Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «**Информатика и системы управления**»

КАФЕДРА «**Программное обеспечение ЭВМ и информационные технологии**»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

Метод реалистичного воспроизведения музыкальных партий с переносом стиля на основе конкатенативного синтеза звука

Студент группы **ИУ7-81Б** _____ **П. А. Шумнов**
(Подпись, дата) (И.О. Фамилия)

Руководитель ВКР _____ **М. В. Ульянов**
(Подпись, дата) (И.О. Фамилия)

Нормоконтролер _____ _____
(Подпись, дата) (И.О. Фамилия)

2023 г.

РЕФЕРАТ

Расчетно-пояснительная записка 80 с., 18 рис., 3 табл., 32 ист.

Целью работы является разработка метода реалистичного воспроизведения музыкальных партий с переносом стиля исполнения.

Для достижения поставленной цели необходимо решить следующие задачи:

— провести анализ предметной области;

— сформулировать постановку проблемы;

— рассмотреть существующие подходы к решению проблемы;

— разработать метод реалистичного воспроизведения музыкальных партий с переносом стиля;

— разработать программное обеспечение на основе предложенного метода;

— исследовать разработанный метод.

Поставленная цель достигнута: в ходе дипломной работы был разработан метод реалистичного воспроизведения музыкальных партий с переносом стиля на основе конкатенативного синтеза звука.

КЛЮЧЕВЫЕ СЛОВА

перенос стиля, конкатенативный синтез звука, автокодировщик, метод главных компонент, скрытое пространство.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Стиль исполнения в музыке является важным средством коммуникации между музыкантом и слушателем. Он способен передать эмоциональные состояния, выразить интенсивность и глубину музыкального произведения. Каждый стиль исполнения отличается своими особенностями и подходами, что делает его уникальным и значимым для искусства музыки.

Область компьютерного моделирования выразительного исполнения музыки активно развивается, о чем свидетельствует появление новых коммерческих продуктов и научных исследований [1].

Перенос стиля является еще одной областью, которая получила большую популярность, особенно в области обработки изображений. Задача заключается в создании нового изображения, которое объединяет содержание одного изображения с эстетическими характеристиками другого, сохраняя семантическую информацию исходного содержания, но принимая стилистические элементы и визуальные особенности другого изображения.

С помощью переноса стиля можно создавать уникальные и впечатляющие визуальные эффекты, а также экспериментировать с различными художественными стилями. Это предоставляет возможность для развития творческого процесса и расширения возможностей в области графического дизайна, визуального искусства и искусственного интеллекта.

Однако проблема переноса стиля не ограничивается только обработкой изображений. Она также находит свое применение в области музыки. Задача переноса стиля в музыке заключается в создании новых композиций, которые сочетают музыкальное содержание одной композиции с музыкальным стилем или характеристиками другой. Это позволяет артистам, музыкантам и композиторам экспериментировать с различными музыкальными стилями и создавать уникальные аранжировки, интерпретации и новые музыкальные произведения.

Целью работы является разработка метода реалистичного воспроизведе-

ния музыкальных партий с переносом стиля исполнения.

Для достижения поставленной цели потребуется решить следующие задачи:

— провести анализ предметной области;
— сформулировать постановку проблемы;
— рассмотреть существующие подходы к решению проблемы;
— разработать метод реалистичного воспроизведения музыкальных партий с переносом стиля;
— разработать программное обеспечение на основе предложенного метода;
— исследовать разработанный метод.

1 Аналитическая часть

В данном разделе проводится анализ предметной области. Проводится обзор существующих методов переноса стиля и определения стилевых особенностей исполнения.

Рассматриваются различные подходы к синтезу звука и проводится сравнение методов. Обосновывается выбор и приводится подробное описание метода синтеза звука.

На основе проведенного анализа формулируется постановка задачи и ограничения на применение разрабатываемого метода.

1.1 Понятие музыкальной партии и стиля исполнения

Партией в музыке называют составную часть многоголосного произведения, предназначенную для исполнения отдельным голосом или на отдельном музыкальном инструменте [2]. Она может быть задана различными способами: в виде нотной записи, оригинальной партитуры, последовательности MIDI-команд.

Обычно партитура описывает не только содержание произведения, но и некоторые параметры его исполнения. Тем не менее, у музыканта остается достаточно свободы для влияния на восприятие слушателя: фразировка, динамика, темп, артикуляция и многое другое.

Стиль исполнения музыки формируется под влиянием различных факторов, включая образование и опыт музыканта, его индивидуальное восприятие и интерпретацию произведения, а также культурный и исторический контекст. Каждый музыкант развивает собственный стиль, который может быть узнаваемым и характерным только для него.

В данной работе под стилем исполнения произведения будем понимать совокупность выразительных средств, характерных для данного музыкального инструмента, которые не были указаны в партитуре, но были использованы при исполнении произведения. Таким образом, стиль исполнения определяется, как

правило, двумя факторами: уникальной интерпретацией музыканта и историческим контекстом произведения.

1.2 Запись и хранение музыкальных произведений

Большую часть музыкальных произведений, за исключением экспериментальной музыки, можно записать при помощи современной 5-линейной тактовой нотации. Существует множество компьютерных программ-нотаторов (нотных редакторов), предназначенных для создания и хранения нотных текстов. Они активно используются композиторами и аранжировщиками, и одной из основных функций подобного ПО является возможность воспроизведения созданной партитуры.

Изначально стандарт MIDI, возникший в 1983, предназначался для обмена данными между электронными музыкальными инструментами [2]. Первое время MIDI позволял описывать лишь основные параметры нотации: выбор инструмента, высоту и длительность ноты. Сейчас помимо этого формат задает также настройку громкости и других акустических параметров каждой ноты, выбор тембра, темпа, тональности и др., с точной привязкой во времени.

Таким образом, современный MIDI-файл несет куда больше информации, чем нотная запись, и даже позволяет для некоторых инструментов полностью описать параметры исполнения произведения. Благодаря этому формат MIDI можно использовать и в качестве партитуры, и для передачи стиля исполнения.

1.3 Реалистичное воспроизведение музыкальных партий

Задача реалистичного воспроизведения музыкальных произведений включает в себя две основные подзадачи, которые важны для достижения максимально точного и естественного исполнения произведения: определение параметров исполнения и синтез звука.

Первая подзадача – определение параметров исполнения – заключается в моделировании воспроизведения с учетом музыкальных элементов выразительности, таких как динамика, артикуляция, фразировка, темп и других характеристик, которые придают уникальность и оригинальность исполнению.

Вторая подзадача – синтез звука – связана непосредственно с созданием звукового сигнала. Основными задачами синтеза в данном случае являются как можно более точное воссоздание тембра музыкального инструмента и передача всех заданных нюансов исполнения.

Определение параметров исполнения является основой для синтеза звука, так как они определяют характер и выразительность музыкальной партии. В свою очередь, синтез звука позволяет воплотить заданные параметры исполнения в аудио-формате, создавая впечатление реального исполнения музыканта. Оба аспекта вместе способствуют достижению реалистичного и качественного воспроизведения музыкальных произведений.

1.4 Распознавание музыкального стиля

В рамках проведенного исследования было выявлено, что область определения стилевых особенностей исполнения в музыке практически не была исследована, и не было найдено значимых методов или подходов, специфических для этой задачи.

Однако, можно отметить, что задача определения стилевых особенностей исполнения имеет сходства с задачей распознавания музыкального стиля произведения в целом. Это связано с тем, что стилевые особенности исполнения часто влияют на общую музыкальную структуру и характер композиции; и наоборот, стиль исполнения нельзя рассматривать вне контекста стиля произведения.

Сходство позволяет предположить, что методы, используемые для определения музыкального стиля, могут быть применимы и в контексте стилевых особенностей исполнения. Необходимо отметить, что применение данных методов требует дополнительной адаптации для специфического контекста: в данных задачах понятие стиля определяется разными, хоть и пересекающимися, множествами характеристик.

Таким образом, на основе аналогии с задачей определения музыкального стиля и отсутствия существующих методов для определения стилевых особен-

ностей исполнения, в данной работе предлагается использовать те же методы и алгоритмы, адаптируя их к конкретным характеристикам стилевых особенностей исполнения.

Среди таких методов можно выделить следующие группы:

— Использование алгоритмов машинного обучения, таких как классификация с помощью метода опорных векторов (SVM) [3], нейронные сети [4] или случайные леса [5]. Эти алгоритмы тренируются на больших наборах данных, содержащих представления различных стилей музыки, и на основе этих данных определяют характерные признаки для каждого стиля. Затем они могут использоваться для классификации, определения или переноса стиля новых музыкальных композиций.

— Анализ спектральных характеристик аудио-сигналов, таких как спектрограммы или спектральные дескрипторы. Эти характеристики предоставляют информацию о распределении энергии в различных частотных диапазонах и позволяют выделить уникальные характеристики каждого стиля музыки [6][7]. На основе этих спектральных характеристик можно разработать алгоритмы для определения стиля композиции.

— Анализ ритмических и темповых характеристик, использование музыкальных дескрипторов, таких как хроматические или тональные дескрипторы [8].

— Статистический анализ нотного текста. Данный метод позволяет определить авторство музыкальных произведений [9] и потенциально может быть использован для классификации произведений по музыкальным стилям.

1.5 Перенос стиля исполнения музыки

1.5.1 Постановка задачи

В рамках данной работы задача переноса стиля заключается в моделировании выразительного исполнения заданного произведения с учетом характерных стилистических особенностей исполнения, присутствующих в образце

исполнения. Основная цель состоит в передаче желаемых выразительных нюансов и эмоциональной сущности исполнения, создавая новую аудио-версию произведения в заданном стиле исполнения.

Перечисленные ранее методы распознавания музыкального стиля являются значимыми в контексте проблемы, однако они не могут быть применены напрямую для решения задачи переноса стиля. Хотя эти алгоритмы позволяют распознать и классифицировать музыкальный стиль, обратная задача, связанная с применением определенного стиля, остается нерешенной. Это связано с тем, что успешный перенос стиля требует учета контекста и особенностей исходного произведения.

В контексте переноса стиля, необходимо применить выявленный стиль к новому произведению, учитывая его уникальные характеристики и структуру. Однако, простое копирование стилевых признаков из одного произведения в другое может привести к несоответствию и потере целостности произведения. В результате, для успешного применения стиля требуется разработка более сложных методов и моделей, которые способны учитывать контекст произведения, сохраняя его музыкальные особенности и выразительность.

Для решения этой проблемы можно обратиться к методу нейронного переноса стиля, который используется в области обработки изображений. Основная идея заключается в использовании оптимизационного алгоритма с целью минимизации разницы между извлеченными стилистическими особенностями и стилем образца, сохраняя при этом содержание исходного изображения.

Для оценки близости двух изображений по содержанию и стилю используются функции потерь $L_{content}$ и L_{style} соответственно. В результате данная задача сводится к задаче оптимизации функции [10]:

$$L_{content}(x_{content}, y) + aL_{style}(x_{style}, y) \to \min_{y}, \tag{1}$$

где $x_{content}$ и x_{style} – изображения-образцы содержания и стиля соответственно, y – синтезируемое изображение, а a – весовой коэффициент стиля.

Для применения метода нейронного переноса стиля к задаче переноса стиля исполнения музыки необходимо адаптировать функцию потери, чтобы она отражала качество исполнения произведения. В отличие от переноса стиля изображений, где функция потери контента оценивает сохранение содержания изображения, в музыкальном контексте необходимо учитывать, насколько синтезируемое исполнение соответствует структуре данного музыкального произведения.

Таким образом, задачу переноса стиля в данной работе можно поставить следующим образом:

$$L_{quality}(x, y) + aL_{style}(x_{style}, y) \to \min_{y}, \tag{2}$$

где:

— x – заданное произведение;

— x_{style} – образец стиля исполнения;

— y – синтезируемые параметры исполнения;

— $L_{quality}$ – функция потери качества исполнения;

— L_{style} – функция потери стиля;

— a – весовой коэффициент стиля.

1.5.2 Оценка качества исполнения

Как упоминалось ранее, для решения задачи переноса стиля необходимо определить функцию потери, которая бы учитывала качество исполнения музыкального произведения.

Качество исполнения во многом является субъективным понятием, поскольку оно включает в себя оценку эстетических и эмоциональных аспектов исполнения, которые могут различаться в зависимости от восприятия каждого слушателя. Однако можно утверждать о существовании некоторых объективных критериев, которые связаны с профессионализмом и выразительностью исполнения. Эти объективные критерии основаны на общепринятых нормах и стандартах в музыкальном сообществе. Например, точность воспроизведения

нот, ритмическая стабильность, динамические вариации, фразировка и артикуляция — все это критерии, которые позволяют экспертам оценить качество исполнения произведения.

Однако формулировка этих критериев может быть сложной задачей, поскольку необходимо учитывать контекст и характер произведения. Различные жанры и стили музыки могут требовать разных подходов к оценке качества исполнения. Например, в классической музыке может быть важным сохранение авторского стиля и традиций, в то время как в современных жанрах акцент может быть сделан на индивидуальности и оригинальности исполнения.

В свете этих сложностей, использование методов машинного обучения представляется логичным подходом. Алгоритмы машинного обучения позволяют анализировать большие объемы данных и выявлять общие закономерности и паттерны. Одним из подходов к решению задачи могло бы быть использование обучения с учителем, то есть обучение модели на размеченных данных, где каждое исполнение оценено по критериям профессионализма.

К сожалению, подобного набора данных, который содержал бы всю необходимую информацию для оценки качества исполнения музыкальных партий, не было найдено. Однако, для решения данной проблемы можно применить другой подход – обучение без учителя.

В данном случае, задачу оценки качества исполнения музыкальных партий можно поставить как задачу одноклассовой классификации. Для этого выделим класс «профессиональное исполнение музыкальной партии» и будем определять принадлежность партии к данному классу. Особенностью одноклассовой классификации является то, что модель обучается только на данных одного класса, в данном случае на наборе данных профессиональных исполнений.

Были найдены два подходящих набора данных: GiantMIDI-Piano [11] и MAESTRO Dataset [12]. Эти наборы данных содержат сотни часов MIDI-записей профессиональных музыкантов, исполняющих классическую музыку, и могут быть использованы для обучения модели одноклассовой классификации.

1.5.3 Автокодировщик как метод анализа музыки

В статье [3] был предложен метод стилизации музыкальных фрагментов на основе нейронной сети автокодировщика.

Автокодировщик - это специальная архитектура искусственных нейронных сетей, используемая для решения задачи понижения размерности. Основной принцип работы и обучения сети автокодировщика – получить на выходном слое отклик, наиболее близкий к входному [13]. На рисунке 1 изображена простейшая схема автокодировщика – сеть прямого распространения, наиболее схожая с многослойным перцептроном и содержащая входной слой, промежуточный слой и выходной слой.

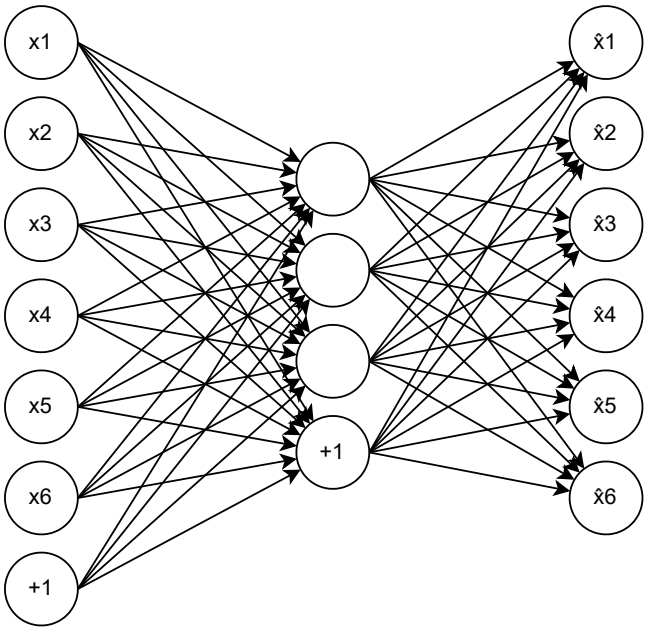

Рисунок 1 – Архитектура автокодировщика

Автокодировщик состоит из двух компонентов: зашифровщика (encoder) и дешифровщика (decoder), которые могут быть определены следующим образом:

$$\begin{cases} enc : \mathbb{X} \to \mathbb{L}, \\ dec : \mathbb{L} \to \mathbb{X}, \\ enc, dec = \arg\min_{enc,dec} \|X - (enc \circ dec)X\|^2, \end{cases} \tag{3}$$

где $\mathbb{X}$ – множество входных значений автокодировщика, $\mathbb{L}$ – множество значений скрытого пространства (latent space).

