Слайд 1. Введение

Задача распознавания музыкальных инструментов востребована при обработке сложных музыкальных сигналов, таких как классическая музыка, эстрадная музыка и другие. Данная задача относится к классу задач Music Information Retrieval (MIR). Результаты работы могут быть использованы для аннотирования медиаконтента, сегментации аудиосигналов и решения других задач обработки звука.

Суть задачи состоит в том, что аудиофайл должен быть разбит на небольшие фрагменты одинаковой длительности - таймфреймы. Из них нужно получить наиболее значимые в контексте решаемой задачи признаки. Далее эти признаки подаются на вход классификатора и решается задача классификации, в результате узнаем тип музыкального инструмента, звучащий в данном фрагменте.

Слайд 2

Пример интерпретации результата решения такой задачи представлены на рисунке. Здесь представлена Прелюдия Шостаковича №5. Мы видим, что для каждого фрагмента, определен звучащий инструмент.

Слайд 3. Выбор датасета

Для решения задачи нужна база данных с записями инструментов.

Мною были использована база музыкальных инструментов с сайта университета Айова Эта база данных образцов включает струнные инструменты, духовые инструменты, медные духовые инструменты, фортепиано. Образцы музыкальных инструментов созданы Лоуренсом Фриттсом, директором студии электронной музыки.

Слайд 4.

Музыкальные файлы представляют собой хроматические гаммы, проигрываемые по нотам на реальных музыкальных инструментах. Записи охватывают полный диапазон звучания инструмента. На некоторых инструментах использовалось несколько техник, такие как пиццикато, вибрато и невибрато.

Все инструменты были записаны в безэховой камере, за исключением фортепиано. Записи производились с помощью одиночного кардиоидного конденсаторного микрофона.

Слайд 5.

Визуально сигналы выглядят следующим образом. Здесь на слайде приведен внешний вид сигналов на временном промежутке 10 секунд.

Каждый аудиофайл в формате wav представляет собой последовательность отсчетов, или сэмплов. Сэмпл, или отсчет это минимальная часть аудио сигнала, содержащая амплитудное значение звуковой волны. Некоторая последовательность семплов образуем таймфрейм, которые мы и будем анализировать.

Слайд 6

Для обработки аудиоданных я воспользовался библиотекой librosa языка python.

Я выбрал для анализа один канал, битовую глубину сэмпла 1 байт, частоту дискретизации 22050 Гц (сэмплов в секунду).

Получившиеся файлы я объединил в рамках каждой категории. Затем удалил тишину функцией split\_on\_silence библиотеки librosa. Экспериментально подобрал оптимальный порог тишины. Получившиеся файлы выровнял по длине. Далее, выполнил нормализацию функцией librosa.normalize(), то есть привел максимальную амплитуду сигнала к единице.

Часть кода показана на картинке, весь код я залил на гитхаб.

Слайд 7. Здесь представлен внешний вид сигналов после обработки.

Слайд 8. Теперь нужно получить признаки. От правильно выбранных признаков очень многое зависит, и здесь нужно подходить ответственно.

У нас есть следующие варианты.

1) Спектральный центроид. (картинка). Указывает, на какой частоте сосредоточена энергия спектра или, другими словами, указывает, где расположен “центр масс” для звука.

Слайд 9.

2) Спектральный спад. Мера формы сигнала представляющая собой частоту, ниже которой лежит определенный процент общей энергии.

Слайд 10.

3) Спектральная ширина. А также скорость пересечения нуля;

Слайд 11.

4) Цветность - обычно представлен вектором признаков из 12 элементов, в котором указано количество энергии каждого высотного класса {C, C#, D, D#, E, …, B} в сигнале. Используется, когда нужно различать ноты.

Слайд 12.

5) Мел-кепстальные коэффициенты. Я остановился именно на них, так этот метод позволяет приблизить процесс выделения признаков к процессу восприятия звуков человеком. Кроме того, этот метод себя хорошо зарекомендовал для решения задач данного класса.

Слайд 13.

5. Мел-кепстральные коэффициенты.

Мел – это единица высоты звука, основанная на восприятии этого звука нашими органами слуха. АЧХ человеческого уха даже отдаленно не напоминает прямую, и амплитуда – не совсем точная мера громкости звука. Поэтому, ввели эмпирически подобранные единицы громкости, например, фон.

Аналогично, воспринимаемая человеческим слухом высота звука не совсем линейно зависит от его частоты. Подобные единицы измерения часто используют при решении задач распознавания, так как они позволяют приблизиться к механизмам человеческого восприятия.

Слайд 14.

