The Next Frontier in DJ Software Innovation Is Analytics Driven

Overview

DJ software have evolved over the past decade to provide DJs with a wide array of features for music manipulation, such as loopers, samplers, beat match, and filters. While all these features were built around ***how*** DJs can perform their music, little advancements have been made on providing DJs with insights on ***what*** music to play.

A key differentiator of a skilled DJ is their ability to blend different songs together into a cohesive set. By executing seamless transitions between songs, DJs can introduce the audience to new ideas while sustaining the energy and momentum of the dancefloor – i.e., keep the crowd dancing! Because two or more songs are simultaneously playing during a transition, **DJs must develop an intuition for rhythmic and harmonic differences between songs to ensure that they blend well together**.

Most DJ software compute two variables to help DJs assess these differences: beats per minute (BPM) and key. While BPM is effective for evaluating rhythmic differences, **key is often an unreliable variable for evaluating harmonic differences** due to complexities within a song such as sudden key changes or unconventional chord progressions. Furthermore, **as key is a categorical variable, it cannot provide insight into the magnitude of harmonic differences between songs**.

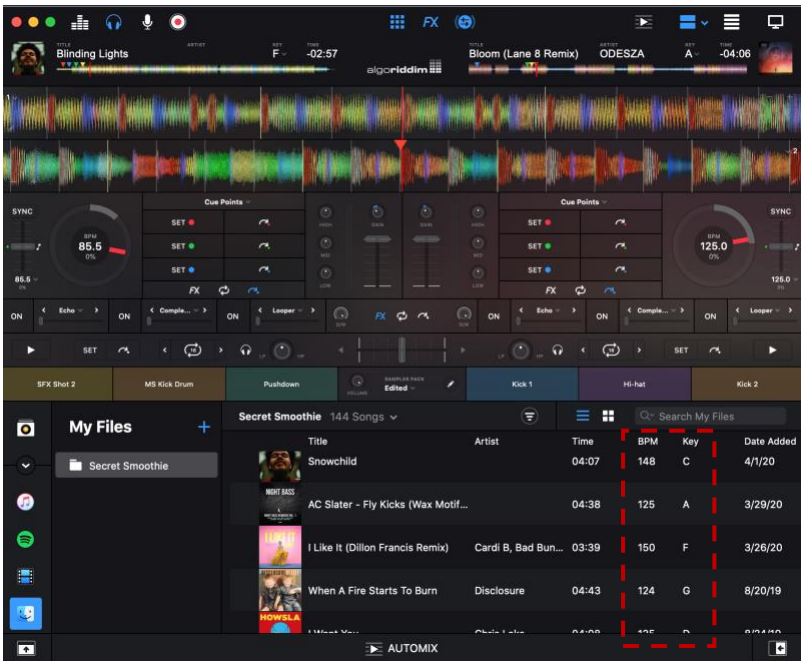

Figure 1: While modern DJ software have many song manipulation features, they lack tools for assessing harmonic compatibility (Algoriddim djay Pro 2)

Instead of relying on key, DJs could manually test the harmonic compatibility of song pairings prior to a performance. **However, it is impractical to experiment with every possible song pair combinations of a substantial music collection.** Even my collection of as little as ~150 tracks has over ten thousand possible pairs.

As my music collection continues to grow, I was inspired to develop more systemic and sophisticated approaches for uncovering harmonic connections in my music library. With the LibROSA and NetworkX packages and Gephi software, it is now easier than ever to experiment with music information retrieval (MIR) techniques and visualize insights from music data.

Through my exploration of data-driven DJ solutions, I was able to:

- Design a more robust metric for evaluating harmonic similarities between songs
- Generate an interactive network visualization to explore harmonic connections within a music library
- Develop functionalities inspired by network theory concepts to assist with song selection and tracklist planning