Одним из ключевых преимуществ автокодировщиков является их способность изучать внутреннее представление данных. На скрытое пространство накладывается ограничение: его размерность должна быть ниже размерности входных данных. Таким образом, автокодировщик обучается извлекать наиболее информативные и значимые признаки.

Автокодировщики, обученные на музыкальных фрагментах, могут применяться для извлечения музыкального стиля. В статье [3] было показано, что скрытое пространство автокодировщика может содержать интерпретируемые характеристики музыкальных произведений. Авторы статьи применили метод главных компонент к скрытому пространству, полученному от предобученного автокодировщика, и смогли выявить определенные направления в этом пространстве, соответствующие различным стилевым свойствам музыкальных фрагментов. Для создания новых музыкальных фрагментов с измененными стилевыми характеристиками исследователи изменяли скрытое пространство, после чего применяли дешифровщик. Этот подход позволяет контролировать степень изменения стиля и получать интересные вариации музыкальных композиций.

Помимо того, автокодировщики широко применяются в задачах одноклассовой классификации [14]. В этом случае модель обучается на образцах, представляющих типичные или нормальные данные. Затем автокодировщик применяется для восстановления этих образцов из их скрытого представления. В процессе обучения автокодировщик стремится минимизировать ошибку восстановления, тем самым позволяя модели изучить основные структуры и закономерности нормальных образцов. При этом, если входной образец является аномалией, то модель будет испытывать трудности в восстановлении такого образца, что приведет к более высокой ошибке реконструкции.

Таким образом, использование автокодировщика в данной работе будет обладать существенными преимуществами перед альтернативными методами

анализа музыки, поскольку он эффективно решает обе задачи, связанные с переносом стиля.

Во-первых, с его помощью можно сравнивать стиль двух музыкальных произведений, анализируя и сопоставляя их скрытые представления. Это обеспечивает возможность оценки степени сходства или различия между музыкальными композициями на основе их стилевых характеристик.

Во-вторых, та же самая модель автокодировщика может быть использована для оценки качества исполнения музыкальных произведений через задачу одноклассовой классификации. Обученный на данных, соответствующих высококачественным исполнениям, автокодировщик будет стремиться восстановить высококачественные характеристики входных данных. Это значит, что малое отличие восстановленной версии от оригинальной записи будет свидетельствовать о высоком качестве исполнения.

1.5.4 Метод главных компонент

В статье [15] было продемонстрировано, что для эффективного анализа скрытого пространства автокодировщика можно использовать метод главных компонент (МГК). Анализ скрытого пространства является сложной задачей из-за смешанных и сложных признаков, полученных в результате работы автокодировщика. Однако метод главных компонент позволяет выделить наиболее значимые компоненты (направления) в скрытом пространстве, которые могут использоваться для оценки различных стилевых признаков.

Метод главных компонент – метод снижения размерности данных с наименьшей потерей информации, изобретенный Карлом Пирсоном в 1901 году [16]. Он заключается в нахождении подпространства меньшей размерности, в ортогональной проекции на которое разброс данных максимален.

Пусть имеется p-мерная величина $X = (x_1, x_2, \ldots, x_p)$ с вектором средних значений $a = (a_1, a_2, \ldots, a_p)$. Определим на множестве признаков всевоз-

можные линейные ортогональные нормированные комбинации:

$$z_i = \sum_{i=1}^{p} l_{ij}(x_i - a_i), \tag{4}$$

где $\sum_{i=1}^{p} l_{ij}^2 = 1$ и $\sum_{i=1}^{p} l_{ji} l_{ki} = 0$ для $j = 1, 2, \ldots, p$ и $k = 1, 2, \ldots, p$, но $j \neq k$. При этом потребуем, чтобы выполнялось условие монотонности дисперсий: $D(z_1) \geqslant D(z_2) \geqslant \cdots \geqslant D(z_p)$.

Полученные таким образом переменные $z_1(X), z_2(X), \ldots, z_p(X)$ и называются главными компонентами.

Видно, что k-я главная компонента некоррелированна с предыдущими $k - 1$ и обладает наибольшей дисперсией среди всех прочих таких линейных комбинаций переменных $x_1, x_2, \ldots, x_p$. Поскольку главные компоненты ранжированы по величине дисперсии, можно отбросить часть компонент, оставив только $m \leqslant p$ из них, которые воспроизводят большую часть дисперсии. Эти свойства и позволяют применять МГК для снижения размерности с наименьшей потерей информации.

Можно показать, что для того, чтобы величина $D(z_i)$ достигала максимума, необходимо, чтобы вектор l_i был i-м по величине собственного значения собственным вектором ковариационной матрицы вектора X. Таким образом, задача вычисления главных компонент сводится к вычислению собственных векторов в порядке уменьшения собственных значений.

В уже упомянутой статье [3] было показано, что МГК позволяет найти в скрытом пространстве автокодировщика главные компоненты, соответствующие стилевым особенностям музыкальных произведений. Это позволяет предположить, что этот метод можно применить также и к задаче определения стилевых особенностей исполнения.

1.5.5 Алгоритм Adam

Для решения поставленной задачи методом оптимизации необходимо выбрать оптимизационный алгоритм.

Оптимизационный алгоритм Adam (Adaptive Moment Estimation) является одним из наиболее распространенных и эффективных методов оптимизации в глубоком обучении и успешно применяется в том числе в задачах переноса стиля. Он был предложен в статье «Adam: A Method for Stochastic Optimization» (2014) авторами Diederik P. Kingma и Jimmy Ba [17].

Adam сочетает в себе преимущества двух других популярных методов оптимизации – адаптивного градиентного спуска (Adagrad) и стохастического градиентного спуска с инерцией (SGD with momentum). Алгоритм Adam позволяет эффективно находить оптимальные значения параметров модели, учитывая особенности каждого параметра и изменяя скорость обновления в зависимости от их вклада в общую функцию потерь.

Основным преимуществом алгоритма Adam является его адаптивность к различным условиям оптимизации. Он автоматически адаптирует скорость обучения для каждого параметра на основе оценки первого и второго моментов градиента:

$$m_k = \beta_1 m_{k-1} + (1 - \beta_1)\nabla f(x_k), \tag{5}$$

$$v_k = \beta_2 v_{k-1} + (1 - \beta_2)(\nabla f(x_k))^2, \tag{6}$$

где x_k – k-я точка градиентного спуска, $f(x)$ – оптимизируемая функция, а β_1 и β_2 – гиперпараметры, как правило, принимаемые равными 0.9 и 0.99 соответственно.

Вычисленные моменты корректируются по формулам:

$$\bar{m_k} = \frac{m_k}{1 - \beta_1^k}, \tag{7}$$

$$\bar{v_k} = \frac{v_k}{1 - \beta_2^k}. \tag{8}$$

Затем вычисляется следующая точка спуска:

$$x_{k+1} = x_k - \frac{\alpha}{\sqrt{\bar{v_k} + \varepsilon}}\bar{m_k}, \tag{9}$$

где α – скорость обучения, а ε – гиперпараметр, принимаемый, как правило, равным $1e{-}8$

Данные модификации градиентного спуска позволяют справляться с проблемами, связанными с выбором оптимального шага обучения и нахождением локальных минимумов в функции потерь. Алгоритм Adam также имеет низкую требовательность к памяти, что позволяет эффективно использовать его для обучения моделей с большим количеством параметров.

1.6 Синтез звука

1.6.1 Сравнение синтеза речи и музыкальных звуков

Несмотря на сходство задач синтеза речи и музыкальных звуков, между ними есть несколько ключевых различий, которые делают нетривиальным применение методов синтеза речи к музыке:

— В музыке часто встречается полифония – например, аккорды – одновременное звучание нескольких равноправных звуков, что невозможно в речи.

— В отличие от речи, музыка часто содержит длинные ноты. Это является серьезной проблемой в том числе в конкатенативном синтезе.

— В речи расположение синтезируемых единиц во времени определяется их естественными длительностями. В музыке необходимо точно соответствовать заданному ритму.

— При синтезе речи первостепенное значение уделяется разборчивости и естественности. В музыке же используется множество способов выразительности, которые необходимо предусмотреть при синтезе.

1.6.2 Абстрактный синтез

Методы абстрактного синтеза звука используют математические функции для генерации звуков без прямой физической интерпретации [18].

Наиболее известным абстрактным методом является частотная модуляция (FM-синтез), которая служила основой для многих ранних цифровых синтезаторов. Принцип работы FM-синтеза заключается в использовании сигна-

ла модулятора для изменения частоты воспроизводимого (несущего) звукового сигнала. В самом простом случае, когда два осциллятора генерируют синусоидальные волны, синтезируемый сигнал определяется формулой:

$$y(t) = A_c sin[2\pi(f_c + dsin(2\pi f_m t))t], \tag{10}$$

где A_c – амплитуда несущего сигнала, f_c и f_m – частоты модулирующего и несущего осцилляторов соответственно, а d – интенсивность модуляции.

Абстрактный синтез требует сравнительно небольшого объема памяти, а также позволяет динамически управлять спектром создаваемого звука при помощи небольшого числа параметров. Однако, поскольку эти параметры не имеют физической интерпретации, их назначение трудно понять обычному пользователю.

При помощи абстрактного синтеза трудно приблизиться к звучанию настоящих музыкальных инструментов, потому его основное назначение – создание новых тембров.

1.6.3 Спектральное моделирование

Методы спектрального моделирования уделяют основное внимание спектру генерируемых звуков. Наиболее известным подходом спектрального моделирования является аддитивный синтез.

Аддитивный синтез предполагает, что любой периодический сигнал можно рассматривать как сумму синусоид с разными огибающими амплитуды и разными
частотами [19]:

$$y(t) = \sum_{i=1}^{p} A_i(t) sin[\Theta_i(t)], \tag{11}$$

где $y(t)$ – синтезируемый сигнал, p – количество синусоид, а $A_i(t)$ и $\Theta_i(t)$ – амплитуда и фаза i-й синусоиды в момент времени t соответственно.

Фаза в момент времени t определяется по формуле:

$$\Theta_i(t) = \Theta_i(0) + \phi_i + \int_0^t \omega_i(t)\, dt, \tag{12}$$

где $\omega_i(t)$ – угловая частота в момент времени t, $\Theta_i(0)$ – начальная фаза волны, а ϕ_i – постоянный фазовый сдвиг.

Основным недостатком этого мощного метода является количество задействованных параметров. Для синтеза сигналов с богатым содержанием гармоник количество необходимых генераторов может быть очень большим, что требует сложной настройки и больших вычислительных ресурсов [20].

1.6.4 Физическое моделирование

Общий принцип данного метода заключается в моделировании физических процессов, отвечающих за воспроизведение звука. Этот процесс требует высоких вычислительных затрат, поэтому такие алгоритмы основаны на упрощающих предположениях.

В случае музыкальных инструментов звук синтезируется путем описания звукоизвлечения и поведения элементов инструмента, отвечающих за звук, например, смычка, струн и корпуса скрипки [21]. Таким образом, музыкальный инструмент можно рассматривать как систему из механизма возбуждения колебаний и резонатора [22].

Пример моделирования скрипки изображен на рисунке 2.

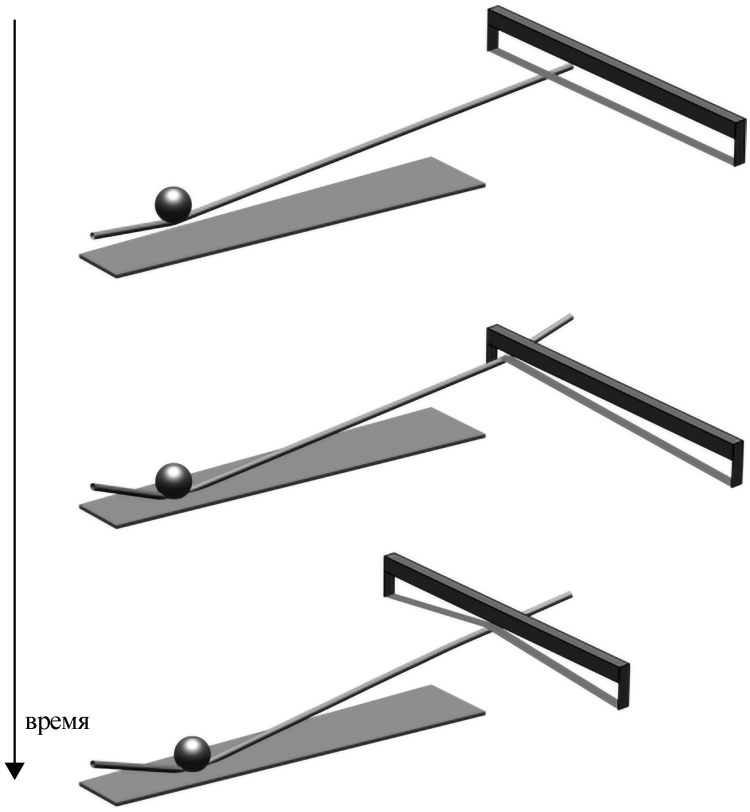

Рисунок 2 – пример моделирования скрипки; смычок опускается на струну, которая зажимается между пальцем и грифом; затем смычок нажимает на струну, чтобы привести ее в движение

Физическое моделирование предоставляет музыкантам осмысленные параметры для настройки звука (к примеру, размеры инструмента, толщина струн, материал корпуса) и возможность синтезировать звуки с высоким качеством [20].

Недостаток подхода заключается в том, что полученные модели очень специфичны и в значительной степени зависят от типа звуков, для которых они предназначены. Даже для описания одного инструмента требуется реализовать несколько моделей для разных типов звукоизвлечения. Помимо того, детальное описание поведения инструмента требует больших вычислительных мощностей.

1.6.5 Конкатенативный синтез

В конкатенативном синтезе используются фрагменты реальных акустических сигналов, из которых путем склейки (конкатенации) создается основа синтезируемого акустического сигнала [23]. Эта основа подвергается модификации, чтобы придать склеенным фрагментам нужный характер, а также замаскировать переходы между ними.

Различные системы конкатенативного синтеза используют в качестве базовых элементов для склейки звуковые единицы различного размера: это могут быть целые ноты и переходы между ними, части нот или короткие отрывки до нескольких сотен миллисекунд в случае гранулярного синтеза.

1.6.6 Сравнение методов

Для оценки методов можно предложить следующие критерии:

— возможность генерации звука в реальном времени;
— объем требуемых вычислительных ресурсов;
— объем занимаемой памяти;
— наличие параметров для настройки звука, имеющих понятную для пользователя физическую интерпретацию;
— универсальность, под которой понимается возможность использования метода с незначительными изменениями для синтеза звуков различных инструментов.
— реалистичность, под которой понимается возможность воссоздания тембров реальных музыкальных инструментов.

Для оценки непосредственно качества работы алгоритма – реалистичности воспроизведения – можно предложить некоторые формальные критерии, например, частоту повторений звуковых фрагментов. Однако такие показатели не дают полного представления о качестве синтезируемого сигнала, поскольку во многом это понятие является субъективным. Иногда для сравнения алгоритмов используют опросы экспертов слепым методом [24] [25].

Объемы требуемых вычислительных ресурсов и памяти варьируются в рамках одного подхода в зависимости от конкретной реализации. Поэтому сравнить рассмотренные методы по этим критериям не представляется возможным. Тем не менее, можно сделать несколько общих наблюдений:

— конкатенативный синтез требует значительно больше места на диске по сравнению с другими методами в связи с необходимостью хранить базу данных звуковых фрагментов. В коммерческих продуктах объем таких данных может достигать сотен гигабайт;
— наиболее затратными с точки зрения времени и оперативной памяти считаются методы физического и спектрального моделирования;
— наименее ресурсоемким методом является абстрактный синтез;

На таблице 1.1 представлена классификация методов синтеза звука в соответствии с предложенными критериями.

