Juggle: Large-scale Discovery in Music Recommendation

Filipe Coelho
Laboratório SAPO/U.Porto
Universidade do Porto
filipe.coelho@fe.up.pt

José Devezas
Laboratório SAPO/U.Porto
Universidade do Porto
jld@fe.up.pt

Cristina Ribeiro
DEI–FEUP & INESC TEC
Universidade do Porto
mcr@fe.up.pt

ABSTRACT

Today's offer of audio content exceeds the human capability of manually searching datasets with hundreds of songs, demanding automated tools capable of handling music recommendation when faced with large-scale collections.

In this work, we address the playlist generation and song discovery tasks with large-scale datasets. It is possible to quickly obtain playlists and explore collections with example-based queries using audio features, lyrics and tags.

We developed a music discovery prototype to demonstrate this content based approach. This demo is based on the Million Song Dataset, a large-scale collection of audio features and associated text data comprising almost 300 GB of information.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*abstracting methods, indexing methods*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*clustering, information filtering, search process, selection process*

General Terms

Performance, Human Factors, Experimentation

Keywords

Music search, song discovery, playlist generation

1. INTRODUCTION

The explosion of multimedia content generated over the last decades, aided by continuous technological advancements in storage and processing hardware, as well as smaller capturing and consuming devices available to the general public, has put us in a situation of information overload.

Much research has been made on content-based analysis of multimedia data [1]. Despite the existence of the Semantic Gap, that is, the the separation between the high-level concepts used in queries and the low level features that can be easily obtained from images or songs, good results have been obtained in tasks such as automatic text illustration and visual exploration of large-scale datasets.

In this work, we use content-based similarity and approximate search methods following a late-fusion approach that can be applied to large-scale datasets and used in highly interactive scenarios where real-time answers are demanded in order to provide a useful user experience.

1.1 The Million Song Dataset

The Million Song Dataset (MSD), released in 2011, is a freely-available collection of audio features and metadata for a million contemporary popular music tracks [2]. This multimedia dataset represents a significant step in this area, with the objective of encouraging research on large-scale algorithms, provide a reference evaluation dataset and help new researchers in Music Information Retrieval.

1.2 Crossmedia Retrieval

We state it is possible to take advantage of the efficiency of textual indexing by mapping audio features to a textual form and indexing them with textual search engines, as previously done for image retrieval [3, 4]. Song metadata is preprocessed and stored in an inverted index. Audio low-level features are analyzed and used to build audio feature vectors, which are then transformed into a textual representation designated as *Surrogate Text Representation* (STR) [3]. These representations are handled in a common index and provide the means to search songs by similarity on both audio and textual features, including existing metadata and lyrics.

2. APPLICATIONS

Given the subjective nature of playlist generation and music discovery retrieval tasks, we present results from some predefined queries in order to demonstrate the usefulness of the content-based approach.

2.1 Playlist generation

Given a text query by the user, be it the name of a song or artist, a lyrics excerpt or emotion tags such as "happiness" or "betrayal", the system performs a textual search over the indexed fields of each song. This results in an initial playlist with no more than twenty songs. A subset of ten, for the "coldplay live" query, is shown in Table 1. "Score" refers to the Lucene text retrieval score. In this example, users wanted live performances from the Coldplay band, but they

OAIR'13 May 22-24, 2013, Lisbon, Portugal.

Table 1: Initial playlist.

Query: "coldplay live"	Score
See You Soon (Live In Sydney) - Coldplay	5.1
Shiver (Live In Sydney) - Coldplay	5.1
One I Love (Live In Sydney) - Coldplay	5.1
Amsterdam (Live In Sydney) - Coldplay	4.2
You Only Live Twice (Live Norway) - Coldplay	4.0
Daylight - Coldplay Tribute	3.9
Moses (Live In Sydney) - Coldplay	3.7
Yellow (Live In Sydney) - Coldplay	3.4
Speed Of Sound (Live) - Coldplay	3.2
Fix You (Live) - Coldplay	3.2

Table 2: Final playlist.

After Audio Re-Rank	Score
Shiver (Live In Sydney) - Coldplay	5.1
Moses (Live In Sydney) - Coldplay	3.7
One I Love (Live In Sydney) - Coldplay	5.1
One I Love	3.2
Pour Me (Live At The Hollywood Bowl)	3.2
See You Soon (Live In Sydney) - Coldplay	5.1
Warning Sign	3.1
Fix You	3.1
Daylight - Coldplay Tribute	3.9
The World Turned Upside Down - Coldplay	3.2

Table 3: Playlist starting with a specific song.