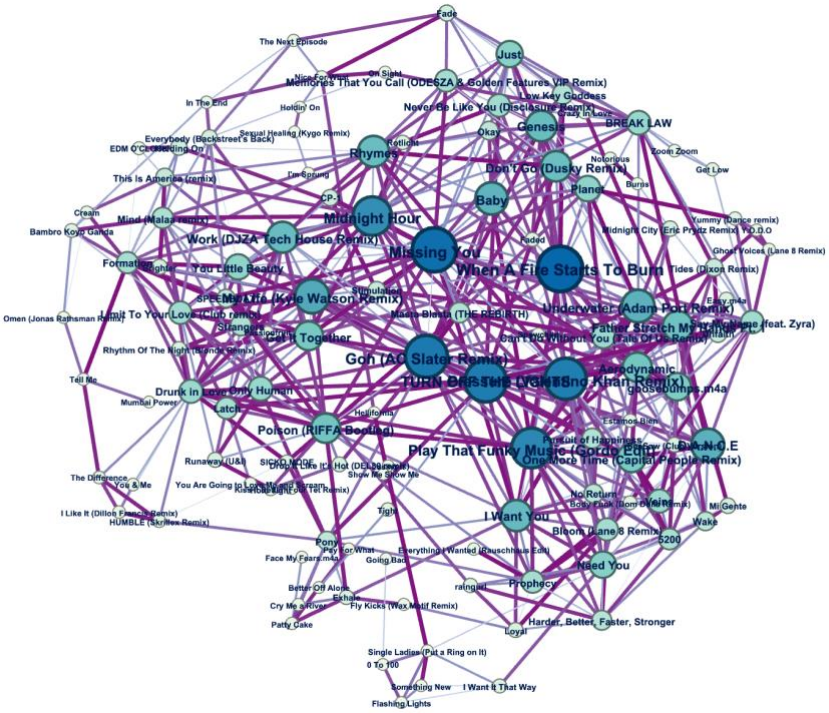

Figure 2: Visualization of my music collection as a network consisting of nodes as song, node size/shade relative to song importance, edges as harmonic connections, and edge thickness relative to harmonic distance between songs

After experimenting with initial prototypes, I am convinced that **the incorporation of more sophisticated analytical functionalities is the next frontier in DJ software innovation**.

Designing a more robust metric for evaluating harmonic similarities

There are twelve chroma classes (i.e., pitches) that exist in music: {C, D♭, D, E♭, E , F, G♭, G, A♭, A, B, B♭}. With LibROSA's chromagram computation function, I can extract the intensity of each pitch over the span of an audio file and calculated the overall distribution of each chroma class. **My hypothesis: songs that share similar chroma distributions have a high likelihood of being harmonically compatible for mixing.**

I first tested LibROSA functionality on a short audio clip, in which the chroma characteristics are easily discernable. The first 15 seconds of Lane 8's remix of Ghost Voices by Virtual Self starts with some white noise followed by a string of A's, then G – F♯ – E – D – G[1].

Ghost Voices (Lane 8 remix) – first 15 seconds

The chromagram function outputs a list of 12 vectors, each vector representing one of the 12 chroma classes and each values representing the intensity [0, 1] of a pitch per audio sample (by default, LibROSA samples at 22050 Hz, or 22 samples per second).

Using LibROSA's plotting function, I was able to visually evaluate the audio clip's chromagram:

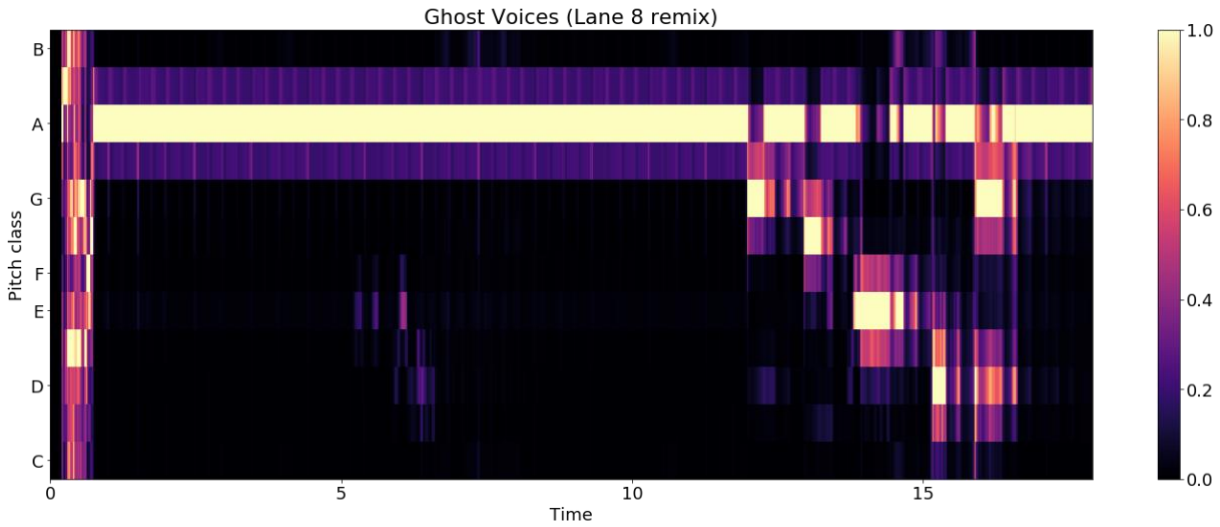

Figure 3: LibROSA captured quite a bit of noise (purple) that were irrelevant to the harmonic footprint of the track