Таблица 1.1 – Сравнение существующих решений

	Реалистичность	Наличие параметров, имеющих физическую интерпретацию	Универсальность
Абстрактный синтез	−	−	+
Спектральное моделирование	+−	−	+
Физическое моделирование	+	+	−
Конкатенативный синтез	+	−	+

В данной работе было решено использовать конкатенативный синтез звука для реализации задачи реалистичного воспроизведения музыкальных партий. Этот метод обеспечивает достаточную степень реалистичности звучания, являясь при этом универсальным. Он может быть применен к различным инструментам, включая фортепиано, скрипку, гитару и другие. Это обеспечивает гибкость в реализации метода и возможность применения его в разных музыкальных контекстах.

В данной работе физическая интерпретация параметров звука не является первоочередной задачей. Вместо этого основным фокусом является достижение реалистичности и сохранение выразительности воспроизведения музыкальных партий. Параметры звука, такие как атака, длительность и тембр, могут быть учтены при подборе и сшивке звуковых образцов и не требуют детального физического моделирования. Это позволяет сосредоточиться на более общих аспектах воспроизведения и переносе стиля.

1.7 Конкатенативный синтез на основе Unit Selection

Синтез звука, основанный на алгоритме Unit Selection, является наиболее распространенной реализацией конкатенативного подхода. Существует множество модификаций этого алгоритма, однако далее будет рассмотрен классический вариант, предложенный в статье [26]:

— фрагменты акустических сигналов хранятся в базе данных, называемой корпусом;
— каждому фрагменту ставится в соответствие дескриптор, описывающий его параметры;
— музыкальная партия, которую необходимо воспроизвести, переводится в последовательность целевых дескрипторов;
— алгоритм Unit Selection подбирает наиболее подходящие фрагменты из базы данных;
— эти фрагменты после необходимых преобразований склеиваются, образуя итоговую звуковую дорожку.

1.7.1 Сегментация

Прежде чем добавить запись в базу данных, ее необходимо сегментировать, чтобы получить массив отдельных нот или фраз. Это может быть выполнено различными способами, например:

— В случае, когда акустический сигнал предоставлен вместе с некоторой нотацией – например, с MIDI-файлом, – это делается тривиально путем сопоставления длительностей нот.
— Партии с устойчивым ритмом могут быть сегментированы на регулярно расположенные блоки с учетом ритмического рисунка.
— Сегментация может быть выполнена вручную или результат автоматической сегментации может быть вручную отредактирован.
— Существует множество более сложных алгоритмов, например машинное

обучение, которые можно использовать в качестве внешнего модуля.

Для каждого полученного сегмента выполняется дополнительное разделение на фазы атаки, звучания и затухания звука. Это также может быть проделано различными способами. В самом простом случае фазам атаки и затухания отводится фиксированное время в начале и конце сегмента соответственно.

Фазы затухания одного сегмента и атаки следующего вместе образуют сегмент перехода.

1.7.2 Анализ

Полученные после сегментации фрагменты анализируются для выражения их характеристик с помощью дескрипторов. Характеристики являются скалярными и могут быть различными в зависимости от реализации.

Характеристики можно разделить на три основные группы [27]: категориальные, статические и динамические.

Категориальные характеристики отражают принадлежность фрагмента к определенной категории или классу. Это может быть тип инструмента, способ исполнения (легато, стаккато, вибрато и т. д.), динамика исполнения (крещендо, диминуэндо) и прочее. К этой же группе относится тип фрагмента, принимающий значения ноты для целой ноты, аттаки, звучания или затухания, или фразы для нескольких нот.

Статические характеристики отражают неизменные свойства фрагментов, такие как номер ноты MIDI, количество сыгранных нот, информация о музыкальной записи, из которой получен фрагмент, начальное время фрагмента в записи, путь к аудио-файлу, длительность фрагмента.

Динамические характеристики вычисляются после анализа акустического сигнала. Могут рассматриваться такие параметры, как высота звука, мощность сигнала, спектральный центроид, спектральный наклон, негармоничность звука и другие. Для каждого такого параметра можно определить следующие характеристики:

1) среднее значение;

2) стандартное отклонение;
3) минимум, максимум и диапазон;
4) средний наклон, отражающий приблизительное направление изменения параметра.

Как уже было сказано, характеристик может быть разное количество. Некоторые системы конкатенативного синтеза позволяют пользователю динамически добавлять дополнительные характеристики или изменять существующие [28].

После того, как все необходимые характеристики сегмента вычислены, эта информация заносится в дескриптор сегмента, который помещается в базу данных.

1.7.3 Целевые дескрипторы

Чтобы синтезировать звук, необходимо сформировать последовательность дескрипторов, описывающую желаемый результат. Эту последовательность можно получить различными способами:

1) Из нотной записи, MIDI или других источников, в явном виде определяющих параметры дескрипторов.
2) Из аудио-файла: в этом случае решается задача переозвучивания партии с использованием звуков из базы данных, или создание так называемой аудио-мозаики.
3) При помощи различных статистических моделей.

1.7.4 Unit Selection

Важнейшей частью описываемого метода является алгоритм Unit Selection. Его задача – подобрать наиболее подходящую последовательность звуков из базы данных.

На вход алгоритм получает базу данных дескрипторов фрагментов u^i и целевую последовательность дескрипторов t^τ, которые описывают n характеристик u^i_f и t^τ_f соответственно.

Алгоритм последовательно подбирает для каждого целевого дескриптора наиболее подходящую запись из БД. Качество подобранного варианта опреде-

ляется двумя значениями: коэффициент различия и стоимость конкатенации.

Коэффициент сходства p_t^i стремится описать перцептивное сходство подобранного фрагмента u^i и целевого дескриптора t^τ. Он определяется функцией перцептивного расстояния D по соотношению:

$$p_t^i = \sum_{j=-r}^{r} w_i D(u^{i+j}, t^{\tau+j}, j), \tag{13}$$

где контекст в диапазоне $[-r; r]$ единиц вокруг текущего дескриптора сравнивается с контекстом вокруг дескриптора из базы данных, при этом веса w_i уменьшаются с расстоянием (см. рис. 3).

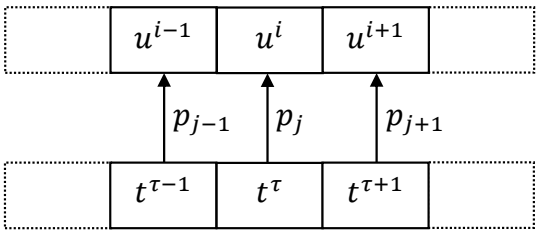

Рисунок 3 – Множители $p_j = D(u^{i+j}, t^{\tau+j}, j)$ для диапазона контекста $r = 1$

Функция перцептивного расстояния D определяется как взвешенная сумма функций d_f расстояния для каждой характеристики:

$$D(u, t, j) = \sum_{f=1}^{n} w_f d_f(u, t, j). \tag{14}$$

Можно заметить, что каждая функция расстояния характеристики имеет доступ ко всем характеристикам фрагментов.

Коэффициент сходства также связан с количеством преобразований фрагмента, необходимых для полного соответствия целевому дескриптору, а следовательно отражает возникающие в связи с этим искажения.

Стоимость конкатенации p_c^i характеризует разрыв, который возникнет при склейке выбранного фрагмента u^i с предыдущим u‘ (см. рис. 4).

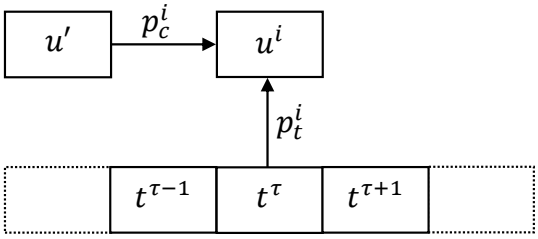

Рисунок 4 – Коэффициент сходства p_t^i и стоимость конкатенации p_c^i

Стоимость конкатенации характеризуется функцией стоимости конкатенации C, которая в свою очередь определяется аналогично функции перцептивного расстояния – как взвешенная сумма функций c_f конкатенации для каждой характеристики:

$$p_c^i = C(u‘, u^i) = \sum_{f=1}^{n} w_f c_f(u‘, u^i). \tag{15}$$

Для последовательных фрагментов стоимость конкатенации равна нулю, поэтому если в базе данных будет найдена фраза, соответствующая последовательности целевых дескрипторов, то она будет выбрана целиком.

Итоговая стоимость, которую стремится минимизировать алгоритм, определяется как взвешенная сумма коэффициента сходства и стоимости конкатенации:

$$\begin{aligned} p_s^i &= w_c p_c^i + w_t p_t^i = \\ &= w_c C(u‘, u^i) + w_t \textstyle\sum_{j=-r}^{r} w_i D(u^{i+j}, t^{\tau+j}, j) = \\ &= w_c \textstyle\sum_{f=1}^{n} w_f c_f(u‘, u^i) + w_t \sum_{j=-r}^{r} w_i \sum_{f=1}^{n} w_f d_f(u, t, j). \end{aligned} \tag{16}$$

Задача алгоритма Unit Selection – найти последовательность фрагментов с минимальным значением p_s^i. В случае большого числа фрагментов возникает проблема эффективного поиска такой последовательности. Часто для этого применяют алгоритм Витерби, а также кластеризацию, например, методом k-ближайших соседей [29].

1.7.5 Синтез

Этап синтеза итогового звукового сигнала выполняется в два этапа: трансформация и конкатенация.

Трансформация применяется к каждому фрагменту в отдельности для достижения полного соответствия дескриптору. Могут быть скорректированы длина фрагмента, громкость, высота тона, характеристики спектра и т. д. Некоторые преобразования могут негативно сказываться на качестве звукового фрагмента, поэтому важным показателем эффективности работы системы является число таких трансформаций. Оно зависит как от алгоритма подбора фрагментов, так и от размера базы данных.

Конкатенация выполняется путем соединения фрагментов с небольшим перекрытием для плавного перехода (crossfade).

1.8 Выбор музыкального инструмента

Поскольку целью работы является разработка метода, учитывающего особенности исполнения музыки, для его разработки необходимо выбрать конкретный музыкальный инструмент. Наиболее подходящим является фортепиано по ряду причин, включая его широкое распространение, универсальность и простоту в реализации стилизации и синтеза.

Во-первых, фортепиано является одним из самых распространенных и популярных музыкальных инструментов в мире. Его звуковой диапазон и возможности выразительности позволяют воспроизводить очень разные стили и жанры музыки. Также фортепиано широко используется в академической музыке, что обеспечивает доступность исследовательского материала и репертуара для изучения и анализа.

Во-вторых, фортепиано обладает хорошо структурированным и документированным звуковым образцом, что делает его идеальным инструментом для реализации метода переноса стиля. Большое количество сэмплов и записей фортепиано доступно для использования в качестве исходного материала для обу-

чения модели и создания реалистичного звучания. Это позволяет улучшить точность и достоверность воспроизведения стиля исполнения и выразительности музыкальных партий.

Наконец, фортепиано предоставляет относительно немного параметров исполнения, которые при этом являются общими для большинства музыкальных инструментов. Это позволит продемонстрировать возможности метода без необходимости адаптировать его под специфичный инструмент.

В связи с вышеперечисленными причинами, выбор фортепиано в качестве инструмента для реализации метода реалистичного воспроизведения музыкальных партий с переносом стиля является логичным и обоснованным решением, обеспечивающим простоту реализации и возможности достижение необходимого уровня реалистичности и качества звучания.

Однако стоит отметить, что принципы и методы, разработанные для синтеза фортепиано, вполне могут быть применимы и для других инструментов. Поэтому выбор фортепиано как начального шага в разработке метода не ограничивает его применение только к этому инструменту. В дальнейшем, при наличии достаточных ресурсов и возможностей, метод можно будет расширить на другие музыкальные инструменты.

1.9 Параметры стиля исполнения

В данной работе было принято решение выбрать в качестве варьируемых параметров стилизации следующие:

— громкость каждой ноты;

— связность исполнения;

— оттяжка.

Первым параметром является громкость каждой ноты. Громкости отдельных нот складываются в общую динамику произведения, позволяя передать эмоциональную окраску и акцентировать важные моменты в музыкальной композиции.

Вторым параметром является связность исполнения. Она характеризует-

ся следующими штрихами:

— стаккато – обозначает отрывистый, резкий звук;

— нонлегато – ровное, не отрывистое, но и не плавное звучание;

— легато – связное звукоизвлечение.

легатность, которая определяет способ звукоизвлечения соседних нот - слитное или раздельное. Легатность влияет на характер и плавность переходов между нотами. Использование слитного звукоизвлечения придает игре плавность, в то время как раздельное звукоизвлечение создает более резкие и выделенные ноты.

Третьим параметром является оттяжка, которая определяет момент начала звучания ноты относительно указанного в нотации времени. Оттяжка позволяет расставлять дополнительные акценты.

Рассматривалась также возможность темповой динамики в качестве дополнительного параметра. Изменение темпа является сложной задачей, поскольку предполагает работу с временными характеристиками. При этом человеческий слух очень чувствителен к изменению темпа, и даже небольшая ошибка в данном параметре может привести к очень существенным и нежелательным искажениям в музыкальном материале. Поэтому для обеспечения более надежного и качественного результата в данной работе было принято решение не включать изменение темпа в рассматриваемые параметры, сосредоточившись на других стилевых характеристиках исполнения.

1.10 Постановка задачи

Необходимо разработать метод реалистичного воспроизведения музыкальных партий с переносом стиля. Входными данными для метода являются два музыкальных произведения: одно произведение является образцом стиля исполнения, второе необходимо исполнить с учетом образца.

На рисунке 5 представлена IDEF0-диаграмма разрабатываемого метода.

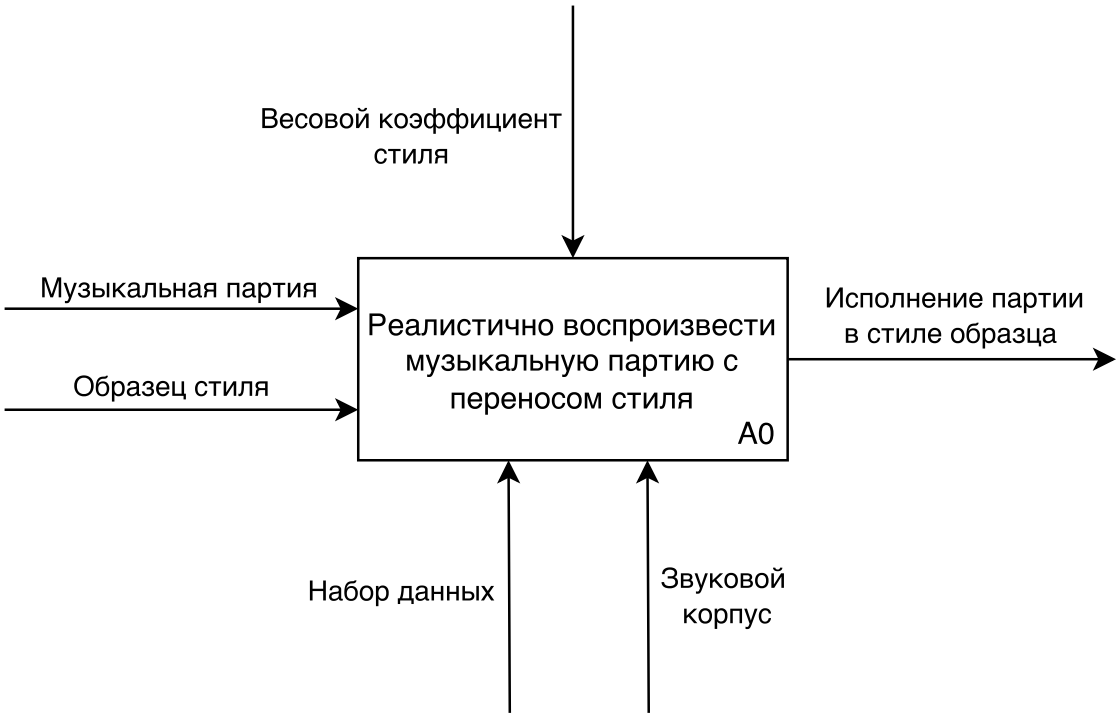

Рисунок 5 – IDEF0 диаграмма разрабатываемого метода

На разрабатываемый метод накладываются следующие ограничения:

— В качестве инструмента воспроизведения выбрано фортепиано.
— Метод должен работать с монофоническими партиями.
— Метод позволяет настраивать такие параметры исполнения, как громкость, легатность и оттяжку.
— В методе не учитывается эффект педали фортепиано.

