МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

ПРИМЕНЕНИЕ ГЕНЕРАТИВНЫХ СОСТЯЗАТЕЛЬНЫХ СЕТЕЙ ДЛЯ ЗАДАЧИ ТРАНСФОРМАЦИИ СТИЛЯ АУДИО ФАЙЛОВ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Яшиной Анны Александровны

Научный руководитель
зав. каф. техн. прогр, к. ф.-м. н. _____ И. А. Батраева

Заведующий кафедрой
к. ф.-м. н., доцент _____ А. С. Иванов

Саратов 2020

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Применение генеративных состязательных сетей может быть полезно для систем распознавания речи, требующих больших объемов информации, за счет увеличения объема данных. Также синтезирование аудио, генерирование звуковых эффектов может иметь практическое применение в звуковом дизайне для музыки и кино. Однако, несмотря на то, что применение генеративных состязательных сетей для задач синтеза и стилизации изображений набирает все большую популярность, использование сетей данной архитектуры для задачи синтезирования звука в условиях обучения без учителя еще не до конца изучено.

Аудиосигналы дискретизируются с высоким разрешением, для обучения синтезированию звука требуется захват структуры сигнала в различных временных масштабах. Применение оконного преобразования Фурье позволяет получить графическое представление аудио сигнала, спектограмму, благодаря чему мы можем работать с аудио файлами, как с изображениями [1].

Однако в генеративном контексте такой подход может быть достаточно проблематичен, так как спектрограммы являются в некотором роде необратимыми и, следовательно, не могут быть прослушаны без оценки потерь.

Целью данной работы является создание и обучение генеративной состязательной сети, нацеленной на трансформацию жанра заданного аудио файла, рассматриваемого в виде спектограммы.

Поставлены следующие задачи:

- изучение принципов работы и разновидностей генеративных состязательных сетей;
- анализ существующих алгоритмов преобразования звуковых сигналов;
- создание и обучение генеративной состязательной сети, нацеленной на трансформацию стиля аудио файла;
- создание программного обеспечения, предоставляющего интерфейс для обученной генеративной нейронной сети.

1 Нейронные сети для задачи трансформации стиля

Глубокие генеративные модели в виду сложностей аппроксимации, трудно разрешимых вероятностных вычислений, возникающих при оценке функции максимального правдоподобия, до недавнего времени оставались менее изученными, чем дискриминационные модели, нацеленные на сопоставление многомерных входных данных с метками определенных классов.

В 2014-ом году была предложена архитектура генеративной сети, которая носит название *генеративная состязательная сеть* [2]. Данная модель состоит из двух противоборствующих сетей — генеративной модели (генератор) и дискриминационной модели (дискриминатор). Задача дискриминатора заключается в определении того, является ли данная выборка выборкой из распределения модели или выборкой из распределения данных. Задача генератора — генерация таких данных, которые бы рассматривались дискриминатором, как истинные (выборкой из распределения данных).

В обучении данных моделей конкуренция между дискриминатором и генератором играет значительную роль. Так, переобучение дискриминатора, т. е. безошибочное распознавание моделью сгенерированных образцов приводит к прекращению обучения генератора, т. к. генеративная сеть не получает информации по поводу того, как минимизировать ошибку, что впоследствии приводит к прекращению обучения дискриминатора.

Генеративные состязательные сети имеют свои преимущества и недостатки по сравнению с предшествующими моделями. Недостатки в первую очередь заключаются в том, что отсутствует явное представление генерирующей функции, также необходимо синхронизировать обучение дискриминатора и генератора (в частности, для генерации разнообразных результатов и избежания «сценария Гельветики», при котором генератор отображает различные входные данные z в одно и то же значение x, не следует тренировать генератор слишком долго без обновления дискриминатора).

Преимущество генеративных состязательных сетей носит в основном вычислительный характер. Также состязательные модели могут получить некоторое статистическое преимущество, так как сеть генератора не обновляется непосредственно с примерами данных, а только с градиентами, проходящими через дискриминатор. Это означает, что компоненты входных данных не копируются непосредственно в параметры генератора.