Для того чтобы получить мел-кепстральные коэффициенты, первым делом нам нужен спектр исходного сигнала, который мы получаем с помощью оконного преобразования Фурье. В данном случае применяется оконное преобразование Фурье, например с гауссовым окном (т.н. преобразование Габора), которое применяется к каждому таймфрейму сигнала. На картинках показано для примера.

Слайд 15.

Теперь начинается самое интересное, полученный в спектр нам нужно расположить на мел-шкале. Для этого мы используем окна, равномерно расположенные на мел-оси. Простым перемножением векторов спектра сигнала и оконной функции найдем энергию сигнала, которая попадает в каждое из окон анализа. Мы получили некоторый набор коэффициентов. Теперь уже из них надо получить мел-кепстральные коэффициенты, для чего обычно используют дискретное косинусное преобразование, благодаря его свойству "уплотнения энергии".

На пайтон все эти операции выполняет функция librosa.features.mfcc().

Количество мел-кепстральных коэффицентов для кодирования одного таймфрейма я выбрал 15. Я пробовал формировать различное число коэффициентов, библиотека librosa позволяет задать от 1 до 128, но именно вариант когда я брал около 15 коэффициентов проявил себя наилучшим образом. Также я пробовал взять большое число коэффициентов и выделить главные компоненты, то есть сократить до 15 число компонентов, но такой способ показал худшую точность при дальнейшей классификации.

Так как мел-кепстральные коэффициенты в librosa рассчитываются по 512 отсчетам, я решил для того чтобы использовать больше информации по временной шкале, объединить в одном прецеденте признаки нескольких таймфреймов. Выбрал продолжительность анализируемого фрагмента пол-секунды. За полсекунды имеем 22050 отсчетов, или (22050/2)/512 = 21 таймфрейм в прецеденте.

Слайд 16.

6. Визуальзация признаков.

Для визуализации признаков я использовал метод главных компонент и метод t-SNE. Визуализацию проводил для 2000 случайно выбранных прецедентов.

Слайд 17.

Визуализация t-SNE

Параметры метода t-SNE: число итераций 1000, perplexity = 30, learning\_rate = 10. Все классы хорошо разделимы, и можем наконец выполнить классификацию.

Слайд 18.

Модели классификации.

Наиболее хорошо в задаче классификации зарекомендовали себя ансамблевые модели, поэтому я решил построить модели на основе деревьев решений.

(Байесовская классификация – потеря точности в случае зависимых друг от друга атрибутов)

Это модель случайного леса и модель градиентного бустинга.

Весь датасет я разделил на обучающую и тестовую выборку с отношением 80/20.

Бэггинг.

В модели бэггинга, мы создаем Н моделей, каждая модель обучается из некоторой выборки из общего набора с повторениями. Затем результаты решения комбинируются путем голосования.

Я использовал следующие параметры:

Количество деревьев = 100,

Максимальная глубина = 10,

критерий разбиения - индекс Джини, энтропия.

Валидацию провел методом кросс-энтропии и отдельно на тестовом датасете, результаты приведены в таблице.

Бустинг.

В отличие от случайного леса, градиентный бустинг строит последовательность деревьев, в которой каждое дерево пытается исправить ошибки предыдущего. Я использовал следующие параметры:

Количество деревьев = 100,

Скорость обучения = 0.1.

Слайд 19

Результаты классификации методом случайного леса приведены на рисунке.

Результаты тестирования методом кросс-валидации на обучающем наборе составили 99%. Результаты тестирования на тестовом наборе при критерии разбиения «индекс джини» чуть выше.

Это можно объяснить так. Разбиение по индексу Джини тяготеет к атрибутам с большим количество значений. Тяготеет к равномощным разбиениям с равным количеством примесей. Здесь именно такой случай.

Слайд 20

В своей работе я опирался на статью [1] по нейросетевому распознаванию музыкальных инструментов, но некоторые подходы авторов пришлось пересмотреть. Например, я отказался от предложенного авторами способа уменьшения числа параметров модели методом главных компонент, так как это отрицательно сказалось на точности классификации.

Часть 2. Датасет с синтезированными инструментами.

**Слайд 2.1. Выбор датасета.**

Исследованные в первой части работы записи представляют собой запись конкретного музыкального инструмента. Это фортепиано, гитара, скрипка и труба.

Но на практике, хотелось бы например под "гитарой" понимать и бас-гитару, акустическую, классическую, элекрогитару и т.д, а их тембр должен быть существенно различен.