Для оценки метода необходимо сформулировать количественные оценки сходства стилей.

1.11 Вывод

В данном разделе был проведен анализ предметной области. В ходе анализа были рассмотрены основные понятия и определено понятие стиля исполнения в контексте данной работы.

Также были рассмотрены алгоритмы, применяемые для решения задачи стилизации музыки и приведена мотивация к их использованию в задаче сти-

лизации исполнения. Анализ методов синтеза звука позволил выбрать конкатенативный синтез в качестве подходящего метода для данного исследования. Кроме того, был рассмотрен и описан алгоритм конкатенативного синтеза на основе Unit Selection.

На основе проведенного анализа были сформулированы постановка задачи и ограничения на применение предлагаемого метода.

2 Конструкторская часть

В данном разделе рассматриваются используемые алгоритмы и их специфические особенности, присущие решаемой задаче. Разрабатываются основные положения предлагаемого метода реалистичного воспроизведения музыкальных партий с переносом стиля. Подробно описывается метод переноса стиля. Проектируется модуль синтеза звука и рассматриваются особенности реализации алгоритма конкатенативного синтеза звука.

2.1 Общая структура метода

Разрабатываемый метод можно разделить на два основных этапа: формирование параметров исполнения и синтез звука. Данная структура изображена на диаграмме 6.

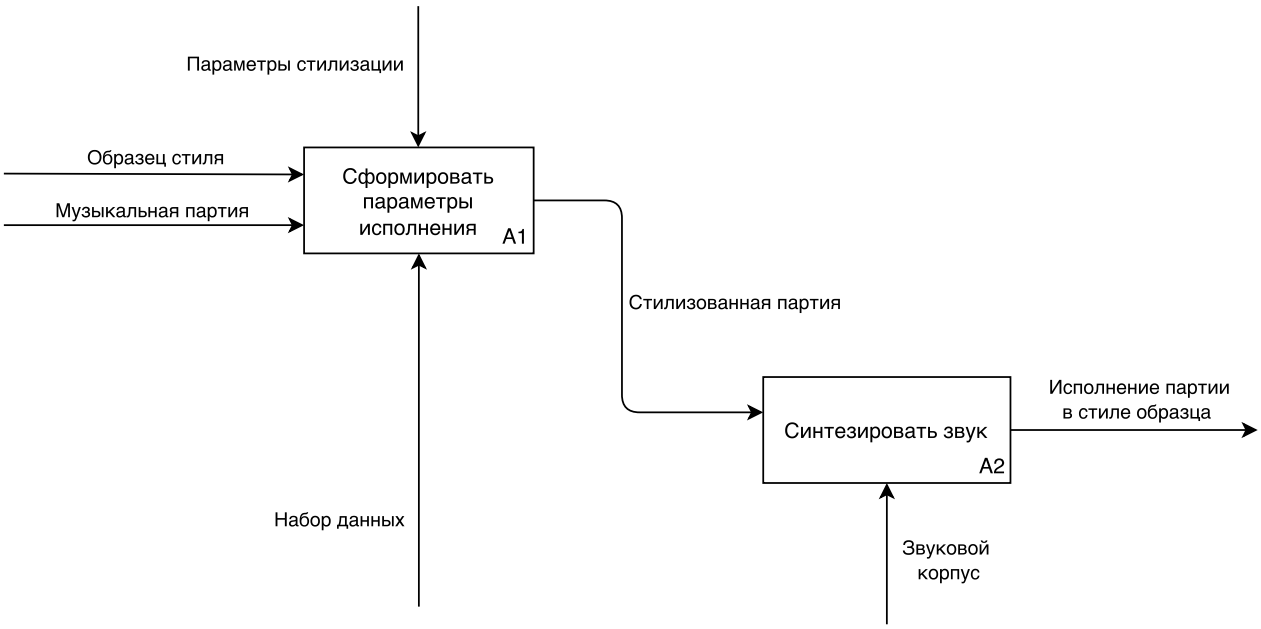

Рисунок 6 – Основные этапы работы метода

Задача первого этапа — формирования параметров исполнения — состоит в переносе стиля с образца на заданную музыкальную партию. На вход модуля, реализующего данный этап, поступают два MIDI-файла: музыкальная партия, которую необходимо исполнить, и образец стиля. В результате работы модуля формируется новый MIDI-файл, содержащий стилизованное исполнение

заданной партии.

Более подробная схема этапа формирования параметров исполнения изображена на диаграмме 7. Обучение модели проводится единожды и не требует повторения при каждом применении разрабатываемого метода.

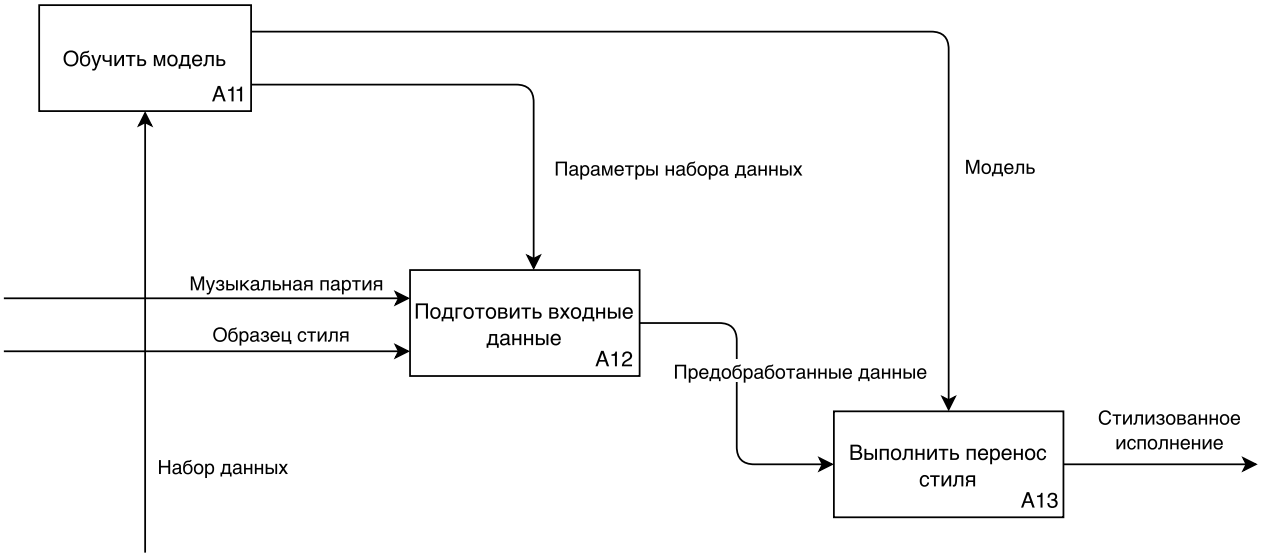

Рисунок 7 – Основные этапы формирования параметров исполнения

Задачей второго этапа — синтеза звука — является создание аудио-файла с исполнением, полученным на первом этапе. На вход соответствующего модуля программы поступает MIDI-файл стилизованного исполнения, а на выходе формируется соответствующий аудио-файл. Схема данного этапа изображена на диаграмме 8. Так же, как и обучение модели, обработка звукового корпуса выполняется единожды.

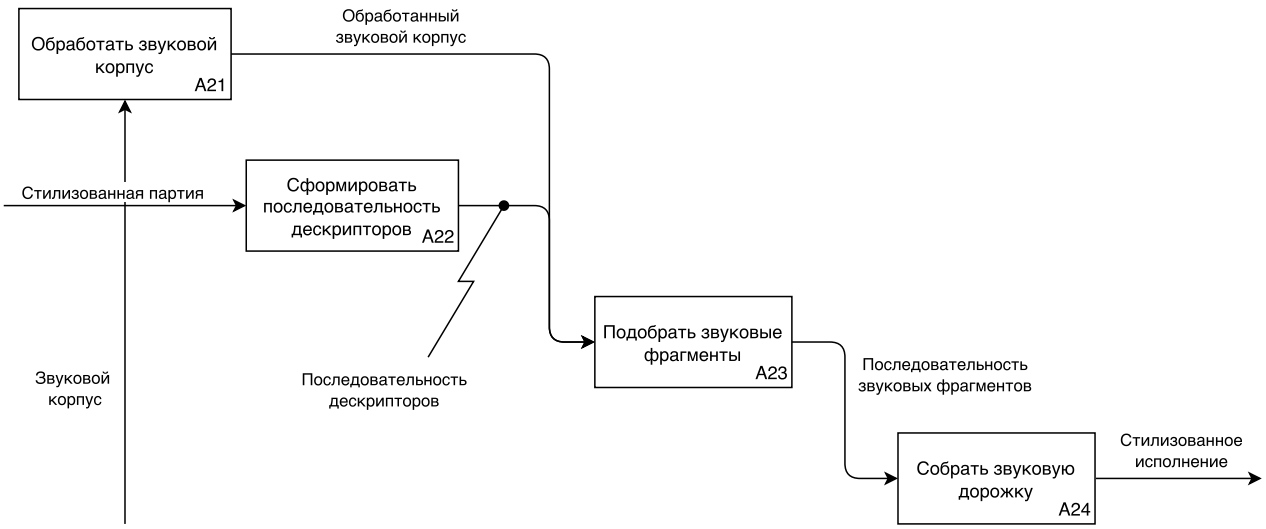

Рисунок 8 – Основные этапы синтеза звука

2.2 Метод переноса стиля

2.2.1 Параметры исполнения

Генерируемое исполнение можно определить векторами vel, leg и dt, длины которых совпадают с количеством нот в обрабатываемом фрагменте. В переменной vel сохраняется интенсивность (velocity) каждой ноты, в leg хранится значение, отражающее легатность (legato), а в переменной dt сохраняется оттяжка.

Интенсивность, она же громкость, ноты определяется числом в интервале от 0 до 127, в соответствии с форматом MIDI, где 0 — отсутствие звука, а 127 — максимально громкий звук.

Легатность ноты определим как отношение реальной длительности ноты к временному интервалу от начала звучания ноты до начала звучания следующей. Таким образом, этот параметр отражает не только факт раздельного или слитного звукоизвлечения, но и степень отрывистости или связности:

— $leg_i \ll 1$ соответствует стаккато;

— $leg_i < 1$ соответствует нонлегато;

— $leg_i \geqslant 1$ соответствует легато.

Оттяжка ноты определяется временным интервалом между указанным в

нотации моментом звучания и фактическим моментом начала звукоизвлечения в секундах.

2.2.2 Архитектура автокодировщика

Для анализа музыкальных фрагментов было решено использовать многослойный полносвязный шумоподавляющий автокодировщик.

Шумоподавляющий автокодировщик - это модификация автокодировщика, в которой к входным данным намеренно добавляется шум в процессе обучения. Чаще всего в качестве шума выбирается Гауссовский шум; в этом случае постановка задачи обучения автокодировщика приобретает следующий вид:

$$\begin{cases} enc : \mathbb{X} \to \mathbb{L}, \\ dec : \mathbb{L} \to \mathbb{X}, \\ noise \sim \mathcal{N}(\mu,\, \sigma^2), \\ enc, dec = \arg\min_{enc,dec} \|X - (enc \circ dec)(X + noise)\|^2. \end{cases} \tag{17}$$

Помимо повышения устойчивости к шуму, такая регуляризация побуждает модель выделять более устойчивые и обобщенные признаки, что в свою очередь делает более информативным скрытое пространство. Именно это свойство является основной мотивацией к использованию шумоподавляющего автокодировщика в данной работе, поскольку он будет использоваться именно для выделения характеристик музыкальных фрагментов. Добавление шума к данным также выполняет в некотором смысле роль аугментации, позволяя бороться с переобучением.

Выбор полносвязного автокодировщика для данной работы обусловлен несколькими факторами, включая простоту обучения по сравнению с рекуррентными моделями. Полносвязные автокодировщики имеют простую архитектуру и обучаются сравнительно быстро, что позволяет сосредоточиться на разработке комплексного метода стилизации исполнения, вместо затраты времени на обучение и настройку конкретной нейронной сети.

2.2.3 Кодирование музыкальных фрагментов

Рассмотрим методику кодирования данных для предложенной модели.

Прежде всего, музыкальный фрагмент должен быть приведен к монофонической последовательности нот. Традиционный подход заключается в отбрасывании нижних голосов произведения, при этом оставляют только верхние ноты аккордов. Такой подход имеет ряд недостатков. Во-первых, это приводит к потере значительного объема информации, что сокращает объем обучающей выборки. Во-вторых, полученная последовательность нот чаще всего не является репрезентативной для музыкального анализа, поскольку мелодические линии могут быть распределены по различным голосам и регистрам.

Поэтому было решено воспользоваться методом разделения голосов на основе скрытых марковских моделей, предложенном в статье [30]. Этот метод обеспечивает разделение многоголосых произведений на отдельные монофонические линии и обладает рядом преимуществ:

1) каждая нота произведения будет включена в одну из полученных последовательностей, что препятствует потере информации;
2) последовательности нот будут более репрезентативными, поскольку будут соответствовать отдельным голосам произведения.

Поскольку выбранная модель оперирует данными фиксированного размера, необходимо привести последовательности нот к заданной фиксированной длине. В данном случае было принято решение установить длину последовательности в $n = 64$ ноты.

Выбранная длина является достаточной для улавливания долговременных паттернов и не слишком велика для входного вектора нейронной сети. Помимо того, в ранее проведенных исследованиях [3] использование длины последовательности в 64 ноты демонстрировало хорошие результаты и показало свою эффективность.

Для реализации этого преобразования применяется скользящее окно с дли-

ной 64 ноты и шагом, равным единице. Такой подход позволяет последовательно обработать данные с перекрытием, пройдя через всю последовательность и захватив все 64-нотные фрагменты.

С учетом определенных ранее характеристик стиля были выбраны следующие параметры для кодирования нот:

1) Высота тона i-й ноты $tone_i$ — целое число от 0 до 127.
2) Длительность ноты $dist_i$ — временной интервал от момента звукоизвлечения i-й ноты до следующей $(i+1)$-й для $i < n$ и фактическая длительность звучания ноты для $i = n$ в секундах.
3) Громкость (интенсивность) i-й ноты vel_i.
4) Легатность i-й ноты leg_i.

Для исключения лишней информации о тональности фрагмента высоту тона можно закодировать в виде музыкальных интервалов, то есть вычислить конечную разность первого порядка:

$$tone\_encoded_j = tone_{j+1} - tone_j, \tag{18}$$

где $j = 1 \dots n$-1.

Конкретные методы нормализации данных будут выбраны в ходе анализа собранных данных в технологическом разделе работы.

2.2.4 Функция потерь

Функция потери качества определяется как среднеквадратическая ошибка (MSE) между входом и выходом автокодировщика:

$$L_{quality}(input) = \|input - (enc \circ dec)input\|^2, \tag{19}$$

где $input$ — входной вектор признаков.

Можно также использовать взвешенную среднеквадратическую ошибку для того, чтобы придать больше веса параметрам, отвечающим за стиль исполнения — vel и leg.

Функция потери стиля вычисляется в три этапа.

Сперва необходимо вычислить главные компоненты скрытого пространства на основе обучающей выборки. Для этого обучающая выборка обрабатывается зашифровщиком, в результате чего получается так называемая матрица данных X размера $N \times d_l$, где N — число образцов в выборке, а d_l — размерность скрытого пространства. После применения МГК к этой матрице будет получено d_l главных компонент, часть из которых необходимо отбросить.

Число оставляемых главных компонент k необходимо подбирать опытным путем. Чем оно больше, тем более стилизованный фрагмент будет похож на образец стиля в деталях; чем оно меньше, тем менее выраженной будет стилизация. Более того, метод позволяет в дальнейшем провести исследование полученных компонент и отобрать вручную те из них, которые будут сочтены наиболее релевантными для стилизации.

После отбрасывания части компонент будет получена так называемая матрица нагрузок P размером $d_l \times k$, каждый столбец которой — вектор главных компонент. Этот этап необходимо выполнить единожды и сохранить матрицу нагрузок и вектор средних значений a в конфигурационном файле.

