервиса, как правило, целесообразно не обучать модели с нуля, а применять готовые предобученные решения, обоснованию выбора конкретных моделей под каждую подзадачу посвящен раздел 2.

1.4 Обзор существующих программных средств и сервисов

К настоящему времени сформировался рынок программных средств анализа музыки – от узкоспециализированных сервисов до открытых библиотек. Рассмотрим основные категории.

Идентификация композиций. Сервисы Shazam и SoundHound распознают звучащую запись по акустическому «отпечатку» (audio fingerprint), сопоставляя ее с базой данных [16, 17]. Эта задача решена на высоком уровне, но не относится к содержательному анализу произведения.

Извлечение характеристик трека. Платформа Echo Nest (вошедшая в состав Spotify) предоставляла программный интерфейс с характеристиками композиции – темпом, тональностью, «энергичностью», «настроением» и др. [18]. Подобные признаки используются стриминговыми сервисами в рекомендательных системах, однако они закрыты, не сопровождаются привязкой к нотному тексту и не позволяют пользователю детально разобрать произведение.

Разделение источников. Коммерческие сервисы Moises.ai, LALAL.AI и AudioShake выделяют из фонограммы отдельные дорожки (вокал, ударные, бас и др.) и ориентированы преимущественно на музыкантов и звукорежиссеров. Качество разделения у современных сервисов высоко, однако они решают лишь одну подзадачу.

Распознавание аккордов и работа с табулатурами. Сервис Chordify автоматически определяет аккорды из аудио или видео и отображает их на временной шкале. Ресурсы Songsterr и Ultimate Guitar предоставляют табулатуры и аккорды, формируемые преимущественно сообществом. Эти средства близки к задачам настоящей работы, но не связывают аккорды и текст с метрической сеткой произведения на уровне долей.

Синхронизированный текст (караоке). Стриминговые сервисы Spotify и Apple Music отображают текст песни, синхронизированный по строкам или словам (как правило, на основе данных сторонних поставщиков). Синхронизация выполняется на уровне слова или строки и не учитывает положение слога относительно доли такта.

Обучающие приложения (Yousician, Simply Piano и аналогичные) помогают разучивать инструмент, оценивая исполнение в реальном времени, но не предоставляют развернутого аналитического разбора произвольной композиции.

Открытые библиотеки. Для исследователей и разработчиков доступны библиотеки общего назначения – librosa (извлечение признаков), madmom (ритмический анализ), Essentia (комплексный анализ, включая тональность, жанр и настроение) [19]. Они предоставляют отдельные алгоритмы, но не образуют законченного пользовательского сервиса.

Сопоставление возможностей рассмотренных средств приведено в таблице 3.

Таблица 3 – Сравнение существующих средств анализа музыки

Средство / сервис	Основная функция	Тональность	Темп / размер	Аккорды	Текст по времени	Слог на доле такта
Shazam, SoundHound	Идентификация трека	–	–	–	–	–
Spotify / Echo Nest	Характеристики трека	+	+	–	–	–
Moises.ai,	Разделение источников	частично	частично	в платной версии	–	–
LALAL.AI, AudioShake	Разделение источников					
Chordify	Аккорды из аудио	+	частично	+	–	–
Songsterr, Ultimate Guitar	Табулатуры, аккорды	+	+	+	–	–

Продолжение таблицы 3

Средство / сервис	Основная функция	Тональность	Темп / размер	Аккорды	Текст по времени	Слог на доле такта
Spotify / Apple Music (тексты)	Синхронизация текста	–	–	–	по словам/строкам	–
librosa, madmom, Essentia	Библиотеки алгоритмов	+	+	частично	частично	–
Разрабатываемый сервис	Комплексный разбор песни	+	+	+	по словам и слогам	+

Анализ таблицы 3 показывает, что отдельные подзадачи (разделение, аккорды, синхронизация текста) уже решаются существующими средствами, однако ни одно из них не предоставляет совмещенного разбора, в котором слоги вокального текста размещены на ритмической сетке по долям такта, а аккорды сопоставлены с конкретными долями. Именно эта возможность наиболее востребована при самостоятельном разучивании песни – например, гитаристом или учеником музыкальной школы – и образует незанятую нишу.

Таким образом, существующие сервисы либо закрыты и решают отдельные подзадачи, либо ограничиваются пословной синхронизацией текста. Совмещение комплексного анализа (тональность, темп, размер, жанр, эмоция, аккорды) с распределением слогов по долям такта в литературе и среди доступных продуктов не представлено, что обосновывает разработку собственного сервиса.

1.5 Постановка задачи разрабатываемого сервиса

На основании проведенного обзора определен перечень задач, решаемых разрабатываемым сервисом. По загруженной аудиозаписи песни сервис должен определять: тональность (тонику и лад), темп (BPM), тактовый размер, ритмическую сетку долей и тактов, жанр, эмоциональную окраску, аккорды с привязкой к долям такта, вокальный текст с распределением слогов по долям ритмической сетки. Перечисленные характеристики могут вычисляться выборочно по запросу пользователя.

Центральной из этих задач является выравнивание слогов вокального текста на ритмическую сетку (далее – задача SRA, Syllable-Rhythm Alignment). Ее содержание состоит в том, чтобы для каждого спетого слога определить, на какую долю такта и в какую ее ритмическую позицию (например, «раз», «и», «два» и т. п.) он приходится. Несмотря на близость к ряду известных задач, SRA не сводится ни к одной из них (таблица 4):

- форсированное выравнивание текста (forced alignment) [20, 21] определяет временные границы слов и фонем, но не соотносит их с метрической сеткой,
- трекинг долей (beat tracking) [22] находит положение долей и сильных долей, но не работает с текстом,
- синхронизация караоке привязывает текст к времени на уровне строки или слова, без учета доли,
- синтез певческого голоса (Singing Voice Synthesis) и транскрипция пения [23] оперируют нотами и фонемами, решая обратную или смежную задачу.

Таблица 4 – Сопоставление задачи SRA со смежными задачами

Задача	Вход	Выход	Привязка к долям
Форсированное выравнивание	Аудио + текст	Границы слов/фонем во времени	Нет
Трекинг долей	Аудио	Доли, сильные доли	– (нет текста)
Синхронизация караоке	Аудио + текст	Текст по строкам/словам во времени	Нет
Транскрипция пения / SVS	Аудио / ноты	Ноты, фонемы	Частично (к нотам)
SRA (настоящая работа)	Аудио + текст	Слог – (время, доля, позиция в доле, сильная доля)	Да

Формальная постановка задачи SRA, метрика ее качества и метод решения приводятся в разделе 3.

Работа развивает тему, которой автор последовательно занимался в ходе обучения в магистратуре, – от обзора предметной области и отдельных подзадач к обоснованному выбору моделей, проектированию и реализации

комплексного сервиса, результаты обзора методов представления звука и архитектур нейронных сетей опубликованы автором в статье [5].

Таким образом, определен перечень решаемых сервисом задач и выделена центральная задача SRA, не покрываемая существующими подходами.

1.6 Выводы по разделу

Интеллектуальный анализ музыкальных произведений – самостоятельная область (MIR) на стыке музыковедения, обработки сигналов и машинного обучения, ее задачи разделяются на низко-, средне- и высокоуровневые, а также задачи на стыке аудио и текста, к которым относится центральная задача работы.

Базовым входом для современных моделей служит мел-спектрограмма, для гармонических задач применяются хромограмма и CQT, для тембровых – MFCC. Выбор представления определяется характером задачи.

Доминирующей парадигмой является применение предобученных трансформерных и self-supervised-моделей с последующим дообучением, что делает целесообразным использование готовых решений вместо обучения с нуля.

Анализ существующих сервисов показал, что комплексный разбор песни с размещением слогов на ритмической сетке по долям такта и привязкой аккордов к долям не реализован ни в одном доступном продукте, это определяет нишу и практическую ценность разрабатываемого сервиса.

Сформулирован перечень задач сервиса и выделена ключевая задача выравнивания слогов на ритмическую сетку (SRA), формализуемая в разделе 3.

Дальнейший анализ конкретных моделей, пригодных для решения каждой из перечисленных подзадач, и обоснование их выбора проводятся в разделе 2.

2 Сравнение и выбор моделей по подзадачам анализа музыки

В разделе 1 показано, что современный анализ музыки опирается на готовые предобученные модели. Цель настоящего раздела – для каждой подзадачи разрабатываемого сервиса рассмотреть доступные решения и обоснованно выбрать модель. Выбор проводился по совокупности критериев:

- качество по опубликованным результатам на общепринятых наборах данных,
- вычислительная стоимость (время обработки и объем видеопамати) с учетом ограничения «одна видеокарта» – целевой пик потребления видеопамати ниже 12 ГБ,
- наличие открытых весов и лицензионная чистота,
- поддержка русского языка (для распознавания и выравнивания вокального текста),
- интегрируемость в единый программный конвейер (совместимость библиотек, формат входа/выхода).

Поскольку приведенные в литературе метрики получены на разных тестовых наборах и по разным протоколам, в сравнительных таблицах сделан акцент на архитектурных свойствах и пригодности моделей, а конкретные числовые значения приводятся как ориентир с указанием источника.

2.1 Разделение источников

Разделение фонограммы на вокал и аккомпанемент – первый содержательный этап конвейера: на выделенном вокале выполняются детекция речи, распознавание и выравнивание текста, а на инструментале – оценка тональности и трекинг долей. Качество разделения напрямую влияет на все последующие этапы [24].

Рассматривались следующие решения. Spleeter [24] – быстрый открытый разделитель на сверточной U-Net-архитектуре, поддерживающий деление на 2/4/5 дорожек, ценится за скорость, но уступает современным моделям по качеству. Demucs и его развитие Hybrid Transformer Demucs (HTDemucs) [25]

– гибридная архитектура, работающая одновременно во временной и спектральной областях, с трансформерными слоями, значительно превосходит Spleeter. Band-Split RoPE Transformer (BS-RoFormer) [26] разбивает спектр на частотные полосы и применяет иерархический трансформер с поворотным позиционным кодированием (RoPE), по данным авторов, на наборе MUSDB18-HQ [27] достигает среднего по дорожкам SDR 9,76 дБ без дополнительных данных, превосходя HTDemucs ($\approx 7,7$ дБ) примерно на 2 дБ, а при обучении с ~ 500 дополнительными композициями – 11,99 дБ, что соответствует уровню лучших результатов и победе в конкурсе Sound Demixing Challenge 2023 [26, 28]. Mel-Band RoFormer [28] – развитие BS-RoFormer, в котором эмпирическое деление на полосы заменено проекцией по мел-шкале, обеспечивает прирост около 0,5 дБ по вокалу, но менее сбалансирован по остальным дорожкам.

Качество разделения принято измерять отношением сигнал/искажение (Signal-to-Distortion Ratio, SDR), чем выше значение в децибелах, тем чище выделенная дорожка [29]. Сопоставление рассмотренных моделей приведено в таблице 5.

Таблица 5 – Сравнение моделей разделения источников

Модель	Архитектура	Качество (SDR, MUSDB18-HQ)	Скорость	Лицензия	Источник
Spleeter	Сверточная U-Net	Базовое (ниже современных)	Очень высокая	MIT	[24]
HTDemucs	Гибридная (время + спектр) + трансформер	$\sim 7,7$ дБ (среднее)	Высокая	MIT	[25]
BS-RoFormer	Band-split + RoPE-трансформер	9,76 дБ (без доп. данных), 11,99 дБ (с доп. данными)	Средняя	открытые веса	[26]
Mel-Band RoFormer	Mel-band + RoPE-трансформер	$\sim 0,5$ дБ по вокалу к BS-RoFormer	Средняя	открытые веса	[28]

Выбор – BS-RoFormer. Несмотря на то что Mel-Band RoFormer дает несколько лучший вокал, для задач сервиса важна сбалансированность: на инструментале (с минимумом протечек вокала) выполняется трекинг долей и оценка тональности, поэтому предпочтительна модель с высоким качеством по всем дорожкам и лучшим инструментом. BS-RoFormer обеспечивает

такой баланс и показал наилучший результат на конкурсе SDX 2023. В сервисе используется предобученный чекпойнт семейства BS-RoFormer (с заявленным значением SDR по вокалу около 12,98 дБ), подключаемый через библиотеку audio-separator [29].

Таким образом, в качестве разделителя источников выбран BS-RoFormer как наиболее сбалансированная по качеству модель с открытыми весами, Spleeter и HTDemucs рассматриваются как более быстрые, но менее точные альтернативы.

2.2 Определение долей, темпа и тактового размера

Определение положения долей (beats) и сильных долей (downbeats) – основа ритмической сетки, на которую затем проецируются слоги и аккорды. Из положения долей детерминированно вычисляются темп и тактовый размер.

Madmom [30] – классическая библиотека, в которой рекуррентная сеть предсказывает функцию активации долей, а динамическая байесовская сеть (DBN) выполняет постобработку, долгое время являлась эталонным решением, но DBN увеличивает задержку. BeatNet [31] объединяет сверточно-рекуррентную сеть с фильтром частиц и работает в реальном времени. Beat Transformer [32] применяет трансформер с расширенным самовниманием к разделенным на источники спектрограммам. Beat This! [22] – трансформерная модель (чередование сверток и блоков внимания по частоте и времени), обученная на большом числе наборов данных с устойчивой к сдвигу функцией потерь, принципиально отказывается от DBN-постобработки и, по данным авторов, достигает наилучшего значения F-меры среди сравниваемых моделей на разнообразном тестовом наборе, включая GTZAN [22]. Качество трекинга оценивается F-мерой с допуском ± 70 мс с помощью библиотеки mir_eval [33].

Сравнение моделей трекинга долей представлено в таблице 6.

Таблица 6 – Сравнение моделей трекинга долей

Модель	Архитектура	Постобработка (DBN)	Режим	Источник
madmom	RNN + DBN	Требуется	Офлайн	[30]

Продолжение таблицы 6

Модель	Архитектура	Постобработка (DBN)	Режим	Источник
BeatNet	CRNN + фильтр частиц	Не требуется	Реального времени	[31]
Beat Transformer	Трансформер на разделенных спектрограммах	Требуется	Офлайн	[32]
Beat This!	Свертки + трансформер, устойчивая к сдвигу потеря	Не требуется	Офлайн	[22]

Выбор – Beat This! Модель дает наилучшее качество среди открытых решений, не требует DBN-постобработки (что упрощает интеграцию и снижает задержку) и устойчива к небольшим неточностям разметки. Используется официальный чекпойнт `final0` с отключенной DBN-постобработкой [33].

На основе найденных долей $\beta_1, \dots, \beta_M$ детерминированно вычисляются темп и тактовый размер. Базовая величина – межбитовый интервал, то есть расстояние между двумя соседними долями:

$$IBI_i = \beta_{i+1} - \beta_i, \quad (1)$$

где IBI_i – межбитовый интервал между i -й и следующей долями;

β_i – время начала i -й доли;

β_{i+1} – время начала следующей доли.

Через межбитовые интервалы выражаются и темп, и стабильность темпа (2, 3). Темп оценивается по медиане межбитовых интервалов – медиана устойчивее среднего к выбросам:

$$BPM = \frac{60}{\text{median}_i IBI_i} \quad (2)$$

где BPM – оценка темпа, ударов в минуту;

60 – число секунд в минуте;

$\text{median}_i \text{IBI}_i$ – медиана по всем межбитовым интервалам.

Полученное значение приводится в целевой диапазон 60–180 ударов в минуту удвоением или делением пополам – для устранения ошибок «двойного/половинного» темпа (когда трекер оценивает темп вдвое выше или ниже истинного).

Дополнительно оценивается стабильность темпа – насколько ровно выдержан пульс:

$$\text{stab} = 1 - \text{CV}, \quad (3)$$

где stab – стабильность темпа, от 0 до 1 (значение, близкое к 1, соответствует ровному темпу);

CV – коэффициент вариации межбитовых интервалов, то есть отношение их стандартного отклонения к среднему.

Тактовый размер определяется отдельно – как наиболее частое число долей между соседними сильными долями, для составных размеров (6/8 и др.) применяется дополнительная эвристика, учитывающая сырой (до нормализации) темп. Эти детерминированные алгоритмы и их улучшения подробно описаны в разделе 3.

В итоге для трекинга долей выбран Beat This! как наиболее качественная модель без DBN-постобработки, темп и размер выводятся из ее результата детерминированными методами.

2.3 Определение тональности

Тональность (тоника и лад) традиционно определяется сопоставлением усредненной по времени хромограммы с эталонными профилями. Классический метод Крумханслы–Шмуклера сопоставляет усредненную хромограмму с 24 эталонными профилями (12 тоник × мажор/минор, в качестве эталонов обычно берут профили Крумханслы–Кесслера) и выбирает тот, корреляция с которым максимальна [34]. Библиотека Essentia реализует алгоритм KeyExtractor, строящий профиль гармонических классов высоты

(НРСР) и сопоставляющий его с одним из наборов профилей (в том числе профилем Темперли) [19]. Существуют и обучаемые подходы – например, сверточная сеть с направленными фильтрами для совместной оценки темпа и тональности [35].

Таблица 7 – Сравнение методов определения тональности

Метод	Принцип	Зависимости	Примечание	Источник
Krumhansl–Schmuckler (librosa)	Хромограмма + корреляция с профилями	librosa (чистый Python)	Прост, без внешних зависимостей	[34]
Essentia KeyExtractor	НРСР + профиль (Temperley)	Essentia	Устойчивее по ладу	[19]
CNN (Schreiber–Müller)	Обучаемая модель	Обучение/веса	Требуется обучения	[35]

Выбор – Essentia KeyExtractor как основной метод, Krumhansl–Schmuckler (librosa) как резервный. Опыт реализации показал, что определение лада по хромограмме `chroma_cqt` в `librosa` из-за вклада гармоник систематически смещается в сторону мажора, тогда как алгоритм Essentia на основе НРСР устойчиво определяет лад. Поэтому основным выбран Essentia, а реализованный на чистом NumPy/librosa алгоритм Крумханслы–Шмуклера используется как независимый от внешних библиотек резервный путь.

Таким образом, тональность определяется методом Essentia KeyExtractor с резервом на основе профилей Крумханслы–Кесслера – комбинация обеспечивает устойчивость и отказоустойчивость без обучения собственной модели.

2.4 Классификация жанра

Жанровая классификация – одна из наиболее изученных задач MIR, стандартным учебным набором служит GTZAN (1000 фрагментов, 10 жанров) [36], а более крупным и современным – FMA [37].

Рассматривались два пути. Первый – обучение собственной модели методом переноса обучения: сверточная сеть (например, CNN14 из семейства PANNs [10] или остаточная сеть ResNet [38]), предобученная на большом аудиокорпусе, дообучается на жанровом наборе, современной альтернативой такому обучению служат модели-основания – трансформер AST [14] и self-

supervised-модель MERT [4], представления которой пригодны для жанровой классификации с помощью линейного классификатора. Этот путь требует размеченных данных и обучения. Второй путь – использование готовой промышленной модели. Библиотека Essentia предоставляет классификатор genre_discogs400, построенный на эмбедингах discogs-effnet и обученный на таксономии Discogs (400 поджанров) [19].

Подходы к классификации жанра представлены в таблице 8.

Таблица 8 – Подходы к классификации жанра

Подход	Представитель	Готовность	Лицензия	Примечание	Источник
Собственная CNN + перенос обучения	CNN14 / ResNet18	Требуется обучения	–	Нужны размеченные данные и обучение	[10, 38]
Foundation-модель + классификатор	AST, MERT	Требуется «голова»	CC BY-NC (веса MERT)	Высокое качество, ресурсоемко	[4, 14]
Готовая промышленная модель	Essentia genre_discogs400	Готова к применению	CC BY-NC-SA	400 поджанров, промышленный уровень	[19]

Выбор – Essentia genre_discogs400. Для сервиса предпочтительна готовая модель промышленного уровня, не требующая собственного обучения и охватывающая широкий перечень стилей. Поскольку модель является многоклассовой с независимыми выходами (сигмоида), ее 400 поджанров сворачиваются в 13 укрупненных групп, внутри группы берется максимум вероятностей (детали реализации – в разделе 4).

Таким образом, жанр определяется готовой моделью Essentia genre_discogs400, обучение собственной сверточной сети рассмотрено как альтернатива, но для итогового сервиса выбран более практичный и воспроизводимый путь.

2.5 Распознавание эмоциональной окраски

Эмоциональную окраску музыки принято описывать в двумерной модели Рассела координатами «валентность» (приятность) и «активация» (энергичность), по знакам этих координат выделяют четыре квадранта [39]. Для обучения и оценки таких моделей существуют размеченные наборы –

DEAM (1802 фрагмента с непрерывными оценками валентности и активации) [40], а также РМЕто и подмножество MTG-Jamendo с тегами настроения.

Двумерная модель Рассела и положение пяти проанализированных треков (подраздел 5.3) в ее координатах показаны на рисунке 3.

