Album Similarity based on Chord Progressions

Sergio Llana and Pau Madrero

Polytechnical University of Catalonia

Abstract. This paper aims to evaluate how similar some albums are based on their harmony. For that, we have modelled the chord progressions of the songs of the albums as graphs. Then, we have compared three different ways of computing the graph similarities between all 2-combinations of albums.

Keywords: chords, chord progression, album similarity, graphs

1 Introduction

Thanks to platforms such as Spotify or SoundCloud, music recommendations have become a trending topic in machine learning and artificial intelligence. The most usual tool used by recommendation systems is collaborative filtering, but regarding music, there are some features such as harmony or lyrics which can be more reliable to find similar songs.

With respect to harmony, we aim to find harmonic similarities between albums using their songs' chord progressions. Note this is not an easy task, because of variable length and complex patterns in the progressions. This is not only interesting for recommendations, but to analyze and understand the way music composers work.

We will use graphs to represent the chord progressions and their patterns. This is so, because their complex structure makes it difficult to represent them in any other way. Furthermore, it will allow us to use graph related algorithms and graph theory to study our data.

In the first sections, a brief summary of the previous related work will be done, as well as some remarks on music theory (specifically in harmony) and graph theory. Then, we will explain how we obtained our data, and the procedures we have followed to process it and analyze it. Finally, we will end up with some conclusions and future work.

1.1 Related Work

Regarding the related work of this paper, it is clear it mixes two fields: graph similarity and chord progression analysis. We have found a lot of bibliography and papers about those fields separated but none of them together.

On the one hand, regarding graphs, we should distinguish between node similarity and graph similarity. For the former one, there are a lot of methods such as kernels from graphs, which take into account node information and graph's structure. For the latter, *Danai Koutra et al.* performed a useful survey of different methods for graph comparison in [1].

T. Ito and M. Shimbo et al. in [2] and [3] developed a kernel-based framework for calculating relatedness and importance. They show that the HITS algorithm could be extended with Neumann Kernels, which are an efficient mechanism for determining the relatedness of nodes.

On the other hand, concerning chord analysis, we have found interesting the work done by *Peter Kiefer and Manda Riehl* in [5] doing Markov Chain Analysis of classical music chord progressions. In addition, we have found some applications of chord analysis. A clear example is [4], that is a clustering of artist based on a n-gram model of chord progressions.

1.2 Theory

Graph concepts We will introduce several concepts that were used in the implementation of our work. Starting by the **diameter** of a graph, which is the length of the longest shortest-path of the network:

$$d = max_{i,j} d_{i,j} \tag{1}$$

Transitivity measures the probability that the adjacent vertices of a vertex are connected. This is sometimes also called the **clustering coefficient**. We can compute it for a node (C_i) and then average over all the network:

$$C_i = \frac{2 * connections - between - neighbours}{k_i(k_i - 1)} \tag{2}$$

$$C = \frac{1}{n} \sum_i C_i \tag{3}$$

Based on the HITS algorithm, **Neuman Kernels** provide a framework for evaluating the relatedness of nodes in an efficient way. It is able of balancing relatedness and importance by tuning the parameter gamma (i.e. decay factor).

In order to compute the Neumann Kernels, the document correlation matrix K and the term correlation matrix T are computed as follows:

$$K = X^T X \tag{4}$$

$$T = XX^T \tag{5}$$

The Neumann Kernel defines two matrices $\hat{K}$ and $\hat{T}$ which give the similarity between documents and the similarity between terms respectively. The previous equations are embedded in an iterative algorithm until convergence.

$$\hat{K} = K(I - \gamma K)^{-1} \tag{6}$$

$$\hat{T} = T(I - \gamma T)^{-1} \tag{7}$$

If we apply this idea to a network, we could obtain a ranking of most related and important chords of an album. When comparing two albums' rankings, we could use **Kendall's Tau correlation coefficient** to compute the similarity of their rankings (a.k.a. top-k lists).

There are several ways to generalize Kendall's Tau to measure distances between top-k lists and we have chosen the minimizing Kendall distance $K_{min}(x1, x2)$. We begin by generalizing the definition of the set P. Given two top-k lists $x1$ and $x2$, the set of all pairs of distinct elements in $D_{x1} \cup D_{x2}$ as $P(x1, x2) = P_{D_{x1} \cup D_{x2}}$.