1.1 CycleGAN

В 2017-ом году учеными Исследовательской лаборатории ИИ Беркли (BAIR) была предложена измененная версия генеративной состязательной сети — *циклическая генеративная состязательная сеть* [3].

Оптимальный генератор G отображает область X в область $\hat{Y}$, распределенную идентично целевой области Y. Тем не менее, данное отображение не гарантирует того, что отдельные входы x и выходы y будут объединены в пару. Также, существует бесконечное множество отображений G, задающих одинаковое распределение по $\hat{y}$.

Оптимизация состязательной потери в условиях изоляции сетей друг от друга затруднительна: стандартные процедуры на практике зачастую приводят к известной проблеме *mode collapse*, когда все входные изображения преобразуются в одни и те же выходные изображения, и оптимизация не прогрессирует.

Объединение потери согласованности цикла, поощряющей $F(G(x)) \approx x$ и $G(F(y)) \approx y$, с состязательными потерями на доменах X и Y формирует конечную цель для непарного преобразования изображения в изображение.

Так, пусть X, Y — непарные домены, $\{x_i\}_{i=1}^{N}$, $x_i \in X$, $\{y_i\}_{i=1}^{N}$, $y_i \in Y$ — тренировочные данные, $x \sim p_{data}(x)$, $y \sim p_{data}(y)$ — распределения данных. Предложенная модель включает два отображения $G : X \to Y$ и $F : Y \to X$, также определены дискриминаторы D_X и D_Y, где целью дискриминатора D_X является обучение распознаванию различий между изображениями домена X $\{x_i\}$ и отображенными изображениями $\{F(y_i)\}$, аналогично дискриминатор D_Y нацелен на распознавание $\{y_i\}$ и $\{F(x_i)\}$.

Цель исследователей заключалась в оптимизации состязательных потерь для сопоставления распределения сгенерированных изображений с распределением данных из целевого домена и потери согласованности цикла для предотвращения противоречивости отображений G и F в процессе обучения.

Так, для отображения $G : X \to Y$ и дискриминатора D_Y была определена следующая функция потери:

$$L_{GAN}(G, D_Y, X, Y) = E_{y \sim p_{data}(y)}\big[\log D_Y(y)\big] + E_{x \sim p_{data}(x)}\big[\log(1 - D_Y(G(x)))\big].$$

Целью генератора G является минимизация данного выражения, в то

время как, цель дискриминатора D_Y состоит в его максимизации, т. е. цель обучения может быть описана в виде следующего выражения:

$$\min_G \max_{D_Y} L_{GAN}(G, D_Y, X, Y).$$

Аналогичным образом определяется состязательная потеря для отображения $F : Y \to X$ и дискриминатора D_X, т. е. $\min_F \max_{D_X} L_{GAN}(F, D_X, Y, X)$.

Состязательные потери, изолированные при обучении отображений друг от друга, не могут гарантировать того, что обученная функция сможет отобразить индивидуальный вход x_i в желаемый выход y_i. С целью сокращения пространства возможных функций отображения, была предложена потеря согласованности цикла: для каждого изображения x цикл преобразований должен быть способен вернуть изображение к его первоначальному представлению, т.е. $x \to G(x) \to F(G(x)) \approx x$. Данное условие носит название *согласованность прямого цикла*. Аналогично для каждого изображения $y \in Y$ отображения G и F должны удовлетворять условию *согласованности обратного цикла* $y \to F(y) \to G(F(y)) \approx y$. Данное поведение стимулируется с помощью потери согласованности цикла:

$$L_{cyc}(G, F) = E_{x \sim p_{data}(x)}\big[\|F(G(x)) - x\|_1\big] + E_{y \sim p_{data}(y)}\big[\|G(F(y)) - y\|_1\big].$$

Таким образом, суммарная функция потери CycleGAN имеет следующий вид:

$$L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F),$$

где параметр λ контролирует зависимость отображений друг от друга.