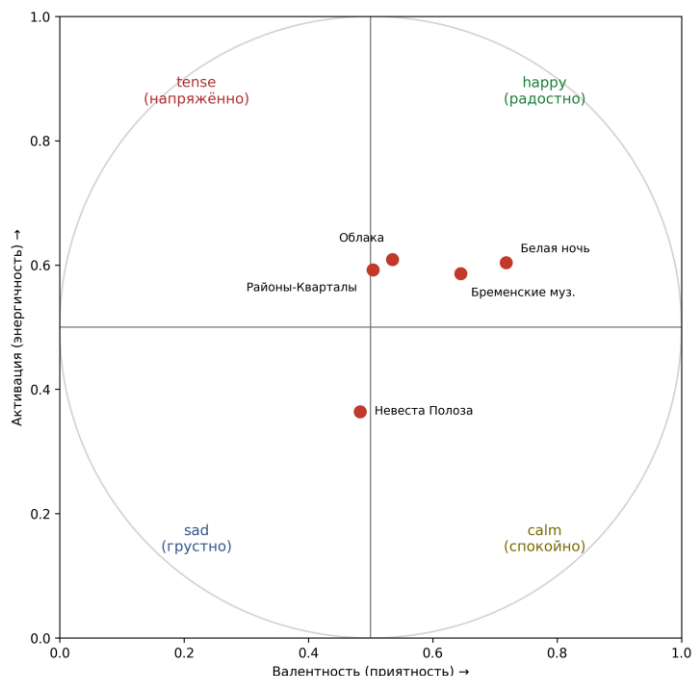


Рисунок 3 – Двумерная модель эмоций Рассела (валентность × активация) с квадрантами happy/tense/sad/calm и точками тестовых треков

Рассматривались следующие решения. Аудио-модель Essentia на основе эмбедингов musicnn [41] включает регрессор emomusic, предсказывающий валентность и активацию, и набор бинарных классификаторов настроения (mood_*) [19]. Модели-основания (MERT [4]) позволяют решать задачу регрессией валентности/активации поверх извлеченных представлений. Наконец, мультимодальный подход, объединяющий аудио- и текстовые признаки (например, эмбединги MERT и языковой модели RuBERT) с механизмом внимания, потенциально точнее для песен с выраженным текстом, этот подход рассматривается как перспективное направление и в настоящей работе не реализован (см. заключение).

Подходы к распознаванию эмоций представлены в таблице 9.

Таблица 9 – Подходы к распознаванию эмоций

Подход	Вход	Готовность	Примечание	Источник
Essentia musicnn + emomusic	Аудио	Готов к применению	Валентность/активация + теги настроения	[19, 41]
MERT + регрессия	Аудио (эмбединги)	Требуется «головы»	Высокое качество представлений	[4]
Мультимодальный (MERT + RuBERT)	Аудио + текст	Не реализован	Перспектива, требует данных	[4]

Выбор – аудио-модель Essentia (emomusic + классификаторы настроения). Решение готово к применению, не требует обучения и согласуется с уже используемым в сервисе стекком Essentia. Выход регрессора (по шкале 1–9) линейно нормируется в диапазон $[0, 1]$ по правилу $x' = (x - 1)/8$, после чего по знакам нормированных валентности и активации относительно 0,5 определяется квадрант (детали – в разделе 4). Мультимодальное распознавание отнесено к направлениям дальнейшей работы.

Итак, эмоциональная окраска определяется аудио-моделью Essentia (валентность/активация по модели Рассела и теги настроения), более сложный мультимодальный подход обозначен как перспектива.

2.6 Распознавание текста и выравнивание во времени

Для размещения слогов на ритмической сетке необходимы две вещи: текст вокальной партии и точные временные метки его элементов. Соответствующие методы делятся на два класса.

Речевое форсированное выравнивание. Модель Whisper [42] обеспечивает устойчивое распознавание речи, но выдает временные метки лишь на уровне сегментов (с точностью порядка сотен миллисекунд). Надстройка WhisperX [21] добавляет пословное и посимвольное выравнивание за счет форсированного выравнивания моделью wav2vec 2.0 [3] (для русского языка применяется соответствующий дообученный чекпойнт). Альтернативой является Montreal Forced Aligner (MFA) [20] – выравниватель на основе скрытых марковских моделей (Kaldi), дающий точные фонемные границы по известному тексту, но работающий медленнее и чувствительный к особенностям пения.

Певческое (singing-oriented) выравнивание. Специализированные системы – SOFA (Singing-Oriented Forced Aligner) [43], STARS [23] и SongTrans [44] – учитывают распевы, мелизмы и удержания, характерные для вокала. Однако их применение для русского языка затруднено: доступные модели SOFA ориентированы преимущественно на мандаринский китайский, а обучение требует фонемно-размеченного певческого корпуса, которого для русского языка нет.

Сравнение методов распознавания текста и выравнивание представлено в таблице 10.

Таблица 10 – Сравнение методов распознавания текста и выравнивания

Метод	Тип	Точность меток	Поддержка рус. пения	Примечание	Источник
Whisper	ASR	Сегментная (~сотни мс)	Хорошая (распознавание)	Без точного выравнивания	[42]
WhisperX (+ wav2vec2)	ASR + выравнивание	Пословная/посимвольная	Хорошая, на GPU	Основной выбор	[3, 21]
MFA (russian_mfa)	Форсир. выравнивание	Фонемная	Дрейфует на пении	Медленный, резервный	[20]
SOFA / STARS / SongTrans	Певческое	Фонемная/нотная	Нет рус. модели/корпуса	Перспектива	[23, 43, 44]

Выбор – WhisperX (Whisper large-v3 + wav2vec 2.0) как основной движок. Он работает на видеокарте, дает пословные и посимвольные временные метки достаточной точности и поддерживает русский язык. MFA с моделью russian_mfa включен как альтернативный движок (дает фонемные границы), однако он медленнее, а его речевая акустическая модель «плывет» на мягких вокальных приступах и распевах, поэтому его уточнения принимаются только при согласии с результатом WhisperX в пределах допуска ± 80 мс (механизм описан в разделах 3–4). SOFA рассмотрена обзорно и не используется ввиду отсутствия русской певческой модели, STARS и SongTrans отнесены к направлениям развития.

Таким образом, текст распознается и выравнивается основным движком WhisperX, MFA служит уточняющей альтернативой с защитным ограничением, певческие системы – перспективой.

2.7 Распознавание аккордов

Распознавание аккордов (Automatic Chord Recognition, ACR) сопоставляет участкам фонограммы аккордовые метки. Классические методы – Chordino/NNLS [45] – основаны на сопоставлении хроматических признаков с шаблонами аккордов, они просты, но ограничены небольшим словарем (мажор/минор). Современный подход BTC (Bi-directional Transformer for Chord recognition) [46] использует двунаправленный трансформер и поддерживает расширенный словарь (включая септаккорды, sus, увеличенные и уменьшенные). Появляются и новые модели (ChordFormer), однако публичные веса для них на момент работы недоступны.

Сравнение методов распознавания аккордов представлено в таблице 11.

Таблица 11 – Сравнение методов распознавания аккордов

Метод	Принцип	Словарь	Реализация	Лицензия	Источник
Chordino / NNLS	Хроматические шаблоны	maj/min	Плагин (Vamp)	GPL	[45]
BTC	Двунаправленный трансформер	Расширенный (170 классов)	PyTorch	MIT	[46]
ChordFormer	Трансформер	Расширенный	Нет публичных весов	–	–

Выбор – BTC. Модель поддерживает большой аккордовый словарь (170 классов), реализована на чистом PyTorch (что обеспечивает совместимость с TensorFlow-моделями Essentia в одном процессе) и распространяется под лицензией MIT. Простой бэкенд autochord рассматривался как базовый, но исключен из итоговой сборки из-за конфликта его TensorFlow-зависимостей с Essentia и ограниченного словаря. Собственного измерения качества распознавания аккордов (по метрике mir_eval.chord [33]) в работе не проводилось – это отмечено как направление дальнейшей оценки.

Таким образом, аккорды распознаются моделью BTC с расширенным словарем, привязка аккордов к долям и слогам выполняется детерминированно (раздел 3).

2.8 Выводы по разделу

Проведен сравнительный анализ моделей по восьми подзадачам анализа музыки и обоснован выбор по каждой из них. Итоговый набор приведен в таблице 12.

Таблица 12 – Итоговый выбор моделей и методов

Подзадача	Выбранное решение	Основание выбора
Разделение источников	BS-RoFormer (audio-separator)	Сбалансированное качество, лучший инструментал, победа на SDX'23
Трекинг долей	Beat This! (final0, без DBN)	Лучший F1, без DBN-постобработки
Темп и размер	Детерминированный вывод из долей	Прозрачность, воспроизводимость
Тональность	Essentia KeyExtractor + librosa (резерв)	Устойчивость по ладу, отказоустойчивость
Жанр	Essentia genre_discogs400	Готовность, 400 поджанров
Эмоции	Essentia emomusic + классификаторы настроения	Готовность, модель Рассела
Текст и выравнивание	WhisperX (Whisper large-v3 + wav2vec2), MFA – альтернатива	GPU, пословные метки, поддержка рус.
Аккорды	BTC (large-voca, 170 классов)	Большой словарь, MIT, совместимость

Большинство выбранных моделей (Essentia/MTG) распространяются под лицензией CC BY-NC-SA, допускающей некоммерческое (в том числе академическое) использование, под некоммерческой лицензией (CC BY-NC) распространяются и веса модели MERT. Для возможной коммерциализации потребовалась бы замена на решения с разрешительными лицензиями (например, CLAP или PANNs). Beat This!, BTC и silero-vad распространяются под лицензией MIT, WhisperX – под BSD, Whisper и wav2vec 2.0 – под лицензией MIT.

Таким образом, для всех подзадач выбраны готовые предобученные модели и детерминированные алгоритмы, не требующие обучения с нуля и укладывающиеся в бюджет одной видеокарты. Спроектированный на их основе конвейер и оригинальный метод выравнивания слогов рассматриваются в разделе 3.

3 Проектирование сервиса и формализация задачи выравнивания слогов

В разделах 1–2 определены задачи сервиса и выбраны модели для каждой подзадачи. В настоящей разделе проектируется архитектура программного сервиса, формализуется центральная задача – выравнивание слогов вокального текста на ритмическую сетку (SRA), и описываются детерминированные алгоритмы, реализующие ее решение: квантизация слогов по долям, регуляризация ритмической сетки и слоговое деление русского текста.

3.1 Функциональные и нефункциональные требования

Функциональные требования. По загруженной аудиозаписи сервис должен определять: тональность (тонику и лад), темп (BPM), тактовый размер, ритмическую сетку долей и тактов, жанр, эмоциональную окраску, аккорды с привязкой к долям такта, вокальный текст с распределением слогов по долям ритмической сетки. Анализ должен запускаться выборочно – пользователь отмечает нужные характеристики из набора {тональность, темп, жанр, эмоция, текст со слогами, аккорды}, и вычисляются только они с учетом зависимостей между этапами. Результат должен предоставляться в машиночитаемом виде (структурированный JSON) и визуализироваться в пользовательском интерфейсе, включая режим ручной коррекции распределения слогов.

Нефункциональные требования. Обработка одного файла длительностью до 5 минут и размером до 50 МБ в распространенных форматах (MP3, WAV, FLAC, OGG, M4A, AAC). Работа в пределах ресурсов одной видеокарты – целевой пик потребления видеопамати ниже 12 ГБ. Асинхронная обработка с отображением прогресса по этапам. Локальное (контейнерное) развертывание без обращения к внешним сервисам в процессе анализа. Воспроизводимость результатов (фиксированное зерно генератора случайных чисел). Отказоустойчивость: сбой отдельного модуля не должен приводить к отказу всего конвейера – недоступная характеристика возвращается как «отсутствует», а не подменяется фиктивным значением.

Пользовательские сценарии. Основные пользователи – музыканты-любители (например, гитаристы), преподаватели и ученики музыкальных школ. Типичный сценарий: загрузка песни – получение тональности, темпа, размера, аккордов по долям и текста, разложенного по ритмической сетке – разучивание или разбор произведения, при необходимости – ручная корректировка раскладки слогов и экспорт результата.

Таким образом, требования определяют облик сервиса как асинхронного многомодульного приложения с выборочным анализом, работающего в пределах одной видеокарты и обеспечивающего честную деградацию при сбоях отдельных моделей.

3.2 Выбор инструментов разработки

Технологический стек подобран при проектировании сервиса и подтвержден при реализации. Серверная часть выполнена на языке Python (версия 3.11) с веб-фреймворком FastAPI, обеспечивающим асинхронную обработку запросов, выбор Python обусловлен тем, что все используемые модели (раздел 2) распространяются как библиотеки для Python. Хранение данных реализовано через абстракцию хранилища с реализациями на основе оперативной памяти и SQLite (через асинхронный драйвер aiosqlite), что оставляет путь к переходу на полноценную СУБД без изменения логики. Контейнеризация выполнена средствами Docker, что обеспечивает воспроизводимость окружения и упаковку тяжелых зависимостей (включая модели и Montreal Forced Aligner). Клиентская часть реализована на React с TypeScript и сборщиком Vite, для отображения формы волны используется WaveSurfer.js, для ритмической сетки – собственный компонент на HTML5 Canvas. Сводка стека приведена в таблице 13.

Таблица 13 – Технологический стек

Уровень	Технология	Назначение
Серверная часть	Python 3.11, FastAPI	Прием запросов, оркестрация конвейера, REST API
Асинхронность	asuncio (очередь, воркер)	Последовательная обработка задач на одной видеокарте

Продолжение таблицы 13

Уровень	Технология	Назначение
Хранение	SQLite (aiosqlite) / память	Кэш результатов, история анализов
Модели	PyTorch, TensorFlow (Essentia), Kaldi (MFA)	Нейросетевые и детерминированные модули анализа
Клиентская часть	React, TypeScript, Vite	Интерфейс, визуализация, редактор сетки
Визуализация	WaveSurfer.js, HTML5 Canvas	Форма волны, ритмическая сетка
Развертывание	Docker, обратный прокси	Контейнеризация, маршрутизация

Таким образом, стек подобран исходя из доступности моделей (Python-экосистема), требований к асинхронности и контейнерному развертыванию, он обеспечивает интеграцию разнородных моделей в единый сервис.

3.3 Архитектура многомодульного конвейера

Сервис построен как многомодульный конвейер (pipeline), а не как единая сквозная (end-to-end) модель. Такой выбор обоснован несколькими соображениями: интерпретируемостью (каждый этап дает проверяемый промежуточный результат), заменяемостью (любую модель можно обновить на более совершенную без переобучения остальных), возможностью кэширования промежуточных артефактов и использования готовых предобученных моделей под каждую подзадачу. Издержками подхода являются накопление ошибок по цепочке (cascade error) и необходимость согласовывать форматы данных между этапами.

Конвейер состоит из десяти измеримых этапов (для каждого фиксируется время выполнения), выполняемых строго последовательно. Порядок и привязка этапов к источнику звука приведены в таблице 14.

Таблица 14 – Этапы конвейера анализа

Этап	Вход (звук)	Модель / метод
Загрузка и нормализация	Файл	Ресемплинг 44,1 кГц, нормализация –14 LUFS
Разделение источников	Микс	BS-RoFormer
Доли, темп, размер	Инструментал	Beat This! + детерминированный вывод
Тональность	Инструментал	Essentia KeyExtractor / librosa
Детекция вокала (VAD)	Вокал	silero-vad
Распознавание и выравнивание текста	Вокал	WhisperX (или MFA)
Слогоделение и квантизация	Текст + тайминги	Детерминированные алгоритмы

Продолжение таблицы 14

Этап	Вход (звук)	Модель / метод
Жанр	Микс	Essentia genre_discogs400
Эмоция	Микс	Essentia emomusic
Аккорды	Микс	BTC

Привязка к источнику звука существенна: трекинг долей и тональность вычисляются по инструменталу (с минимумом протечек вокала), детекция и распознавание текста – по выделенному вокалу, а жанр, эмоция и аккорды – по полному миксу (на котором обучались соответствующие модели). Этапы запускаются по зависимостям: запрос текста со слогами требует разделения, трекинга долей, VAD, распознавания и квантизации, запрос темпа – только трекинга долей, запрос аккордов – трекинга долей и распознавания аккордов. Распознавание текста выполняется лишь при наличии вокала (по результату VAD), иначе трек считается инструментальным.

Общая схема конвейера с указанием источника звука и зависимостей между этапами приведена на рисунке 4.

В итоге многомодульная архитектура обеспечивает интерпретируемость, заменяемость моделей и выборочный анализ, порядок этапов и привязка к источнику звука согласованы с требованиями отдельных моделей.

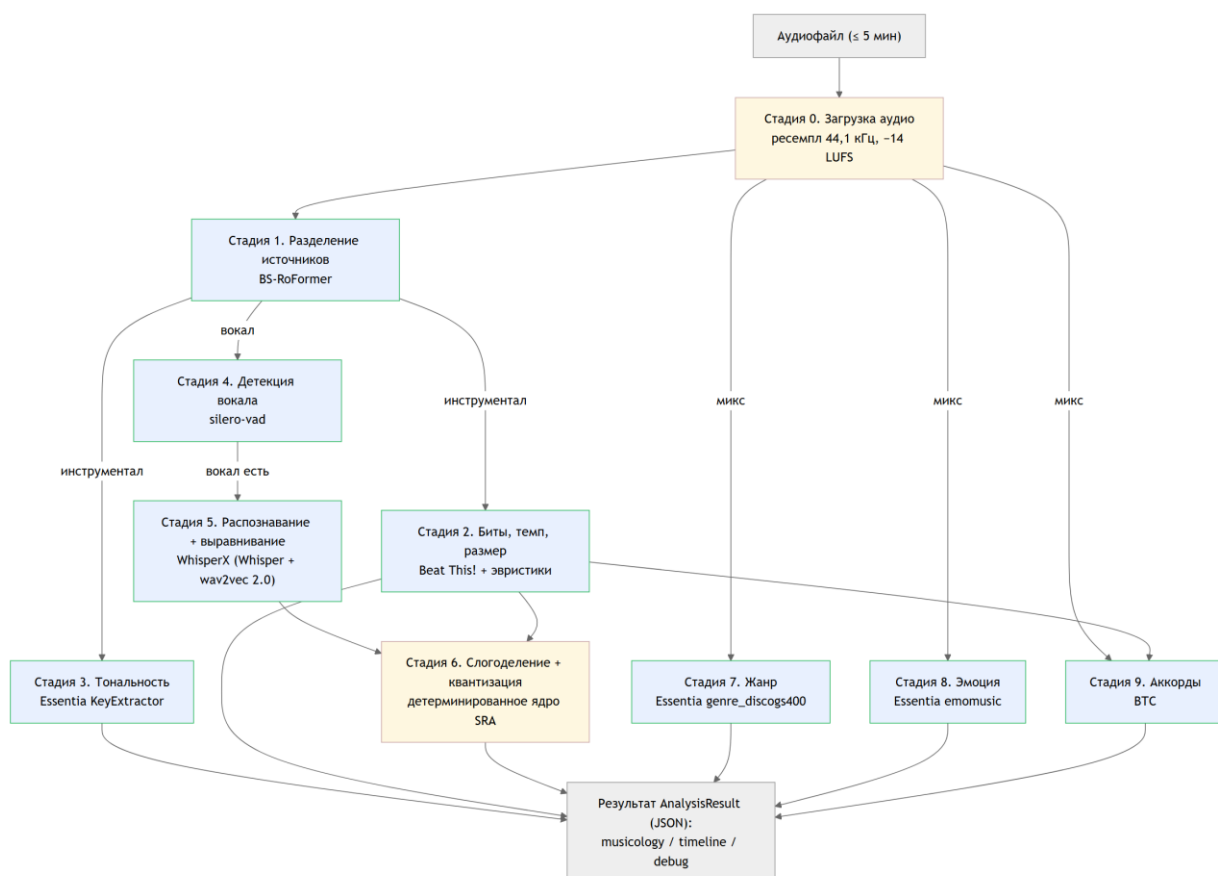


Рисунок 4 – Схема многомодульного конвейера анализа (10 стадий)

3.4 Управление ресурсами и профили моделей

Чтобы удерживать пик потребления видеопамати ниже 12 ГБ при использовании нескольких тяжелых моделей, применяется ленивая загрузка с последующей выгрузкой: каждая модель загружается в память непосредственно перед своим этапом и выгружается по завершении анализа с очисткой кэша видеопамати. Поскольку обработка идет строго последовательно (один асинхронный воркер на одну видеокарту), одновременно в памяти находится не более одной-двух тяжелых моделей.

Для адаптации к доступному оборудованию предусмотрены профили моделей: «cpu» (без видеокарты, облегченные модели), «balanced» (средняя видеокарта), «quality» (максимальное качество) и «quality_mfa» (выравнивание через MFA). Профиль задает устройство, модель распознавания речи и тип вычислений, движок выравнивания и другие параметры, при смене профиля модели перезагружаются.

Таким образом, ленивая загрузка и профили моделей позволяют выполнять ресурсоемкий конвейер в пределах одной видеокарты и адаптировать сервис к различному оборудованию.