Selected song: "Sleeping Sun"	Score
Sleeping Sun - Coldplay	3.2
You Only Live Twice (Live Norway) - Coldplay	4.0
Shiver (Live In Sydney) - Coldplay	5.1
Moses (Live In Sydney) - Coldplay	3.7
One I Love (Live In Sydney) One I Love	3.2
Pour Me (Live At The Hollywood Bowl)	3.2
The World Turned Upside Down - Coldplay	3.2

could also insert parts of songs or tags describing their mood or a specific music genre.

We apply a content-based rerank on the feature vectors of this initial playlist, based on the audio similarity between songs. Using the Hamming distance, we sum the distances between songs. Songs with a smaller total distance value, that is, with greater similarity to all the others, become more "central" in the playlist.

We state that tracks that are more similar between them represent a "cluster" of songs that may appeal the user, while "outliers" will be pulled to the end of the playlist. The result is displayed on Table 2.

Another option is to pick a song and reorder the list "by closest". We add the initial chosen song to an empty list, and the following song becomes the closest song that's not already on the list. By using this option, we allow the user to start with a favorite song and progress through the playlist with minimum disruption, as each next song is the closest to the current one, as shown in Table 3.

2.2 Song discovery

The main advantage of the content-based approach is its user-independent nature, not affected by item popularity.

Table 4: Similar songs – audio features.

Query: "One I Love (Live In Sydney)"	Score
One I Love (Live In Sydney) - Coldplay	17.0
Banda De Rock & Roll - Ratones Paranoicos	17.0
Silver Strand (Album Version) - The Corrs	17.0
Time (24-Bit Digitally Remastered 05) - Blind ...	14.8
Amplified Ohm - Melting Euphoria	14.7
Tomorrow Is Coming - Ocha la Rocha	12.5
Jalousi (Igen!) - Peter Sommer	12.2
Are You Anywhere? (edit) - Padded Cell	11.5
One I Love - Coldplay	11.5
Let The Sky Fall - Ten Years After	11.5

Table 5: Similar songs – lyrics and tags.

Query: "One I Love (Live In Sydney)"	Score
One I Love - Coldplay	4.7
One I Love (Live In Sydney) - Coldplay	3.9
You - Mr. Sancho	3.1
Desperado - Journey South	3.1
You Are The One Lalala - Morten Abel	2.8
You Are Everything - Dru Hill	2.8
Blame It On Me - Aaron Watson	2.7
Just The Way (Explicit) - Alfonzo Hunter	2.7
Sprung - B2K	2.6
If Work Permits - The Format	2.6

As shown in Table 4, by using the live performance of the "One I Love" song, we were able to also retrieve the original version using the audio-only query, which indicates that even with the approximate indexing algorithm and binary hashing scheme, the audio features used in this work do capture meaningful information. Table 5 shows the same attempt in discovering new songs, but using tags and lyrics information.

From a system performance point of view, the playlist generation and song discovery tasks are performed in seconds using the Lucene library. The full index, with metadata and audio features, takes over 12 GB and is not fully loaded to memory. Instead, Lucene handles its cache and retrieves only documents that are considered relevant for the query.

For even larger-scale collections, it is possible to split the index and content information between Lucene and a separate database, loading the index to RAM and retrieving data for visualization in a separate process.

3. REFERENCES

[1] M.S. Lew, N. Sebe, C. Djeraba, and R. Jain: "Content-based Multimedia Information Retrieval: State of the Art and Challenges," *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2011.

[2] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere: "The Million Song Dataset," *Proceedings of the International Society for Music Information Retrieval Conference*, 2011.

[3] G. Amato, P. Bolettieri, F. Falchi, C. Gennaro, and F. Rabitti: "Combining Local and Global Visual Feature Similarity Using a Text Search Engine," *Proceedings of the International Workshop on Content-based Multimedia Indexing*, 2011.

[4] F. Coelho, and C. Ribeiro: "Image Abstraction in Crossmedia Retrieval for Text Illustration," *Proceedings of the European Conference on Information Retrieval*, 2012.