Цель оптимизации:

$$G^*, F^* = arg \min_{G,F} \max_{D_X, D_Y} L(G, F, D_X, D_Y).$$

Качественные результаты были продемонстрированы для нескольких задач, таких как преобразование стиля коллекции изображений, трансформация объектов, улучшение качества фотографий и т. д.

1.2 TraVeLGAN

В 2019-ом году исследователями Йельского университета был предложен новый подход для решения задачи отображения изображений из одного домена в другой. Данный подход был назван *Translation by Transformation Vector Learning* [4].

Так как существует бесконечное множество отображений между двумя доменами, то отсутствуют гарантии, что после отображения некоторое отдельное изображение из исходного домена будет обладать какими-либо общими характеристиками с его представлением в целевом домене. Некоторые методы решают эту проблему путем регуляризации семейства генераторов. Так, регуляризация введенная CycleGAN, накладывает на отображения ограничение быть обратными друг другу, известное как свойство согласованности цикла. Однако подобные ограничения, накладываемые на генераторы, могут препятствовать обучению более сложным функциям отображения между различными доменами, заставляя генераторы быть близкими функциям тождества.

Модель TraVeLGAN в дополнение к генератору и дискриминатору использует вспомогательную сиамскую сеть с целью создания латентного пространства данных для определения высокоуровневых семантических признаков, характеризующих домены. Данное пространство направляет генератор во время обучения, заставляя его сохранять векторную арифметику между точками в этом пространстве. Вектор, преобразующий одно изображение в другое в исходном домене, должен являться тем же самым вектором, который преобразует сгенерированную версию этого изображения в сгенерированную версию другого в целевом домене.

TraVeLGAN отличается от предыдущих работ несколькими особенностями:

1. отсутствует необходимость в ограничении архитектуры генератора, а также в обучении согласованию циклов или связыванию весов генератора;
2. для оценки сходства между оригинальными и сгенерированными изображениями применяется отдельная сеть;
3. TraVeLGAN добавляет возможность отображения в неконтролируемую область благодаря применению латентного пространства, объясняющего какие аспекты конкретного изображения были использованы для опре-

деления его представления в целевом домене.

Таким образом, модель TraVeLGAN способна обучаться отображениям между сложными, неоднородными доменами, требующими значительных и разнообразных трансформаций.

2 Разработка приложения для задачи стилизации аудио файла

В ходе практической части дипломной работы были созданы две генеративных состязательных сети, основанные на архитектуре CycleGAN и TraVeLGAN для решения задачи трансформации стиля аудио файла по средствам представления его в виде mel-спектограммы.

2.1 Выбор средств и библиотек разработки

В качестве языка программирования был использован язык Python 3.6.0. Для разработки использовались следующие пакеты и библиотеки:

- NumPy — фундаментальный пакет для вычислений на Python, добавляющий поддержку многомерных массивов, высокоуровневых математических функций, предназначенных для работы с многомерными массивами.
- Keras — открытая библиотека, предназначенная для работы с нейронными сетями, написанная на языке Python. Keras представляет собой надстройку над фреймворками Deeplearning4j, TensorFlow и Theano. Эта библиотека содержит многочисленные реализации широко применяемых строительных блоков нейронных сетей, таких как слои, целевые функции, оптимизаторы и т. д.
- LibROSA — библиотека, предназначенная для обработки аудио и музыкальных сигналов. На высоком уровне LibROSA обеспечивает реализации различных общих функций, используемых в области поиска информации из музыки (MIR).