For top-k lists $x1$ and $x2$, the minimizing Kendall distance $K_{min}(x1, x2)$ between $x1$ and $x2$ is defined to be the minimum value of $K(\sigma 1; \sigma 2)$, where $\sigma 1$ and $\sigma 2$ are each permutations of $D_{x1} \cup D_{x2}$.

Finally, we want to define what a **edit distance** is for graphs. Graph edit distances are graph similarity measures analogous to Levenshtein distance for strings. It is defined as the minimum cost of edit path (sequence of node and edge edit operations) transforming graph G1 to graph isomorphic to G2.

Music theory In musical terms, **harmony** is the study of simultaneous sounding tones, which usually are **chords**.

Chords are combinations of three or more pitches sounding at the same time which can be a major or minor scale. They are said to be in a **key**, which identifies by their tonic. A **chord progression** is understood as the movement from one chord to the next (i.e. a succession of chords).

Finally, **transposing** is the process of rewriting a piece of music so that is sounds higher or lower in pitch. This involves raising or lowering each pitch by the same interval.

This definitions have been extracted from [6].

2 Own work

Our work aims to evaluate how similar some albums from three arbitrarily chosen groups are based on their harmony. For that, we have modelled the chord progressions of the songs of each album, and build a graph. Then, we have compared three different ways of computing the similarities between all 2-combinations of graphs.

In this section we will introduce the structure used to represent our data, the procedures followed to analyze it, and we will finally discuss the results.

2.1 Data set

We will start by talking about our data set, since all the experiments performed use it and it is imperative to know about it in order to fully understand the procedures. As previously mentioned, our data set contained song progressions from albums of three different bands: *Arctic Monkeys*, *The Last Shadow Puppets* and *Franz Ferdinand*. The first two bands share the same lead singer and main songwriter (Alex Turner), whereas the third only shares the music style with the rest (Indie Rock).

We have used a homemade scraper program to download all the chord progressions of our chosen songs. Firstly, we used Spotify's API [9] to retrieve information about the songs belonging to the albums we wanted to use. Then, we used Ultimate Guitar's [10] search engine to find the webpage with the chords of our songs with highest rating. Finally, the scraper extracted the chord progression from the HTML and stored it for our later use.

At this point, our data set just contained a list of songs, with their corresponding chord progression stored as an ordered list. In order to structure this information in the form of a graph, we grouped the songs by album and created a new graph per album. Then, a unique vertex was created for each distinct chord that appeared in any of the songs of the album.

The edges were created in order to represent the consecutiveness of chords in a chord progression. In other words, a new edge between a chord node A and a chord node B was created every time the chord A appeared as the strict predecessor of B in any chord progression of the album. Figure 1 shows the actual graph of one of the selected albums.

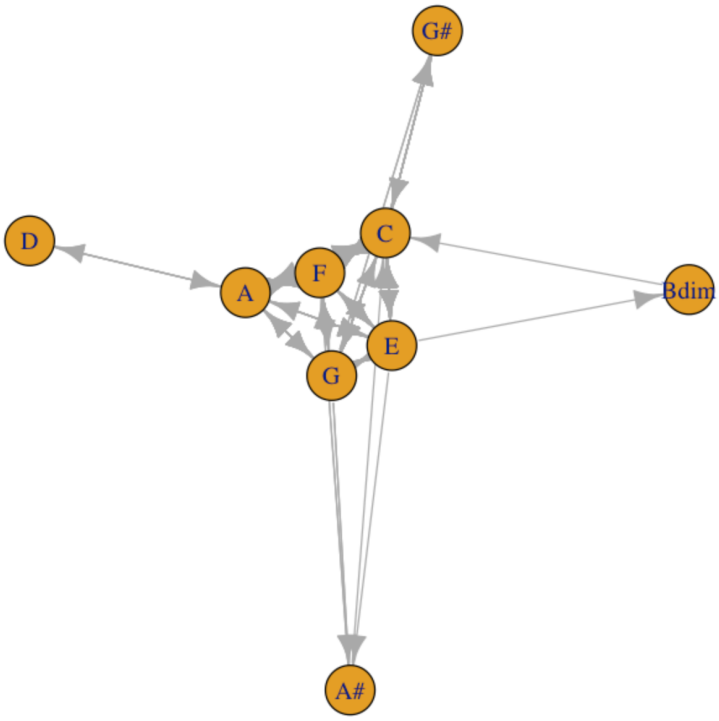

Fig. 1: Graph for the album *You Could Have Done So Much Better*.

