УДК 004.021-047.27:78.087.6

АЛГОРИТМ РАСПОЗНАВАНИЯ МУЗЫКАЛЬНЫХ КОМПОЗИЦИЙ ПО ВОКАЛЬНОМУ ИСПОЛНЕНИЮ

АНИСИМОВА С. В., ГОЛОВАТЫЙ А. И., АДУЦКЕВИЧ И. А.

Белорусский государственный университет
(г. Минск, Республика Беларусь)

E-mail: sofiaanisimova@yandex.by, dutsik@gmail.com, alex.halavatyi@gmail.com

Аннотация. В работе проанализированы существующие алгоритмы распознавания песен для оригинального и акапельного исполнения. На основе проведенного анализа разработан алгоритм, состоящий из двух этапов: хеширования и поиска. Подобраны параметры хеширования и показана экспоненциальная зависимость объёма занимаемой памяти от данных параметров. Проведено экспериментальное исследование алгоритма и определена степень достоверности результатов его работы. Предложены возможные пути для повышения достоверности полученных результатов.

Abstract. The paper discusses existing algorithms for music recognition based on either original or acapella performance. Based on the conducted analysis, an algorithm is developed, comprising hashing and querying steps. The paper proposes hashing feature selection and establishes an exponential dependency between the features and the amount of consumed memory. The algorithm is studied experimentally, and its resulting accuracy is determined. The possible ways of increasing resulting accuracy further are proposed.

В связи с тем, что в современной индустрии распознавания человеческой речи практически не существует алгоритмов, которые способны качественно распознавать песню, напетую человеком акапельно, а существующие не находятся в открытом доступе, данное направление исследований представляет особый интерес. Основной целью работы является разработка алгоритма для распознавания пения, на основе изучения и модификации существующих алгоритмов распознавания. Из существующих программ для распознавания песен наиболее популярной является Shazam, поэтому при анализе существующих решений был взят именно её алгоритм [1]. Однако в ходе проведенного анализа этой программы выявлена низкая достоверность результатов для акапельного исполнения. Основными его недостатками являются использование при анализе для распознавания полного частотного диапазона и неустойчивость к условиям дефицита частотного диапазона записи. Так как запись каждого исполнения даже одной и той же песни отличается уровнем шума, основными частотами исполнения, темпом и громкостью, необходимо выделить параметры аудиозаписи, которые будут неизменны для каждого исполнителя. Для практической реализации выбран язык разработки Python, как язык с большим количеством библиотек с открытым исходным кодом для обработки различных данных, в том числе и аудиосигналов.

Алгоритм состоит из двух основных этапов: хеширование аудиозаписи и поиск по базе данных. Алгоритм хеширования достаточно неплохо описан в статье [1], поиска – не описан. В представленной реализации предлагается обрезать частотный диапазон до диапазона человеческого голоса, так как при попытке распознать акапельное пение весь звук в других диапазонах априори является шумом, поскольку человек не может издать звук в другом диапазоне.

Стоит отметить, что самого алгоритма Shazam нет в открытом доступе. Поэтому нет оснований достоверно утверждать, что в данной работе удалось воспроизвести его абсолютно точно. Данный алгоритм рассматривался не как конкретная разработка, а как «чёрный ящик», разработанный по принципу, описанному далее.

Запись приходит на вход алгоритма хеширования в формате MP3, с помощью Python-библиотеки [2] она преобразуется в амплитудно-частотную зависимость от времени. В ней выделяются локальные экстремумы по принципу суперпозиции – точка считается подозрительной на локальный экстремум, если она обладает большей интенсивностью, чем точки в окрестности. Выделенные локальные экстремумы используются в качестве хэш-представления.

Помимо простоты сопоставления, в таком виде хранимые данные занимают намного меньше места. Визуализированные хеши для оригинала записи песни и её акапельного исполнения представлены на рис. 1. Каждая точка на данных визуализациях является парой ключ-значение: время-частота экстремальных точек.

