Федеральное государственное бюджетное образовательное учреждение
высшего образования
«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА И
ГОСУДАРСТВЕННОЙ СЛУЖБЫ при Президенте Российской Федерации»
ИНСТИТУТ ЭКОНОМИКИ, МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ
ЭКОНОМИЧЕСКИЙ ФАКУЛЬТЕТ
НАПРАВЛЕНИЕ 38.03.01 ЭКОНОМИКА

Проект в рамках курса «Глубокое обучение»
На тему:
«Генерация MIDI бэнгеров»

студент-бакалавр
Охина Алина Сергеевна,
Волков Иван Дмитриевич
Преподаватель:
Чухров Никита Сергеевич

Москва, 2025

Введение

Генерация музыки с использованием нейросетевых моделей — активно развивающееся направление в области машинного обучения. Современные модели уже способны создавать музыкальные фрагменты, имитируя стиль композитора или заданный жанр. Однако большинство решений нацелены на полностью автоматическую генерацию и не учитывают участие пользователя в процессе.

В рамках данного проекта мы поставили цель разработать систему, которая позволяет генерировать многотрековые MIDI-композиции с возможностью выбора инструментов. Такой подход делает модель более гибкой и адаптируемой под творческие задачи, где важна не только структура, но и звучание каждого трека.

Проект включает в себя анализ архитектур генерации последовательностей, подготовку и токенизацию MIDI-данных, обучение трансформерной модели и разработку веб-интерфейса для визуализации результатов обучения модели.Основная часть

Основная часть

1. Цель и задачи проекта

Целью проекта является разработка нейросетевой модели, способной генерировать музыкальные композиции в формате MIDI с учётом заданных пользователем параметров, в частности — выбора инструментов. Такой подход позволяет объединить творческий потенциал нейросетей с контролем со стороны человека, что делает результат более осмысленным и применимым на практике.

Для достижения этой цели были поставлены следующие задачи:

- Изучить существующие подходы к генерации последовательностей в задачах машинного обучения;
- Сравнить архитектуры RNN, GAN и Transformer с точки зрения применимости к задаче генерации музыки;
- Подготовить датасет MIDI-композиций: выполнить фильтрацию, нормализацию и токенизацию;
- Обучить трансформерную модель на основе подготовленных данных;
- Реализовать интерфейс, позволяющий пользователю генерировать композиции с учётом заданных параметров.

2. Обзор архитектур и выбор модели

Перед тем как приступить к реализации модели, мы изучили несколько подходов к генерации последовательностей, которые используются в машинном обучении. Нас в первую очередь интересовали архитектуры, способные работать с музыкальными данными — в частности, с MIDI-последовательностями. Мы сосредоточились на трёх вариантах: RNN/LSTM, GAN и Transformer.

Рекуррентные нейронные сети хорошо справляются с последовательной информацией и могут учитывать временные зависимости. Такой подход применялся, например, в модели BachBot, которая генерирует музыку в стиле Баха. Однако у RNN есть важные ограничения. Главная проблема — они слабо работают с длинными зависимостями. Это особенно критично для музыки, где повторения и развитие темы могут происходить через десятки или сотни событий. Кроме того, генерация в RNN происходит поэтапно — это делает модель медленной и трудной для масштабирования.

Генеративные состязательные сети (GAN) тоже применяются в музыкальных задачах. Их плюс в том, что они могут генерировать сразу всю композицию, а не по одной ноте за раз. Примеры таких моделей — MuseGAN или GANSynth. Но у GAN много сложностей: они требуют чрезвычайно больших ресурсов, чувствительны к настройке, и, что особенно важно, не дают точного контроля над результатом. Это делает их менее удобными, если нужно управлять звучанием — например, выбирать инструмент или влиять на структуру трека.

Transformer-модель оказалась наиболее подходящей для нашей задачи. В отличие от RNN, трансформеры умеют работать с глобальным контекстом: они "видят" всю последовательность целиком благодаря механизму внимания (attention). Это важно в музыке — ведь аккорды, мелодии и ритмические рисунки могут зависеть от событий, находящихся далеко друг от друга. Кроме того, трансформеры обрабатывают данные параллельно, что позволяет быстрее обучать модель даже на больших объёмах данных.