2.2 Pre-processing

Before generating the final graphs, we needed to apply normalization to the chord sequences. As our goal was to estimate the harmonic similarity of albums, the tonality was not important. Then, each song was transposed to the C key.

This procedure was somehow naive. After doing some research on the topic, we realised that it is quite difficult to find out the actual key of a song by just having its chord sequence. Knowing more about the structure of the song (e.g. verse, chorus...) would have helped to figure it out, but it was hard to do without trespassing the scope of our project.

Our solution consisted on comparing the song's sequence to different arrays of chords, each one containing the chords most frequently used when playing in each different key. Note that we have also normalized the existence of sharps and flats. In order to avoid having the same chord twice in the graph (e.g. C and Db) we converted all flats into their equivalent sharps.

2.3 Experiments

Once our data set was ready to be used, we proceeded to perform a few experiments. First of all, in order to have a useful first view of the data, which could help us later understand the results, we computed some metrics of each graph. Table 1 shows this results. N corresponds to the number of vertices of the graph, E to the number of edges, k corresponds to the mean degree and delta to the density of the network. Finally, we computed the diameter of each graph, as well as its clustering coefficient.

	N	E	k	delta	diameter	cc
AM	10	820	164.00000	18.222222	3	0.6000000
Favourite Worst Nightmare	22	648	58.90909	2.805195	6	0.4333333
Humbug	15	636	84.80000	6.057143	5	0.5185185
Suck It and See	27	1527	113.11111	4.350427	6	0.5151515
Tranquility Base Hotel & Casino	50	1659	66.36000	1.354286	5	0.3423512
Whatever People Say I Am, That's What I'm Not	24	788	65.66667	2.855072	6	0.4520548
Always Ascending	11	402	73.09091	7.309091	3	0.7288136
Franz Ferdinand	24	336	28.00000	1.217391	7	0.4308511
Right Thoughts, Right Words, Right Action	11	674	122.54545	12.254545	3	0.6532258
Tonight: Franz Ferdinand	14	320	45.71429	3.516484	3	0.5806452
You Could Have It So Much Better	9	194	43.11111	5.388889	3	0.6835443
Everything You've Come To Expect	18	558	62.00000	3.647059	5	0.5378151
The Age Of The Understatement	12	650	108.33333	9.848485	4	0.6132597

Table 1: Graph metrics for each one of the albums.

Then, we started the actual comparison of methods for computing the similarity between the albums.

Similarity based on graph metrics This first method is the simplest one, but the less related with graphs. Using the previous table of metrics, we computed the correlation matrix of each combination of albums to create a heat-map. Note that more complex metrics could have been used.

Neumann Kernels + Kendall's Tau The second method was composed by two phases. Firstly, we computed the K and T correlation matrices with the Neumann Kernel algorithm which gave us a ranking per album indicating the most important chords. Then, using the Kendall's tau correlation coefficient [7], we computed the distance between all combinations of albums.

We faced a problem when comparing the rankings, since some chords appeared only on one of the rankings. Despite some other and better approaches proposed in [8], due to time limitations we decided to go for the simpler and more naive approach of using only those chords which were common in both rankings. This

resulted in comparisons with lists of rankings with different lengths. To diminish the effect of these differences, we chose to use the normalized version of the Kendall tau distance, explained in [7].

Note that, in order to be able to compare the results with the previous method, the distance must be converted into a similarity. We have used the following equation:

$$sim(x, y) = \frac{1}{1 + d(x, y)} \tag{8}$$

Edit distance for graphs Finally, this last method was programmed in Python to take advantage of the *networkx* library, as *igraph* doesn't contain this functionality. As it was described in the Theory section, it consisted on counting how many nodes and edges we must add/modify/remove to get from one graph to another.