LibROSA successfully captured the key distinguishable notes that appeared in the audio clip (yellow bars). However, it also captured quite a bit of noise (purple/orange bars surrounding the yellow bars). To prevent the noise from biasing the actual distribution of chroma classes, I set all intensity values less than 1 to 0.

[1] F♯ is synonymous to G♭; F♯ is the proper notation in the key of G major

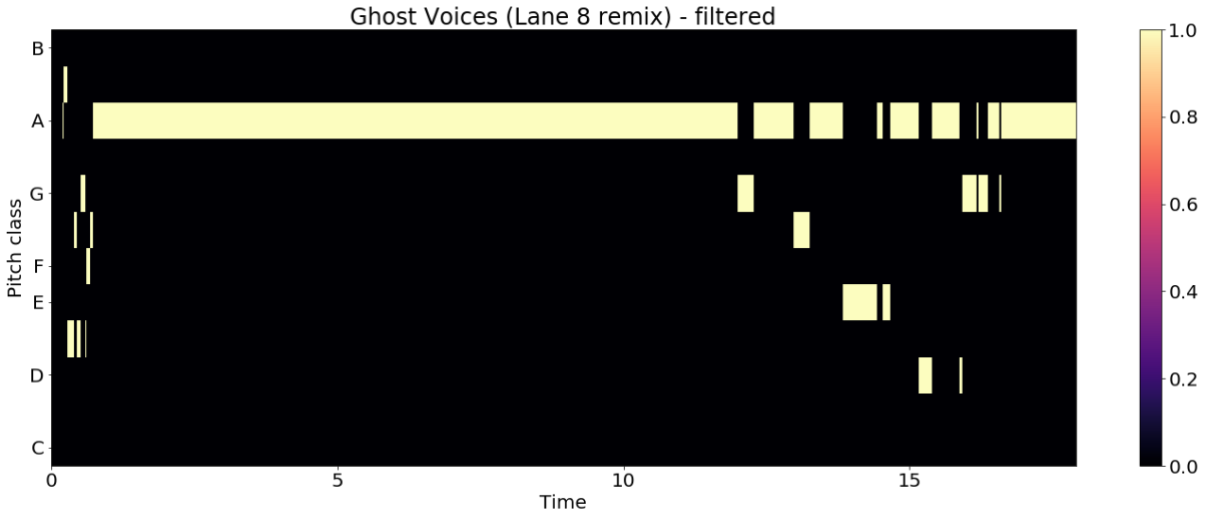

Figure 4: Filtering out noise to prevent biased calculations of chroma distribution

While some of the white noise from the start of the audio clip were still captured, the noise surrounding the high intensity notes were successfully filtered out.

The **chroma distribution** $\boldsymbol{p}$ of a song is then derived by summing the intensity values i_s of all samples $s \in S$ for each of the 12 chroma classes $c \in C$, then divided by the sum of all intensity values:

$$p_c = \frac{\sum_{s \in S} i_{sc}}{\sum_{s \in S, c \in C} i_{sc}} \quad for\ c \in C$$

The impact of filtering out intensity values less than 1 was evident when comparing the chroma distributions between the filtered and unfiltered audio sample:

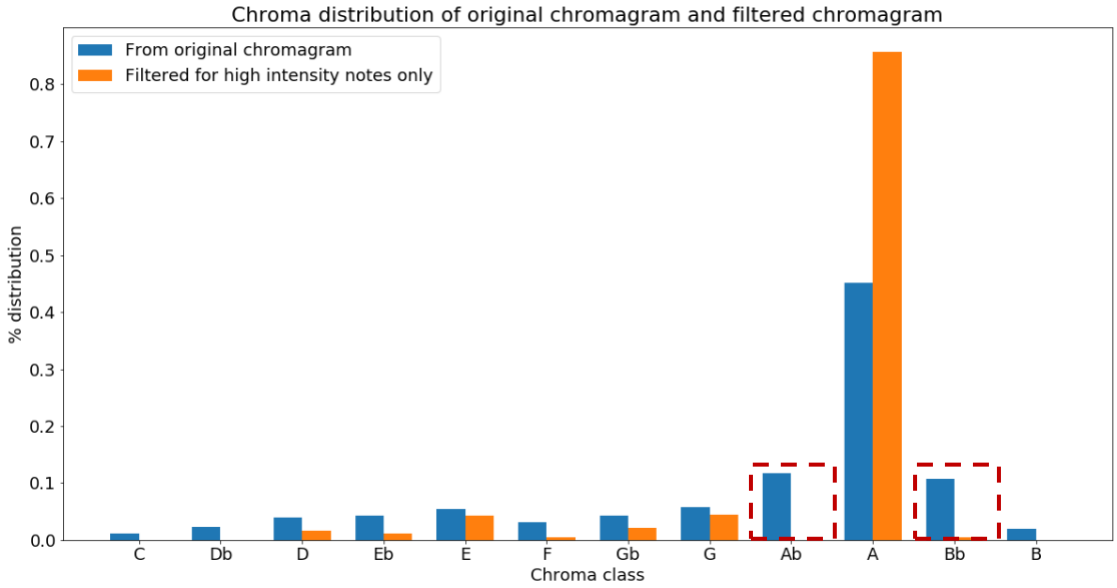

Figure 5: Ab and Bb would have been the second and third most common pitches if the noise were not filtered out

Note that the unfiltered distribution suggested that A♭ and B♭ were the second and third most common pitches from the audio clip, when in fact they were attributed to noise leaking from the string of A's.

The **harmonic distance** d between two songs can then be derived by calculating the sum of squares between their chroma distributions $(\boldsymbol{p}, \boldsymbol{q})$:

$$d = \sum_{c \in C} (p_c - q_c)^2$$

By deriving the chroma distributions of all my music files, I can effectively calculate and rank the harmonic distances of all possible song pair combinations in my library and prioritize the exploration of pairs with the smallest harmonic distances.

One of the more unexpected pairs at the top of the list was I Want It That Way by Backstreet Boys and Flashing Lights by Kanye West. These songs share four of their top five chroma classes and have a harmonic distance of 0.016, which falls in the ~0.5% percentile of ~10k pairs:

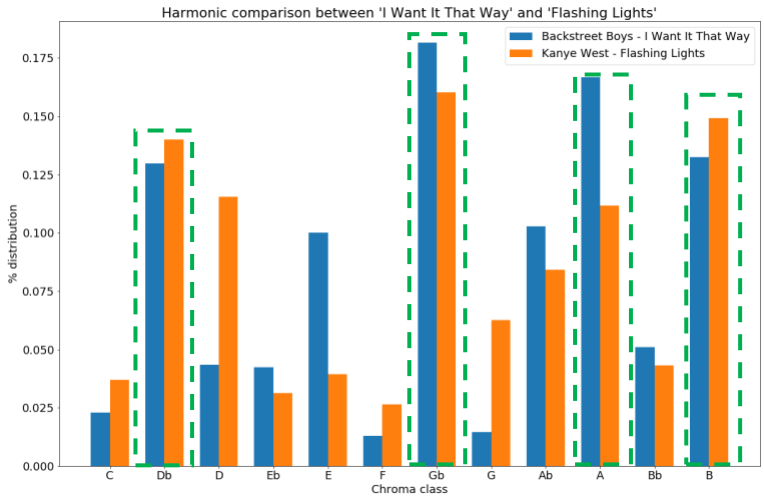

Figure 6: I Want It That Way and Flashing Lights share four out of their top five chroma classes