За основу мы выбрали готовую архитектуру Allegro Transformer Maker, которая уже рассчитана на работу с MIDI-файлами. Это облегчило старт: нам не пришлось строить модель с нуля, но мы смогли адаптировать её под свои задачи. Мы переработали предварительную обработку файлов, схему токенов, сгруппировали инструменты по категориям, добавили специальные управляющие токены и подключили оптимизации (Flash Attention и DataParallel), чтобы ускорить обучение.

В итоге трансформер дал нам необходимый баланс между качеством, гибкостью и стабильностью — и стал основой нашей модели генерации.

3. Подготовка данных

Работа с музыкальными данными требует внимательной предобработки. В отличие от текста или изображений, MIDI-файлы представляют собой события с временными координатами, длительностью, высотой ноты, громкостью и указанием инструмента. Прежде чем подавать такие данные в модель, их нужно привести к унифицированному виду. Мы разбили этот процесс на несколько этапов: фильтрация, нормализация и токенизация.

Первым шагом была очистка датасета. Мы использовали Lakh MIDI Dataset — крупную коллекцию MIDI-композиций разных жанров. Однако далеко не все треки были пригодны для обучения. Мы исключили:

- слишком длинные треки, которые перегружали память при обучении;
- композиции без разметки по инструментам — для нас важно было точно знать, какой инструмент играет партию;
- треки с несколькими ведущими партиями — чтобы модель училась на "чистых" линиях и не путалась в голосах.

После фильтрации остались наиболее структурированные и понятные примеры, подходящие для обучения нашей модели.

На этапе нормализации мы привели данные к стандартному формату. Основные шаги:

- Квантование времени: все события привязали к сетке из 1/32 долей — это избавило от дробных значений и помогло выстроить чёткую ритмическую структуру.
- Округление длительности: длительность нот также округлялась к ближайшему значению по сетке, чтобы уменьшить количество уникальных вариантов.
- Упрощение громкости: значения velocity округлялись с шагом 15 — это позволило сохранить динамику, но сократить число токенов.
- Сокращение числа инструментов: мы сгруппировали стандартные 128 MIDI-инструментов в 12 категорий, таких как фортепиано, гитара, духовые и т. д. Это сделало модель менее чувствительной к шуму и упростило управление звучанием.

Для подачи данных в модель каждое музыкальное событие мы представили в виде трёх токенов: время начала (в пределах 0–225 шагов), комбинация длительности и громкости (один токен на основе формулы длительность * 8 + velocity), комбинация инструмента и высоты ноты (инструмент * 128 + нота).

Также мы добавили специальные токены, указывающие на: начало последовательности (START), наличие или отсутствие ударных, токены-индикаторы выбранного инструмента.

Такая схема токенизации сделала формат данных похожим на язык — где каждое событие, как "слово", может быть подано в трансформер. Это позволило использовать проверенные подходы из NLP и сделать модель более универсальной и расширяемой.

4. Обучение модели

После подготовки данных мы перешли к обучению модели. Мы использовали трансформерную архитектуру, адаптированную под задачу генерации MIDI-треков. Модель работала в авторегрессионном режиме, то есть училась по текущим токенам предсказывать следующий. По своей структуре она близка к архитектурам, применяемым в языковом моделировании, например GPT.

В основе модели стоит модуль TransformerWrapper с 12 слоями декодера. С учётом доступных вычислительных ресурсов, были выбраны следующие параметры обучения:

- размер эмбеддингов: 512;
- количество голов внимания: 10;
- максимальная длина последовательности: 512 токенов (плюс 1 токен ответа);
- Flash Attention — для ускорения вычислений и экономии памяти;
- DataParallel — для обучения на нескольких GPU одновременно.