Затем необходимо обработать зашифровщиком образец стиля, получив отображение на скрытое пространство, и спроецировать его на k главных компонент:

$$s\_proj = (enc(style) - a) \times P, \tag{20}$$

где $style$ — отображение образца стиля на скрытое пространство.

Этот этап выполняется каждый раз перед началом стилизации.

Наконец, потеря стиля может быть вычислена как среднеквадратичная ошибка между проекциями предсказанного фрагмента и образца стиля:

$$L_{style}(s\_proj, predicted) = \|s\_proj - (enc(predicted) - a) \times P\|^2, \tag{21}$$

где $predicted$ — предсказанный музыкальный фрагмент.

Итоговая функция потерь определяется как сумма функции потери стиля

с коэффициентом a и функции потери качества:

$$L_{total}(s\_proj, predicted) = aL_{style}(s\_proj, predicted) + L_{quality}(predicted). \quad (22)$$

2.2.5 Структура алгоритма

Алгоритм переноса стиля основан на соотношении 2.

Входные данные:

— MIDI-файл стилизуемой партии $content\_file$;

— MIDI-файл с образцом стиля $content\_file$;

— число шагов градиентного спуска $steps$;

— коэффициент стилизации A;

— коэффициент качества B.

Выходные данные: стилизованная партия $stylized$.

Алгоритм 1 Алгоритм переноса стиля исполнения

$content \leftarrow preprocess(content\_file)$ */* Подготовить входную партию */*

$style \leftarrow preprocess(style\_file)$ */* Подготовить образец стиля */*

$vel \leftarrow$ *начальное значение громкости*

$leg \leftarrow$ *начальное значение легатности*

$dt \leftarrow$ *начальное значение оттяжки*

for $i = 1, \ldots, steps$ **do**

    $predicted \leftarrow predict(dt, vel, leg, content)$

    / Вычислить значение функции потерь */*

    $style\_loss \leftarrow get\_style\_loss(predicted, style)$

    $quality\_loss \leftarrow get\_quality\_loss(predicted)$

    $overall\_loss \leftarrow a * style\_loss + quality\_loss$

    / Вычислить градиенты функции потерь относительно переменных* vel, leg *и* dt **/*

    $vel\_grad \leftarrow gradient(overall\_loss, vel)$

    $leg\_grad \leftarrow gradient(overall\_loss, leg)$

    $dt\_grad \leftarrow gradient(overall\_loss, dt)$

    / Применить градиенты к соответствующим переменным и обновить их значения */*

    $vel \leftarrow apply\_gradient(vel, vel\_grad)$

    $leg \leftarrow apply\_gradient(leg, leg\_grad)$

    $dt \leftarrow apply\_gradient(dt, dt\_grad)$

end for

$predicted \leftarrow predict(dt, vel, leg, content)$

$stylized \leftarrow reconstruct(predicted)$ */* Перевести партию в формат MIDI */*

output $predicted$

Для инициализации переменных vel, leg и dt можно использовать случайные значения, выбранные из распределений, соответствующих набору данных.

2.3 Синтез звука

2.3.1 Обработка звукового корпуса

Для достижения наилучшего качества синтезируемого фрагмента звуковые файлы необходимо обработать таким образом, чтобы оставить минимальный интервал времени до момента начала звука. Для этого требуется достаточно точно определить этот момент, чтобы сохранить фазу атаки звука

Один из эффективных подходов заключается в поиске точки максимального перепада громкости, которая указывает на начало атаки звука после тишины, поскольку обычно фортепиано имеет резкую атаку. Для этого применяется алгоритм троичного поиска.

Для эффективного поиска звуковых фрагментов в процессе синтеза необходимо сформировать и сохранить для каждого фрагмента дескриптор со следующей информацией:

— id — уникальный идентификатор фрагмента;

— path — путь к аудио-файлу;

— tone — MIDI-значение высоты тона;

— velocity — MIDI-значение интенсивности звука;

— volume — громкость фрагмента.

Для вычисления громкости дискретного аудио-сигнала на участке $[a; b)$ можно использовать метрику RMSE — квадратный корень из среднеквадратической ошибки — которая часто используется в аудио-инженерии для оценки громкости звука:

$$loudness = \sqrt{\frac{1}{b-a}\sum_{x=a}^{x<b}(signal(x) - mean)^2}, \tag{23}$$

где $mean$ - среднее значение сигнала на участке $[a; b)$.

2.3.2 Особенности реализации

В отличие от некоторых других музыкальных инструментов, у фортепиано отсутствуют плавные переходы между нотами. Каждая нота на фортепиано звучит отдельно и имеет свою характерную атаку и затухание. Благодаря этому факту можно принять стоимость конкатенации p_c^i равной нулю.

На рисунке 9 изображена UML-диаграмма классов модуля синтеза звука.

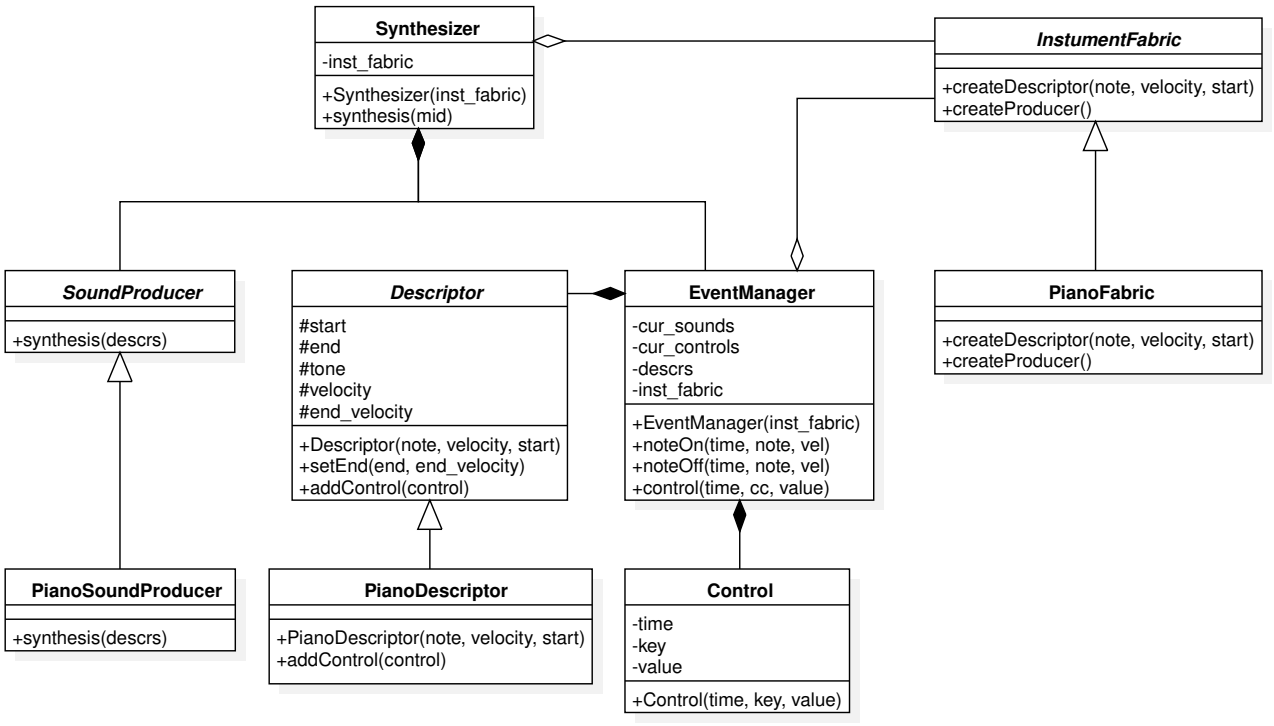

Рисунок 9 – Диаграмма классов модуля синтеза звука

В целях обеспечения универсальности и гибкости системы, при проектировании было принято решение минимизировать зависимость от конкретного музыкального инструмента. Для этого был применен паттерн «абстрактная фабрика». Все классы, связанные с инструментом, были объявлены абстрактными, а конкретная реализация функционала для фортепиано была вынесена в дочерние классы.

В классе «Synthesiser», который по сути представляет собой фасад для работы с синтезатором, используется ссылка на объект класса «InstrumentFabric». Благодаря данной реализации, классы системы могут создавать объекты, соответствующие нужному инструменту, в данном случае фортепиано. Это позволяет легко расширять систему и добавлять поддержку других инструментов в будущем без изменения основной структуры.

Коэффициент различия было решено определить следующим образом:

$$p(descr, sound) = w(descr_{tone} - sound_{tone})^2 + |descr_{velocity} - sound_{velocity}|, \quad (24)$$

где $descr_{tone}$ и $sound_{tone}$ — высота тона дескриптора и звука соответственно, $descr_{velocity}$ и $sound_{velocity}$ — громкость дескриптора и звука соответственно, а w — весовой коэффициент.

После того, как для дескриптора подобран звуковой фрагмент с наименьшим коэффициентом различия, его необходимо преобразовать для полного соответствия дескриптору. Для этого он транспонируется на $descr_{tone} - sound_{tone}$ полутонов, а затем сигнал умножается на $\frac{descr_{velocity}}{sound_{velocity}}$ для преобразования громкости.

2.4 Вывод

Были разработаны основные положения предлагаемого метода реалистичного воспроизведения музыкальных партий с переносом стиля. Были изложены особенности предлагаемого метода переноса стиля, а также сформулированы и описаны ключевые этапы метода. Были описаны специфические особенности реализации конкатенативного синтеза звука, присущие решаемой задаче. Был спроектирован модуль синтеза звука.

3 Технологическая часть

В данном разделе описываются основные моменты разработки программного обеспечения. Перечисляются средства разработки, описываются также формат конфигурационного файла, пользовательский интерфейс, приводится инструкция по запуску ПО.

Рассматривается процесс обучения нейронной сети: сбор и предварительная обработка данных, методы валидации и кривые обучения. Также приводится итоговая архитектура модели.

3.1 Выбор средств разработки

В рамках данной дипломной работы были выбраны следующие средства разработки:

1) Python. Python является высокоуровневым языком программирования с динамической типизацией, который отличается простотой синтаксиса и ясностью кода. Python широко используется в области анализа данных и машинного обучения, и имеет огромное сообщество разработчиков и множество библиотек для различных задач.
2) TensorFlow. Эта библиотека, разработанная командой Google Brain, была выбрана в связи с её гибкостью, мощностью и широкими возможностями для создания и обучения нейронных сетей. TensorFlow предоставляет интуитивно понятный API для автоматического дифференцирования и позволяет использовать аппаратное ускорение с помощью GPU.
3) NumPy. Эта библиотека Python предоставляет поддержку больших многомерных массивов, вместе с большой библиотекой высокоуровневых математических функций для операций с этими массивами. NumPy является фундаментальной библиотекой для научных вычислений в Python.
4) Matplotlib. Matplotlib - это библиотека для создания статических, анимированных и интерактивных визуализаций в Python. Matplotlib был выбран из-за его гибкости и мощных средств визуализации, которые оказываются

крайне полезными при анализе и отображении результатов модели.

5) Mido. Mido - это библиотека для работы с MIDI-файлами и сообщениями. Она предоставляет удобные средства для чтения, записи и обработки MIDI-файлов, что было критически важно для данной работы.
6) PyQt5. PyQt5 - это библиотека для создания графического пользовательского интерфейса (GUI). Она была выбрана для реализации интерфейса приложения.
7) SQLite3. Библиотека была выбрана для реализации базы данных звукового корпуса.

3.2 Обучение нейронной сети

3.2.1 Сбор данных

Сбор данных является важным этапом работы, поскольку качество, объем и разнообразие данных имеют прямое влияние на эффективность и обобщающую способность модели. Поэтому для обучения модели было решено использовать два набора данных: GiantMIDI-Piano и MAESTRO Dataset.

Набор данных GiantMIDI-Piano [11] является одним из крупнейших открытых наборов данных с информацией о музыкальных произведениях. Он содержит 10855 MIDI-файлов с профессиональными исполнениями классических фортепианных произведений 2786 композиторов.

Набор данных MAESTRO Dataset [12] в свою очередь содержит более 200 часов виртуозных исполнений классических произведений, а также соответствующих им файлов MIDI. Этот набор данных также предоставляет разнообразие в исполнительском стиле и содержит музыку разных эпох и жанров.

Для преобразования MIDI-файлов, полученных путем объединения двух наборов данных, в монофонические последовательности нот было найдено программное обеспечение [31] для разделения MIDI-файла на отдельные голоса.

3.2.2 Нормализация данных

На рисунке 10 представлены гистограммы параметров ноты. Исследование плотности распределения параметров позволяет определить подход к их

нормализации. В данном случае было решено применить стандартизацию для высоты тона и громкости ноты, то есть привести их к нулевому среднему и единичному среднеквадратичному отклонению; длительность нормализована к интервалу $[0; 1]$, а из значения легатности было решено вычесть единицу, чтобы получить значение в интервале $[-1; 1]$.

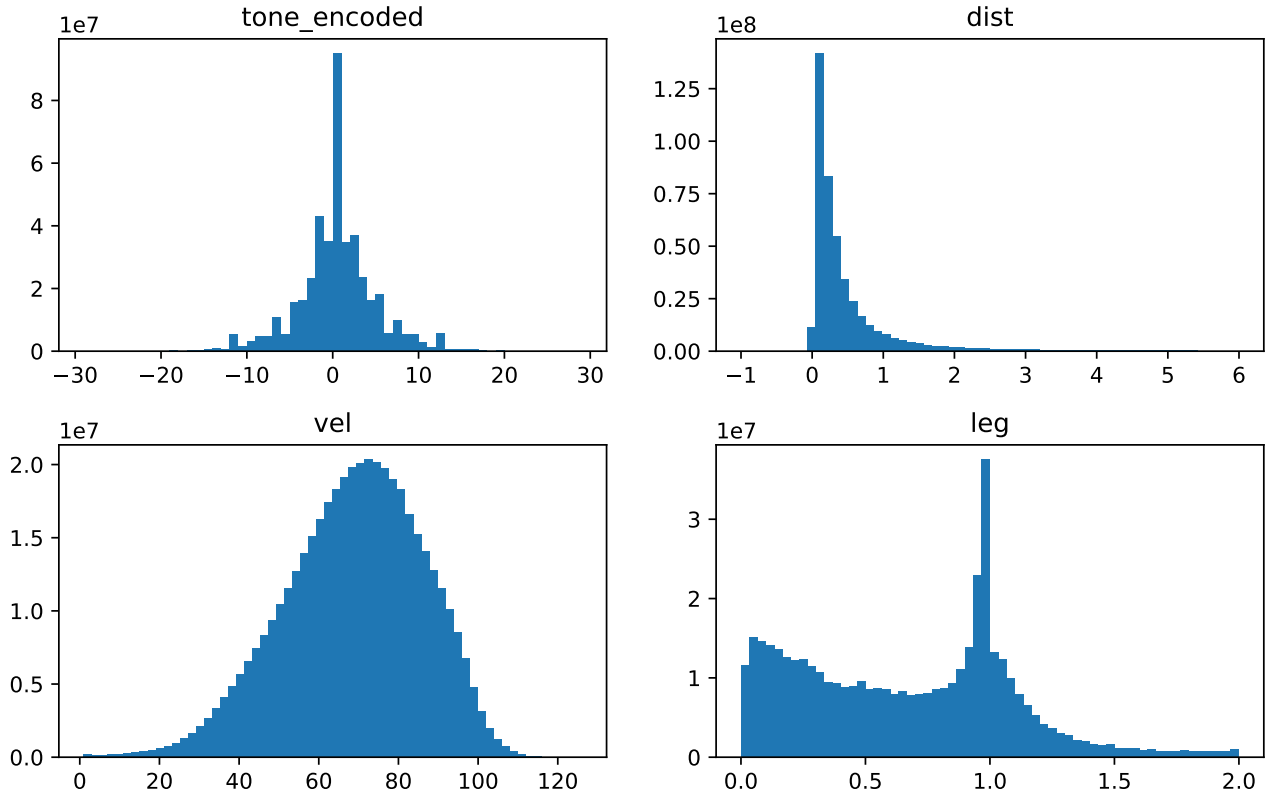

Рисунок 10 – Гистограммы параметров ноты

Итоговые преобразования параметров можно записать следующим образом:

$$tone\_preprocessed_j = \frac{tone\_encoded_j - \mu_{tone\_encoded}}{\sigma_{tone\_encoded}}, \tag{25}$$

$$dist\_preprocessed_i = \frac{dist_i}{max_{dist}}, \tag{26}$$

$$vel\_preprocessed_i = \frac{vel_i - \mu_{vel}}{\sigma_{vel}}, \tag{27}$$

$$leg\_preprocessed_i = leg_i - 1, \tag{28}$$

где $\mu_{tone\_encoded}$ и μ_{vel} – средние закодированная высота тона и интенсивность ноты по обучающей выборке соответственно, $\sigma_{tone\_encoded}$ и σ_{vel} – среднеквадратичные отклонения высоты тона и интенсивности ноты по обучающей выборке соответственно, max_{dist} – максимальная длительность ноты по обучающей выборке, $i = 1 \ldots n$, $j = 1 \ldots n$-1.