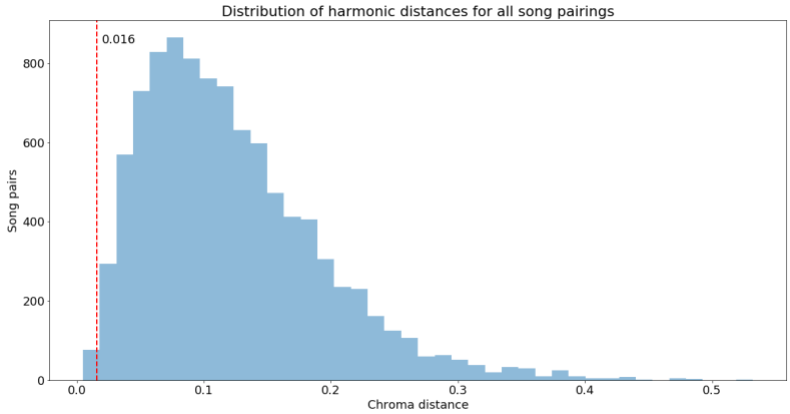

Figure 7: Their harmonic distance value falls in the 0.5% percentile of all pair combinations

Despite the fact that these songs have very few commonalities in terms of style, mood, message, etc., they share a striking resemblance in terms of harmonic characteristics. Here is a mashup I made of the two songs:

I Want It That Way x Flashing Lights mashup

Testing the performance of the harmonic distance metric

To test the effectiveness of this harmonic distance metric, I evaluated 4 sets of 25 song pair recommendations generated from the following models:

Model	Filter out pairs without matching keys	Rank pairs by harmonic distance
Random	✗	✗
Key Match	✓	✗
Harmonic Distance	✗	✓
Key Match + Harmonic Distance	✓	✓

To generate recommendations, I first calculated the percent difference in BPM between all song pair combinations and filtered out any pairs with differences greater than 15% to ensure that all recommendations are at least rhythmically compatible[2]. For models with Key Match, I filtered out any pairs that were not in the same key[3]. For models with the ranking feature, the top 25 pairs were selected based on smallest harmonic distance. For models without ranking, 25 pairs were randomly selected after applying filter(s).

In the evaluation of each recommendation, I attempted to create a mashup of the two songs with my DJ software and equipment. If I was able to create a harmonically agreeable mashup, I assigned a compatibility score of 1, else 0.

After scoring all recommendations, I compared the total and unique pairs of compatible songs across each model:

[2] The 15% threshold is based on my prior DJ experience; BPM adjustments greater than 15% distorts the rhythmic integrity of the song

[3] The BPM and key metadata were derived from the Spotify Web API using the Spotipy python library.

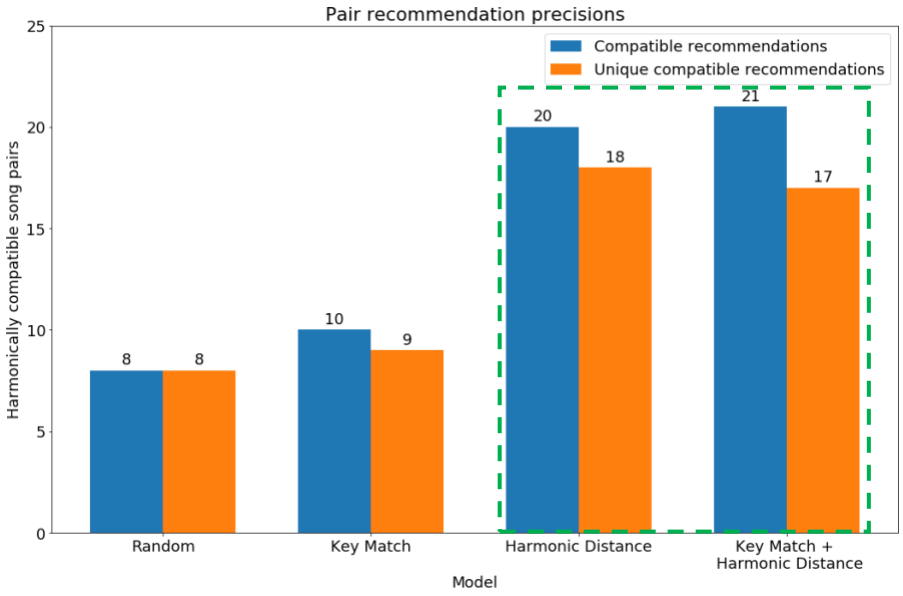

Figure 8: Harmonic Distance and Key Match + Harmonic Distance were most successful at identifying compatible pairs

As expected, Random performed the worst while the full model, Key Match + Harmonic Distance, performed exceptionally at identifying compatible pairs. Harmonic Distance by itself also performed well, generating nearly the same number of compatible pairs as the full model.