Поэтому я решил пойти дальше и взять сразу еще один датасет, и на этот раз изучить СИНТЕЗИРОВАННЫЕ инструменты. Я выбрал для исследования синтезатор Casio с новейшим мощным музыкальным процессором AiX. Процессор позволяет синтезировать 600 музыкальных инструментов. Все музыкальные инструменты можно разделить на 14 категорий.

Я выбрал из них 4 категории - piano-фортепиано, guitar-гитара, strings-скрипка, brass-труба. То есть это те категории, в которые как раз входят музыкальные инструменты из предыдущего датасета.

**Слайд 2.2**

В каждой категории у нас 20 инструментов.

Деление здесь на категории с одной стороны довольно условное, так как по сути и гитара и скрипка струнные инструменты. Но человек эти категории инструментов может разделить, и можно ожидать, что найдутся признаки, характерные для гитары, и не характерные для инструментов близких скрипке. Нахождение таких признаков позволило бы более эффективно определять тип инструмента в электронной музыке.

Частотный для этого датасета - это хроматическая гамма большой октавы.

Датасет я записывал сам с реального синтезатора в программу на языке python, использовал библиотеку pyaudio и wave.

**Слайд 2.3**

Получившиеся записи представлены на рисунках (по категориям).

**Слайд 2.4**

Далее я провел предобработку аудиоданных и визуализировал признаки. Здесь уже не все так радужно, как на предыдущем датасете. В частности, фортепиано и скрипка сильно перекрываются, остальные инструменты тоже не в лучшем виде.

**Слайд 2.5**

Попробуем запустить ансамбли деревьев.

Случайный лес джини 92%

Энтропия 93%

Градиентный бустинг f\_measure\_mean 82 %,

Причем фортепиано распознается лучше всех, в 96 % случаев.

Часть 3.

Тестирование полученных моделей.

Итак, мы получили отличные результаты классификации, но что если датасет реальных инструментов использовать при тестировании моделей синтезированных инструментов, и наоборот? Полученные данные показаны на рисунке. Проблема в том, что мы можем не обладать в должной мере всеми данными о инструменте, чтобы обучить модель на них. Но модели "знают" много избыточной информации о инструментах.

Как видно, тестирование первой модели на данных из второго датасета, и тестирование второй модели на данных из первого датасета оказалось провальным, реальные инструменты и синтезированные слишком сильно отличаются.

Я вижу здесь три пути решения.

Первое, это дообучить модель на другом датасете, но я предполагаю что у нас нужных нам данных для обучения нет. Датасеты предполагаются изолированными.

Во-вторых, можно обрезать деревья, чтобы не возникало переобучения. Это было мной сделано но результаты тестирования не улучшились.

В-третьих, можно взять другие признаки или выделить особенности в имеющихся.

В-четвертых, можно взять другую модель (нейросетевую).

Выделить локальные структуры сигнала позволяет вейвлет анализ. Вейвлеты позволяют получить из сигнала временнУю и частотную информацию, сужая окно для выделения коротких высокочастотных и расширяя для более длительных низкочастотных участков. Этот инструмент по своей природе лучше оконного преобразования Фурье, которое делит сигнал на равные интервалы.

Сам вейвлет представляется двумя функциями: аппроксимирующей и детализирующей. Не вдаваясь в подробности, они дают нам грубое и точное приближение сигнала соответственно. Обычно многие получаемые коэффициенты получаемого таким образом сигнала - малые числа, и их можно отбросить.

Я применил следующие преобразования: это простейший вейвлет Хаара, вейвлет Добеши порядка 1,2,3,4,8,20, вейвлет Мейера.

Все опыты я проводил с пайтон-библиотекой pywavelets.

На рисунках показаны вейвлеты и результаты применения их к аудиоданным.

Затем, я применил последовательно все шаги из первой части доклада к полученным коэффициентам. И результат, полученный на детализирующих коэффициентах с помощью случайного леса, меня обрадовал, это 45 процентов accuracy.

В итоге я решил попробовать отфильтровать исходные аудиоданные, выявив их локальные особенности. Это принесло определенные результаты. При предварительной фильтрации сигнала с помощью вейвлет-преобразования (вейвлет Добеши 2 порядка), удалось добиться 45% accuracy.

Теперь модель точно определяет из всех 4-х инструментов гитару и трубу, однако скрипку считает за трубу, а фортепиано за скрипку. Но тем не менее, получен осмысленный результат, и фильтрация работает.

Результаты полученные после обрезки деревьев, не смогли настолько улучшить модель.

Выводы: выбор обучающих данных, процесс выделения признаков, предобработка данных играют основополагающую роль в процессе обучения модели.

Весь проект с кодом выложен на гитхаб https://github.com/aleks1212v/audio\_video\_analysis