A Probabilistic Approach to Content-Based Audio Analysis for Music Clustering and Search

Nader Al-Naji
Department of Computer Science
Princeton University

Abstract

Music recommendation and search based on textual metadata has become of large-scale commercial value. However, analysis of music based purely on audio content is an area of research still in its relative infancy. In this paper, we discuss, explore, implement, and augment state of the art methods for extracting significant features from songs, computing similarity between songs, and clustering songs. We then demonstrate the effectiveness of these methods by running them on a collection of music. We begin by detailing the methods used for extracting features from small chunks of audio using MFCC's, a method common in the speech recognition literature [1]. Segmenting a song and extracting the features from each segment yields a large number of feature vectors for each song. We then cluster these vectors and represent the song as a parameterized probability distribution over feature space returned by this clustering. The crucial bag of windows assumption made in doing this clustering of vectors will be discussed. Then, considering songs to be probability distributions over feature space, we define a similarity measure based on Kullback-Leibler divergence, a non-symmetric measure of the difference between two probability distributions [2]. We then explore the collection in the framework of information retrieval using this measure, as well as discuss an innovative new clustering technique we developed based on this measure. We emphasize that we implement all of the methods used for feature extraction and clustering ourselves and that these methods are run on full-length pieces of music.

Analysis using human-labeled textual metadata for songs has become a technique widely used in many industrial-strength music analysis systems. Such methods are appealing because they allow one to make full use of a large corpus of text-based information retrieval literature. However, textual labels can be very subjective, depending on who is doing the labeling, and such schemes inherently leave behind a lot of useful information by ignoring the signal itself. By analyzing the actual audio signal, so-called content-based approaches have the potential to pick up on what text-based approaches miss. This paper will discuss and build upon previous research by Perry Cook et al. [3] in the relatively new field of content-based audio analysis.

Our analysis is centered around the idea that songs can be represented as probability distributions over feature space. Using this idea, we then seek a similarity measure $dist(A, B)$ (not to be confused with a similarity *metric*) between two songs A and B, modeled as probability distributions, such that $dist(A, B) < dist(A, C)$ if and only if song A is perceptually more similar to B than to C. Of course, because the performance of such a similarity measure will rely heavily on the methods used to extract features, a large part of the paper will be devoted to outlining exactly how features are extracted from songs.

Establishing a similarity measure between songs allows us to evaluate our model using standard information retrieval metrics. Taking one song out of a collection of songs, we then "search" for the songs closest to that song based on our similarity measure. We then compare the results of this search to what songs would be most similar perceptually.

Finally, we present a novel method we developed to cluster music under this probabilistic framework, and show the results of clustering using this method on an actual collection of songs.

I. Feature Extraction

Computing content-based similarity between songs relies heavily on extracting relevant features from audio. Thus, we detail the exact steps taken to condense a piece of music into a large collection of feature vectors, to be used in further analysis. Mel-Frequency Cepstral Coefficients, or MFCC's, have been show by Logan [1] to be a reasonable way to extract perceptually relevant features from music as well as speech. The methods described here for completeness are thus the standard implementation of MFCC's described in [1] applied to a whole song.

Song Feature Extraction Steps

1. **Break song up into $50ms$ windows with a hop size of $20ms$.**
   The reasons for using these time frames are described below.
2. **For each window:**