2.2 Подготовка данных

В качестве исходного датасета был использован датасет GTZAN Genre Collection, состоящий из 1000 аудио файлов продолжительностью 30 секунд. Набор данных содержит 10 жанров, каждый из которых представлен 100 треками, являющимися монофоническими 16-битными аудиофайлами 22050 Гц в формате `.au` [5]. Для увеличения набора данных применялась аугментация — каждый трек был разделен на 10 треков по 3 секунды. С помощью возможностей библиотеки libROSA каждый трек был представлен в виде mel-спектограммы. Спектрограммы нормализовались в диапазоне $[-1, 1]$, чтобы соответствовать выходу функции активации генератора $\tanh$.

2.3 Модель, основанная на архитектуре TraVeLGAN

Так как спектограммы композиций различных музыкальных жанров зачастую являются достаточно разнообразными, модель для задачи трансформации стиля музыкального произведения была основана на подходе, описанном в [4]. Так, модель состоит из трех нейронных сетей: генератора, дискриминатора и вспомогательной сиамской сети.

Модель генератора основана на U-Net архитектуре нейронных сетей, шаг смещения фильтра $strides = 2$. В качестве функции активации после сверточных слоев использована функция ReLU, функция активации на выходном слое — $\tanh$. Также после каждого сверточного слоя для стабилизации обучения генератора применена *Instance*-нормализация [6].

Модель дискриминатора состоит из трех сверточных слоев, функция активации — LeakyReLU с параметром $alpha = 0.2$. Для ускорения обучения после каждого сверточного слоя применена *Batch*— нормализация. Код функции создания модели дискриминатора представлен ниже:

Сиамская нейронная сеть также состоит из трех сверточных слоев, функция активации — LeakyReLU, шаг смещения фильтра — $strides = 2$, размерность вектора признаков — 128.

2.4 Обучение модели

Для обучения модели использовалась платформа Google Colab.

Продолжительность обучения модели составила 2000 эпох. Для моделей генератора и дискриминатора использовался оптимизатор Adam, шаг обучения дискриминатора был выбран равным $l_{r_D} = 0.0002$, шаг обучения генератора и сиамской сети — $l_{r_{G,S}} = 0.0004$ [7].

Алгоритм обучения генеративной состязательной сети:

Пусть X — исходный домен (жанр), Y — целевой домен, $x \in X$ — произвольный представитель данного жанра (сегмент аудио трека, представленный в виде спектограммы), $y \in Y$ — произвольный представитель целевого жанра.

1. Спектограмма x размерности $M \times L$, где M — количество mel каналов ($M = 128$), $L < t$, делится на две равные части x_1 и x_2 размерности $M \times \frac{L}{2}$.
2. Вычисление результата работы генеративной сети G на входных данных x_1 и x_2, $\hat{y_1} = G(x_1)$ и $\hat{y_2} = G(x_2)$.

3. Вычисление векторов признаков сиамской сетью S: $S(x_1)$, $S(x_2)$, $S(\hat{y_1})$, $S(\hat{y_2})$, где $\hat{y_1} = G(x_1)$, $\hat{y_2} = G(x_2)$.
4. Объединение спектограмм $\hat{y_1}$ и $\hat{y_2}$ в спектограмму $\hat{y}$ размерности $M \times L$.
5. Вычисление значения функции потери дискриминатора L_{D_r} на спектограмме $y \in Y$ из целевого домена.
6. Вычисление значения функции потери дискриминатора L_{D_f} на сгенерированной спектограмме $\hat{y} \in G(X)$.
7. Вычисление значения функции потери генератора L_G по ошибке дискриминатора L_{D_f}.
8. Вычисление значения функции потери сиамской сети L_S для пары (x_1, x_2).
9. Вычисление значения функции потери L_{TraVeL} по входным данным $(S(x_1), S(x_2), S(\hat{y_1}), S(\hat{y_2}))$.
10. Обновление дискриминатора D по ошибке $L_D = \frac{L_{D_r} + L_{D_r}}{2}$, обновление шага обучения l_{r_D}.
11. Обновление генератора G по ошибке $L_G + L_{TraVeL} + \alpha L_S$, обновление шага обучения $l_{r_{G,S}}$.