3.2.3 Валидация

После формирования итоговой выборки данных она была разделена на две части: обучающую и тестовую.

Для валидации модели была определена собственная метрика качества, дополнительно к использованию среднеквадратичной ошибки (MSE). Методика оценки качества включала следующие шаги:

1) Тестовая выборка была разделена на n групп, где n представляет собой количество нот в последовательности.
2) В каждом образце из i-й группы случайным образом выбирались i нот, и параметры исполнения для этих нот заменялись случайными значениями, сэмплированными из соответствующего распределения параметров.

3) Для каждой группы вычислялась разность между входом и выходом автокодировщика для зашумленных образцов и соответствующих нетронутых образцов (оригиналов).
4) Была вычислена разность между полученными показателями.
5) После этого были рассчитаны среднее значение и коэффициент вариации для полученных величин.

Значение метрики можно вычислить по формуле:

$$metric_i = \frac{1}{group\_size} \sum_{j=1}^{group\_size} (L_{quality}(noisy_{ij}) - L_{quality}(original_{ij})), \quad (29)$$

где $group\_size$ – размер группы, а $noisy_{ij}$ и $original_{ij}$ зашумленный и первоначальный фрагменты соответственно.

Данная методика позволяет оценить, насколько большие значения $L_{quality}$ выдает модель для зашумленных музыкальных фрагментов в зависимости от того, сколько нот было искажено.

3.2.4 Обучение модели

Далее представлен обзор итоговой модели, в то время как процесс подбора ее параметров будет описан в экспериментальном разделе работы.

Размерность скрытого пространства d_l выбрана равной 64. Входной вектор признаков имеет длину $4n - 1 = 255$.

Архитектура зашифровщика представлена в таблице 3.1. Сеть содержит 3 скрытых полносвязных слоя с функцией активации ReLu. Функция активации выходного слоя – Tanh. Общее количество обучаемых параметров в зашифровщике составляет 61 696.

Таблица 3.1 – Архитектура зашифровщика

Номер слоя	Тип слоя	Кол-во нейронов	Функция активации	Число обучаемых параметров
1	Входной	255	–	0
2	Скрытый	128	ReLu	32 768
3	Скрытый	128	ReLu	16 512
4	Скрытый	64	ReLu	8 256
5	Выходной	64	Tanh	4 160

В таблице 3.2 представлена архитектура дешифровщика, который получает на вход выходной вектор зашифровщика. Архитектура дешифровщика симметрична архитектуре зашифровщика. Общее число обучаемых параметров дешифровщика составляет 61 887.

Таблица 3.2 – Архитектура дешифровщика

Номер слоя	Тип слоя	Кол-во нейронов	Функция активации	Число обучаемых параметров
1	Входной	64	–	0
2	Скрытый	64	ReLu	4 160
3	Скрытый	128	ReLu	8 320
4	Скрытый	128	ReLu	16 512
5	Выходной	255	Linear	32 895

Таким образом, общее число обучаемых параметров модели составляет 123 583.

Модель была обучена с использованием размера пакета (batch size) 64. С учетом размера обучающей выборки 5 267 234 образцов, количество шагов в одной эпохе составляло 82 300.

Обучение модели проводилось в течение 60 эпох.

На рисунке 11 представлены кривые обучения. Можно заметить, что ошибка на тестовой выборке ниже, чем ошибка на обучающей выборке. Это объясняется тем, что в процессе валидации не используется регуляризация, в частности, добавление шума к данным.

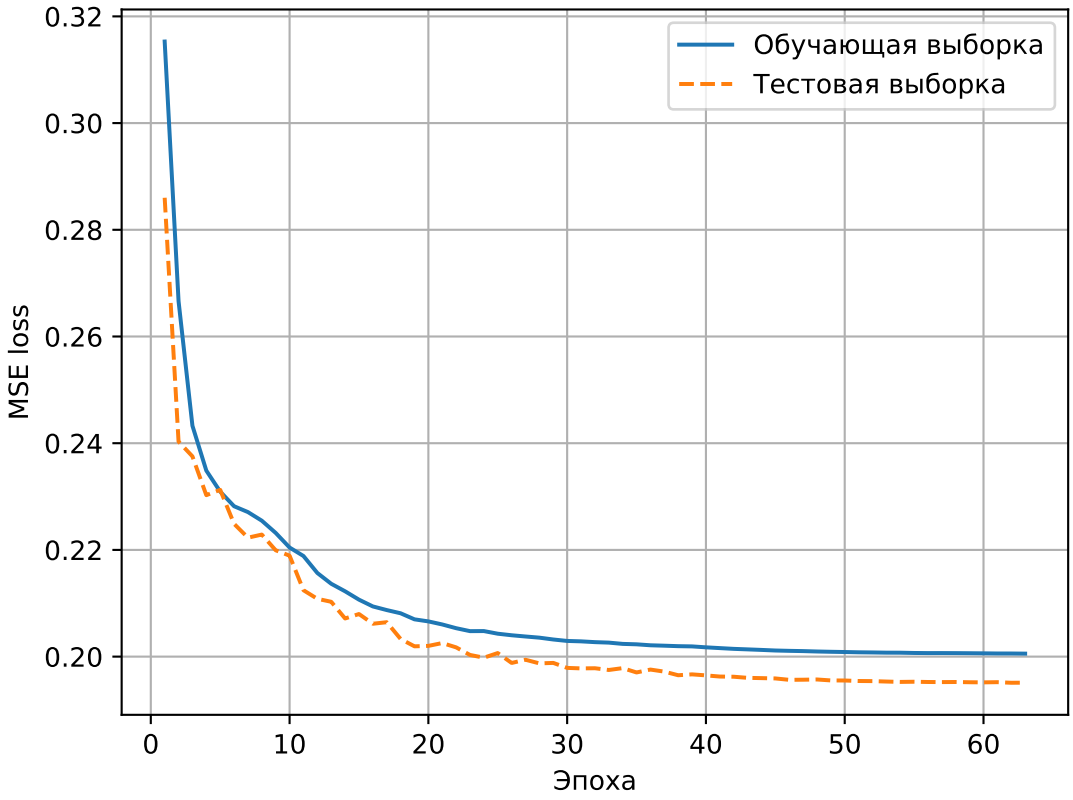

Рисунок 11 – Кривые обучения

На рисунке 12 представлен анализ итоговой модели с точки зрения собственной метрики качества. Верхние два графика демонстрируют зависимость среднего значения и коэффициента вариации для разности $L_{quality}$ между искаженными и нетронутыми фрагментами в зависимости от количества искаженных нот. На нижних графиках представлены гистограммы той же величины для различного числа искаженных нот (1, 4, 32 и 64). На каждом графике присутствуют две кривые, отражающие искажение громкости и легатности нот.

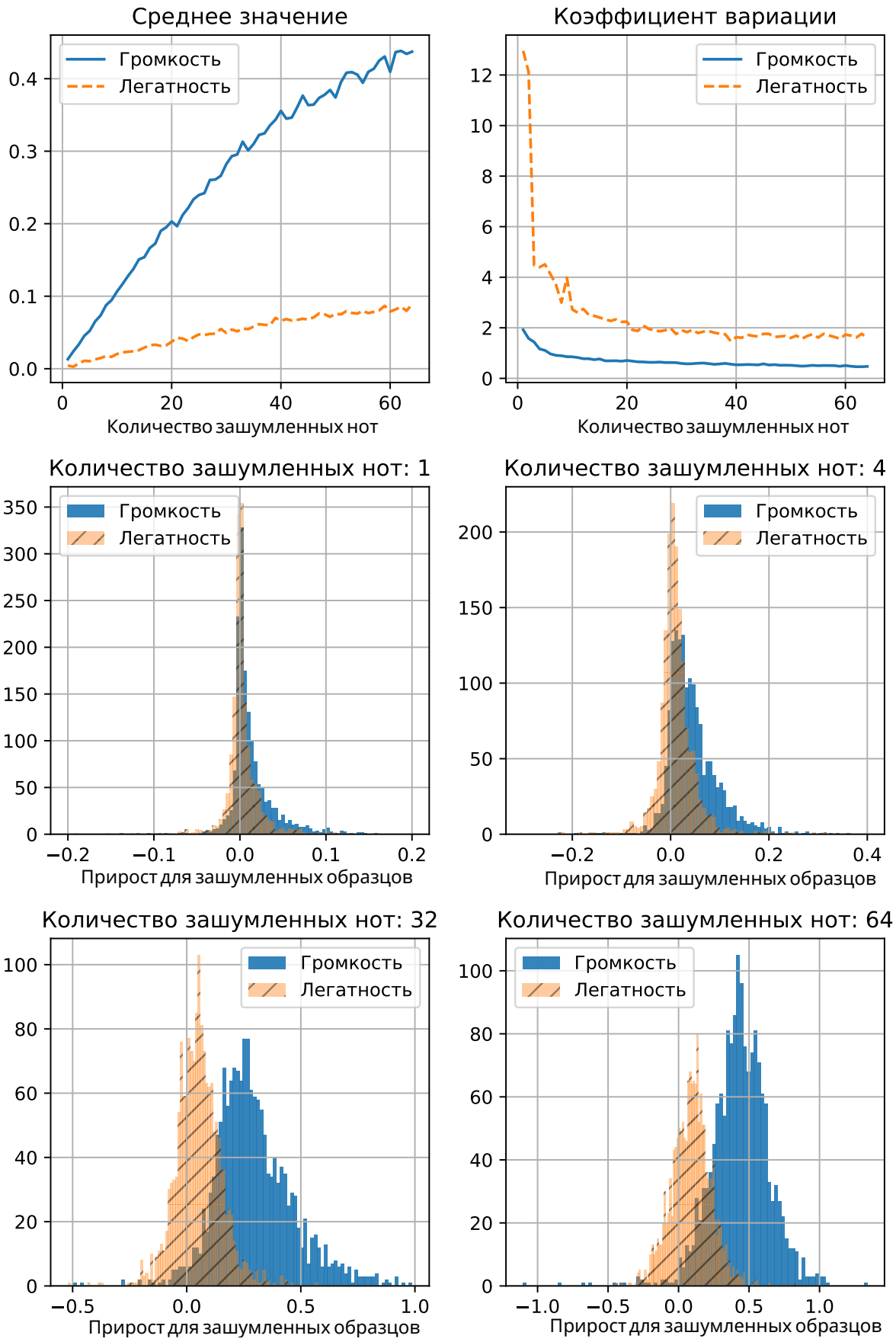

Рисунок 12 – Собственная метрика качества

На графиках отчетливо видно, что $L_{quality}$ тем выше, чем больше нот искажено; при этом даже при одной искаженной ноте $L_{quality}$ зашумленного фрагмента в среднем выше. Это явно свидетельствует о работоспособности модели.

Также можно заметить, что модель лучше справляется с искажениями громкости, чем с искажениями легатности. Это объясняется тем, что легатность является временной характеристикой и тесно связана с длительностью ноты, что делает ее обработку более сложной для полносвязной нейронной сети.

3.3 Руководство пользователя

3.3.1 Формат конфигурационного файла

Конфигурационный файл программы имеет формат json. Пример такого файла представлен на листинге 2.

Листинг 1: Пример конфигурационного файла

```
{
    "data_processor": {
        "notes_qty": 64,
        "config_path": "./norm_params.txt"
    },
    "autoencoder": {
        "type": "MLPAutoEncoder",
        "latent_dim": 64,
        "encoder_layers": (128, 128, 64),
        "decoder_layers": (64, 128, 128)
    },
    "occ": {
        "dist_weight": 4,
        "vel_weight": 1,
        "leg_weight": 4,
        "checkpoint_path": "./ckpt-60"
    },
    "pca": {
        "pca_dim": 32,
        "config_path": "./pca_params_32.txt"
    },
    "synth": {
        "type": "piano",
        "sounds_dir": "./corpus_wav_lufs"
    }
}
```

В разделе «data_processor» указываются параметры обработки данных, такие как тип обработчика, количество нот в последовательности, использование абсолютной громкости и путь к файлу с сохраненными параметрами нормализации.

Раздел «autoencoder» определяет параметры автокодировщика, включая его тип, размер скрытого пространства и конфигурации слоев для кодировщика и декодировщика.

Раздел «occ» содержит веса для расчета метрики качества. Также указывается путь к контрольной точке (checkpoint) для загрузки модели.

Раздел «pca» определяет параметры метода главных компонент (PCA), включая число компонент и путь к файлу с матрицей нагрузок.

В разделе «synth» указывается тип синтезатора (в данном случае «piano») и путь к директории, содержащей звуковые файлы для синтеза.

3.3.2 Интерфейс пользователя

Для запуска программы необходимо установить интерпретатор Python 3, после чего выполнить следующие команды:

Листинг 2: Запуск программы

```
> cd src
> pip install -r requirements.txt
> python main.py
```

Графический пользовательский интерфейс представлен на рисунке 13. Пользователю необходимо загрузить конфигурационный файл, нажав на кнопку «Load config», MIDI-файл с партией (кнопка «open...» напротив «Content MIDI») и MIDI-файл с образцом стиля (кнопка «open...» напротив «Style MIDI»). Настроить параметры стилизации можно при помощи ползунков:

— «Style ratio» соответствует коэффициенту стилизации A.

— «Max tenuto» задает максимальную оттяжку в секундах.

— «Time limit» задает ограничение по времени работы метода стилизации в минутах.

После настройки программы можно нажать кнопку «Create MIDI» для стилизации партии и сохранения результата в формате MIDI, либо «Create WAV» для стилизации партии, синтеза звука и сохранения результата в формате WAV.