3.5 Формализация задачи выравнивания слогов

Центральная задача работы – выравнивание слогов вокального текста на ритмическую сетку (Syllable-Rhythm Alignment, SRA). Неформально: для каждого спетого слога нужно определить, на какую долю такта и в какую ритмическую позицию внутри доли он приходится, и является ли эта доля сильной. В отличие от форсированного выравнивания [20, 21], дающего лишь временные границы, и трекинга долей [22], не работающего с текстом, SRA связывает текст с метрической сеткой.

Вход. Пара (A, L) , где A – аудиосигнал, а L – вокальный текст, разбитый на слова и далее на слоги $\sigma_{i,j}$ (i – индекс слова, j – индекс слога в слове). Предшествующими этапами конвейера из A извлекаются множество долей $B = \{\beta_1, \dots, \beta_M\}$ (моменты времени в секундах) и множество сильных долей $D \subseteq B$.

Выход. Каждому слогу σ сопоставляется кортеж (t, b, p, d) , где t – время начала слога, b – индекс доли, для которой $\beta_b \leq t < \beta_{b+1}$, p – фазовая позиция внутри доли, $d \in \{0,1\}$ – признак сильной доли. Что принять за долю, определяет тактовый размер: его знаменатель задает длительность доли (в размерах вида $X/4$ доля – четверть, в $X/8$ – восьмая), а числитель – число долей в такте. Долю можно дробить мельче, вплоть до шестнадцатой, и это разрешение сетки используется в любом размере. Так, в простом размере доля-четверть делится на четыре шестнадцатые, что дает позиции *on*, *e*, *and*, *a*, составной размер $6/8$ имеет свое дробление доли (детали и другие случаи – в подразделе 3.6). Точное время онсета переводится в непрерывную фазу внутри доли и округляется до ближайшего узла сетки.

Метрика качества (4). Предсказание $(\hat{t}, \hat{b}, \hat{p}, \hat{d})$ для слога считается правильным при допуске τ , если одновременно близко время начала и верно

определены доля, фазовая позиция и признак сильной доли. Качество на множестве слогов S – доля правильно выровненных слогов:

$$\text{SRA-Acc}(\tau) = \frac{1}{|S|} \sum_{\sigma \in S} 1 [|\hat{t} - t^*| < \tau, \hat{b} = b^*, \hat{p} = p^*, \hat{d} = d^*] \quad (4)$$

где $\text{SRA-Acc}(\tau)$ – точность слога-долевого выравнивания при допуске τ ,
 S – множество оцениваемых слогов, $|S|$ – число слогов в нем,
 $1[\cdot]$ – индикатор: 1, если все условия в скобках выполнены, иначе 0,
 $\hat{t}, \hat{b}, \hat{p}, \hat{d}$ – предсказанные время, индекс доли, фазовая позиция и признак сильной доли слога,
 t^*, b^*, p^*, d^* – соответствующие эталонные значения,
 τ – временной допуск.

По построению метрика принимает значения от 0 до 1 и не убывает с ростом допуска τ . Кроме общей точности удобно отдельно смотреть точность по каждому из четырех условий – по времени, доле, позиции и сильной доле.

Выбор допуска $\tau = 80$ мс. Ориентиром служит общепринятый в MIR стандарт оценки трекинга долей: доля считается верной, если попадает в окно ± 70 мс вокруг эталонной [47], и тот же допуск используется в библиотеке `mir_eval` [33]. Значение $\tau = 80$ мс близко к нему и слегка ослаблено с учетом точности пословного выравнивания систем распознавания речи, чувствительность слуха к ритмическим отклонениям также лежит в области десятков миллисекунд [48, 49]. Для анализа чувствительности предусмотрены значения $\tau = 50$ и 100 мс.

Метрика реализована программно (модуль `benchmarks/metrics/sra_accuracy.py`) с сопоставлением предсказанных и эталонных слогов «один к одному» (венгерский алгоритм), то есть операциональна, ее количественная оценка на размеченном корпусе отнесена к дальнейшей работе (раздел 5, заключение).

Таким образом, задача SRA формализована: определены вход, выход в виде кортежа «время – доля – фазовая позиция – признак сильной доли» и метрика SRA-Аccuracy(τ), описаны ее основные свойства и обоснован выбор допуска.

3.6 Алгоритм квантизации слогов на ритмическую сетку

Решение задачи реализовано как детерминированный постпроцессинг поверх результатов конвейера. Для слога с онсетом t методом двоичного поиска находится содержащая его доля β_b (где $\beta_b \leq t < \beta_{b+1}$) и вычисляется нормированная фаза онсета внутри доли:

$$\varphi = \frac{t - \beta_b}{\beta_{b+1} - \beta_b}, \varphi \in [0,1) \quad (5)$$

где φ – нормированная фаза онсета внутри доли,

t – время начала (онсет) слога, с,

β_b – время начала доли, в которую попал слог,

β_{b+1} – время начала следующей доли.

Полученная фаза округляется до ближайшей субдоли. Базовое число субдолей K определяется тактовым размером: $K = 4$ для простых размеров, 6 или 2 для составного 6/8.

Алгоритм квантизации в целом представлен блок-схемой на рисунке 5.

Округление фазы выполняется с гранулярностью onset_K , после чего определяется номер субдоли:

$$\hat{\varphi} = \frac{\text{round}(\varphi \cdot \text{onset_K})}{\text{onset_K}}, \text{sub} = \text{round}(\hat{\varphi} \cdot K) \quad (6)$$

где $\hat{\varphi}$ – округленная фаза онсета,

round – округление до ближайшего целого,

onset_K – гранулярность округления онсета (2 – к восьмым, 4 – к шестнадцатым),

K – базовое число субдолей в доле ($K = 4$ для простых размеров, 6 или 2 для 6/8),

sub – номер субдоли, на которую попадает слог.

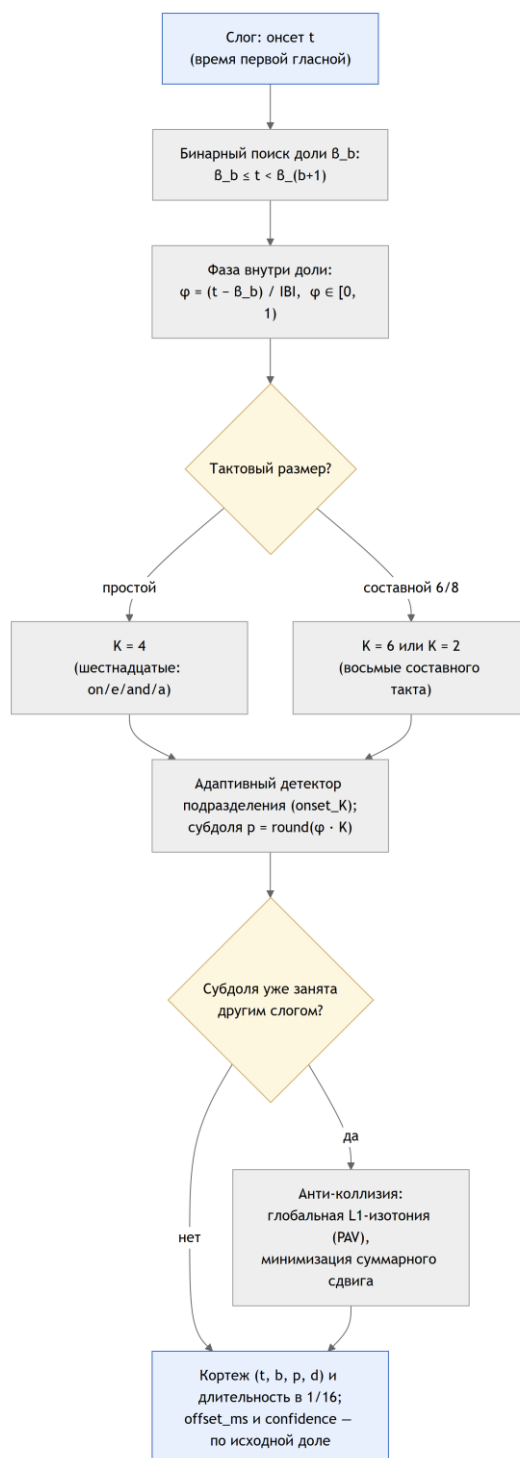


Рисунок 5 – Блок-схема алгоритма квантизации слога на ритмическую сетку (ядро SRA)

По исходной (до разведения наложений) субдоле вычисляются отклонение онсета от идеального положения доли и уверенность (7):

$$\begin{aligned} offset_ms &= \left(t - \left(\beta_b + \frac{sub}{K} (\beta_{b+1} - \beta_b) \right) \right) \cdot 1000, \\ conf &= \max \left(0, 1 - \frac{2 |offset_ms|}{IBI \cdot 1000} \right) \end{aligned} \quad (7)$$

где $offset_ms$ – отклонение онсета от идеального положения субдоли, мс,

$conf$ – уверенность в попадании, от 0 до 1,

t – онсет слога,

β_b – время начала доли,

sub – номер субдоли, K – число субдолей в доле,

IBI – межбитовый интервал доли (см. формулу (1)).

Уверенность равна 1 при точном попадании в субдолю и убывает до 0 при отклонении на половину межбитового интервала. Отклонение и уверенность считаются по исходной ближайшей субдоле и отражают качество пения, а не последующий сдвиг клетки.

Разрешение наложений. Несколько слогов могут претендовать на одну клетку сетки (например, при быстром распеве). Чтобы соблюсти принцип «один слог – одна клетка», такие слоги разводятся по соседним свободным клеткам с наименьшим суммарным сдвигом (метод изотонической регрессии). В отличие от простого сдвига только вперед, разведение идет в обе стороны – в том числе в свободную клетку перед группой, – поэтому не возникает «эффекта домино», когда смещается вся фраза. Для составного размера шаг разведения ограничен так, чтобы слог оставался в пределах своего такта и не уходил в соседний (это устранило характерный для 6/8 «уход» слогов на несколько тактов вперед). При переполнении сетки применяется запасная плотная укладка. Поскольку отклонение и уверенность вычислены по исходной клетке, разведение не искажает оценку качества выравнивания.

Длительность слога. Если для слогов известно время окончания, оно квантуется на ту же глобальную сетку шестнадцатых, и длительность слога в шестнадцатых равна числу клеток между квантованными началом и концом (не менее одной и не более 64). Длительность дополнительно ограничивается, чтобы прямоугольник слога не накрывал следующий слог. При отсутствии времени окончания длительность по умолчанию равна одной клетке.

Геометрия одного шага квантизации (фаза слога внутри доли и ее проекция на ближайшую субдолю) показана на рисунке 6.

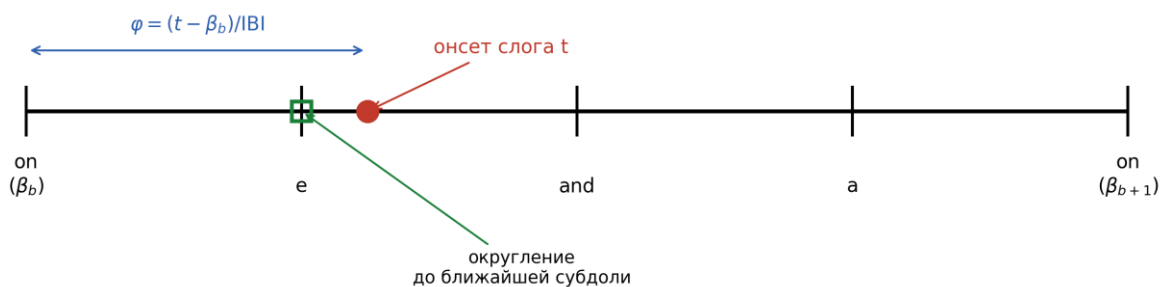


Рисунок 6 – Геометрия квантизации: фаза ϕ онсета слога внутри доли и округление до ближайшей субдоли (on/e/and/a)

Краевые случаи. Онсеты вне диапазона долей (раньше первой доли с допуском 5 мс или позже последней) отбрасываются, а не привязываются к краевой доле с искажением отклонения. Вырожденные межбитовые интервалы пропускаются. Признак сильной доли устанавливается, если время доли совпадает с одной из сильных долей (с допуском 5 мс) и субдоля нулевая.

Сложность. Двоичный поиск доли для каждого из N слогов выполняется за $O(\log M)$, разведение наложений – за $O(N)$, суммарная вычислительная сложность алгоритма составляет порядка $O((N + M) \log M)$, что обеспечивает обработку за доли секунды.

Таким образом, квантизация слогов реализована детерминированным алгоритмом с автоматическим выбором дробности округления, оценкой отклонения и уверенности и разведением наложившихся слогов, алгоритм воспроизводим, объясним и вычислительно дешев.

3.7 Регуляризация ритмической сетки

Качество квантизации напрямую зависит от качества сетки долей и сильных долей, выдаваемой Beat This! [22]. На практике эта сетка содержит характерные артефакты, для устранения которых применяются два детерминированных алгоритма регуляризации.

Регуляризация сильных долей (простые размеры). Beat This! в зонах низкой уверенности иногда выдает паразитные лишние сильные доли – например, одну каждые две доли вместо каждых четырех, – из-за чего отдельные такты охватывают меньше долей, чем определенный размер, и на сетке появляются пустые клетки. Алгоритм привязывает сильные доли к каждой numerator-й доле, закрепляясь на модальной (наиболее частой) фазе тактовой черты среди обнаруженных сильных долей, внефазовые паразитные сильные доли отбрасываются, а сетка перестраивается равномерно. Алгоритм идемпотентен на уже равномерной сетке и применяется только при высокой стабильности темпа и простом размере, чтобы не сгладить настоящую смену размера.

Регуляризация составной сетки (6/8). Для составного размера сильные доли Beat This! определяет надежно (примерно один такт на интервал), но восьмые внутри такта оказываются неравномерными, из-за чего слоги садятся на чужие восьмые, а сильные доли (первая и четвертая) смещаются. Алгоритм вычисляет канонический такт как медиану интервалов между сильными долями, делит чрезмерно длинные интервалы (пропущенные сильные доли) на целое число равных тактов и нарезает каждый такт ровно на заданное число равных восьмых. На проверочном фрагменте (песня «Невеста Полоза», размер 6/8) такая регуляризация привела к совпадению сильных долей с эталоном.

Для корректного выбора между простым и составным размером используется сырой (до нормализации) темп: нормализация ВРМ в диапазон [60, 180] удвоением или делением пополам (см. подраздел 2.2) маскировала бы действительно быстрый 6/8 и срывала пороги определения составного размера. Кроме того, при неоднозначной моде числа долей применяется

устойчивая коррекция числителя по отношению медиан интервалов сильных долей и долей.

Эффект регуляризации – устранение паразитного «полутакта» и переход к равномерной сетке тактов – иллюстрирует рисунок 7.

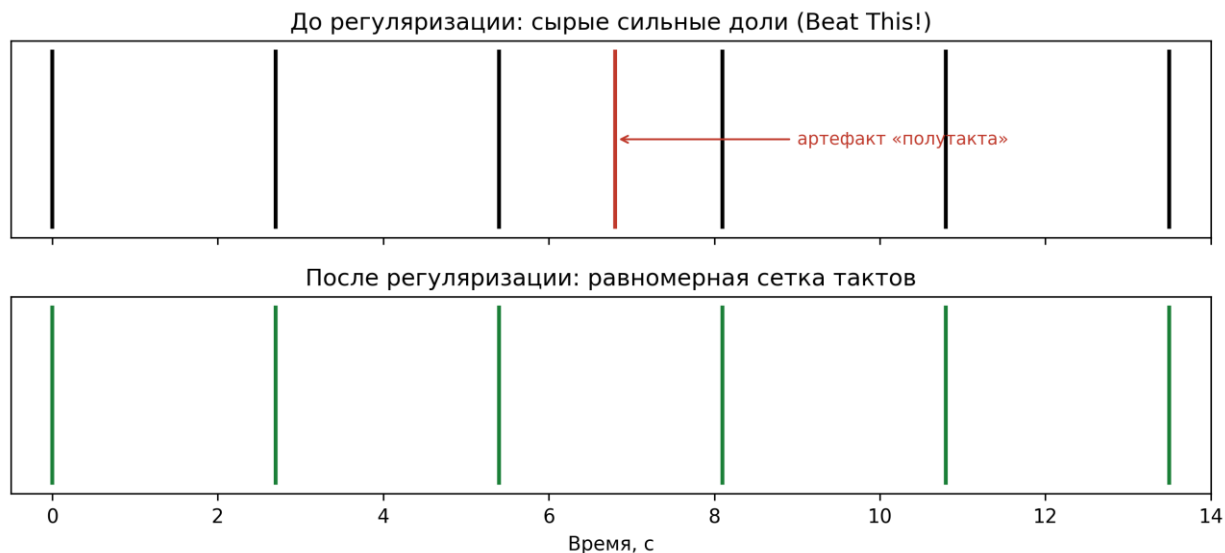


Рисунок 7 – Регуляризация ритмической сетки: сырые сильные доли с артефактом «полутакта» (сверху) и равномерная сетка после регуляризации (снизу)

Таким образом, алгоритмы регуляризации устраняют типичные артефакты трекера долей (паразитные сильные доли в простых размерах и неравномерные восьмые в составном 6/8), что повышает корректность ритмической сетки и, как следствие, качество квантизации слогов.

3.8 Слоговоеделение русского текста

Для размещения текста на сетке необходимо разбить слова на слоги. Используется библиотека `ruphen` (словарь `ru_RU`), дополненная правилами для вокального слогоделения. Базовый инвариант: число слогов равно числу гласных в слове (гласные: а, е, ё, и, о, у, ы, э, ю, я), буквы ь, ъ, й не образуют отдельного слога и присоединяются к предыдущему. Алгоритм выполняется в несколько проходов: разбиение средствами `ruphen`, проверка инварианта с резервным разбиением по группам гласных при его нарушении, постобработка (присоединение неслогообразующих букв) и восстановление пунктуации. Время начала слога определяется как время первой гласной слога по

результатам посимвольного выравнивания (WhisperX), при отсутствии посимвольных меток применяется пропорциональное распределение по числу букв.

Таким образом, слоговое деление реализовано детерминированно на основе словаря и правил с проверкой инварианта «число слогов равно числу гласных», онсет слога привязывается к первой гласной, что точнее простого пропорционального распределения.

3.9 Выводы по разделу

Сформулированы функциональные и нефункциональные требования к сервису и определен технологический стек (Python/FastAPI, SQLite, Docker, React), обеспечивающий интеграцию разнородных моделей.

Спроектирована многомодульная архитектура конвейера из десяти этапов с привязкой каждого этапа к источнику звука (инструментал/вокал/микс) и выборочным выполнением по зависимостям, управление видеопамью реализовано ленивой загрузкой и профилями моделей.

Формализована задача выравнивания слогов на ритмическую сетку (SRA): определены вход, выход (кортеж «время – доля – фазовая позиция – признак сильной доли») и метрика SRA-Ассурасу(τ), описаны ее основные свойства и обоснован допуск $\tau = 80$ мс.

Разработан детерминированный алгоритм квантизации слогов с автоматическим выбором дробности округления, оценкой отклонения и уверенности и разведением слогов, попавших в одну клетку (с наименьшим суммарным сдвигом), вычислительная сложность – $O((N + M) \log M)$.

Предложены алгоритмы регуляризации ритмической сетки, устраняющие артефакты трека долей в простых и составном (6/8) размерах, и детерминированный алгоритм слогового деления русского текста.

Реализация спроектированного сервиса и описанных алгоритмов рассматривается в разделе 4.

4 Программная реализация сервиса

В разделе 3 спроектированы архитектура конвейера и алгоритмы решения задачи SRA. В настоящей разделе описывается их программная реализация: структура приложения, реализация модулей анализа, оркестратор и формат результата, программный интерфейс, пользовательский интерфейс с редактором ритмической сетки, а также экспорт и развертывание. Реализация подтверждена прогоном сервиса в контейнере на наборе тестовых композиций (раздел 5).

4.1 Общая структура приложения

Сервис реализован как клиент-серверное приложение, разворачиваемое в Docker и состоящее из трех контейнеров:

- backend – серверная часть на Python/FastAPI, выполняющая анализ (образ на базе CUDA),
- frontend – клиентское одностраничное приложение на React, собранное в статические файлы и отдаваемое веб-сервером nginx,
- проху – обратный прокси (Caddy), маршрутизирующий запросы: статику отдает frontend, обращения к /api/... направляет на backend.

Веса моделей хранятся в постоянном томе (/data/cache), что позволяет скачивать их один раз и переиспользовать при перезапуске. Поток обработки: клиент загружает аудиофайл – backend ставит задачу в очередь и возвращает ее идентификатор – асинхронный воркер выполняет конвейер – клиент получает прогресс по событиям и итоговый результат. Схема развертывания приведена на рисунке 8.