Unfortunately, Key Match showed little improvement over Random. While recommendations from Random may have had a higher number of harmonic connections by chance, it is more likely that Spotify's key detection algorithm generated too many inaccurate predictions. Out of 57 pairs that Spotify metadata predicted were in the same key, my DJ software only believed 25 of which were actually in the same key. While it was unclear which algorithm was more accurate, this highlights the fact that **key predictions algorithms tend to have varying degrees of success.**

It is important to note that Harmonic Distance and Key Match + Harmonic Distance combined found 39 harmonically compatible pairs out of 48 unique recommendations. Thus, **harmonic distance can not only uncover harmonically compatible pairs but also enhance the utilization of key detection algorithms**. In analyses of future music libraries, I will certainly evaluate recommendations from both models.

From the Key Match + Harmonic Distance recommendations, my favorite mashup that I made was with Pony by Ginuwine and Exhale by Zhu:

Pony x Exhale mashup

I was pleasantly surprised by not only their harmonic compatibility, but stylistic compatibility as well.

Reimagining my music library as a network

The potential of using harmonic distance for DJing really comes to life when the data is applied to network theory concepts. With the top 5% song pairs by harmonic distance, I used the NetworkX package to construct an undirected network with nodes as songs, edges as harmonic connections between songs, and edge weights as harmonic distances.

In addition, I calculated the **eigenvalue centrality score** of each node to evaluate the most important songs based on their connection to other songs with high number of harmonic connections.

With the Gephi software, I created an interactive visualization of this network:

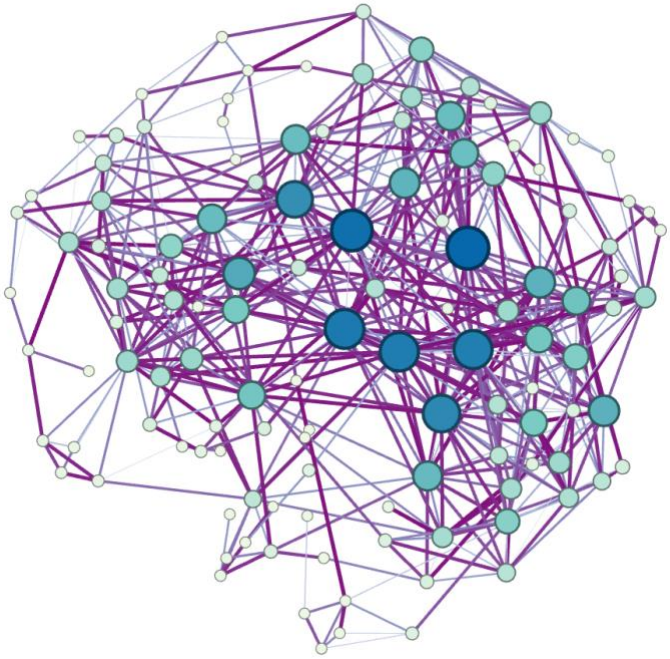

Figure 9: Network of 122 songs and 523 harmonically compatible pairs

I also formatted the visualization such that node size and shade were relative to the eigenvalue centrality score and edge thickness was relative to harmonic distance.

With this visualization, I can make more strategic choices with song selections during a DJ set, as I can evaluate harmonic neighbors of any given track in real-time and be cognizant of "versatile transition tracks", or tracks that can lead to many transition options.

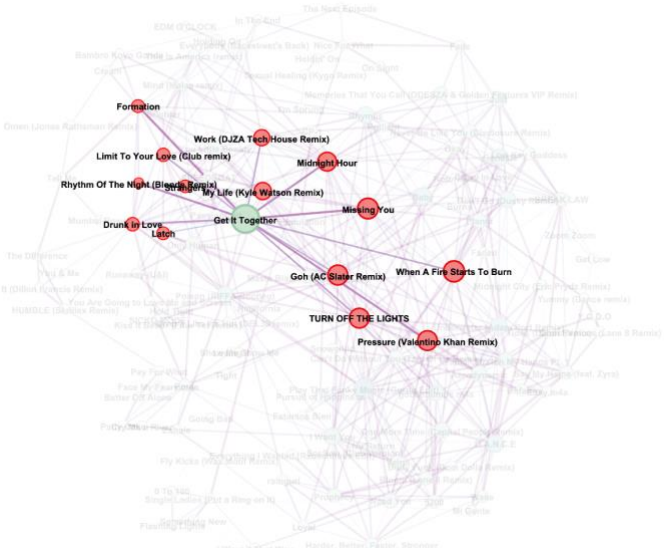

Figure 10: "Get It Together by Drake" is a versatile transition track, as it can smoothly transition to many other songs that have many transition options

The shortest path concept of network theory is also applicable in the context of DJing. Say I'm currently playing Formation by Beyoncé and want to transition to Say My Name by ODESZA, but the two songs are too harmonically dissimilar for an immediate transition. With **Dijkstra's Shortest Path First algorithm**, I can find the shortest weighted path to Say My Name, in which each transition would sound harmonically agreeable.