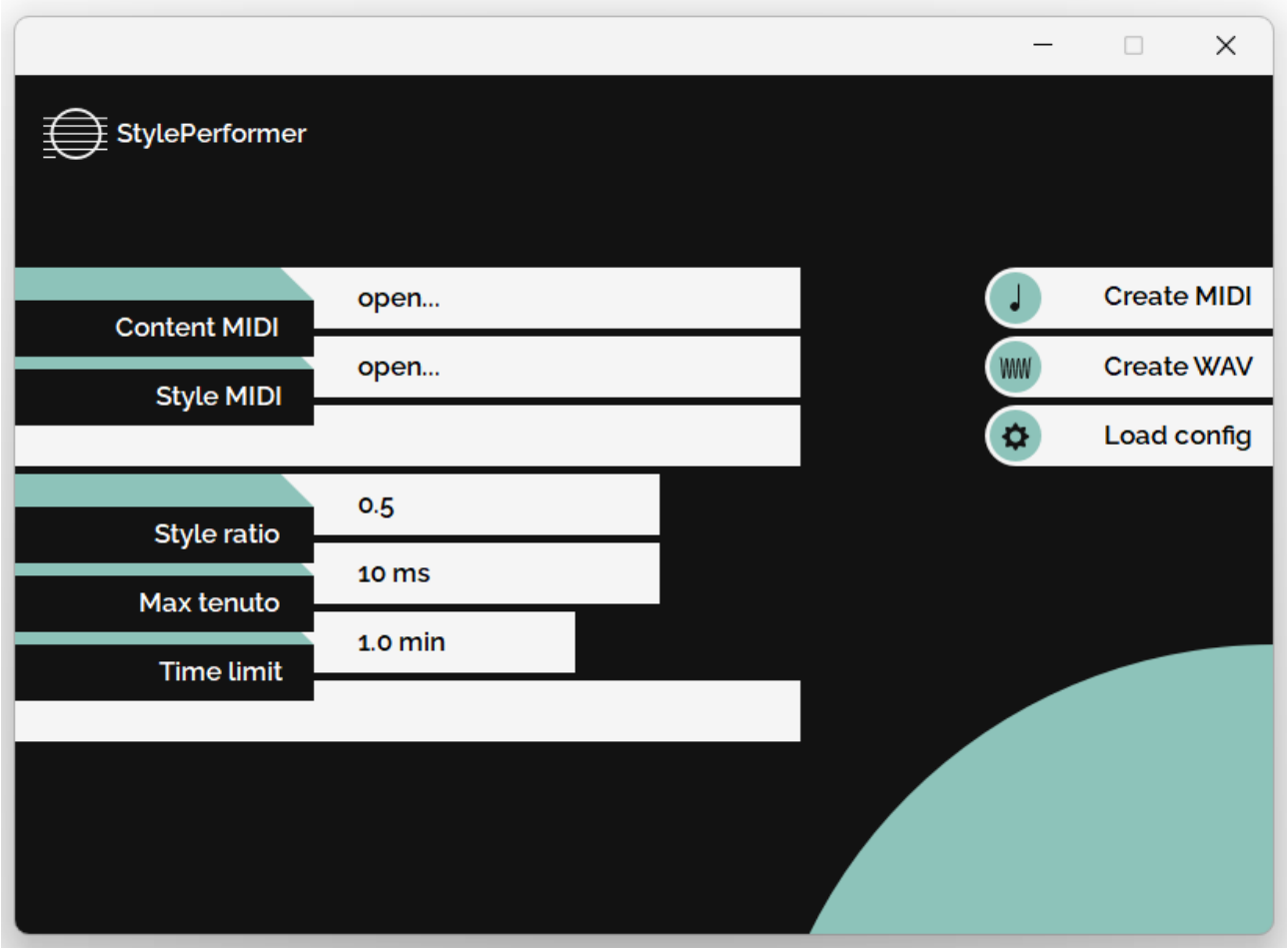

Рисунок 13 – Интерфейс программы

3.4 Вывод

В данном разделе были рассмотрены основные моменты разработки ПО, а также средства разработки, формат конфигурационного файла и пользовательский интерфейс программы.

В данном разделе были описаны основные моменты разработки программного обеспечения. Были перечислены средства разработки, описаны формат

конфигурационного файла, пользовательский интерфейс, приведена инструкция по запуску ПО.

Был рассмотрен процесс обучения нейронной сети: сбор и предварительная обработка данных, методы валидации и кривые обучения. Также была приведена итоговая архитектура модели.

4 Исследовательская часть

В данном разделе исследуются зависимости результата работы программного обеспечения от различных параметров системы. Приводится описание процесса подбора гиперпараметров модели. Исследуется метод переноса стиля.

Проводится экспертная оценка разработанного метода реалистичного воспроизведения музыкальных партий с переносом стиля. Исследуется влияние размера звукового корпуса на качество звука.

4.1 Подбор гиперпараметров модели

При настройке гиперпараметров нейронной сети был применен метод поиска по сетке. При этом основное внимание было сосредоточено на изучении влияния на точность модели следующих параметров: размерность скрытого пространства, количество слоев и нейронов и стандартное отклонение шума на входе.

Для определения оптимальной размерности скрытого пространства были опробованы значения 32 и 64.

Следующим параметром является количество скрытых слоев и нейронов в каждом из них. Здесь были проведены испытания сетей с 1, 2 и 3 слоями, где количество нейронов в каждом слое определялось степенями двойки, и каждый последующий слой не превышал предыдущий по количеству нейронов.

Для стандартного отклонения шума на входе были опробованы значения 0.2, 0.1, 0.05.

Процесс перебора гиперпараметров был автоматизирован. Для сравнения различных конфигурация применялась описанная в конструкторском разделе собственная метрика. .

4.2 Оценка качества стилизации

Для оценки качества стилизации был проведен следующий эксперимент. Случайным образом из тестовой выборки были выбраны два произведения в качестве образцов стиля. Далее, так же случайным образом, из тестовой вы-

борки были выбраны 20 произведений. Каждое из этих 20 произведений было стилизовано в каждом из двух стилей.

Затем было определено трехмерное пространство стиля на основе трех примитивных признаков: среднего значения и среднеквадратичного отклонения громкости, а также среднего значение легатности. Для всех произведений — двух образцов стиля, 20 случайных произведений и 40 стилизованных произведений — были вычислены соответствующие компоненты.

На рисунке 14 изображен эпюр — визуализация этого пространства стиля для числа главных компонент $k = 32$.

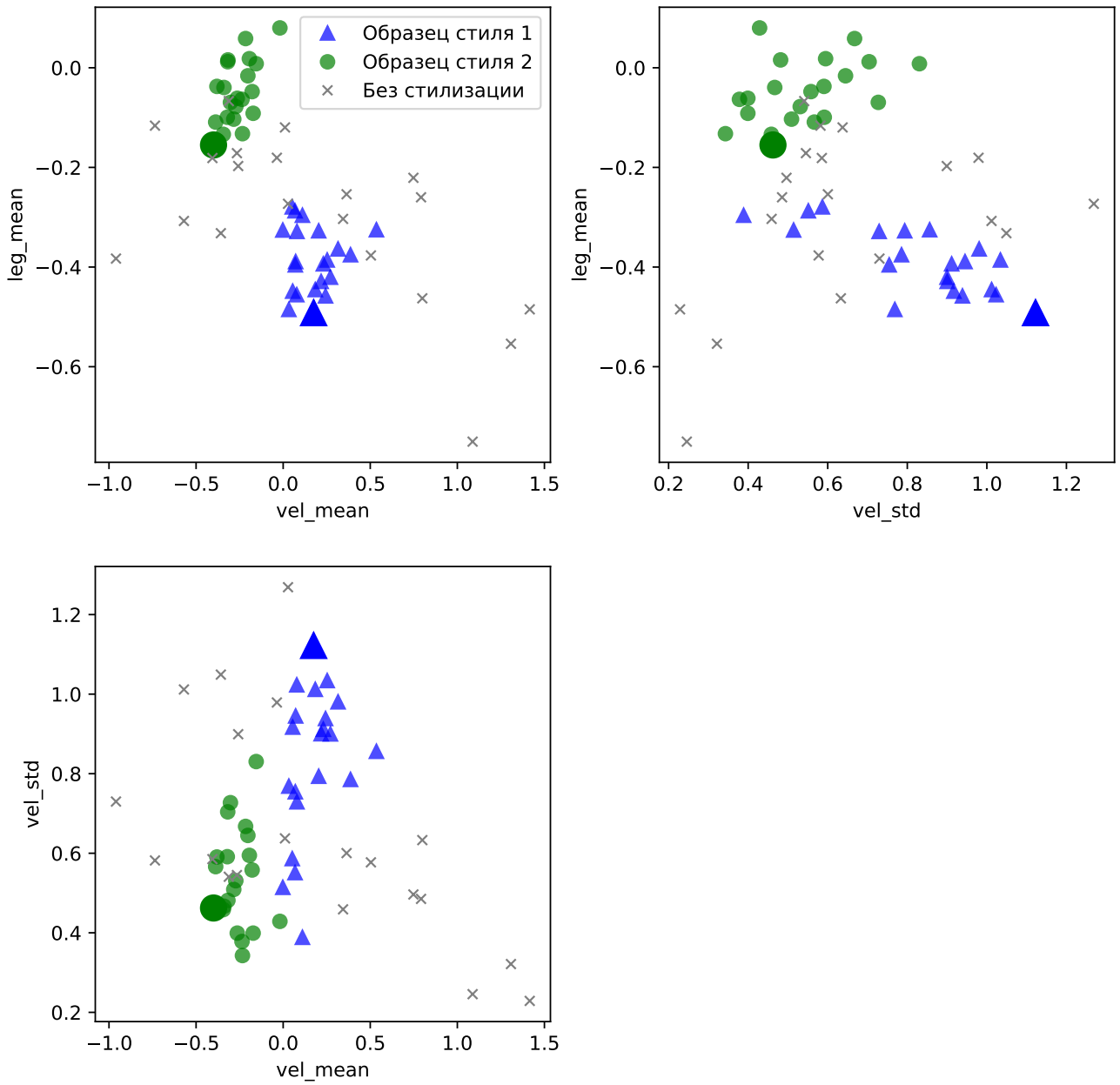

Рисунок 14 – Результаты стилизации; $k = 32$

Анализируя графики, можно заметить, что стилизованные произведения образуют кластеры, расположенные близко к соответствующим образцам стиля, что подтверждается средним значением силуэта для образцов стиля, равным 0.83. Это говорит о том, что алгоритм стилизации успешно переносит стиль с образцов на другие произведения.

На рисунке 15 представлены аналогичные графики для $k = 1$. Заметно, что кластеры имеют меньшую выраженность, и число силуэта равно 0.23. Это объясняется тем, что при отбрасывании главных компонент теряется часть информации о стиле.

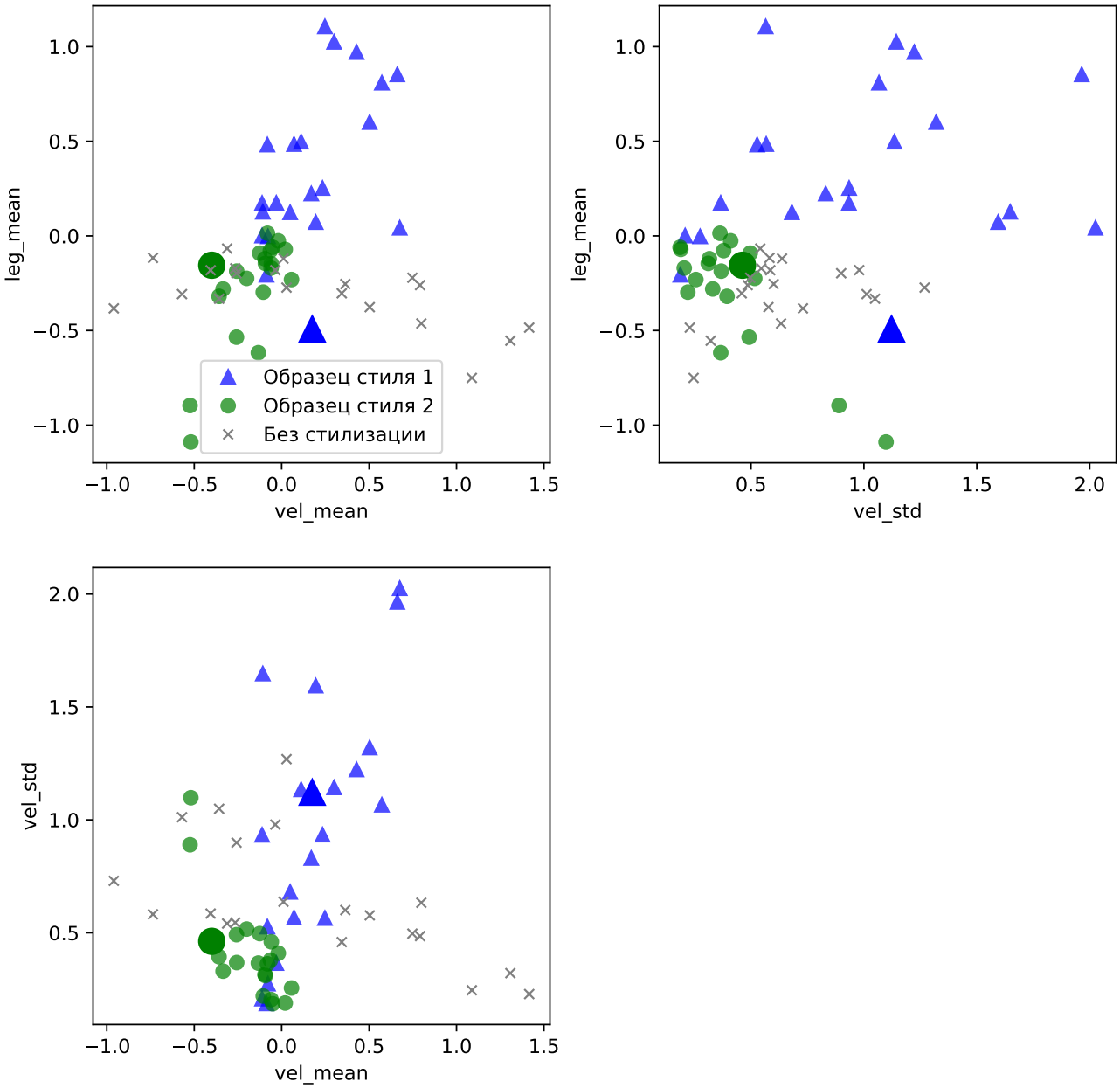

Рисунок 15 – Результаты стилизации; $k = 1$

Выбор числа главных компонент зависит от желаемой степени детализации стилизации. Например, при использовании $k = 64$ можно добиться очень подробного переноса динамики, однако это не всегда имеет смысл: если обра-

зец стиля содержит крещендо, это не означает, что в стилизованном фрагменте также должно быть крещендо.

В данной работе опытным путем было выбрано значение $k = 32$. Такое значение обеспечивает достаточную субъективную и объективную оценку схожести стилей, при этом не стремясь к точному подражанию образцу стиля во всех деталях.

4.3 Экспертная оценка

Для анализа результатов работы разработанной системы была проведена экспертная оценка. В ней приняли участие шесть выпускников Российской академии музыки имени Гнесиных [32], имеющих профессиональные музыкальные навыки и опыт, что делает их мнение крайне ценным в контексте данного исследования.

Экспертам было предложено оценить два ключевых аспекта системы: реалистичность звучания инструмента и реалистичность исполнения музыкальных композиций.

4.3.1 Оценка реалистичности звучания инструмента

В первом разделе опроса участникам было предложено оценить звучание двух музыкальных произведений, синтезированных различными системами:

— системой, разработанной в рамках настоящего исследования;
— профессиональным ПО XLN Audio Addictive Keys;
— профессиональным ПО ProjectSAM Swing.

Выбранные для оценки произведения представляют различные стили игры: первое - «Прелюдия соль минор» из цикла «24 прелюдии» Op. 23, No. 5 С. Рахманинова, характеризующееся быстрым темпом и отрывистым звукоизвлечением; второе - «Gnossienne No. 1» Э. Сати, в котором, наоборот, преобладает медленное, легатное исполнение.

Участникам опроса было предложено оценить реалистичность звучания инструмента в каждом из прослушанных фрагментов по семиступенчатой шкале: от 1, что соответствует полностью искусственному звучанию, до 7, обозна-

чающего звучание, неотличимое от настоящего инструмента. Усредненные результаты данной части опроса представлены на диаграмме 16.

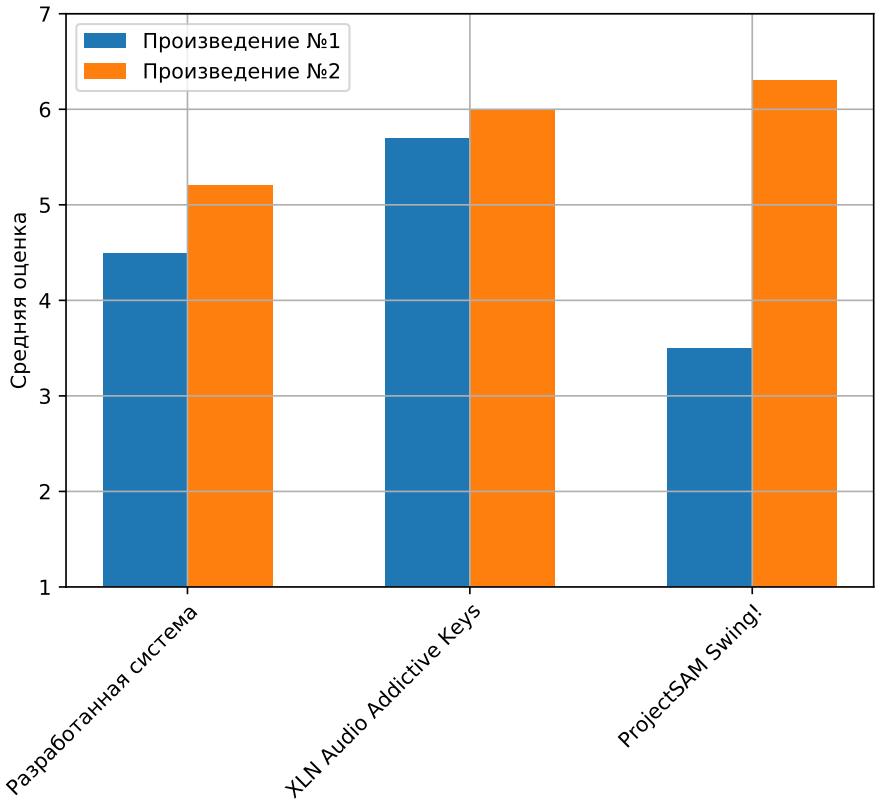

Рисунок 16 – Результаты первой части опроса

Исходя из результатов опроса, можно заключить, что разработанная в рамках данной работы система позволила добиться реалистичности звучания на уровне профессиональных продуктов, и это даже при использовании бесплатных звуковых образцов. Однако, стоит отметить, что профессиональные продукты все еще имеют ряд преимуществ, включая высококачественную запись звука.