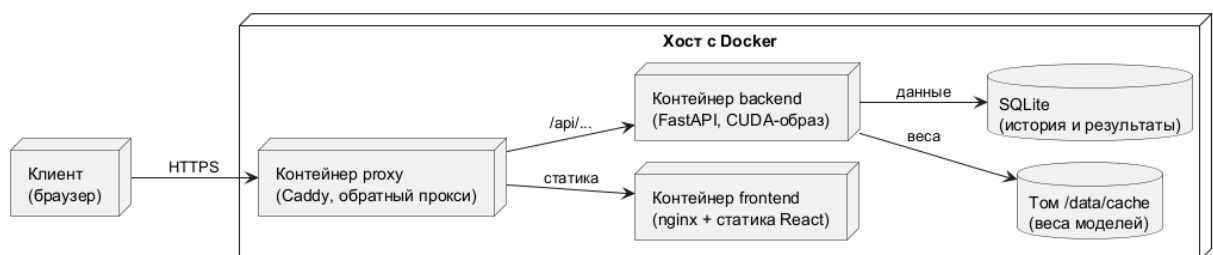


Рисунок 8 – Схема развертывания: клиент – прокси (Caddy) – frontend (nginx) и backend (FastAPI, CUDA) – том с весами моделей

Таким образом, приложение имеет трехконтейнерную архитектуру (backend, frontend, proxy) с постоянным томом для весов моделей и асинхронной обработкой задач, что обеспечивает разделение ответственности и воспроизводимое развертывание.

4.2 Реализация модулей анализа

Каждый модуль анализа реализован как обертка над соответствующей моделью (раздел 2) с единым интерфейсом загрузки и выгрузки, что позволяет оркестратору загружать модели по требованию. Ниже кратко описана реализация по подзадам. Состав модулей серверной части и их связи показаны на рисунке 9.

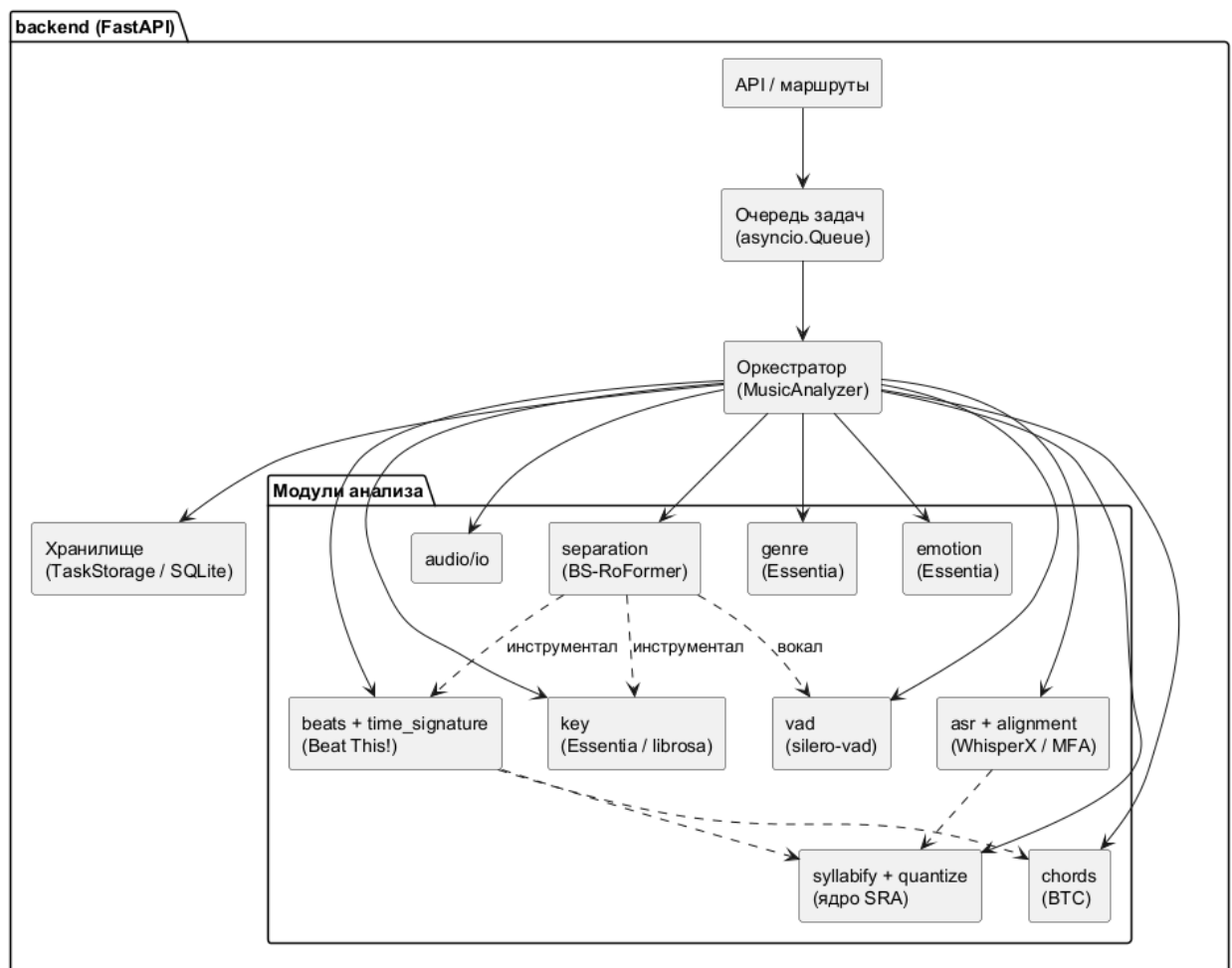


Рисунок 9 – Диаграмма компонентов серверной части: модули анализа, оркестратор, очередь, программный интерфейс, хранилище

Разделение источников (models/separation.py) – модель BS-RoFormer подключается через библиотеку audio-separator, на вход подается микс (44,1

кГц), на выходе – дорожки вокала и аккомпанемента, сохраняемые во временные файлы для последующих этапов.

Доли, темп, размер (models/beats.py, alignment/time_signature.py) – модель Beat This! (чекпойнт final0, без DBN) по инструменталу возвращает доли и сильные доли, из них детерминированными функциями выводятся BPM (медиана межбитовых интервалов с нормализацией в целевой диапазон), тактовый размер (мода числа долей между сильными долями с устойчивой коррекцией) и сегменты тактов, при необходимости применяется регуляризация сетки (подраздел 3.7).

Тональность (models/key.py) – основной путь через Essentia KeyExtractor (профиль temperley, НРСР) по инструменталу, резервный – собственная реализация метода Крумханслы–Шмуклера на librosa.

Детекция вокала (audio/vad.py) – модель silero-vad по дорожке вокала определяет наличие речи, при ее отсутствии трек считается инструментальным и распознавание текста пропускается.

Распознавание и выравнивание текста (models/asr.py, alignment/providers.py, providers_mfa.py) – основной движок WhisperX (распознавание моделью Whisper, форсированное выравнивание моделью wav2vec 2.0 с посимвольными метками) по дорожке вокала, альтернативный движок – Montreal Forced Aligner с моделью russian_mfa, применяемый с защитным ограничением (подраздел 2.6).

Слогоделение и квантизация (alignment/syllabify.py, quantize.py) – детерминированные алгоритмы из разделов 3.6 и 3.8.

Жанр (models/genre.py) – Essentia (эмбединги discogs-effnet и классификатор genre_discogs400) по миксу, 400 поджанров с независимыми (сигмоидными) выходами сворачиваются в укрупненные группы, причем внутри группы берется максимум вероятностей (суммирование давало бы значения больше единицы), и выдается отсортированный список.

Эмоция (models/emotion.py) – Essentia (эмбединги musicnn, регрессор emomusic и классификаторы настроения) по миксу, выход регрессора (шкала

1–9) нормируется в диапазон $[0, 1]$, после чего по знакам валентности и активации относительно 0,5 определяется квадрант модели Рассела (happy, tense, sad, calm) и формируется список тегов настроения.

Аккорды (models/chords.py, third_party/btc/) – модель BTC (расширенный словарь, 170 классов) по миксу, полученные аккордовые сегменты детерминированно привязываются к долям, тактам и слогам (доминирующий по перекрытию аккорд на доле).

Каждый этап измеряется по времени. В таблице 15 приведены роль, источник звука и измеренное время этапов на одном из тестовых треков (трек «Районы-Кварталы», ≈ 40 с – тот же, что на скриншотах интерфейса), это иллюстративный профиль, полные измерения по набору – в разделе 5.

Таблица 15 – Модули анализа и измеренное время этапа (трек «Районы-Кварталы»)

Этап	Модель / метод	Источник звука	Время, мс
Загрузка аудио	librosa/soundfile, –14 LUFS	файл	113
Разделение источников	BS-RoFormer	микс	16605
Доли, темп, размер	Beat This! + эвристики	инструментал	984
Тональность	Essentia KeyExtractor	инструментал	63
Детекция вокала	silero-vad	вокал	601
Распознавание + выравнивание	WhisperX (Whisper + wav2vec2)	вокал	11652
Слогоделение + квантизация	Детерминированные алгоритмы	текст	6
Жанр	Essentia genre_discogs400	микс	2305
Эмоция	Essentia emomusic	микс	981
Аккорды	BTC	микс	637
Итого			33947

Существенно, что детерминированный алгоритм квантизации слогов (ядро SRA) выполняется за единицы миллисекунд, тогда как основное время приходится на разделение источников и распознавание текста.

Таким образом, все модули реализованы как обертки с единым интерфейсом, время анализа определяется тяжелыми нейросетевыми этапами (разделение и распознавание), а детерминированные алгоритмы SRA практически бесплатны.

4.3 Оркестратор и формат результата

Оркестратор (pipeline/orchestrator.py) последовательно выполняет этапы по их зависимостям (подраздел 3.3), загружая и выгружая модели, измеряя

время каждого этапа и фиксируя пик видеопамати. При сбое отдельного модуля оркестратор не прерывает работу: соответствующая характеристика помечается как отсутствующая, что реализует требование честной деградации.

Результат анализа представляется структурой `AnalysisResult` (Pydantic) из четырех блоков:

- `metadata` – имя файла, длительность, частота дискретизации, SHA-256-хэш (ключ кэша),
- `musicology` – тональность (тоника, лад, уверенность), темп (BPM, сырой BPM, стабильность, уверенность), тактовый размер (числитель, знаменатель, флаг составного, конвенция доли), жанр (основной и список с вероятностями), эмоция (валентность, активация, квадрант, теги настроения),
- `timeline` – доли, сильные доли, такты, текст (язык, флаг инструментала, слова и слоги), аккорды (сегменты, аккорд на долю, аккорд на такт),
- `debug` – версии моделей по этапам, времена этапов, пик видеопамати, зерно генератора, заметки о сбоях.

Ключевой объект – выравненный слог `SyllableAlignment` – содержит поля: текст, время начала и конца, индекс доли, индекс такта, доля в такте, шестнадцатая в доле и в такте, длительность в шестнадцатых, метку позиции (on/e/and/a или 1–6), флаг сильной доли, знаковое отклонение от идеальной доли (мс), уверенность, индекс родительского слова и аккорд доли. Именно этот объект визуализируется на ритмической сетке. Сокращенный пример результата приведен в приложении Г.

Таким образом, результат имеет строго типизированную четырехблочную структуру, центральный объект `SyllableAlignment` несет полную информацию о положении слога на сетке, достаточную для визуализации и экспорта.

4.4 Программный интерфейс и асинхронная обработка

Серверная часть предоставляет REST-интерфейс (таблица 16). Загруженный файл проходит валидацию (проверка MIME-типа и сигнатуры

файла, ограничения 50 МБ и 5 минут), помещается в очередь (asyncio.Queue), а единственный асинхронный воркер обрабатывает задачи строго последовательно (один графический ускоритель – одна задача). Прогресс транслируется клиенту через поток событий Server-Sent Events, итоговый результат запрашивается отдельным GET-запросом. Результаты кэшируются по паре (хэш файла, профиль): повторная загрузка того же файла возвращает готовый результат без повторного прогона.

Последовательность обработки запроса (очередь, воркер, трансляция прогресса, кэш) показана на рисунке 10.

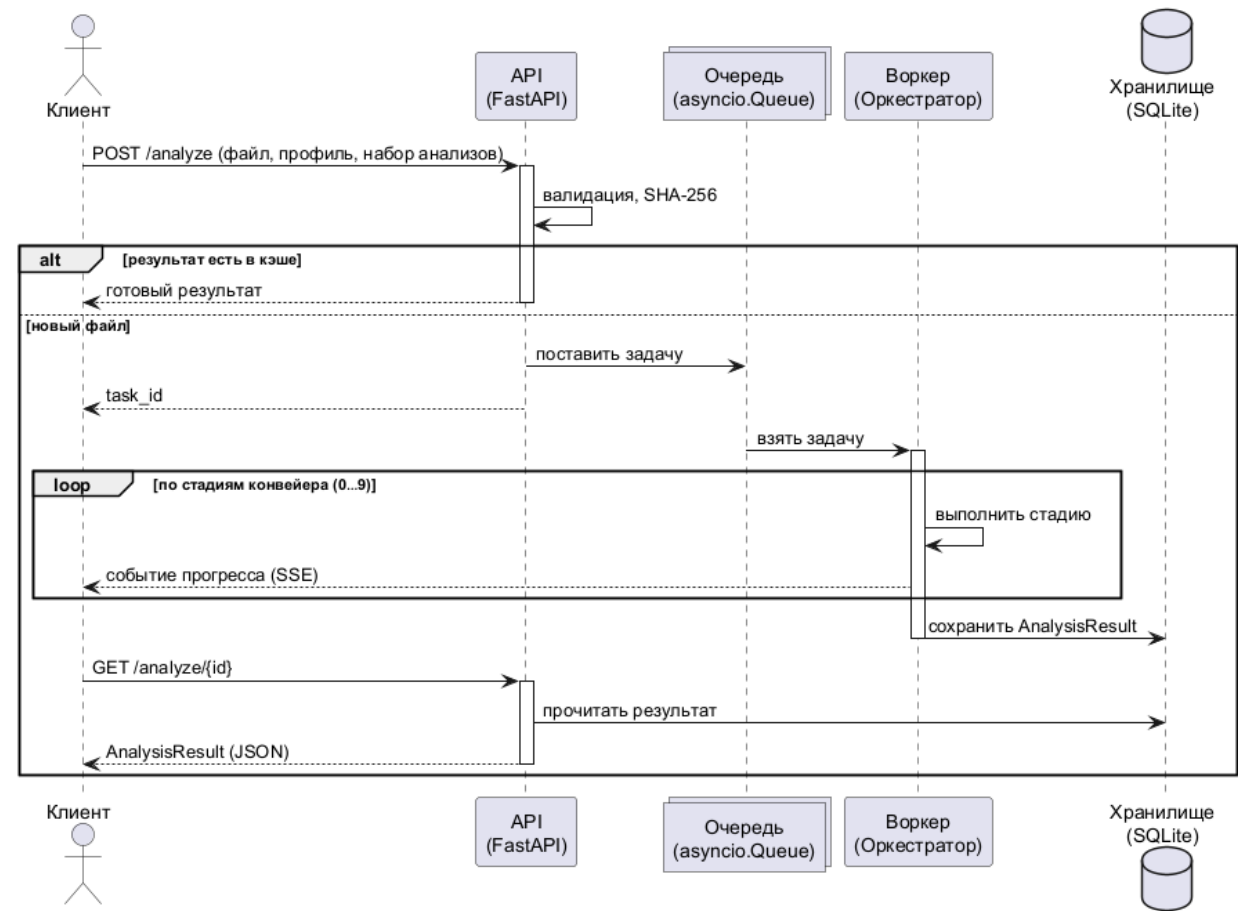


Рисунок 10 – Диаграмма последовательности обработки запроса

Таблица 16 – Основные эндпоинты REST API

Метод и путь	Назначение
POST /analyze	Отправить аудиофайл на анализ, параметры – профиль, набор анализов
GET /analyze/{id}	Статус задачи и результат по завершении
GET /analyze/{id}/events	Поток прогресса (SSE)

Продолжение таблицы 16

Метод и путь	Назначение
GET /analyze/{id}/audio, /peaks, /stems/{kind}	Исходное аудио, пики формы волны, стемы
GET /analyze/{id}/lrc	Экспорт текста в формате LRC
POST /analyze/{id}/cancel, DELETE /analyze/{id}	Отмена и удаление задачи
GET /library	История анализов с фильтрами
GET /pipeline/stages	Список этапов конвейера
GET /config/profiles, PUT /config/profile	Профили моделей
GET /healthz, GET /readyz	Проверки работоспособности и готовности

Итак, программный интерфейс обеспечивает асинхронную обработку с очередью и трансляцией прогресса, выборочный анализ, кэширование по хэшу и экспорт, единственный воркер согласован с ограничением «одна задача на ускоритель».

4.5 Пользовательский интерфейс

Клиентское приложение реализовано на React с TypeScript. На главной странице (рисунок 11) пользователь отмечает нужные анализы (тональность, темп, жанр, эмоция, слоги по долям, аккорды), загружает аудиофайл перетаскиванием или выбором, после чего видит живой прогресс по этапам конвейера, ниже выводится история недавних анализов.

Страница результата (рисунок 12) показывает сводные карточки: тональность (тоника, лад, уверенность), темп (BPM, стабильность, уверенность), тактовый размер (с обозначением составного), жанр (основной и распределение по группам) и эмоцию – в виде круга Рассела с отмеченной точкой валентности/активации и тегами настроения. Ниже расположен аудиоплеер с формой волны, переключателем источника (микс / вокал / инструментал) и регулировкой скорости, а под ним – вкладки с детальным представлением: ритмическая сетка (BeatGrid), караоке, таблица слогов и аккорды.

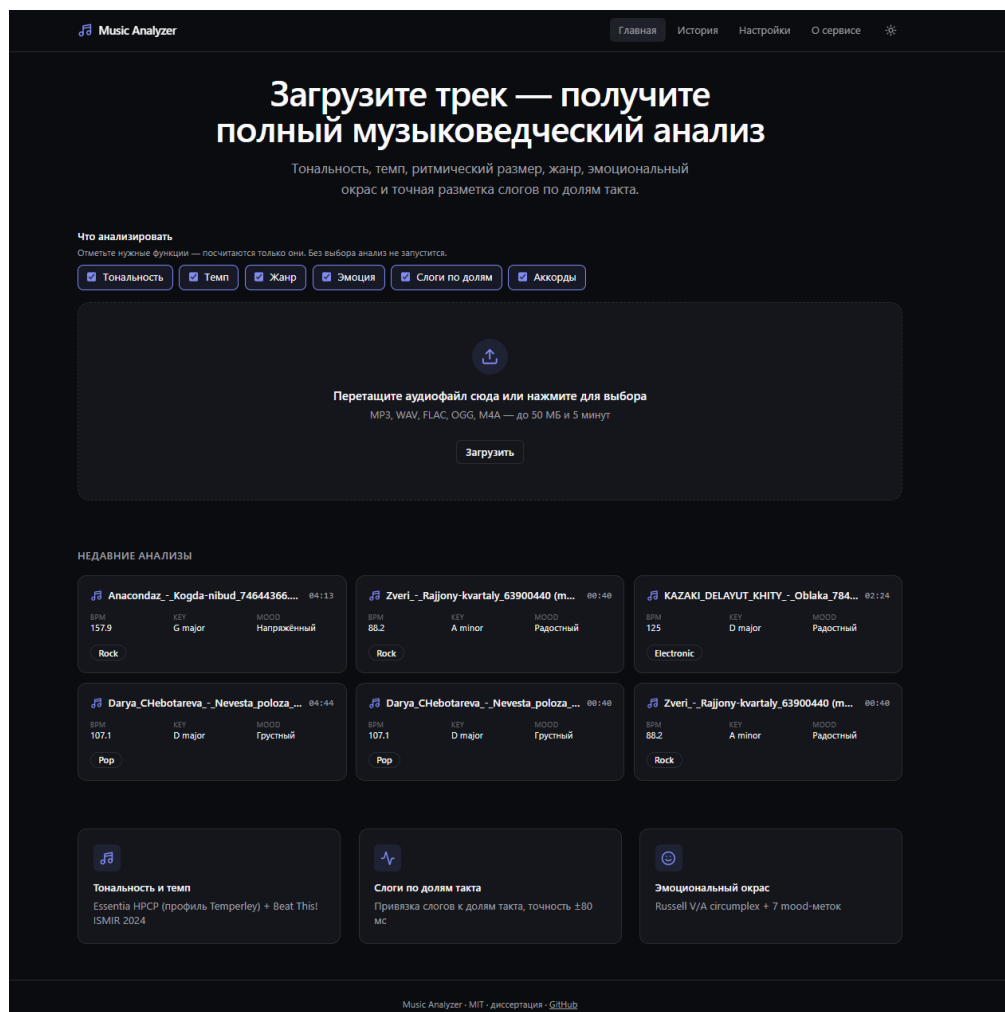


Рисунок 11 — Главная страница: выбор анализов, область загрузки, недавние анализы