Чтобы модель могла работать в авторегрессионном режиме, мы обернули её в AutoregressiveWrapper. Это позволило обучать её на задаче следующего токена, как в

языковых моделях: на вход подавались 512 токенов, а выходом был 513-й, который нужно было предсказать.

Обучение проходило в течение 5 эпох. Мы использовали следующие настройки:

- оптимизатор: Adam;
- начальная скорость обучения (learning rate): 2×10^{-4};
- размер батча: 64;
- шаг градиентного обновления — после каждого батча.

Для контроля качества мы добавили регулярную проверку на валидационном наборе:

- каждые 300 шагов выводилась статистика по метрикам;
- каждые 1000 шагов модель сохранялась, если качество улучшалось.

Обучающие данные подавались с помощью кастомного датасета, который случайным образом выбирал отрезки длиной 513 токенов из полной последовательности. Это позволило эффективно использовать даже длинные треки и сделать обучение стабильным.

Основными метриками обучения были:

- Cross-Entropy Loss — показывает, насколько предсказанный токен близок к реальному;
- Accuracy — процент полных совпадений по токенам (включая ноту, длительность, громкость и инструмент).

Во время обучения мы наблюдали стабильное снижение loss и рост accuracy на обеих выборках (обучающей и валидационной), обе метрики стабилизировались к окончанию обучения. Финальные значения:

- loss: около 1.0–1.5;
- accuracy: около 60%, что считается хорошим результатом для генеративной задачи с множеством возможных исходов.

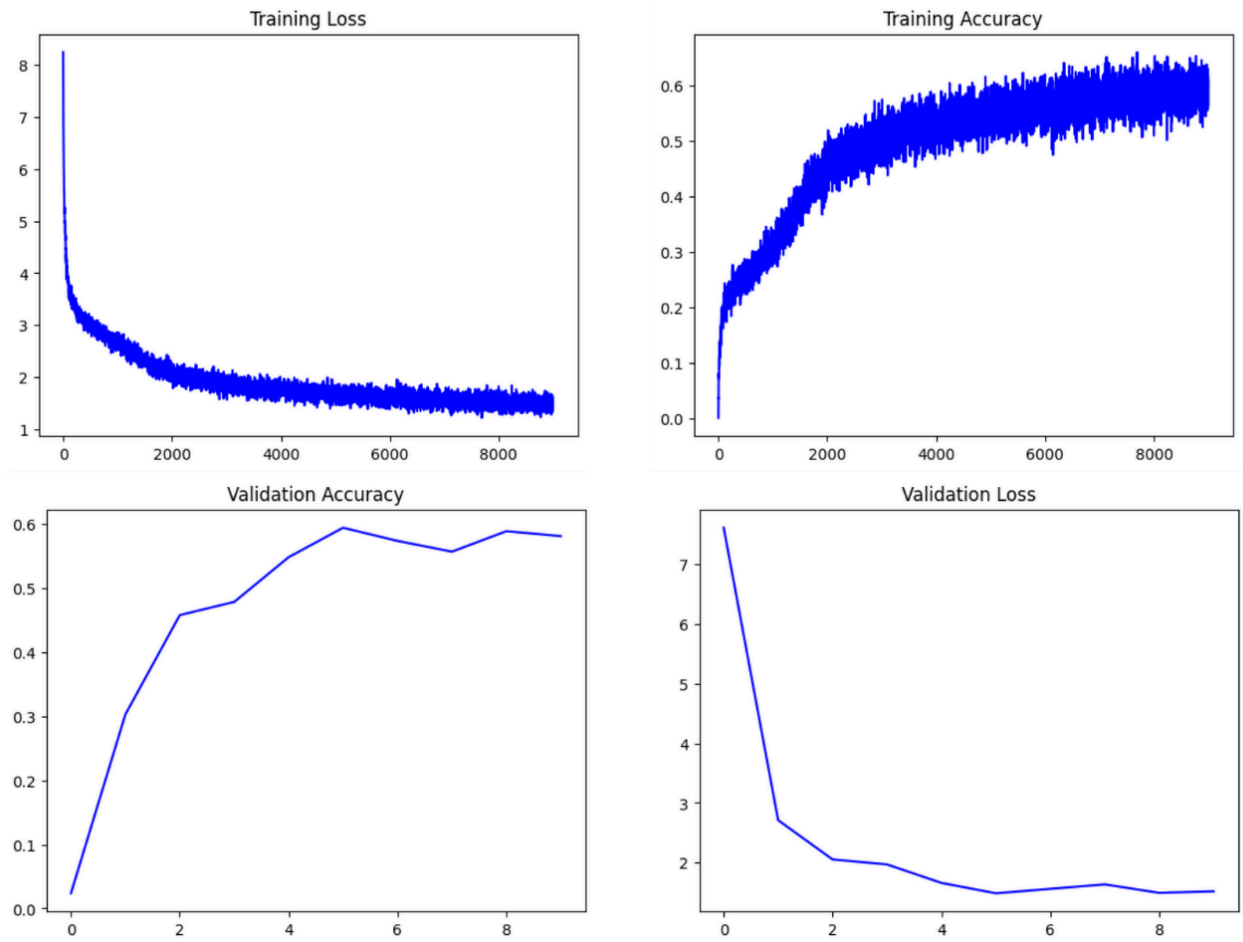
На графиках представлены значения метрик на обучающей и валидационной выборках:

5. Презентация результатов

Для взаимодействия с моделью без необходимости запускать код вручную, мы реализовали простой веб-интерфейс. Для этого была использована библиотека Gradio, которая позволяет создавать интерфейсы для моделей машинного обучения и разворачивать их в браузере.

Интерфейс был развернут на платформе Hugging Face Spaces, что позволило сделать его доступным в любое время без установки дополнительных зависимостей. Пользователь может зайти на страницу, выбрать параметры и получить сгенерированную музыкальную композицию.

Основные функции нашего интерфейса:



- Выбор музыкального инструмента
  Перед началом генерации пользователь может указать, какой инструмент должен быть основным в композиции. Модель поддерживает 11 категорий инструментов (например, фортепиано, гитара, скрипка, духовые и др.).

- Добавление ударных
  При необходимости к выбранному инструменту можно добавить ударную дорожку. Это позволяет получить более полное и ритмически оформленное звучание.

- Настройка длины трека
  Пользователь может задать желаемую длину композиции в тактах или в количестве токенов, чтобы получить более короткий фрагмент или, наоборот, развернутую структуру.

- Генерация нескольких вариантов
  Интерфейс поддерживает одновременную генерацию нескольких треков. Это удобно, когда хочется выбрать наиболее удачный результат из нескольких попыток.

- Температура генерации
  Этот параметр отвечает за степень случайности: при низком значении модель будет действовать более предсказуемо, при высоком — добавит больше креативности. Это позволяет управлять характером музыки — от аккуратных мелодий до экспериментальных решений.

Сгенерированные треки можно скачать в формате MIDI — для дальнейшего редактирования или использования в цифровых аудиостанциях. Также доступен быстрый предварительный прослушиваемый вариант в формате MP3, который удобно использовать для оценки результата на слух.

Заключение

В ходе проекта была разработана трансформерная модель для генерации MIDI-композиций с возможностью выбора инструментов и настройки параметров звучания. Мы прошли все этапы: от анализа архитектур и подготовки данных до обучения модели и создания удобного веб-интерфейса для презентации результатов.

Использование Transformer-архитектуры позволило эффективно работать с музыкальной структурой и обеспечило стабильное качество генерации. Благодаря токенизации и оптимизациям нам удалось адаптировать модель под многотрековые композиции, а через Gradio-интерфейс — сделать её доступной и управляемой для пользователя.

Полученная система может использоваться как вспомогательный инструмент в музыкальном творчестве или как основа для дальнейших исследований в области генеративной музыки. У нашего проекта возможны расширения: добавление аккордовой разметки, более гибкое управление динамикой и длиной трека, а также дообучение модели на определённом жанре или стиле.