4.3.2 Оценка реалистичности и выразительности исполнения

В рамках второго раздела опроса участникам было предложено оценить различные исполнения трех музыкальных произведений, представленных в следующих вариациях:

— фрагменты, стилизованные с использованием разработанного в ходе дан-

ной работы ПО;

— исходные MIDI-файлы, не содержащие исполнительских особенностей;
— фрагменты, исполненные человеком.

Для обеспечения объективности сравнения, звук во всех трех вариантах генерировался одной и той же программой. Кроме того, в исполнениях, выполненных человеком, были убраны ноты, которые система не может воспроизвести из-за монофонического ограничения.

Произведения, выбранные для опроса, включают в себя следующие композиции: Op. 76, No. 2 Яна Сибелиуса, Op. 37a Петра Чайковского, и Op. 7, No. 2 Фредерика Шопена. Перенос стиля осуществлялся с прелюдии «Бергамасской сюиты» Клода Дебюсси.

Участникам опроса было предложено оценить каждый из представленных фрагментов по степени реалистичности и выразительности на 7-балльной шкале. Затем вычислялось среднее из этих двух показателей.

Результаты этого раздела опроса представлены на рисунке 17

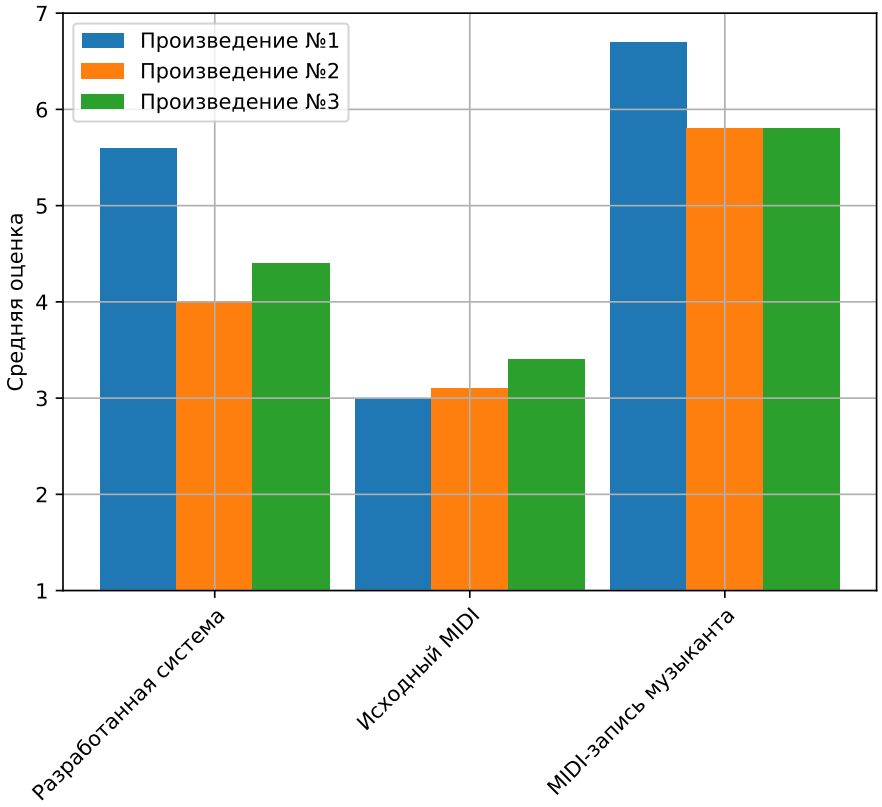

Рисунок 17 – Результаты второй части опроса

Бесспорно, в сравнении с человеческим исполнением разработанное ПО уступает, однако, введение стилизации заметно повышает реалистичность и выразительность исполнения.

4.4 Влияние размера звукового корпуса

В соответствии с техническим заданием необходимо также установить зависимость результата работы метода от размера звукового корпуса. Для этого посчитаем средний коэффициент различия для одной и той же синтезируемой партии, но при использовании звуковых корпусов разного объема.

При формировании нового звукового корпуса меньшего размера мы не просто урезаем существующий корпус, а выбираем элементы из него случайным образом.

На рисунке 18 изображена зависимость среднего значения коэффициента различия от размера звукового корпуса.

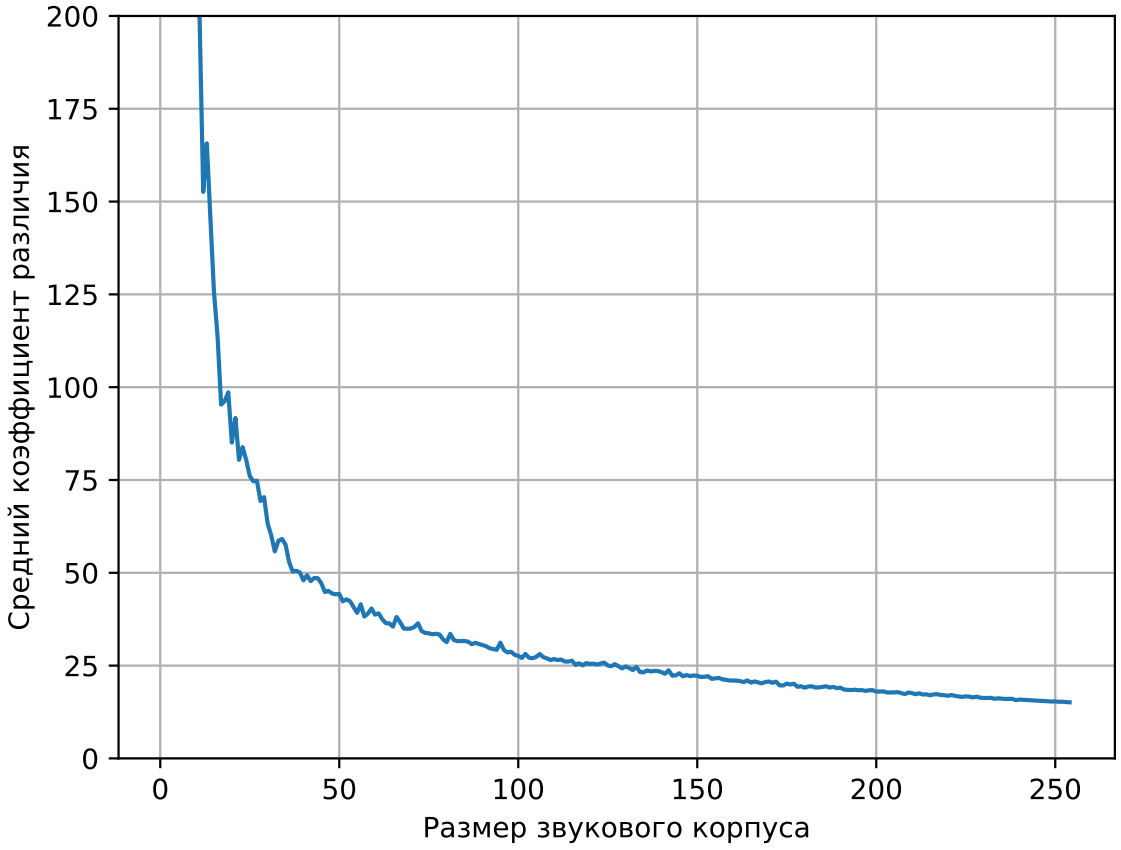

Рисунок 18 – Зависимость среднего значения коэффициента различия от размера звукового корпуса

Согласно полученным результатам, с ростом размера звукового корпуса наблюдается снижение среднего коэффициента различия. Это связано с тем, что с увеличением числа звуковых фрагментов в корпусе требуется меньше преобразований для сопоставления фрагментов с дескрипторами, что ведет к улучшению сходства. При этом темп уменьшения коэффициента постепенно снижается. Это указывает на то, что для дальнейшего улучшения качества звука потребуется существенное увеличение объема звукового корпуса.

Идеальным сценарием, при котором коэффициент различия достигнет нуля, будет ситуация, в которой корпус включает все возможные комбинации высоты ноты и интенсивности. Это означает, что каждый возможный фрагмент уже представлен в корпусе, обеспечивая максимальную точность сопоставле-

ния.

4.5 Вывод

В данном разделе были исследованы зависимости результата работы программного обеспечения от различных параметров системы. Было приведено описание процесса подбора гиперпараметров модели. Был исследован метод переноса стиля.

Была проведена экспертная оценка разработанного метода реалистичного воспроизведения музыкальных партий с переносом стиля и проанализированы ее результаты. Было исследовано влияние размера звукового корпуса на качество звука.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы был разработан метод реалистичного воспроизведения музыкальных партий с переносом стиля исполнения.

Были успешно выполнены все поставленные задачи, в том числе:

— проведен анализ предметной области;
— сформулирована постановка проблемы;
— рассмотрены существующие подходы к решению проблемы;
— разработан метод реалистичного воспроизведения музыкальных партий с переносом стиля;
— разработано программное обеспечение на основе предложенного метода;
— исследованы характеристики разработанного метода;
— проведена экспертная оценка метода.

Полученные результаты открывают перспективы для дальнейших исследований и разработок. В качестве возможных направлений для будущей работы можно предложить следующие:

— реализация поддержки полифонических произведений;
— увеличение числа варьируемых параметров исполнения;
— реализация других музыкальных инструментов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Bontempi P. et al. Research in Computational Expressive Music Performance and Popular Music Production: A Potential Field of Application? //Multimodal Technologies and Interaction. – 2023. – Т. 7. – №. 2. – С. 15.

2. Лебедев С. Н., Трубинов П. Ю. Нотация // Большая российская энциклопедия. М.: 2013. – Т. 23. – С. 353-355.

3. Turker M., Dirik A., Yanardag P. MIDISpace: Finding Linear Directions in Latent Space for Music Generation //Creativity and Cognition. 2022. С. 420-427.

4. Brunner G. et al. MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer [Электронный ресурс] – Режим доступа: https://arxiv.org/abs/1809.07600, свободный (дата обращения: 09.05.2023).

5. Wang L. et al. Music genre classification based on multiple classifier fusion //2008 Fourth International Conference on Natural Computation. IEEE – 2008. – Т. 5. – С. 580-583.

6. Tzanetakis G., Cook P. Musical genre classification of audio signals //IEEE Transactions on speech and audio processing. 2002. – Т. 10. – №. 5. – С. 293-302.

7. Logan B., Salomon A. A Music Similarity Function Based on Signal Analysis //ICME. – 2001. – С. 22-25.

8. Gouyon F. et al. Evaluating rhythmic descriptors for musical genre classification //Proceedings of the AES 25th International Conference. 2004. – Т. 196. – С. 204.

9. Борисенкова А. С., Корухова Ю. С. Об одном подходе к определению авторства музыкальных произведений //Научный сервис в сети Интернет. – 2015. – С. 34-41.

10. Окунев С. В. Перенос стиля изображения с применением генеративно-состязательной нейронной сети // Решетневские чтения: Материалы XXIV

Международной научно-практической конференции, посвященной памяти генерального конструктора ракетно-космических систем академика М. Ф. Решетнева. Часть 2. – Красноярск: ФГБОУ ВО ”Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева”. – 2020. – С. 426-427.

11. Kong Q. et al. Giantmidi-piano: A large-scale midi dataset for classical piano music [Электронный ресурс] – Режим доступа: https://arxiv.org/abs/2010.07061, свободный (дата обращения: 09.05.2023).

12. Hawthorne C. et al. Enabling factorized piano music modeling and generation with the MAESTRO dataset [Электронный ресурс] – Режим доступа: https://arxiv.org/abs/1810.12247, свободный (дата обращения: 09.05.2023).

13. Лазарев А. С. Автокодировщик // Том 3 : Инновационные технологии в науке нового времени. Сборник статей Международной научно-практической конференции. — 2017. – С. 82-84.

14. Perera P., Nallapati R., Xiang B. Ocgan: One-class novelty detection using gans with constrained latent representations //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. – 2019. – С. 2898-2906.

15. Härkönen E. et al. Ganspace: Discovering interpretable gan controls //Advances in Neural Information Processing Systems. – 2020. – Т. 33. – С. 9841-9850.

16. Hasan B. M. S., Abdulazeez A. M. A review of principal component analysis algorithm for dimensionality reduction //Journal of Soft Computing and Data Mining. – 2021. – Т. 2. – №. 1. – С. 20-30.

17. Kingma D. P., Ba J. Adam: A method for stochastic optimization [Электронный ресурс] – Режим доступа: https://arxiv.org/abs/1412.6980, свободный (дата обращения: 09.05.2023).

18. Kleimola J. и др. Nonlinear abstract sound synthesis algorithms // Aalto University. – 2013.

19. Axen U., Choi I. Using additive sound synthesis to analyze simplicial complexes // Proceedings of the second International Conference on Auditory Display, ICAD. – 1995. – T. 94.

20. Roche F. Music sound synthesis using machine learning: Towards a perceptually relevant control space. Université Grenoble Alpes. – 2020.

21. Serra X. State of the art and future directions in musical sound synthesis // 2007 IEEE 9th Workshop on Multimedia Signal Processing. IEEE, – 2007. – C. 9-12.

22. McIntyre M. E., Schumacher R. T., Woodhouse J. On the oscillations of musical instruments // The Journal of the Acoustical Society of America. – 1983. – T. 74. – №. 5. – C. 1325-1345.

23. Соломенник А. И. Технология синтеза речи: история и методология исследований // Вестник Московского университета. Серия 9. Филология. – 2013. – №. 6. – С. 149-162.

24. Wang B., Yang Y. H. PerformanceNet: Score-to-audio music generation with multi-band convolutional residual network // Proceedings of the AAAI Conference on Artificial Intelligence. – 2019. – T. 33. – №. 01. – C. 1174-1181.

25. Dong H. W. et al. Deep Performer: Score-to-Audio Music Performance Synthesis // ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE. – 2022. – C. 951-955.

26. Schwarz D. A system for data-driven concatenative sound synthesis // Digital Audio Effects (DAFx). – 2000. – C. 97-102.

27. Schwarz D. Corpus-based concatenative synthesis // IEEE signal processing magazine. – 2007. – T. 24. – №. 2. – C. 92-104.

28. Schwarz D. Distance mapping for corpus-based concatenative synthesis // Sound and Music Computing (SMC). – 2011. – С. 1-1.

29. Nuanáin C. Ó., Herrera P., Jordá S. k-Best Unit Selection Strategies for Musical Concatenative Synthesis // International Symposium on Computer Music Multidisciplinary Research. Springer, Cham. — 2017. – С. 76-97.

30. McLeod A., Steedman M. HMM-based voice separation of MIDI performance //Journal of New Music Research. – 2016. – Т. 45. – №. 1. – С. 17-26.

31. GitHub. Voice-splitting [Электронный ресурс] / A. McLeod. — Электрон. дан. — Режим доступа: https://github.com/apmcleod/voice-splitting, свободный (дата обращения: 09.05.2023).

32. Гнесинская Российская академия музыки [Электронный ресурс]. — Электрон. дан. — Режим доступа: http://www.gnesin-academy.ru, свободный (дата обращения: 09.05.2023).

ПРИЛОЖЕНИЕ А