    Функции состязательной потери дискриминатора и генератора:

$$L_{D,adv} = -E_{y \sim Y}[\min(0, -1 + D(y))] - E_{x \sim X}[\min(0, -1 - D(G(x)))]$$

$$L_{G,adv} = -E_{x \sim X} D(G(x))$$

    Функция потери TraVeL:

$$L_{(G,S),TraVeL} = E_{(x_1,x_2) \sim X}\left[cos\_similarity(t_{12}, t'_{12}) + \|t_{12} - t'_{12}\|_2^2\right]$$

$$t_{ij} = S(x_i) - S(x_j)$$

$$t'_{ij} = S(G(x_i)) - S(G(x_j)).$$

    Функция сравнительной потери сиамской сети (Contrastive Loss):

$$L_{S,margin} = E_{(x_1,x_2) \sim X} \max(0, (\delta - \|t_{12}\|_2)).$$

    Таким образом, суммарные функции потерь для генератора, дискрими-

натора, сиамской сети определяются следующим образом:

$$L_D = L_{D,adv}$$

$$L_G = L_{G,adv} + \alpha L_{(G,S),TraVeL}$$

$$L_S = \alpha L_{(G,S),TraVeL} + \beta L_{S,margin}.$$

После окончания процесса обучения результат работы генеративной сети для заданной композиции определяется следующим образом:

1. Композиция разбивается на сегменты t_i продолжительности 3 секунды.
2. Для каждого трека определяется mel-спектограмма $spectogram$, нормализованная в диапазоне $[-1, 1]$.
3. Каждая спектограмма $spectogram$ разделяется на 2 равные части s_l и s_r, каждая из которых поступает на вход генеративной сети G.
4. Сгенерированные спектограммы $g_l = G(s_l)$ и $g_r = G(s_r)$, полученные на выходе генератора объединяются в единую спектограмму $g\_spectogram_i$.
5. Спектограммы $g\_spectogram_i$, вычисленные для всех сегментов t_i, переводятся в wave формат и объединяются в единую композицию.

2.5 Модель, основанная на архитектуре CycleGAN

Модель, основанная на архитектуре CycleGAN состоит из двух генераторов G_A, G_B, дискриминаторов D_A, D_B и составных моделей C_A, C_B.

Составная модель требуется для каждой модели генератора и отвечает за обновление весов генеративной сети, в также сообщение данной информации соответствующей модели дискриминатора и другого генератора. Это может быть достигнуто путем маркировки весов других моделей как неспособных к обучению в контексте составной модели, чтобы гарантировать, что обновляются только веса предполагаемого генератора.

Составная модель генератора G_A:

— Состязательная потеря: $B \to G_A \to A \to D_A \to [true/false]$

— Потеря идентичности: $A \to G_A \to A$

— Потеря прямого цикла: $B \to G_A \to A \to G_B \to B$

— Потеря обратного цикла: $A \to G_B \to B \to G_A \to A$

Модели дискриминатора и генератора аналогичны соответствующим моделям, описанным для сети, основанной на архитектуре TraVeLGAN.

2.6 Обучение модели

Обучение осуществлялось следующим образом:

1. Формируется выборка из представителей доменов A, $B - \{a_i\}, \{b_i\}$ с меткой 1.
2. Генерируются представили противоположных доменов $\{\hat{a}_i\}, \{\hat{b_i}\}$, где $\hat{a}_i = G_A(b_i)$, $\hat{b_i} = G_B(a_i)$ с меткой 0.
3. Обновляются веса генератора G_X, дискриминатора D_X;
4. Обновляются веса генератора G_Y, дискриминатора D_Y.

2.7 Результаты эксперимента

В ходе эксперимента обе модели, основанные на архитектуре TraVelGAN и CycleGAN, были обучены для задачи отображения mel-спектограмм композиций жанра «Классика» в mel-спектограммы жанра «Джаз». Обучение некоторому стабильному универсальному отображению может являться довольно затруднительной задачей в связи с тем, что mel-спектограммы аудио файлов, являющихся представителями не только различных, но и одинаковых жанров, могут достаточно сильно отличаться.