2.4 Discussion

The result of the first method is showed by *Fig. 2*:

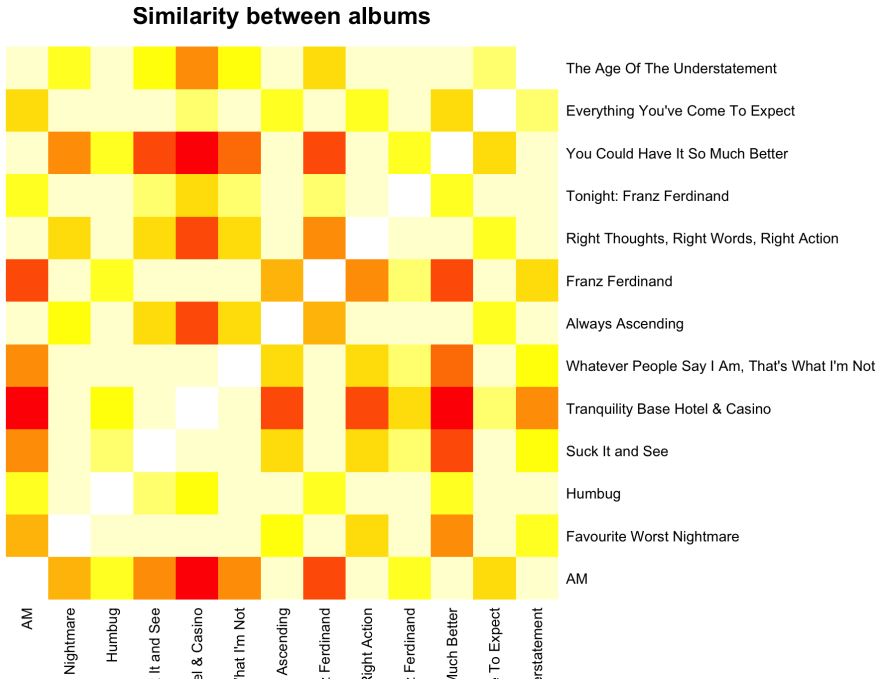

Fig. 2: Similarity of albums based on graph metrics.

It can be seen how albums such as *AM* and *Favourite Worst Nightmare* or *Tranquility Base Hotel & Casino* and *Humbug* seem to be quite similar and it make sense because they all belong to *Arctic Monkeys*. In addition, it also captures that *You Could Have It So Much Better* and *Franz Ferdinand* are similar, both written by *Franz Ferdinand*.

On the other hand, the result of the second method can be seen in *Fig. 3*:

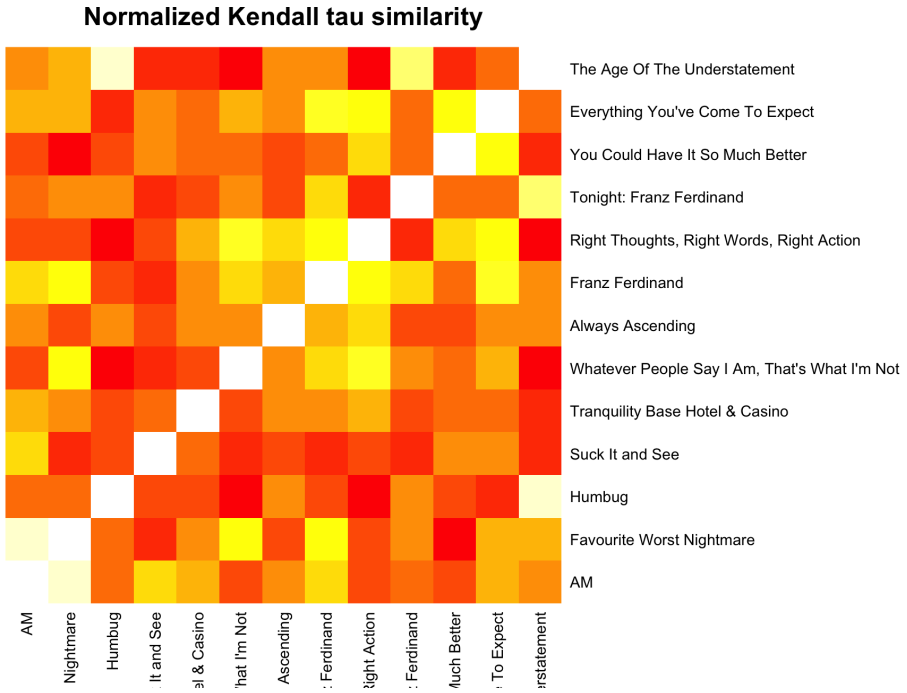

Fig. 3: Distance of albums based on Neumann Kernels + Kendall's tau.

Albums by *Arctic Monkeys* keep looking quite similar between them, however the ones by *Franz Ferdinand* doesn't appear to be as similar as with the previous method. It is worth noting that albums by *The Last Shadow Puppets* are more similar to the ones by *Arctic Monkeys* than to the ones by *Franz Ferdinand*, specially with *Tranquility Base Hotel & Casino* which was also pointed out by music critics when it was released last year. Recall that this two groups share the same songwriter.

Finally, provided the time constraints and that we are using our own laptops to run this project, the last method could not be tested with the available data because of it is large computational cost. However, it is implemented in the delivered code.

3 Conclusions

A first and introductory analysis of chord progressions in modern music has been done. First, we have collected and preprocessed the sequence of chords of the discography of three well-known bands and modelled as directed graphs. Finally, we have compared several methods for computing its pair-wise similarity.

Note all insights extracted in the Discussion section should be interpreted with caution because of three reasons. Firstly, we have done some assumptions in the process of normalization of the chord progressions (e.g. the key has been estimated). Secondly, we assume that the chords, updated by the users of *UltimateGuitar* are accurate. Finally, we are working with "one musical dimension" by focusing only on harmony.

3.1 Future work

As we have mentioned before, we have done some assumptions and simplifications due to time limitations. In the future, some of the procedures performed could be improved. Starting by the pre-processing phase, with the help of people with a music background, parts such as the search of the key of each song could be enhanced with more complex algorithms.

The experiment of the Neumann Kernels could probably be improved by using a better method of dealing with mismatched chords when comparing rankings with Kendall's Tau distance. As proposed by [8], a better approach could be to add the mismatched chords at the last position of the other ranking, and then adding dummy chords in all the ranking positions to compensate for the "overpopulation" on the last one.

Finally, with more time and better computing resources, the edit distance experiment could be executed to get another view on the comparisons made.

References

1. Danai Koutra et al. Algorithms for Graph Similarity and Subgraph Matching. 2011. https://www.cs.cmu.edu/~jingx/docs/DBreport.pdf
2. T. Ito, M. Shimbo, T. Kudo, and Y. Matsumoto. Application of kernels to link analysis. In Proc. 11th ACM SIGKDD. 2005
3. T. Ito, M. Shimbo, D. Mochihashi, and Y. Matsumoto. Exploring communities with kernel-based citation analysis. In Proc. PKDD'06. 2006.
4. Brandt Absolu, Tao Li, and Mitsunori Ogihara. Analysis of Chord Progression Data. 2010 https://www.researchgate.net/publication/227323875_Analysis_of_Chord_Progression_Data
5. Peter Kiefer and Manda Riehl. Markov Chains of Chord Progressions. 2016. http://lib.bsu.edu/beneficencepress/mathexchange/10-01/markovchainschordprogressions.pdf

6. The Method Behind The Music. https://method-behind-the-music.com/theory/dictionary/
7. Ronald Fagin, Ravi Kumar and D. Sivakumar. Comparing top-k lists. 2003. http://www.cs.tau.ac.il/~matias/courses/Seminar_Spring03/topk.pdf
8. Rogier van der Geer. Using Kendall's tau to compare recommendations. 2018. https://blog.godatadriven.com/kendall-tau-recommendations
9. Spotify API — Spotify for Developers https://developer.spotify.com
10. ULTIMATE GUITAR TABS — 1,100,000 songs catalogue https://www.ultimate-guitar.com