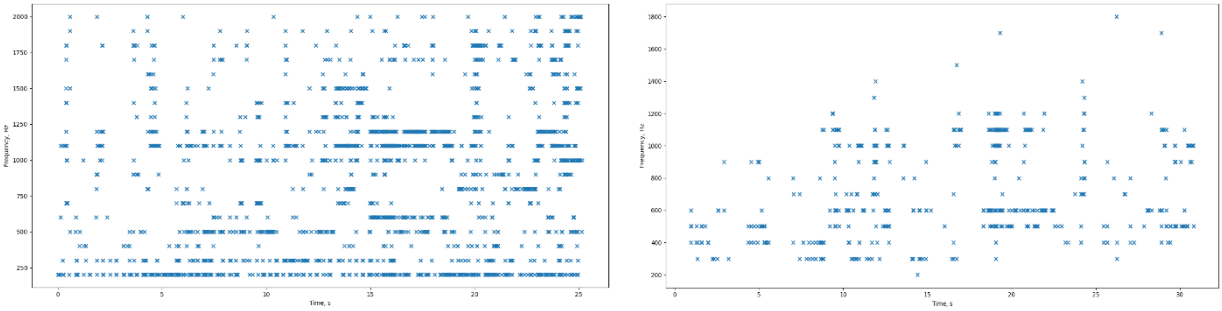

Рис. 1. Визуализированные хеши

Чувствительность алгоритма можно регулировать с помощью специального показателя толерантности. Он влияет на разницу интенсивности сигнала в точке, необходимую для того, чтобы считать точку экстремальной, тем самым влияет на точность анализа и на занимаемый объём памяти. При нулевом уровне этого показателя абсолютно каждая точка будет являться экстремумом, а при очень большом практически никакая, кроме экстремальных шумов. Так же при увеличении показателя толерантности объём необходимой памяти для хранения данных о записи будет экспоненциально уменьшаться по закону:

$$y = y_0 + A \cdot e^{R_0 \cdot x} \qquad (1)$$

Однако, на вход поискового алгоритма подаётся не вся запись целиком, а лишь небольшой фрагмент до 30 секунд, который, кроме того, может быть значительно зашумлён. Из-за этого сопоставление хеши напрямую, ведя отсчёт от временного нуля, в общем случае некорректно. Поэтому данные о времени, хранимые в хеше в исходном виде, не несут ценной информации. Куда важнее временные интервалы между экстремальными точками. Поэтому после получения хешей записи на вход поискового алгоритма они описываются с помощью коллекции объектов типа Target. Эти объекты накладываются на хеши, полученные по алгоритму, описанному выше, внахлёст. В структуре объектов хранятся данные о расстоянии от точки, находящейся в левом нижнем углу, до всех остальных. Далее наборы таких объектов сравниваются с имеющейся базой данных.

Результатом работы программы являются две-три песни с примерно равным количеством совпадений объектов Target. На данном этапе алгоритм ещё недостаточно точен. Одним из вариантов улучшения качества работы алгоритма является ситуативное исключение экстремальных точек из эталонного образца в зависимости от диапазона экспериментальной записи. Основной проблемой данного алгоритма является зависимость результата от частотного диапазона исполнителя, а он может колебаться от 75 Гц (бас) до 1100 Гц (сопрано) [3]. При полном или хотя бы частичном перекрытии диапазонов оригинального исполнителя и акапельного поисковый алгоритм мог справится с поставленной задачей. При несовпадении тембров исполнителей распознать песню с помощью такого метода становится невозможным. Экспериментально установлено, что самым удобным решением данной проблемы является использование логарифмической шкалы описания звука. В частности, в качестве одной из шкал можно использовать нотное описание. Одинаковые ноты разных октав являются обертонами и воспринимаются человеком как один и тот же звук. [4] То есть, при исполнении одной и той же партии в разных октавах, мелодический рисунок для слушателя не искажается. На основании этого вводится предположение, что, если использовать алгоритм, распознающий ноты по записи вокального исполнения, возможно будет не только распознавать акапельное пение, но и напев без слов. Кроме того, использование нотации для описания звука уменьшит объём хранимой хеш-таблицы, чем значительно увеличит скорость поиска по базе.

Список использованных источников

1. Li-Chun Wang, An Industrial-Strength Audio Search Algorithm, Shazam Entertaiment, Ltd 2003.
2. Веб-сервис для хостинга IT проектов [Электронный ресурс] – Режим доступа https://github.com/AllenDowney/ThinkDSP, – Дата доступа: 14.06.2013
3. Мефферт Б., Хохмут О., Инструменты обработки сигналов – основы, примеры применения и задачи, Берлин: Изд-во HUB, 2019. – 320 с.
4. Гельмгольц Г., Учение о слуховых ощущениях как физиологическая основа для теории музыки, Москва: Книжный дом «ЛИБРОКОМ», Изд. 3-е, 2013. – 592 с.
5. Allen B. Downey, Think DSP. Digital Signal Processing in Python. Version 1.0.9., Massachusetts: Green Tea Press, 2014. – 152.