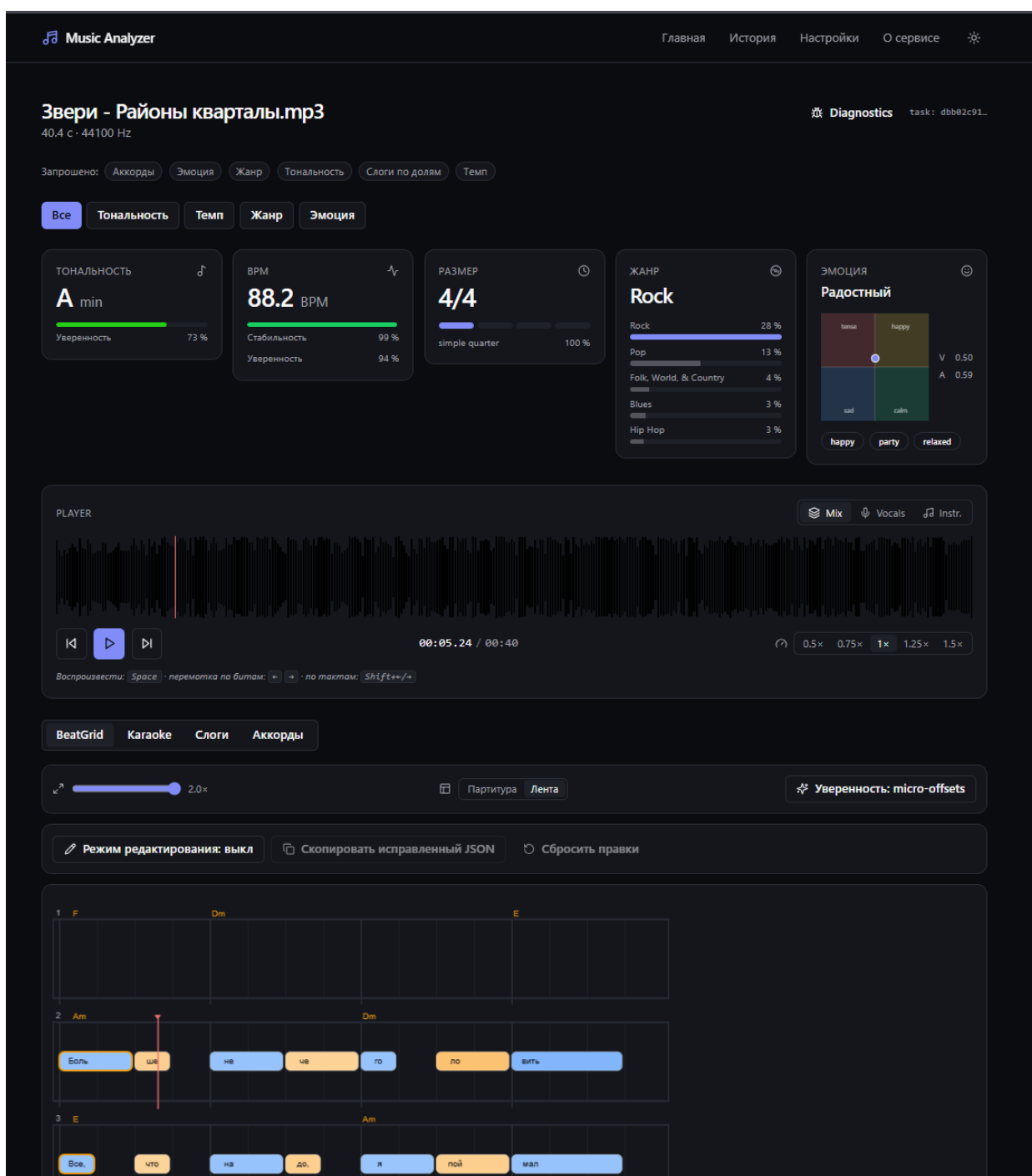


Рисунок 12 – Страница результата: сводные карточки (тональность, темп, размер, жанр, эмоция – круг Рассела), плеер и вкладки

Порядок работы с сервисом. Пользоваться приложением просто. На главной странице пользователь галочками отмечает, что именно проанализировать, – доступны тональность, темп, тактовый размер, жанр, эмоция, текст со слогами по долям и аккорды, отметить можно как все сразу, так и только нужные характеристики (остальные этапы не запускаются, что экономит время). Затем аудиофайл загружается перетаскиванием или

выбором, и пользователь наблюдает прогресс по этапам конвейера. По завершении открывается страница результата со сводными карточками и детальными вкладками. Полученную раскладку можно донастроить: в редакторе ритмической сетки слоги перетаскиваются в нужные клетки (подраздел 4.6), в плеере переключается источник звука (микс / вокал / инструментал) и скорость воспроизведения, у сетки регулируется масштаб и режим отображения. Итог экспортируется в нужном формате (подраздел 4.7) или сохраняется в истории анализов для повторного открытия без пересчета.

Таким образом, интерфейс предоставляет как сводное представление результата (карточки и круг Рассела), так и детальные виды (сетка, караоке, таблица слогов, аккорды), синхронизированные с аудиоплеером, и позволяет выборочно задавать состав анализа и донастраивать результат.

Вкладка караоке подсвечивает активную строку и слог при воспроизведении, вкладка слоги показывает таблицу всех слогов с временем, тактом, долей, шестнадцатой, меткой позиции и отклонением, вкладка аккорды отображает последовательность аккордов и аккорд на каждый такт. Больше скриншотов интерфейса можно посмотреть в презентации в приложении В.

4.6 Редактор ритмической сетки слогов

Центральный элемент интерфейса – компонент BeatGrid, отображающий ритмическую сетку: такты, доли и шестнадцатые, метки аккордов над тактом и плитки слогов в соответствующих клетках. Ширина плитки слога пропорциональна его длительности в шестнадцатых, цвет кодирует позицию в доле, а яркость – уверенность, сетка отображается в режиме «партитуры» (такты переносятся по строкам) или «ленты» (горизонтальная прокрутка) с регулируемым масштабом (рисунок 13).

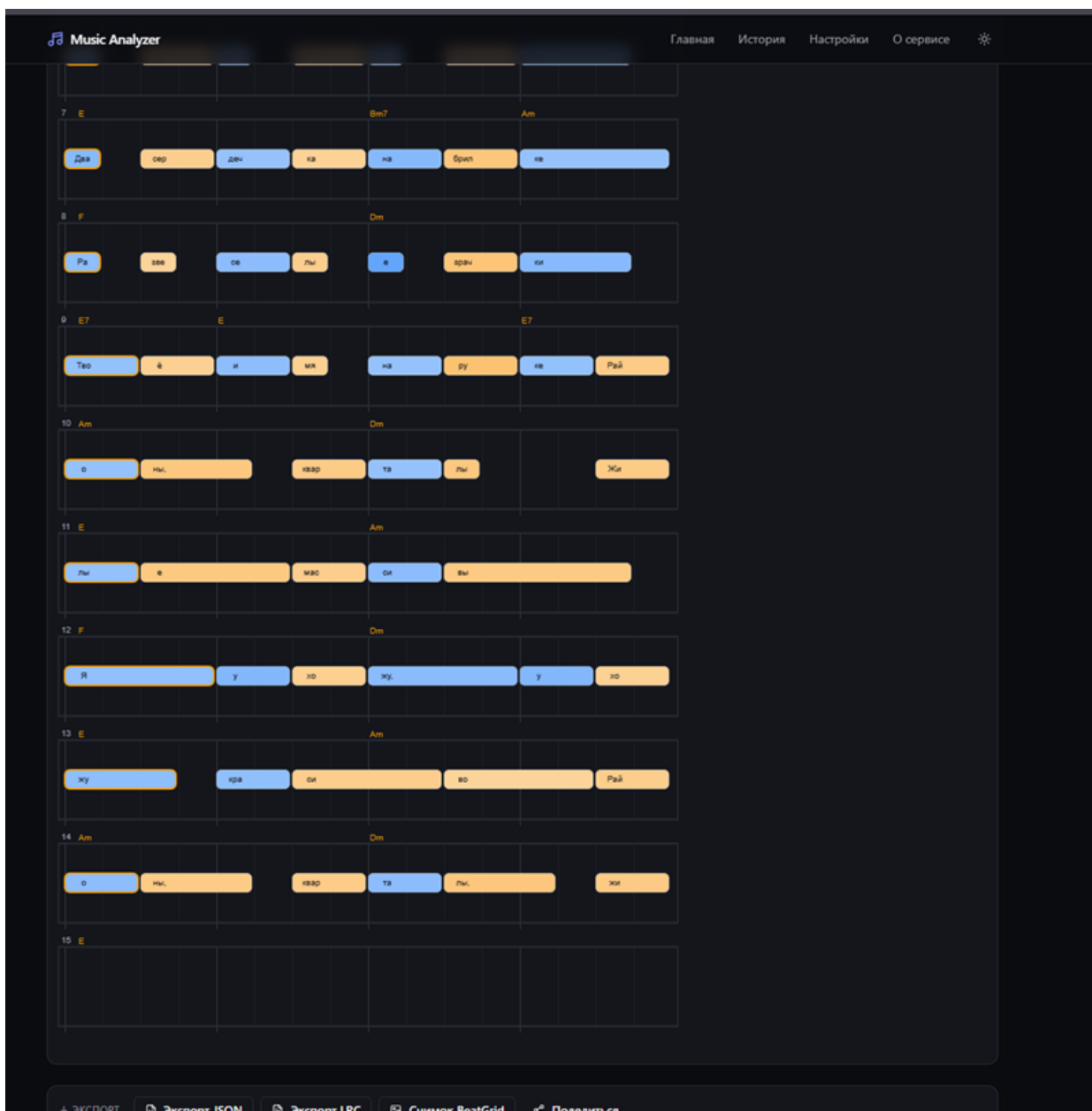


Рисунок 13 – Ритмическая сетка (BeatGrid): такты, доли, аккорды над тактом и плитки слогов

Особенность реализации – режим ручной коррекции (рисунок 14). При его включении пользователь перетаскивает плитки слогов в правильные клетки, при перемещении пересчитываются все зависимые поля слога (индексы такта, доли, шестнадцатой, метка позиции, флаг сильной доли, время, а отклонение обнуляется и уверенность выставляется в единицу как для эталонной разметки). Исправленный результат можно скопировать в виде полного JSON или сбросить к автоматической раскладке. Этот режим служит инструментом отладки и подготовки эталонной разметки: он позволяет

вручную выверить распределение слогов по долям и использовать выверенный вариант как образец для оценки качества автоматического алгоритма.

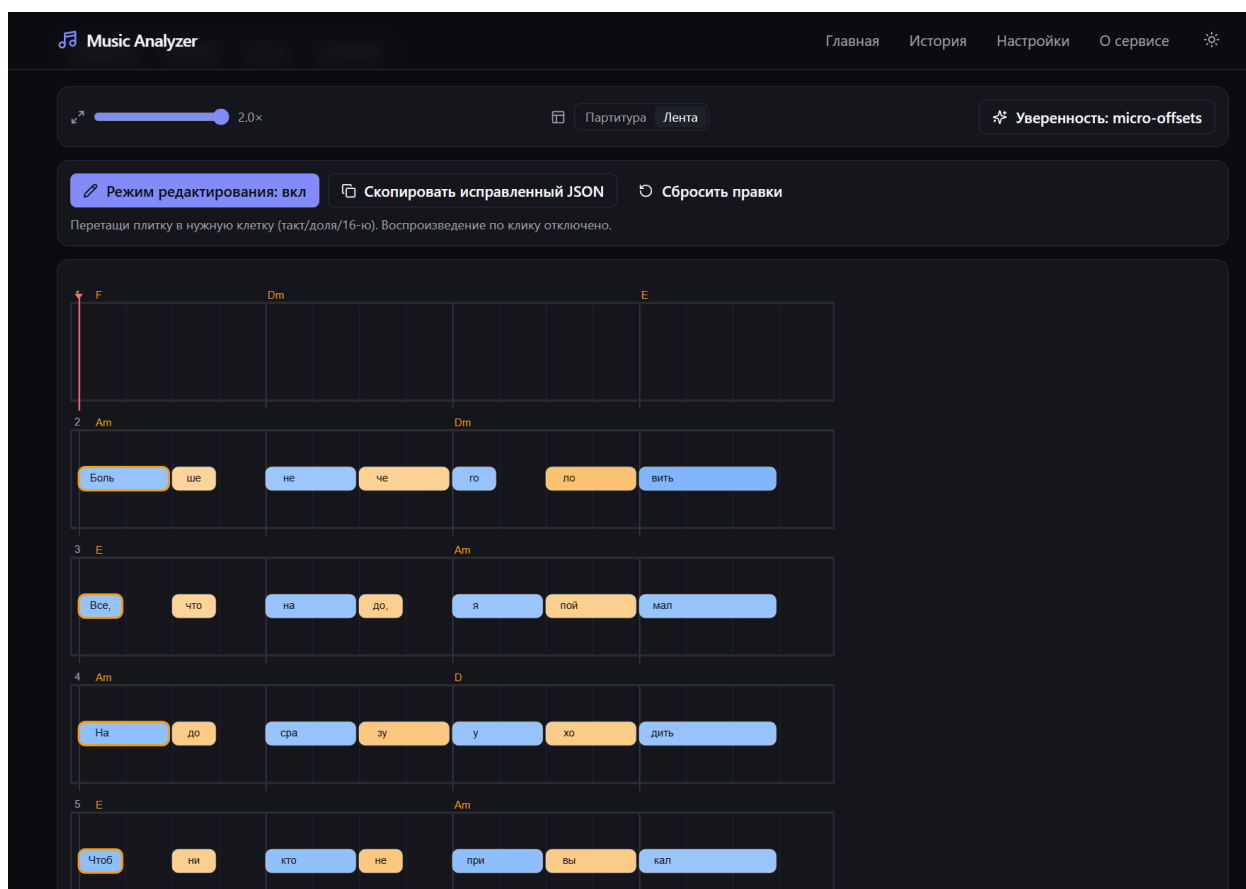


Рисунок 14 – Ритмическая сетка во включенном режиме редактирования

В итоге редактор ритмической сетки реализует центральную прикладную функцию сервиса (наглядное распределение слогов по долям) и одновременно служит инструментом подготовки эталонной разметки для оценки качества SRA.

Укрупненная схема пайплайна и диаграмма классов отображены в приложении А.

4.7 Экспорт, диагностика и развертывание

Экспорт. Из пользовательского интерфейса результат экспортируется в нескольких форматах: полный JSON, караоке-субтитры LRC (построчный и пословный варианты), снимок ритмической сетки в формате PNG, а также формируется ссылка для совместного доступа. В серверной части

дополнительно реализованы функции экспорта аккордов (формат ChordPro и «песенник» – строка аккордов над строкой слогов), которые пока не выведены в интерфейс. Экспорт позволяет использовать результат во внешних программах и как иллюстрацию при разборе песни.

Диагностика. Боковая панель диагностики (приложение В) показывает времена этапов, версии моделей по этапам, пик видеопамати, версию кода и заметки конвейера, а также сырой JSON результата – что использовалось при отладке и сборе данных для оценки.

Развертывание. Приложение разворачивается командой сборки и запуска контейнеров. Образ backend основан на CUDA-окружении, тяжелые зависимости (включая Montreal Forced Aligner) устанавливаются на этапе сборки, веса моделей скачиваются при первом запуске в постоянный том и далее переиспользуются. Профиль моделей (подраздел 3.4) задает используемые модели и устройство, при недоступности графического ускорителя предусмотрен режим работы на центральном процессоре.

Таким образом, сервис поддерживает экспорт результата в практически полезные форматы, предоставляет диагностику для отладки и собирается в воспроизводимое контейнерное развертывание с постоянным кэшем моделей.

4.8 Выводы по разделу

Сервис реализован как трехконтейнерное клиент-серверное приложение (backend на FastAPI, frontend на React, прокси Caddy) с постоянным томом для весов моделей и асинхронной обработкой.

Все модули анализа реализованы как обертки над выбранными моделями с единым интерфейсом загрузки/выгрузки, измеренный профиль времени подтверждает, что основные затраты приходятся на разделение источников и распознавание текста, а детерминированные алгоритмы SRA выполняются за единицы миллисекунд.

Результат представлен строго типизированной структурой из четырех блоков с центральным объектом выравненного слога, программный

интерфейс обеспечивает асинхронную обработку, выборочный анализ, кэширование и экспорт.

Реализован пользовательский интерфейс со сводными карточками, аудиоплеером и детальными видами, а также редактор ритмической сетки – центральная прикладная функция и инструмент подготовки эталонной разметки.

Сервис поддерживает экспорт в практически полезные форматы и собирается в воспроизводимое контейнерное развертывание.

Экспериментальная проверка работоспособности сервиса и качества анализа рассматривается в разделе 5.

5 Оценка качества моделей и работоспособности сервиса

В предыдущих разделах обоснован выбор моделей (раздел 2), спроектированы конвейер и метод выравнивания слогов (раздел 3) и описана программная реализация (раздел 4). Настоящая раздел посвящена экспериментальной проверке: насколько качественно работают отдельные модели по опубликованным данным, насколько работоспособен сервис в целом и как ведет себя предложенный метод слога-долевого выравнивания на реальных записях. Отдельно рассмотрены сравнение движков выравнивания и известные ограничения.

5.1 Методика оценки

Качество системы анализа музыки складывается из качества отдельных моделей и работоспособности сервиса в целом. С учетом этого принята трехуровневая методика оценки, отражающая реально проверяемое:

Качество отдельных моделей – по публичным бенчмаркам из статей-первоисточников (подраздел 5.2). Для каждой подзадачи приводятся опубликованные значения на общепринятых наборах данных с указанием источника и набора у каждого числа. Подчеркивается, что это литературные значения, а не собственные замеры, и что протоколы оценки у разных работ различаются, поэтому числа служат ориентиром, а не строго сопоставимым рейтингом.

Работоспособность сервиса – функциональным тестированием на наборе реальных композиций (подраздел 5.3). Здесь измеряются объективно проверяемые величины: правильность определения тональности и тактового размера, отклонение темпа, среднее время обработки и пик потребления видеопамати. Эти данные получены реальным прогоном сервиса в контейнере.

Слого-долевое выравнивание – качественной демонстрацией (подраздел 5.4): разбором автоматической раскладки слогов по долям на конкретных треках, в том числе в режиме ручной коррекции редактора (раздел 4.6). Результат оценивается содержательно («слоги встают на свои доли, не

накладываются, не уходят в чужой такт»)), без введения собственных числовых метрик.

Для понимания приводимых в разделе 5.2 величин ниже кратко поясняется смысл основных метрик MIR.

Отношение сигнал/искажение (SDR, дБ) характеризует качество разделения источников – это логарифмическое отношение энергии целевого сигнала s к энергии искажений $s - \hat{s}$:

$$SDR = 10 \lg \frac{\sum s^2}{\sum (s - \hat{s})^2} \quad (8)$$

где SDR – отношение сигнал/искажение, дБ,

s – отсчеты целевого (эталонного) сигнала,

$\hat{s}$ – отсчеты выделенного сигнала,

$s - \hat{s}$ – сигнал искажений (ошибка разделения).

Чем выше значение SDR, тем чище выделенная дорожка. Эту метрику применяют для оценки моделей разделения вокала и аккомпанемента [29].

F-мера трекинга долей (beat / downbeat F-measure) – гармоническое среднее точности и полноты обнаружения долей:

$$F_1 = \frac{2PR}{P+R} \quad (9)$$

где F_1 – F-мера трекинга долей,

P – точность (доля верно найденных долей среди всех найденных),

R – полнота (доля верно найденных долей среди эталонных).

Найденная доля считается верной, если попадает в окно ± 70 мс вокруг эталонной. Чем ближе F-мера к 1, тем точнее трекинг долей [33, 47].

Доля ошибочных слов (WER, %) – стандартная метрика распознавания речи: отношение числа вставок, удалений и замен к числу эталонных слов, чем ниже, тем лучше. Точность определения тональности в практике MIREX (ежегодная кампания сравнительной оценки алгоритмов MIR) оценивается взвешенной мерой, начисляющей частичный балл за музыкально близкие

ошибки (доминанта, параллельная и одноименная тональности). Точность классификации жанра – доля верно классифицированных фрагментов. Качество распознавания эмоций в модели «валентность–активация» измеряют корреляцией Пирсона (а также среднеквадратичной ошибкой RMSE и согласованной корреляцией CCC) отдельно по каждой из двух осей. Качество распознавания аккордов измеряют взвешенным по времени совпадением аккордовых меток (weighted chord symbol recall) при разных уровнях детализации словаря.

Таким образом, принята трехуровневая методика: качество моделей берется из публичных источников, работоспособность сервиса измеряется функциональным тестом, а слога-долевое выравнивание оценивается качественно.

5.2 Качество моделей по публичным данным

Качество выбранных моделей оценивается по результатам, опубликованным в статьях-первоисточниках на общепринятых наборах данных. Эти значения являются литературными ориентирами, а не собственными замерами, протоколы оценки (тестовые наборы, объем обучающих данных, словарь меток) у разных работ различаются, поэтому числа не образуют строго сопоставимого рейтинга. Для прозрачности у каждого значения в таблице 17 указаны метрика, набор данных и источник.