В результате эксперимента было выявлено, что примеры, сгенерированные сетью, основанной на архитектуре TraVelGAN значительно превосходят результаты, сгенерированные моделью, основанной на архитектуре CycleGAN. Одним из основных обоснований полученных результатов может являться то, что ограничения, накладываемые архитектурой CycleGAN на генеративные сети быть обратными друг другу отображениями, приводит к тому, что модель генератора становится близкой функции тождества. Так, на практике было получено, что композиции, сгенерированные моделью, основанной на архитектуре CycleGAN, являются очень схожими с исходными аудио файлами.

2.8 Описание приложения

Для нейронной сети, основанной на архитектуре TraVeLGAN, показавшей лучший результат, был написан web-интерфейс. Для клиентской части использовались HTLM, CSS, JavaScript, JQuery, серверная часть была написана с использованием легковесного фреймворка Flask.

При выборе композиции происходит ее отправка на сервер для последующей обработки. При получении запроса на обработку композиции по адресу

/transfer/{genre} на сервере вызывается метод process сервиса, который при первом обращении загружает соответствующую модель в память.

Результирующий аудио файл, сгенерированный моделью, сохраняется в формате wave, ссылка на сгенерированный файл передается клиенту для загрузки и дальнейшего воспроизведения.

ЗАКЛЮЧЕНИЕ

Таким образом, в ходе данной работы были рассмотрены основные принципы работы генеративных состязательных сетей, модели, построенные на основе архитектуры GAN, добившиеся в недавнее время выдающихся результатов для генерации и стилизации изображений, такие как CycleGAN, TraVeLGAN.

Были созданы генеративные состязательные сети, основанные на архитектуре TraVeLGAN и CycleGAN, для решения задачи трансформирования жанра музыкального произведения. В качестве исходного домена был выбран жанр «Классическая музыка», для результирующего — «Джаз». Продолжительность обучения составила 2000 эпох.

Основная сложность задачи отображения аудио файлов из одного стиля в другой путем представления их в виде спектограм связана с тем, что в результате данного преобразования часть информации теряется, в связи с чем, возможно некоторое зашумление восстановленного сгенерированного сигнала.

Тем не менее, полученные результаты подтвердили возможность применения генеративных состязательных сетей для задачи трансформирования стиля аудио файлов, представленных в графическом виде (спектограммы).

Также на практике было подтверждено преимущество подхода, описанного для TraVeLGAN, над моделями, основанными на архитектуре CycleGAN, для решения задачи отображения между достаточно сильно различающимися доменами, в частности для отображения mel-спектограмм одного жанра в другой.

Возможными альтернативными способами, обходящими проблемы, возникающие при работе со звуком, представленном в виде спектограмм, является представление композиций в виде последовательности нот и аккордов, использование `midi` файлов для обучения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Donahue, C.* Adversial audio synthesis / C. Donahue, J. McAuley, M. Puckette. — 2019.

2 Generative adversarial networks / I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. — 2014.

3 *Zhu, J.-Y.* Unpaired image-to-image translation using cycle-consistent adversarial networks / J.-Y. Zhu, T. Park, P. Isola, A. A. Efros. — 2017.

4 *Amodio, M.* Travelgan: Image-to-image translation by transformation vector learning / M. Amodio, S. Krishnaswamy. — 2019.

5 GTZAN Genre Collection [Электронный ресурс]. — URL: https://www.kaggle.com/carlthome/gtzan-genre-collection (Дата обращения 01.05.2020).

6 *Ulyanov, D.* Instance normalization: The missing ingredient for fast stylization / D. Ulyanov, A. Vedaldi. — 2017.

7 *Heusel, M.* Gans trained by a two time-scale update rule converge to a local nash equilibrium // Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems. — Vol. 1. — Long Beach, CA, USA: 2017. — Pp. 6626–6637.