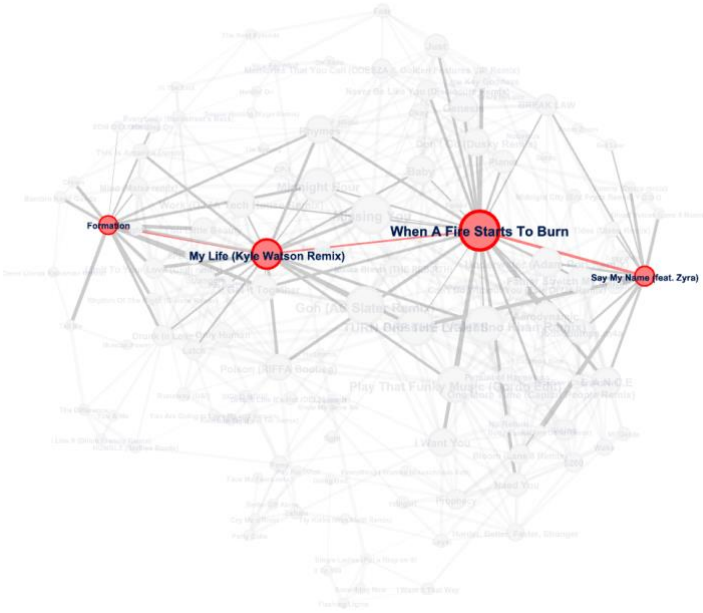

Figure 11: Shortest path from "Formation by Beyoncé" to "Say My Name by ODESZA"

Finally, **the shortest path concept can also assist with creating a tracklist for performances.** Say I have been tasked to perform a 1-hour set of ~20 songs. With my network, I can algorithmically construct a path of 20 unique songs that minimizes harmonic distance. To solidify my experiments into a creative piece of work, I ran this algorithm a few times, each time starting

with a random song. Out of the iterations, I picked the tracklist I found most interesting and created this DJ mix. Because of the harmonic compatibility throughout the tracklist, I was able to fully extend my mixing capabilities to create seamless and elaborate transitions between songs. Furthermore, mood, style, or genre were not taken into account when creating the tracklist, so there was a sense of randomness with the song selections that really contributed to the novelty of the set.

Figure 12: My DJ set with tracklist created using a shortest path algorithm, published on Mixcloud

Conclusion

With the lack of song analysis and recommendation capabilities in today's DJ software, I believe that these features can be key differentiators for next generation DJ platforms:

- A more robust metric for evaluating harmonic distances
- An interactive visualization for exploring harmonic connections
- Recommendations for song selection and tracklist planning

Throughout the development of these features, I have gained a much deeper understanding of my music library, uncovered novel mashup possibilities, and created my most technically advanced DJ set to date – accomplishments that any DJ, professional or enthusiast, would strive for in the advancement of their craft.

As there are plenty of future work for this analysis, I would love to continue refining my features and someday bake them into DJ software so that other DJs can incorporate data-driven solutions into their arsenal of DJ tools.

Future work

- Reproduce analysis with a completely different music library
- Evaluate computation time and model accuracy from reducing the sampling rate for chroma computation; it currently takes 1.3 hours to compute 2 GB worth of music at 22 samples per second
- Derive chroma distribution across different sections of a song (e.g., intro, verse, chorus) to make more granular comparisons
- Test alternative metrics for harmonic distance, e.g., a rank-based approach in which a score is generated based on the number of shared chromas between each song's top 5 most common chroma classes
- Build front-end interface to load music files, generate visualizations, and obtain recommendations

Resources

Repository: GitHub

Python libraries:

- LibROSA
- spotipy
- NetworkX

Network visualization: Gephi

DJ software: Algoriddim djay Pro 2

DJ equipment: AKAI MPK25, Numark Mixtrack Pro 3