Таблица 17 – Качество выбранных моделей по публичным бенчмаркам

Подзадача	Модель / метод	Метрика	Значение	Набор данных	Источник
Разделение источников	BS-RoFormer	SDR, среднее по дорожкам	9,76 дБ (без доп. данных), 11,99 дБ (с доп. данными)	MUSDB18-HQ	[26]
Разделение (для сравнения)	HTDemucs	SDR, среднее по дорожкам	≈7,7 дБ	MUSDB18-HQ	[25]
Трекинг долей	Beat This!	F-мера долей / сильных долей (±70 мс)	89,1 / 78,3	GTZAN	[22]

Продолжение таблицы 17

Подзадача	Модель / метод	Метрика	Значение	Набор данных	Источники
Распознавание/выравнивание (рус.)	wav2vec2-large-xlsr-53-russian	WER (CER)	13,3 % (2,88 %), 9,57 % с языковой моделью	Common Voice (рус.)	[50]
Тональность	Темплейтные методы (профили Крумханслы / Темперли)	Взвешенная точность (MIREX)	высокая на тонально ясной музыке, ниже на гармонический и неоднозначной (сильно зависит от жанра)	MIREX-наборы	[34, 35]
Жанр	MERT-330M (foundation-модель)	Точность (accuracy)	65,6 %	GTZAN	[4]
Жанр (классический базовый)	Спектральные признаки + классификатор (Tzanetakis–Cook)	Точность (accuracy)	≈61 %	GTZAN	[36]
Эмоция	Динамическая регрессия валентности/активации	Корреляция (Пирсон)	активация ≈0,5–0,7, валентность – существенно ниже	DEAM (MediaEval)	[40]
Аккорды	BTC (большой словарь, 170 классов)	WCSR: Root / MajMin / Sevenths / MIREX	83,5 / 82,3 / 71,8 / 80,8	Isophonics + Robbie Williams + UsPop2002 (5-fold)	[46]

Прокомментируем результаты по подзадачам.

Разделение источников. Выбранная модель BS-RoFormer достигает среднего по дорожкам SDR 9,76 дБ без дополнительных данных и 11,99 дБ при обучении с дополнительными композициями, превосходя HTDemucs (≈7,7 дБ) [25, 26], используемый в сервисе чекпойнт семейства заявляет SDR по вокалу около 12,98 дБ. Это соответствует современному уровню и обеспечивает достаточно чистое разделение для последующих этапов.

Трекинг долей. Beat This! показывает на GTZAN F-меру 89,1 для долей и 78,3 для сильных долей (допуск ± 70 мс по mir_eval) [22, 33]. Заметная разница между метриками отражает общую закономерность: определение сильных долей (а значит, и тактового размера) сложнее определения собственно долей, что согласуется с обсуждаемыми далее трудностями на составных размерах.

Распознавание и выравнивание текста. Применяемая для русского форсированного выравнивания акустическая модель wav2vec 2.0 (large-xlsr-53-russian) дает на тестовой части Common Voice WER 13,3 % (с языковой моделью – 9,57 %) при CER 2,88 % [50], распознавание моделью Whisper отличается устойчивостью к шуму и многоязычностью [42]. Эти значения относятся к разборчивой речи, на пении (распевы, мягкие приступы) фактическая точность ниже, что учтено защитным механизмом сравнения движков (подраздел 5.5).

Тональность. В практике MIREX точность определения тональности оценивается взвешенной мерой, начисляющей частичный балл за музыкально близкие ошибки (доминанту, параллельную и одноименную тональности). Темплейтные методы (профили Крумханслы, Темперли), к которым относится используемый Essentia KeyExtractor, надежно работают на тонально ясной популярной и классической музыке, однако их точность сильно зависит от материала и заметно снижается на гармонически неоднозначных стилях (например, электронной танцевальной музыке), современные обучаемые модели (сверточные сети) превосходят профильные методы и лучше обобщаются на разные жанры [34, 35]. Конкретное числовое значение здесь не приводится, так как оно сильно зависит от тестового набора и не сопоставимо между протоколами.

Жанр. Жанровая классификация на эталонном наборе GTZAN остается умеренно трудной задачей: классический подход на спектральных признаках достигает около 61 % [36], а современная foundation-модель MERT-330M – 65,6 % при линейном классификаторе поверх представлений [4]. Эти значения

приведены как ориентир уровня задачи: используемая в сервисе модель *Essentia genre_discogs400* обучена на иной таксономии (*Discogs*, 400 поджанров) и на *GTZAN* не бенчмаркалась, поэтому прямое сопоставление некорректно, ее практическое поведение на русском репертуаре обсуждается в подразделах 5.3 и 5.6.

Эмоция. На наборе *DEAM* (динамические оценки валентности и активации, используемые метрики – корреляция Пирсона, *RMSE* и согласованная корреляция *CCC*) проявляется устойчивая закономерность: активация предсказывается заметно лучше валентности. По итогам кампании *MediaEval* лучшие системы достигают по активации корреляции порядка 0,5–0,7, тогда как качество по валентности существенно ниже – на тестовом наборе 2015 года, где оси валентности и активации были декоррелированы, моделирование валентности оказалось практически неудовлетворительным [40]. Причина в том, что валентность хуже выражается через низкоуровневые акустические признаки, используемая модель *Essentia (emomusic)* относится к тому же семейству аудио-моделей и наследует это свойство, что следует учитывать при интерпретации эмоционального квадранта.

Аккорды. Модель *BTC* с большим словарем (170 классов) на стандартных наборах (*Isophonics*, *Robbie Williams*, *UsPop2002*, 5-кратная кросс-валидация) достигает взвешенного совпадения аккордовых меток (*WCSR*) 83,5 % по корню, 82,3 % по мажор/минор, 71,8 % по септаккордам и 80,8 % по метрике *MIREX* [46]. Закономерно, что качество падает при росте детализации словаря (септаккорды распознаются хуже трезвучий) – это объясняет отдельные неточности на сложных гармониях (подраздел 5.4).

Таким образом, по опубликованным данным выбранные модели соответствуют современному уровню по каждой подзадаче: разделение источников и трекинг долей – на уровне лучших открытых решений, распознавание/выравнивание и аккорды – с практически приемлемой точностью, а распознавание тональности, жанра и эмоции – с известными по литературе ограничениями (зависимость тональности от жанра, умеренная

точность жанровой классификации, заведомо худшее предсказание валентности по сравнению с активацией). Приведенные значения – литературные ориентиры с поясечной атрибуцией, а не собственные замеры.

5.3 Функциональное тестирование сервиса

Работоспособность сервиса проверена прогоном в контейнере (Docker, графический ускоритель) на наборе из пяти композиций разного характера, этого набора достаточно и для проверки работоспособности всех модулей, и для качественного разбора слога-долевой раскладки (подраздел 5.4). В набор вошли:

- «Районы-Кварталы» (исп. «Звери») – рок, размер 4/4,
- «Облака» (исп. «Казаки делают хиты») – эстрадная композиция, 4/4,
- «Белая ночь» (исп. В. Салтыков) – эстрадная композиция, 4/4,
- «Ничего на свете лучше нету» (из м/ф «Бременские музыканты», исп. М. Магомаев) – детская/эстрадная, 4/4,
- «Невеста Полоза» (исп. Д. Чеботарева) – композиция в составном размере 6/8 (на фрагменте).

Для каждого трека запускался полный набор анализов (принудительно, без обращения к кэшу). Объективно проверяемые результаты сведены в таблицу 18.

Таблица 18 – Результаты функционального тестирования на наборе из пяти композиций

Композиция	Тональность (уверенность)	Темп, BPM	Размер	Жанр (основной)	Эмоция: квадрант (V/A)
Районы-Кварталы	ля минор (0,73)	88,2	4/4	rock	happy (0,50 / 0,59)
Облака	ре мажор (0,83)	125,0	4/4	electronic	happy (0,54 / 0,61)
Белая ночь	фа-диез мажор (0,81)	130,4	4/4	rock	happy (0,72 / 0,60)
Ничего на свете лучше нету	до мажор (0,76)	125,0	4/4	children's	happy (0,65 / 0,59)
Невеста Полоза	ре мажор (0,80)	107,1	6/8	pop	sad (0,48 / 0,36)

Анализ результатов.

Тональность. Определена правдоподобно для всех пяти треков, для «Районы-Кварталы» результат (ля минор) совпадает с известной тональностью композиции. Уверенность модели находится в диапазоне 0,73–0,83, что соответствует устойчивому, но не безусловному распознаванию.

Темп и размер. Тактовый размер определен верно для всех треков: 4/4 для четырех композиций (с максимальной уверенностью) и составной 6/8 для «Невеста Полоза» (признак `is_compound_meter` установлен, конвенция доли – восьмая). Корректное определение 6/8 подтверждает работоспособность доработанной логики метра (подраздел 3.7). Темп для треков в 4/4 устойчив (стабильность сетки долей около 0,99). Для трека в 6/8 итоговый темп составил 107 ударов в минуту при сыром значении 214 (темп приведен в целевой диапазон делением пополам), пониженная уверенность темпа на 6/8 ожидаема и обсуждается в подразделе 5.6.

Жанр. Результат правдоподобен, но не идеален на русском репертуаре. «Бременские музыканты» уверенно отнесены к категории «children's» (детская музыка), «Районы-Кварталы» – к «rock», в то же время отнесение «Облака» к «electronic» спорно. Причина – таксономия Discogs (подраздел 2.4), в которой нет адекватных категорий для русской эстрады и шансона, это ограничение отмечено в подразделе 5.6.

Эмоция. Для четырех энергичных композиций определен квадрант «happy» (положительная валентность, высокая активация), для лиричной «Невеста Полоза» – «sad» (валентность около 0,48, пониженная активация 0,36), что согласуется с характером записи.

Время и ресурсы. Среднее время полного анализа составило около 57 секунд на трек (диапазон 34–79 с в зависимости от длительности композиции). Основное время приходится на разделение источников (16–48 с, выполняется на центральном процессоре, так как CUDA-провайдер библиотеки `onnxruntime` в данном образе недоступен) и распознавание текста (около 10–12 с). Детерминированные алгоритмы слоговой деления и квантизации (ядро SRA) выполняются за 3–24 мс, что подтверждает их вычислительную дешевизну.

Пик потребления видеопамати стабильно составил около 2,4-2,5 ГБ – значительно ниже бюджета одной видеокарты (12 ГБ): в активном профиле распознавание (whisperx-medium) и форсированное выравнивание (wav2vec 2.0) выполнялись на графическом ускорителе, а разделение источников – на центральном процессоре. Подробные тайминги по этапам приведены в таблице 19.

Таблица 19 – Время этапов конвейера по композициям набора, мс

Композиция	Загрузка	Разделение	Доли	Тональность	VAD	Распозн.+выравни.	Слоги+кв.	Жанр	Эмоция	Аккорды	Итого	Видеопамать, МБ
Районы-Кварталы	113	16605	984	63	601	11652	6	2305	981	637	33947	2459
Облака	616	44632	1680	216	4388	11935	24	4966	3217	3654	75326	2474
Белая ночь	568	32638	1237	140	2145	11243	16	3716	2337	2247	56287	2484
Ничего на свете...	759	48049	1644	210	3233	12321	23	5612	3242	3891	78983	2435
Невеста Полоза	294	20133	1174	102	1584	10072	3	3117	1450	1554	39484	2480

Таким образом, сервис устойчиво выполняет полный конвейер из шести анализов на разнохарактерных композициях: тональность и тактовый размер (включая составной 6/8) определяются верно, эмоция согласуется с характером записи, жанр правдоподобен с учетом ограничений таксономии, полный анализ укладывается в десятки секунд при пике видеопамати около 2,5 ГБ, а детерминированное ядро SRA практически не нагружает систему.

5.4 Качественная демонстрация слога-долевого выравнивания

Центральная прикладная задача сервиса – раскладка слогов вокального текста по долям такта (подраздел 3.5). Ее работа показана качественно: разбором автоматической раскладки на реальных треках и ее выверкой в редакторе ритмической сетки (подраздел 4.6).

Размер 4/4 («Районы-Кварталы»). На композиции в 4/4 автоматическая раскладка точна. Так, первая вокальная строка «Больше нечего ловить» распределяется по сетке естественно: слог «Боль» встает на сильную долю

такта (первая доля, позиция «оп») с отклонением +12 мс от идеального положения, «-ше» – на слабую позицию той же доли, «не-», «-че-», «-го» – на следующие доли. Подавляющее большинство слогов попадает на свои шестнадцатые с отклонением в пределах принятого допуска (± 80 мс), а уверенность раскладки превышает 0,9. Большие отклонения возникают лишь на быстрых и распетых слогах (например, «от-» с отклонением 113 мс при пониженной уверенности 0,67), однако и они остаются в пределах правильной доли и не порождают наложений. Каждому слогу детерминировано сопоставлены такт, доля, шестнадцатая, метка позиции, длительность в шестнадцатых и аккорд доли, например, гармония по тактам распознана как «Am – Am – E – D – E – Am – E – F ...», и эти аккорды привязаны к соответствующим слограм.

Размер 6/8 («Невеста Полоза»). Составной размер традиционно сложен для трекеров долей (подраздел 5.6), поэтому он выбран как проверочный случай. После доработки логики определения метра и регуляризации сетки (подраздел 3.7) раскладка на 6/8 работает корректно: размер распознан как составной (числитель 6, знаменатель 8), слоги встают на свои восьмые внутри такта, не накладываются друг на друга и не «перетекают» в соседний такт. Это подтверждает, что предложенные детерминированные правила (автоматический выбор дробности округления, разведение наложившихся слогов, исправление сетки) решают задачу не только для простого размера, но и для составного.

Роль редактора. Там, где автоматическая раскладка расходится со слуховым восприятием (распевы, мягкие вокальные приступы, затакт), пользователь исправляет положение слога перетаскиванием в редакторе ритмической сетки, при этом все зависимые поля слога (такт, доля, шестнадцатая, метка позиции, время) пересчитываются автоматически, а отклонение обнуляется. Тем самым редактор служит как инструментом наглядного разбора песни, так и средством подготовки эталонной разметки –

той самой, что необходима для будущей количественной оценки по метрике SRA-Accuracy.

Таким образом, на размере 4/4 автоматическая раскладка слогов по долям точна (отклонения в пределах допуска, редкие выбросы не выходят за границу доли), а на составном размере 6/8 после доработки логики метра она также работает корректно, типовые расхождения (распевы, затакт) устраняются в редакторе ручной коррекции. Качественно метод подтверждает свою работоспособность, количественная оценка отнесена к дальнейшей работе.

5.5 Сравнение движков выравнивания: WhisperX и MFA

В сервисе предусмотрены два движка выравнивания текста (подраздел 2.6): основной – WhisperX (распознавание Whisper + форсированное выравнивание wav2vec 2.0) [3, 21] – и альтернативный – Montreal Forced Aligner с моделью russian_mfa [20]. Их строгое количественное сравнение здесь не проводится, вместо этого приводится качественное сопоставление, опирающееся на механизм их совместного использования.

MFA дает фонемные границы (более детальные, чем посимвольные метки WhisperX), что в принципе полезно для точного определения момента начала слога. Однако его акустическая модель обучена на речи, и на пении – особенно на распевках, мелизмах и мягких вокальных приступах – она «плывет», смещая границы. Поэтому в сервисе MFA подключен не как замена, а как уточняющий движок с защитным ограничением: его границы принимаются только тогда, когда они согласуются с результатом WhisperX в пределах допуска ± 80 мс, при расхождении (дрейфе) система откатывается на метки WhisperX. По построению такая схема **не ухудшает** результат относительно базового движка: в худшем случае используется WhisperX, в лучшем – уточненные фонемные границы MFA там, где оба движка согласны.

Отсюда осторожная формулировка итога: MFA «не хуже WhisperX и фонемно точнее там, где согласуется», – но утверждать, что он точнее в целом, без измерения на размеченных данных нельзя. Кроме того, MFA заметно

медленнее, поэтому в качестве движка по умолчанию выбран WhisperX, работающий на графическом ускорителе.

Таким образом, движки соотносятся так: WhisperX – быстрый основной движок с пословными и посимвольными метками, MFA – более детальный, но медленный и чувствительный к пению, применяемый с защитным гейтом ± 80 мс, который гарантирует, что его подключение не ухудшает результат. Количественное сравнение отнесено к дальнейшей работе.

5.6 Известные ограничения

Проведенная оценка и опыт реализации позволяют очертить границы применимости сервиса.

Тактовый размер. Размер 4/4 определяется надежно. Составной 6/8 после доработки логики метра определяется корректно на проверенных записях, но остается сложным случаем: трекер долей может путать 6/8 с 3/4 и 4/4, а уверенность и стабильность темпа на составном размере ниже (для «Невеста Полоза» уверенность темпа около 0,38). Размеры 5/4, 7/8 и иные нерегулярные метры не поддерживаются.

Жанр. Классификатор обучен на таксономии Discogs, в которой отсутствуют адекватные категории для русской эстрады и шансона, такие композиции относятся к ближайшим доступным группам (pop, electronic, world), что снижает интерпретируемость результата на русском репертуаре.

Сложный вокал. Распевы (один слог на нескольких нотах), мелизмы, синкопы и затакт остаются трудными для автоматической раскладки и могут требовать ручной коррекции, речевая природа моделей распознавания и выравнивания плохо описывает удержания и глиссандо.

Языковая зависимость. Точное посимвольное выравнивание опирается на дообученную модель wav2vec 2.0 для конкретного языка, для языков без такой модели качество выравнивания снизится.

Гармония и тональность. Качество распознавания аккордов (модель BTC) в работе не измерялось на размеченных данных – это отмечено как направление дальнейшей оценки, кроме того, метод оценки тональности

усредняет гармонию по всей композиции и не отслеживает модуляции, а rubato (плавающий темп) ухудшает трекинг долей.

Платформенные ограничения. Модели Essentia доступны практически только в среде Linux/Docker, что ограничивает запуск сервиса на других платформах без контейнеризации, кроме того, часть моделей (Essentia/MTG) распространяется под некоммерческой лицензией (подраздел 2.8).

Количественная оценка выравнивания. В настоящей работе качество слога-долевого выравнивания показано качественно, его количественная оценка по метрике SRA-Ассурасу – первоочередное направление продолжения работы.

В итоге сервис надежно работает в типовых условиях (песни в 4/4 и 6/8 с выраженным вокалом на русском языке), а его ограничения связаны с редкими размерами, особенностями таксономии жанров, сложным вокалом, языковой и платформенной зависимостью, количественная оценка выравнивания отнесена к дальнейшей работе.

5.7 Выводы по разделу

Принята трехуровневая методика оценки: качество моделей – по публичным бенчмаркам из статей, работоспособность сервиса – функциональным тестом, слога-долевого выравнивание – качественной демонстрацией.

Качество выбранных моделей по опубликованным данным соответствует современному уровню по каждой подзадаче (подраздел 5.2), приведенные значения – литературные, с пометкой атрибуции к источникам, а не собственные замеры.

Функциональное тестирование на пяти разнохарактерных композициях подтвердило работоспособность сервиса: тональность и тактовый размер (включая составной 6/8) определяются верно, эмоция согласуется с характером записи, жанр правдоподобен с учетом ограничений таксономии, среднее время анализа – около 57 с на трек при пике видеопамати около 2,5 ГБ, а детерминированное ядро SRA выполняется за единицы миллисекунд.

Качественная демонстрация показала, что автоматическая раскладка слогов по долям точна на размере 4/4 и корректна на составном 6/8 после доработки логики метра, типовые расхождения устраняются в редакторе ручной коррекции, который одновременно служит средством подготовки эталонной разметки.

Сравнение движков выравнивания показало, что подключение MFA с защитным гейтом ± 80 мс по построению не ухудшает результат относительно WhisperX, строгое количественное сравнение отнесено к дальнейшей работе.

Очерчены границы применимости сервиса, количественная оценка слога- долевого выравнивания по метрике SRA-Accuracy вынесена в направления дальнейшей работы.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы исследованы современные подходы прикладного искусственного интеллекта к анализу музыкальных произведений и на их основе разработан сервис комплексного анализа песни. Поставленная цель достигнута, все сформулированные во введении задачи решены.

По задачам работы получены следующие результаты.

Проведен анализ предметной области извлечения музыкальной информации (MIR): рассмотрены основные задачи анализа музыки, способы представления аудиосигнала (спектрограмма, мел-спектрограмма, MFCC, хромограмма, постоянное-Q преобразование) и эволюция нейросетевых архитектур (сверточные сети, сверточно-рекуррентные сети, трансформеры, модели-основания, обучаемые без учителя). Выполнен обзор существующих сервисов-аналогов, который показал, что они решают задачи анализа музыки разрозненно и не предоставляют целостного инструмента, объединяющего музыкально-теоретический разбор песни с раскладкой слогов и аккордов по ритмической сетке такта, это определило нишу работы.

Для каждой подзадачи сервиса выполнен сравнительный анализ доступных моделей и обоснован выбор по совокупности критериев (качество по публичным данным, вычислительная стоимость, наличие открытых весов, поддержка русского языка, интегрируемость). В результате выбраны: для разделения источников – BS-RoFormer, для трекинга долей – Beat This! (без постобработки динамической байесовской сетью), для определения тональности – Essentia KeyExtractor с резервом на методе Крумханслы–Шмуклера, для детекции вокала – silero-vad, для распознавания и выравнивания текста – WhisperX (распознавание Whisper, форсированное выравнивание wav2vec 2.0) с альтернативным движком Montreal Forced Aligner, для слоговой деления – rpyphen с правилами, для классификации жанра – Essentia genre_discogs400, для распознавания эмоций – Essentia emomusic, для

распознавания аккордов – BTC. Темп и тактовый размер выводятся из положения долей детерминированно.

Формализована задача выравнивания слогов с долями такта (Syllable-Rhythm Alignment, SRA): для каждого спетого слога определяются время, доля такта, позиция внутри доли и признак сильной доли, а качество решения оценивается введенной метрикой SRA-Accuracy(τ). Предложен детерминированный объяснимый алгоритм решения задачи – раскладка слогов по клеткам ритмической сетки (с точностью до шестнадцатой доли): с автоматическим выбором дробности округления, разведением слогов, попавших в одну клетку, по соседним клеткам с наименьшим суммарным сдвигом и исправлением ритмической сетки, что обеспечивает корректную работу в том числе на составном размере 6/8.

Спроектирована архитектура многомодульного конвейера анализа из десяти измеримых стадий, рассчитанная на бюджет одной видеокарты: с ленивой загрузкой и выгрузкой моделей, профилями моделей под разное оборудование и честной деградацией при сбое отдельных модулей. Обоснован выбор модульной (а не сквозной) организации конвейера: интерпретируемость, заменяемость отдельных моделей и детерминированный постпроцессинг.

Сервис реализован как клиент-серверное приложение: серверная часть на Python/FastAPI с модулями анализа, асинхронным воркером и оркестратором, программный интерфейс (REST) с очередью задач, кэшированием по хэшу файла, выборочным анализом и трансляцией прогресса, пользовательский интерфейс на React с наглядными карточками результата и интерактивной ритмической сеткой, включая режим ручной коррекции раскладки слогов, контейнерное развертывание в Docker. Центральная прикладная функция – наглядное распределение слогов по долям такта с привязкой аккордов – реализована в редакторе ритмической сетки.

Проведена экспериментальная оценка. Качество выбранных моделей охарактеризовано по опубликованным данным на общепринятых наборах (с

поясечной атрибуцией к источникам), что подтверждает их соответствие современному уровню. Работоспособность сервиса проверена реальным прогоном в контейнере на наборе из пяти разнохарактерных композиций: тональность и тактовый размер (включая составной 6/8) определены верно, эмоциональная окраска согласуется с характером записей, жанр правдоподобен с учетом ограничений таксономии, среднее время полного анализа составило около 57 секунд на композицию при пике потребления видеопамати около 2,5 ГБ, а детерминированное ядро выравнивания слогов выполняется за единицы миллисекунд. Качество слога-долевого выравнивания продемонстрировано качественно: на размере 4/4 раскладка точна, на составном 6/8 после доработки логики метра работает корректно (слоги встают на свои доли, не накладываются и не уходят в чужой такт).

Научная новизна работы заключается: в формализации задачи слога-долевого выравнивания и метрике SRA-Ассурасу(τ) с описанными свойствами, в детерминированном объяснимом методе раскладки слогов по ритмической сетке (с автоматическим выбором дробности, разведением слогов, попавших в одну клетку, и исправлением сетки) и детерминированной привязке «аккорд – доля – слог», в архитектуре многомодульного MIR-конвейера под бюджет одной видеокарты с честной деградацией и сменными движками выравнивания.

Практическим результатом работы является действующий сервис комплексного анализа песни, пригодный для применения музыкантами, педагогами и учениками музыкальных школ при разучивании и разборе произведений. Реализованный редактор ритмической сетки, помимо прикладной функции, служит инструментом подготовки эталонной разметки, необходимой для будущей количественной оценки качества выравнивания.

Направления дальнейшей работы. Развитие работы целесообразно вести по следующим направлениям:

- сбор размеченного корпуса русскоязычного вокала с эталонной слога-долевой разметкой и проведение количественной оценки

предложенного метода по метрике SRA-Accuracy(τ), включая анализ чувствительности к допуску и сравнение движков выравнивания на размеченных данных,

- мультимодальное распознавание эмоций, объединяющее аудио- и текстовые признаки (например, представления MERT и языковой модели RuBERT с механизмом внимания), для повышения точности по оси валентности, которая хуже предсказывается по одному лишь аудио,

- уточнение временных границ слогов по характеристикам атаки и спектральным признакам для более точной работы на распевах и мягких вокальных приступах,

- дообучение моделей распознавания и выравнивания под особенности русского пения и применение специализированных певческих выравнивателей по мере появления русскоязычных певческих корпусов,

- расширение поддержки тактовых размеров (5/4, 7/8 и иные нерегулярные метры) и учет модуляций тональности,

- количественная оценка качества распознавания аккордов на размеченных данных,

- замена моделей с некоммерческой лицензией на решения с разрешительными лицензиями в случае коммерческого применения, а также развитие пользовательского интерфейса, включая мобильную версию.

Таким образом, в работе исследованы современные методы прикладного искусственного интеллекта для анализа музыкальных произведений, формализована задача слога-долевого выравнивания и предложен детерминированный метод ее решения, работоспособность которого подтверждена качественно, а также реализован действующий сервис комплексного анализа песни, что в совокупности подтверждает достижение поставленной цели.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абросимов К. И., Рыбин С. В. Music Information Retrieval: современные задачи и технологии // Компьютерные инструменты в образовании. — 2023. — № 1. — С. 74–95.
2. Müller M. Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications. — Cham: Springer, 2015. — 487 p.
3. Baevski A., Zhou H., Mohamed A., Auli M. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations // Advances in Neural Information Processing Systems (NeurIPS). — 2020. — Vol. 33. — P. 12449–12460.
4. Li Y., Yuan R., Zhang G. [et al.] MERT: Acoustic Music Understanding Model with Large-Scale Self-Supervised Training // Proc. 12th Int. Conf. on Learning Representations (ICLR). — Vienna, 2024. — arXiv:2306.00107.
5. Чапаев И. А. Современные методы представления звука и архитектуры нейронных сетей для интеллектуального анализа музыкальных произведений // Вестник науки. — 2025. — Т. 1, № 11 (92).
6. Иванова Н. А. Анализ музыкальных произведений. Часть 1: учеб. пособие. — Саратов, 2018. — 50 с.
7. Никитин Н. А., Розалиев В. Л., Орлова Ю. А., Заболеева-Зотова А. В. Применение методов машинного обучения для задачи генерации музыкальных композиций // Прикаспийский журнал: управление и высокие технологии. — 2018. — № 2 (42). — С. 140–149.
8. Музыкальная информатика: учеб. пособие / Е. Н. Бажукова, И. Б. Горбунова, М. С. Заливадный, С. В. Чибирёв. — СПб.: Планета музыки, 2017. — 224 с.
9. Goodfellow I., Bengio Y., Courville A. Deep Learning. — Cambridge, MA: MIT Press, 2016. — 800 p.
10. Kong Q., Cao Y., Iqbal T. [et al.] PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition // IEEE/ACM Transactions

on Audio, Speech, and Language Processing. — 2020. — Vol. 28. — P. 2880–2894. — DOI: 10.1109/TASLP.2020.3030497.

11. Gemmeke J. F., Ellis D. P. W., Freedman D. [et al.] Audio Set: An Ontology and Human-Labeled Dataset for Audio Events // Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). — New Orleans, 2017. — P. 776–780. — DOI: 10.1109/ICASSP.2017.7952261.

12. Choi K., Fazekas G., Sandler M., Cho K. Convolutional Recurrent Neural Networks for Music Classification // Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). — New Orleans, 2017. — P. 2392–2396. — DOI: 10.1109/ICASSP.2017.7952585.

13. Vaswani A., Shazeer N., Parmar N. [et al.] Attention Is All You Need // Advances in Neural Information Processing Systems (NeurIPS). — 2017. — Vol. 30. — P. 5998–6008.

14. Gong Y., Chung Y.-A., Glass J. AST: Audio Spectrogram Transformer // Proc. Interspeech. — 2021. — P. 571–575. — arXiv:2104.01778.

15. Hsu W.-N., Bolte B., Tsai Y.-H. [et al.] HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units // IEEE/ACM Transactions on Audio, Speech, and Language Processing. — 2021. — Vol. 29. — P. 3451–3460. — DOI: 10.1109/TASLP.2021.3122291.

16. Wang A. An Industrial-Strength Audio Search Algorithm // Proc. 4th Int. Conf. on Music Information Retrieval (ISMIR). — Baltimore, 2003. — P. 7–13.

17. SoundHound [Электронный ресурс]. — URL: <https://www.soundhound.com/> (дата обращения: 20.05.2026).

18. The Echo Nest [Электронный ресурс]. — URL: <https://the.echonest.com/> (дата обращения: 20.05.2026).

19. Bogdanov D., Wack N., Gómez E. [et al.] ESSENTIA: an Audio Analysis Library for Music Information Retrieval // Proc. 14th Int. Society for Music Information Retrieval Conf. (ISMIR). — Curitiba, 2013. — P. 493–498 ; McFee B., Raffel C., Liang D. [et al.] librosa: Audio and Music Signal Analysis in Python // Proc. 14th Python in Science Conf. (SciPy). — Austin, 2015. — P. 18–25.

20. McAuliffe M., Socolof M., Mihuc S. [et al.] Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi // Proc. Interspeech. — Stockholm, 2017. — P. 498–502. — DOI: 10.21437/Interspeech.2017-1386.
21. Bain M., Huh J., Han T., Zisserman A. WhisperX: Time-Accurate Speech Transcription of Long-Form Audio // Proc. Interspeech. — 2023. — P. 4489–4493. — arXiv:2303.00747.
22. Foscarin F., Schlüter J., Widmer G. Beat This! Accurate Beat Tracking Without DBN Postprocessing // Proc. Int. Society for Music Information Retrieval Conf. (ISMIR). — 2024. — arXiv:2407.21658.
23. Guo W., Zhang Y., Pan C. [et al.] STARS: A Unified Framework for Singing Transcription, Alignment, and Refined Style Annotation. — 2025. — arXiv:2507.06670.
24. Hennequin R., Khlif A., Voituret F., Moussallam M. Spleeter: A Fast and Efficient Music Source Separation Tool with Pre-trained Models // Journal of Open Source Software. — 2020. — Vol. 5, № 50. — Art. 2154. — DOI: 10.21105/joss.02154.
25. Rouard S., Massa F., Défossez A. Hybrid Transformers for Music Source Separation // Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). — 2023. — arXiv:2211.08553.
26. Lu W.-T., Wang J.-C., Kong Q., Hung Y.-N. Music Source Separation with Band-Split RoPE Transformer // Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). — 2024. — arXiv:2309.02612.
27. Rafii Z., Liutkus A., Stöter F.-R., Mimilakis S. I., Bittner R. MUSDB18 — a Corpus for Music Separation. — 2017. — DOI: 10.5281/zenodo.1117372.
28. Wang J.-C., Lu W.-T., Won M. Mel-Band RoFormer for Music Source Separation. — 2023. — arXiv:2310.01809.
29. Le Roux J., Wisdom S., Erdogan H., Hershey J. R. SDR — Half-Baked or Well Done? // Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). — Brighton, 2019. — P. 626–630. — DOI: 10.1109/ICASSP.2019.8683855.

30. Böck S., Korzeniowski F., Schlüter J., Krebs F., Widmer G. madmom: A New Python Audio and Music Signal Processing Library // Proc. 24th ACM Int. Conf. on Multimedia. — Amsterdam, 2016. — P. 1174–1178. — DOI: 10.1145/2964284.2973795.
31. Heydari M., Cwitkowitz F., Duan Z. BeatNet: CRNN and Particle Filtering for Online Joint Beat, Downbeat and Meter Tracking // Proc. 22nd Int. Society for Music Information Retrieval Conf. (ISMIR). — 2021. — arXiv:2108.03576.
32. Zhao J., Xia G., Wang Y. Beat Transformer: Demixed Beat and Downbeat Tracking with Dilated Self-Attention // Proc. 23rd Int. Society for Music Information Retrieval Conf. (ISMIR). — 2022. — arXiv:2209.07140.
33. Raffel C., McFee B., Humphrey E. J. [et al.] mir_eval: A Transparent Implementation of Common MIR Metrics // Proc. 15th Int. Society for Music Information Retrieval Conf. (ISMIR). — Taipei, 2014. — P. 367–372.
34. Krumhansl C. L., Kessler E. J. Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys // Psychological Review. — 1982. — Vol. 89, № 4. — P. 334–368. — DOI: 10.1037/0033-295X.89.4.334.
35. Schreiber H., Müller M. Musical Tempo and Key Estimation Using Convolutional Neural Networks with Directional Filters // Proc. 16th Sound and Music Computing Conf. (SMC). — Málaga, 2019. — P. 47–54. — arXiv:1903.10839.
36. Tzanetakis G., Cook P. Musical Genre Classification of Audio Signals // IEEE Transactions on Speech and Audio Processing. — 2002. — Vol. 10, № 5. — P. 293–302. — DOI: 10.1109/TSA.2002.800560.
37. Defferrard M., Benzi K., Vandergheynst P., Bresson X. FMA: A Dataset for Music Analysis // Proc. 18th Int. Society for Music Information Retrieval Conf. (ISMIR). — Suzhou, 2017. — P. 316–323. — arXiv:1612.01840.

38. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). — Las Vegas, 2016. — P. 770–778.
39. Russell J. A. A Circumplex Model of Affect // Journal of Personality and Social Psychology. — 1980. — Vol. 39, № 6. — P. 1161–1178.
40. Aljanaki A., Yang Y.-H., Soleymani M. Developing a Benchmark for Emotional Analysis of Music // PLOS ONE. — 2017. — Vol. 12, № 3. — Art. e0173392. — DOI: 10.1371/journal.pone.0173392.
41. Pons J., Serra X. musicnn: Pre-trained Convolutional Neural Networks for Music Audio Tagging. — 2019. — arXiv:1909.06654.
42. Radford A., Kim J. W., Xu T. [et al.] Robust Speech Recognition via Large-Scale Weak Supervision (Whisper). — 2022. — arXiv:2212.04356.
43. SOFA: Singing-Oriented Forced Aligner [Электронный ресурс]. — URL: <https://github.com/qiuqiao/SOFA> (дата обращения: 20.05.2026).
44. Wu S., He J., Yuan R. [et al.] SongTrans: An Unified Song Transcription and Alignment Method for Lyrics and Notes. — 2024. — arXiv:2409.14619.
45. Mauch M., Dixon S. Approximate Note Transcription for the Improved Identification of Difficult Chords // Proc. 11th Int. Society for Music Information Retrieval Conf. (ISMIR). — Utrecht, 2010. — P. 135–140.
46. Park J., Choi K., Jeon S. [et al.] A Bi-directional Transformer for Musical Chord Recognition // Proc. 20th Int. Society for Music Information Retrieval Conf. (ISMIR). — Delft, 2019. — P. 620–627. — arXiv:1907.02698.
47. Davies M. E. P., Degara N., Plumbley M. D. Evaluation Methods for Musical Audio Beat Tracking Algorithms. — Technical Report C4DM-TR-09-06. — London: Centre for Digital Music, Queen Mary University of London, 2009.
48. Honing H. Structure and Interpretation of Rhythm and Timing // Tijdschrift voor Muziektheorie (Dutch Journal of Music Theory). — 2002. — Vol. 7, № 3. — P. 227–232.

49. Repp B. H. Sensorimotor Synchronization: A Review of the Tapping Literature // Psychonomic Bulletin & Review. — 2005. — Vol. 12, № 6. — P. 969–992. — DOI: 10.3758/BF03206433.

50. Grosman J. Fine-tuned XLSR-53 Large Model for Speech Recognition in Russian (wav2vec2-large-xlsr-53-russian) [Электронный ресурс]. — Hugging Face, 2021. — URL: <https://huggingface.co/jonatasgrosman/wav2vec2-large-xlsr-53-russian> (дата обращения: 20.05.2026).

ПРИЛОЖЕНИЕ А

(рекомендуемое)

Интеллектуальный анализ музыкальных произведений

Иллюстрационные материалы

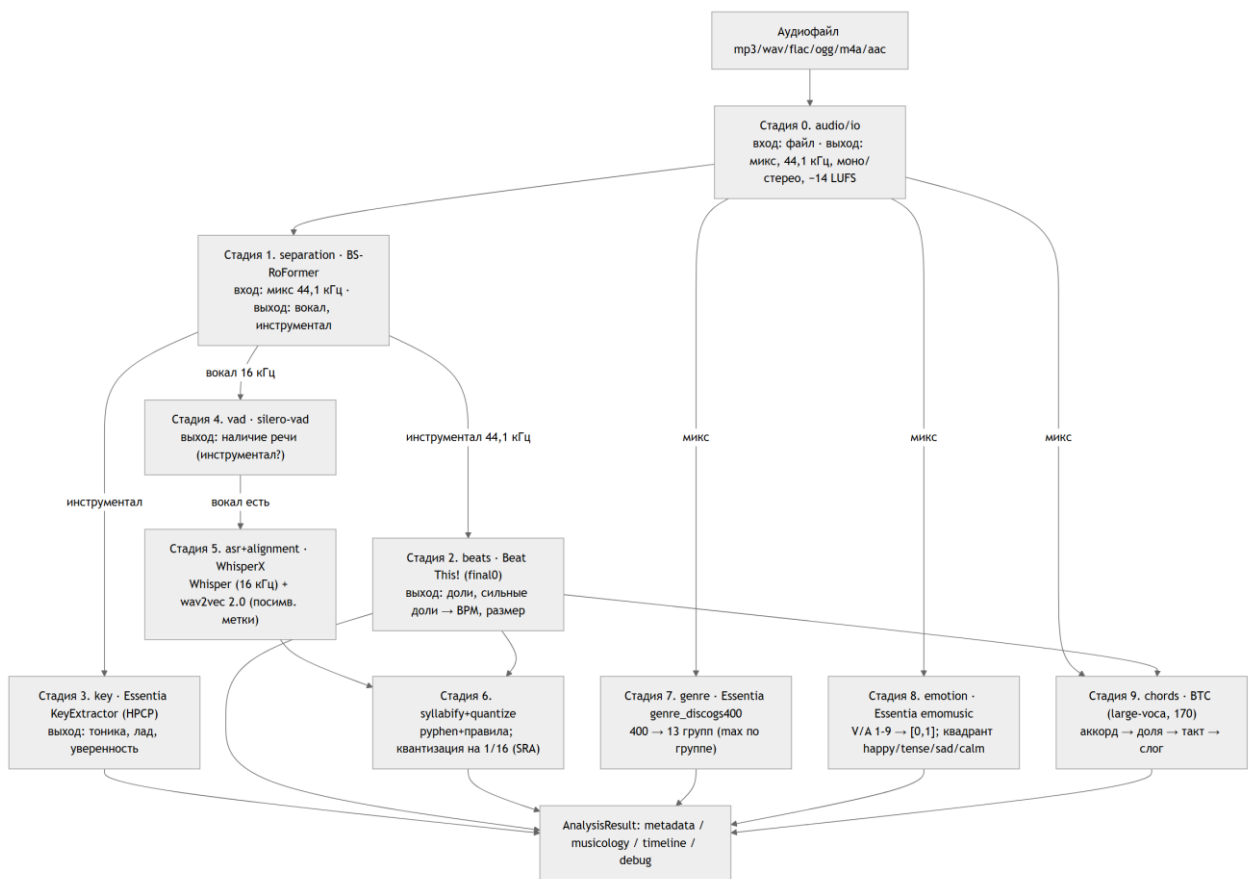


Рисунок 15 – Детализированная схема конвейера анализа: десять стадий, источник звука и частоты дискретизации

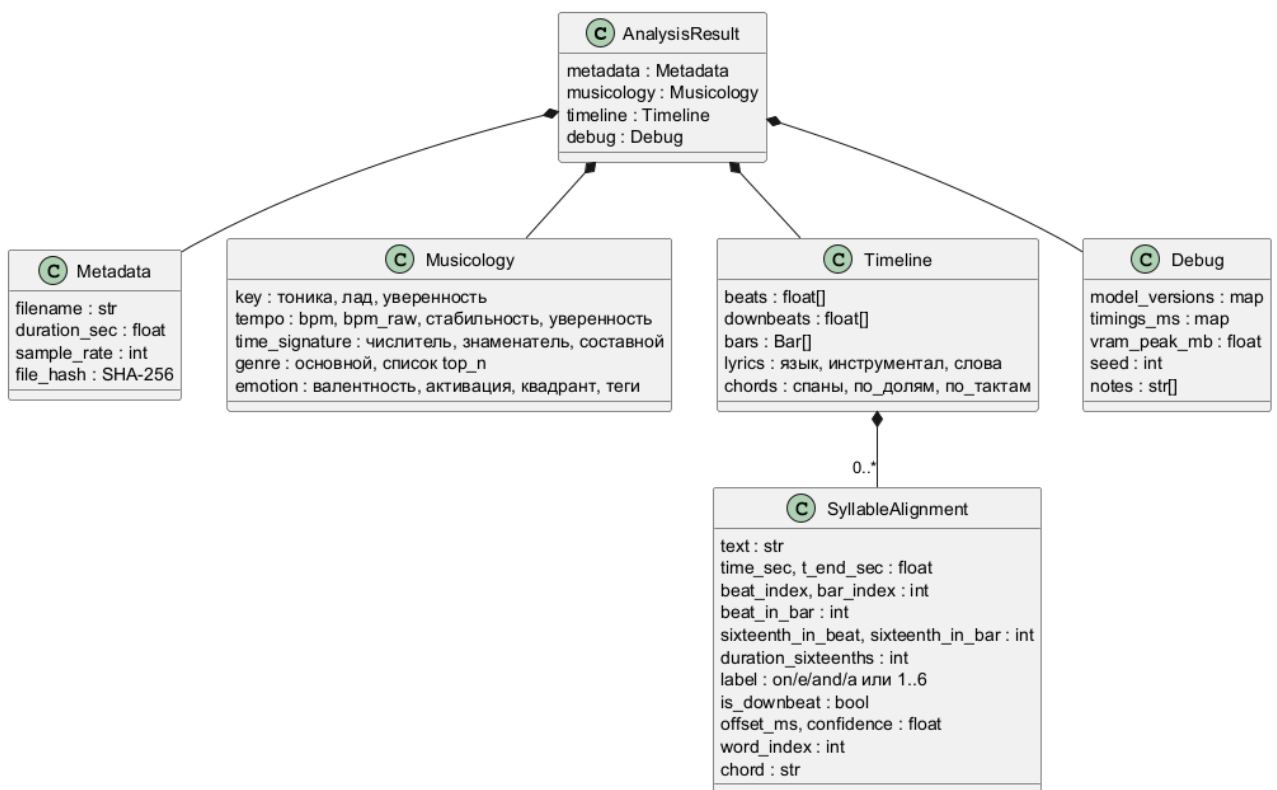


Рисунок 16 – Диаграмма классов модели результата анализа (AnalysisResult)

ПРИЛОЖЕНИЕ Б

(обязательное)

Интеллектуальный анализ музыкальных произведений

Листинг программы

В приложении приведены ключевые фрагменты исходного кода сервиса, иллюстрирующие детерминированные алгоритмы, описанные в разделах 2 и 3. Код приведен на языке Python. Полный исходный текст всех модулей размещен в репозитории проекта на GitHub <https://github.com/ch1llay/IntellectualAnalysisOfMusicalCompositions> ^{здесь} показаны только наиболее существенные для понимания работы фрагменты.

Файл `alignment/time_signature.py`

```
def derive_bpm(beats: np.ndarray) -> TempoResult:
    """Оценить BPM из массива времен долей (с): медиана IBI,
    нормализация
    темпа, стабильность IBI и confidence со штрафом за коррекцию
    темпа."""
    if beats is None or len(beats) < 2:
        return TempoResult(bpm=0.0, confidence=0.0,
        stability=0.0)
    intervals = np.diff(np.asarray(beats, dtype=np.float64))
    intervals = intervals[intervals > 1e-6]
    if len(intervals) == 0:
        return TempoResult(bpm=0.0, confidence=0.0,
        stability=0.0)
    median_ibi = float(np.median(intervals))
    raw_bpm = 60.0 / max(median_ibi, 1e-6)
    bpm = _normalize_octave(raw_bpm) #
    приведение в [60, 180]
    cv = float(np.std(intervals) / max(np.mean(intervals), 1e-
    6))
    stability = max(0.0, 1.0 - cv) #
    стабильность = 1 - CV
    confidence = stability * (0.95 if abs(bpm - raw_bpm) < 1e-6
    else 0.75)
    return TempoResult(bpm=round(bpm, 3), bpm_raw=round(raw_bpm,
    3),
    confidence=round(min(confidence, 1.0),
    3),
```

```
stability=round(stability, 3))
```

Файл alignment/quantize.py

```
    quantize_syllables_to_grid:

phase = (t - b_curr) / ibi                                # фаза онсета
внутри доли, [0, 1)

if phase < 0:
    phase = 0.0

nearest = round(phase * onset_K) / onset_K                # округление к
8-й или 16-й

sub_nat = int(round(nearest * K))                          # номер субдоли
(16-я в доле)

# Отклонение и уверенность считаются по ИСХОДНОЙ ближайшей
клетке - это честная

# оценка качества пения, а не последующий сдвиг клетки при
разведении наложений.

ideal_t = b_curr + (sub_nat / K) * ibi
offset_ms = (t - ideal_t) * 1000.0
conf = max(0.0, 1.0 - 2.0 * abs(offset_ms) / max(ib_i * 1000.0,
1e-6))

def _pav_l1_isotonic(a: list[int]) -> list[int]:
    """L1 (медианная) изотоническая регрессия методом Pool-
Adjacent-Violators.

    Возвращает неубывающую последовательность w, минимизирующую
sum|w_i - a_i|.

    Для блоков четного размера берется верхняя медиана (сдвиг
вперед при равенстве)."""
    blocks: list[tuple[list[int], int]] = []
    for x in a:
        members, val = [x], x
        while blocks and blocks[-1][1] > val:            # нарушение
монотонности - сливаем блоки
            prev_members, _ = blocks.pop()
            members = sorted(prev_members + members)
```

```

        val = members[len(members) // 2]                # верхняя
медиана блока
        blocks.append((members, val))
    out: list[int] = []
    for members, val in blocks:
        out.extend([val] * len(members))
    return out

```

Файл `models/base.py`

```

class ModelWrapper(ABC):
    """Жизненный цикл: ленивая загрузка, предсказание, выгрузка
    для освобождения GPU."""
    name: str = "model"

    def __init__(self, device: str = "cuda") -> None:
        self.device = device
        self._loaded = False

    @abstractmethod
    def _load(self) -> None: ...      # конкретная загрузка
    модели

    @abstractmethod
    def _unload(self) -> None: ...    # очистка внутренних ссылок

    def load(self) -> None:
        if self._loaded:
            return
        self._load()
        self._loaded = True

    def unload(self) -> None:
        if not self._loaded:
            return
        self._unload()
        self._loaded = False

```

```

import torch

if torch.cuda.is_available():
    torch.cuda.empty_cache()    # освобождаем видеопамять
gc.collect()

```

Файл `pipeline/orchestrator.py`

```

# --- тональность ---
if "key" in needed:
    with timed("key", timings):
        key = self._stage_key(instr_path)
    executed.append("key")

# --- распознавание и выравнивание текста (только при наличии
# вокала) ---
if "asr" in needed and not vad_result.is_instrumental:
    try:
        with timed("asr", timings):
            prov = self._get_provider().run(vocals_path,
            self.cfg.asr_default_language, lyrics=lyrics)
            words, language, spans = prov.words, prov.language,
            prov.spans
        executed.append("asr")
        except Exception as exc:          # честная деградация:
        модуль упал - считаем трек инструментальным
            logger.warning(f"lyrics alignment failed: {exc},
            treating as instrumental")
            notes.append(f"asr_failed: {exc}")
            vad_result = VadResult(0.0, 0.0, True, [])

```

Файл `api/events.py`

```

@router.get("/analyze/{task_id}/events", summary="SSE-поток
прогресса задачи")

async def stream_events(request: Request, task_id: str):
    storage = request.app.state.storage
    if (await storage.get_task(task_id)) is None:

```

```

        raise HTTPException(404, "task not found")

    async def event_generator() -> AsyncIterator[dict]:
        for ev in await _replay_state(storage, task_id):      #
отдать текущее состояние
            yield ev

            rec = await storage.get_task(task_id)

            if rec is None or rec.status in ("done", "error",
"cancelled"):

                return

            queue = await bus.subscribe(task_id)              #
подписка на события задачи

            while True:                                       #
трансляция прогресса по стадиям
                ev = await queue.get()

                yield ev

                if ev.get("type") in ("done", "error", "cancelled"):
                    break

    return EventSourceResponse(event_generator())

```

ПРИЛОЖЕНИЕ В

(обязательное)

Интеллектуальный анализ музыкальных произведений

Презентация



МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Интеллектуальный анализ музыкальных произведений

Направление 09.04.01 «Информатика и вычислительная техника»
Магистерская программа «Прикладной искусственный интеллект»

Автор: Чапаев И. А.

Научный руководитель Митрохина Н.Ю.

Пенза, 2026

Актуальность. Зачем анализировать музыку

Музыки очень много— десятки миллионов треков, и разобрать их вручную невозможно. Поэтому музыку всё чаще анализируют программно нейросетями. У такого анализа две большие цели

цель 1 — упорядочить

Организовать музыку

Определить жанр, настроение, тональность, тему и по ним каталогизировать треки, искать похожее, собирать подборки.

цель 2 — понять

Разобрать произведение

Распознать аккорды, текст и ритм конкретной песни, чтобы её было легче понять, разучить и сыграть.

Мне как музыканту ближе вторая цель, но в работе я постарался охватить обе.

Цель и задачи работы

Исследовательский вопрос: как с помощью нейросетей разносторонне анализировать музыку, какие для этого есть подходы и готовые модели— и можно ли собрать из них единый сервис.

ОБЪЕКТ

Методы и средства нейросетевого анализа музыки.

ПРЕДМЕТ

Конвейер анализа песни и метод выравнивания слогов.

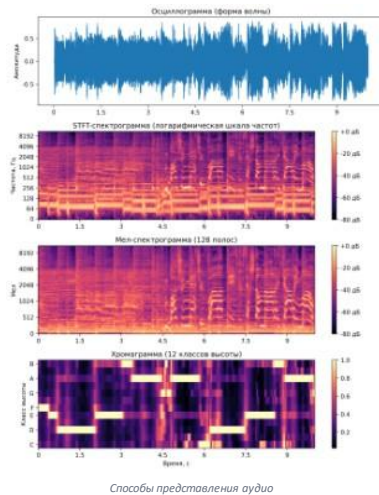
ЦЕЛЬ

Сервис комплексного анализа песни (8 характеристик).

Задачи

- 1 Обзор области MIR: задачи, представления, архитектуры
- 2 Сравнение и выбор моделей по подзадачам
- 3 Формализация задачи выравнивания слогов (SRA)
- 4 Проектирование архитектуры конвейера
- 5 Реализация сервиса: бэкенд, API, интерфейс
- 6 Экспериментальная проверка и апробация

Как нейросети «слышат» музыку



- **Звук → спектрограмма**
сигнал превращают в «картинку» частот во времени — её и анализирует нейросеть.
- **По ней работает модель**
сеть выдаёт нужную характеристику: жанр, аккорд, текст...
- **Эволюция архитектур**
от ручных признаков — к CNN, CRNN и трансформерам.



Признаки → CNN → CRNN → Transformer → большие предобученные модели

Вывод: сегодня почти всё решается готовыми предобученными моделями — обучать сети с нуля не нужно.

Что извлекает сервис: восемь характеристик

Восемь характеристик распадаются ровно на те же две цели:

для МЕДИАТЕКИ

Описать в целом

- жанр
- эмоция (круг Рассела)
- тональность
- темп и тактовый размер

→ каталогизация, фильтры, подборки похожего

для РАЗБОРА

Раскрыть изнутри

- распознанный текст
- аккорды по долям такта
- слоги по долям такта

→ понимание и разучивание произведения

Выбор моделей по подзадачам

Критерии: качество · скорость · лицензия · поддержка русского. Обучение с нуля не требуется; выбираем и собираем готовые модели

Подзадача	Выбранная модель	Тип / архитектура
Разделение источников	BS-RoFormer	Трансформер (band-split + RoPE)
Трекинг долей	Beat This!	CNN + трансформер
Темп и тактовый размер	Вывод из долей	Детерминированный
Тональность	Essentia KeyExtractor	Классический (профили)
Жанр	Essentia Discogs - EffNet	Свёрточная сеть (CNN)
Эмоции	Essentia musicnn / emomusic	Свёрточная сеть (CNN)
Текст и выравнивание	WhisperX (Whisper + wav2vec2)	Трансформер + SSL
Аккорды	BTC (170 классов)	Двунаправл. трансформер

Архитектура: конвейер из 10 этапов

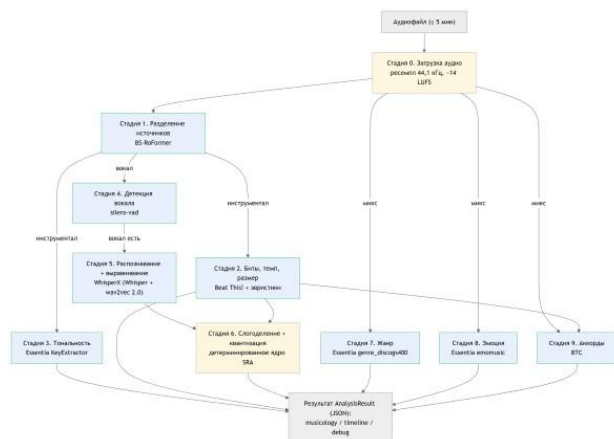


Схема конвейера анализа (10 стадий)

- **Конвейер, а не end-to-end**
каждый этап даёт проверяемый результат, любую модель легко заменить.
- **Свой источник звука на этапе**
инструментал — доли и тональность; вокал — текст; микс — жанр, эмоция, аккорды.
- **Бюджет одной видеокарты**
модели грузятся по очереди перед своим этапом и выгружаются после.
- **Честная деградация**
сбой модуля не рушит конвейер — характеристика помечается как отсутствующая.

Ритмическая раскладка слогов —зачем

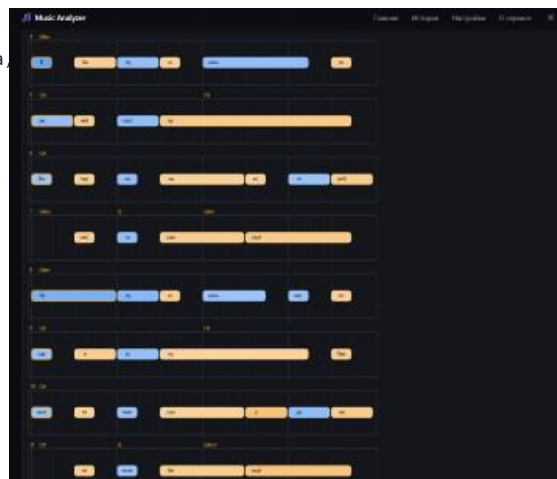
В пении ударный слог слова часто попадает не на сильную долю такта, а на слабую— между долями. Это синкопа: поётся не так, как читается.

ПРИМЕР

«*Ве-тер ка-ча-ет, то рОб-кий, то смЕ-лый*»

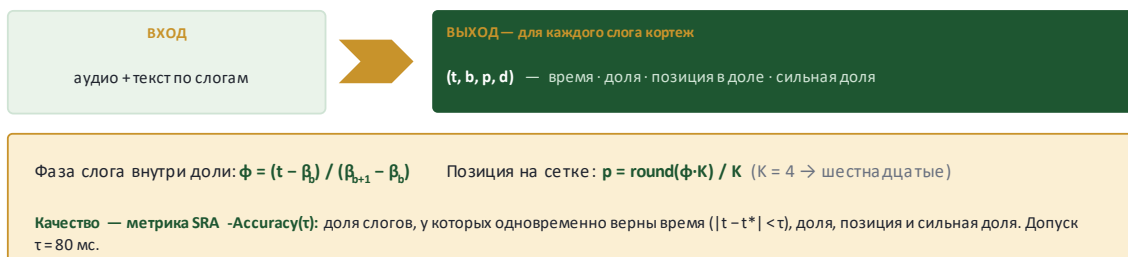
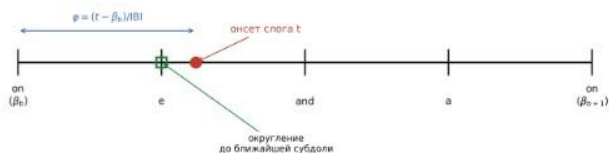
выделенные слоги ложатся на слабые доли такта

Такие места я разбираю с преподавателем по вокалу. На слух их уловить трудно, а наглядно нигде не показано захотелось сделать инструмент, который раскладывает слоги по долям и показывает это глазами.



Слоги, разложенные по долям такта в сервисе

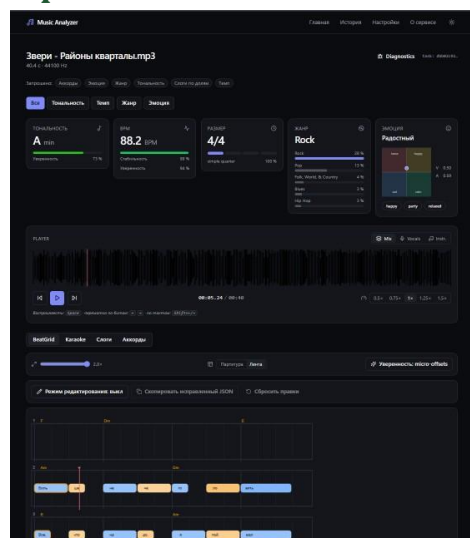
Выравнивание слогов (SRA): как формализовано



Геометрия одного шага: фаза ϕ онсета слога и округление до ближайшей субдоли (on / e / and / a)

Сервис в работе: разносторонний разбор

- **Один прогон — пять характеристик**
жанр, эмоция, тональность, темп и тактовый размер — карточками.
- **Эмоция — на круге Рассела**
по осям «валентность — активация», наглядно и сопоставимо.
- **Основа медиатеки**
по этим характеристикам анализа можно фильтровать и собирать подборки — история работает как небольшой каталог.



Страница результата

Сервис в работе: разбор для разучивания

Ритмическая сетка

текст разложен по слогам на долях такта, над тактами — аккорды.

Караоке-режим

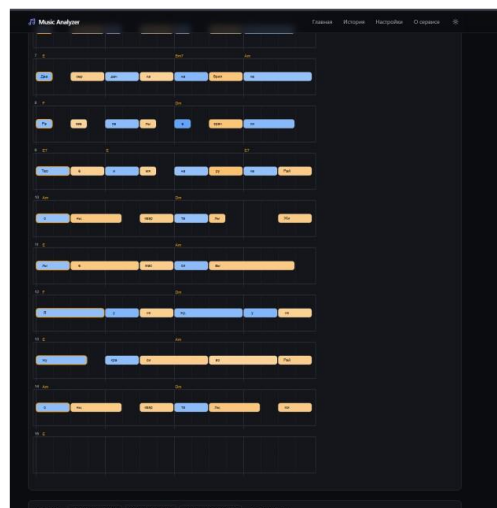
слова подсвечиваются синхронно со звучанием.

Ручная правка

слог можно перетащить в нужную клетку → готовая эталонная разметка.

Видно синкопы

сразу заметно, когда ударный слог стоит на слабой доле.



Ритмическая сетка со слогами по долям и аккордами

Экспериментальная проверка

Три уровня: модели — по публичным бенчмаркам · сервис — реальный прогон в Docker (5 песен) · раскладка слогов — качественно.

Композиция	Тональность	Темп	Размер	Жанр	Эмоция
Районы-Кварталы	ля минор	88	4/4	rock	happy
Облака	ре мажор	125	4/4	electronic	happy
Белая ночь	фа-диез мажор	130	4/4	rock	happy
Ничего на свете лучше нету	до мажор	125	4/4	children's	happy
Невеста Полоза	ре мажор	107	6/8	pop	sad

≈ 57 с

среднее время полного анализа

≈ 2,5 ГБ

пик видеопамати (бюджет — 12 ГБ)

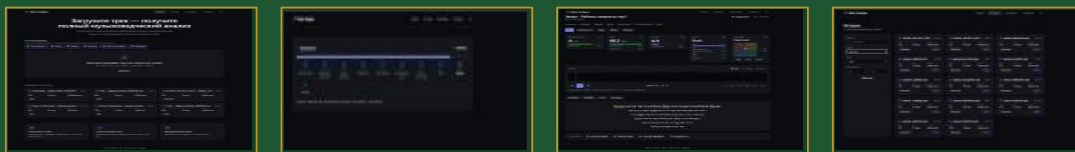
6/8

составной размер определён верно

Раскладка слогов: точная на размере 4/4, корректная на составном 6/8 — слоги встают на свои доли, не накладываются и не уходят в чужой такт.



Демонстрация работы сервиса



Заключение

Итоги

- Цель достигнута, все шесть задач решены.
- Сервис разносторонне анализирует песню: одна часть данных помогает упорядочивать музыку, другая — понять и разучить произведение.
- Научное ядро — формализация задачи SRA, метрика и детерминированный алгоритм раскладки слогов.

АПРОБАЦИЯ

Статья в журнале «Вестник науки» (2025); доклады в рамках научной исследовательской работы в ПГУ.

ДАЛЬНЕЙШАЯ РАБОТА

- размеченный корпус и оценка по метрике SRA - Ассигагу;
- мультимодальные эмоции (аудио + текст);
- поддержка нестандартных тактовых размеров.



Спасибо за внимание!

Чапаев И. А. · Пензенский государственный университет · 2026