- **Apply a Hamming window to samples**
  This amounts to smoothing the window with a "raised cosine" function to reduce edge effects when taking the FFT.
- **Take the Fast Fourier Transform of the window**
  This turns the song window into a vector of complex values, each corresponding to a frequency. If the song was sampled at $44.1kHz$ (a common sampling rate) and a window size of $50ms$ is used, then the maximum frequency that will have a coefficient in the resulting vector will be $22kHz$, just at the limit of what humans can hear. The windows overlap to compensate for the fact that we apply a Hamming window to eliminate edge effects.
- **Drop the phase and log the amplitudes**
  We drop the phase because, as mentioned in [1], phase values have been shown to be perceptually insignificant. We log the amplitudes because humans perceive volume logarithmically, hence the Decibel scale.
- **Bin the frequencies according to the Mel Scale**
  Humans perceive increases in pitch non-linearly with increases in frequency. For example, an increase in one octave from middle C to high C corresponds to a *doubling* of frequency. The Mel Scale maps frequencies to Mel values such that linear increases in Mel values represent linear increases in perceived pitch. Thus, at this step we take a set of mel-spaced bins and populate them with the average over their corresponding frequency ranges.
- **Take the Discrete Cosine Transform of the resulting Mel spectrum**
  This final step is done as a form of principal component analysis to compress the Mel vector into a lower-dimensional vector with minimal loss of information. This works because the DCT concentrates most of the information of a given signal into a few low-frequency components, approaching the KL Transform which is optimal in terms of loss of information [1].
- **Take the first 20 coefficients of the DCT to be a 20-dimensional feature vector**
  The other DCT coefficients can be dropped with minimal loss of information since, as mentioned, most of the Mel vector's information will be concentrated into the lower-frequency components after applying the DCT. We discard the 0^{th} component of the DCT because it corresponds to volume, to which we want to be invariant.

For each song we end up with a large number of 20-dimensional feature vectors, where each vector corresponds to a particular window of time. Figure I illustrates feature extraction for a single window.

II. Songs as Probability Distributions

After feature extraction, every song has a large number of 20-dimensional feature vectors (on the order of thousands) corresponding to it, with each vector representing a specific window within the song. Considering songs simply as a list of ordered feature vectors makes them difficult to deal with as well as difficult to store. In fact, this representation under a naive implementation could require more space to store than the actual song. Thus, we make some careful simplifying assumptions in order to be able to condense this large number of feature vectors into a more meaningful representation that is also easier to deal with.

The first simplifying assumption we make is that feature vectors within a song can be reordered with only a small loss in meaningful information. Recall that one of our main goals is to develop a similarity measure with which to compare songs, so whether or not songs remain similar upon reordering of their windows is

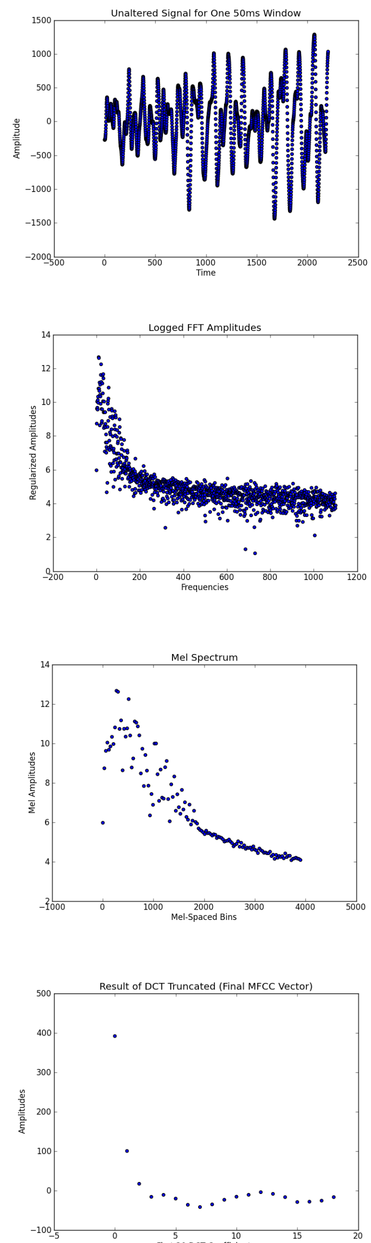


Figure I: An illustration of the feature extraction process.

crucial. This assumption certainly causes our model to neglect some useful information, namely how a song evolves over time. However, it is not unreasonable to think that two similar songs, which presumably contain many feature vectors with features unique to their genre in common, could still be deemed similar even if order is disregarded. For example, one can still tell that two songs by Metallica are similar, even if they only hear random chunks out of order, since random chunks from both of these songs will contain features unique to heavy metal music with high probability. This "bag of windows" assumption thus appears to be justified in the sense that similarity between songs, while not completely unscathed, is preserved to a sufficient extent.

Next, assuming songs can be thought of as bags of windows, we consider a generative probability model for songs, which allows us to represent each song using only the parameters for its probability distribution. This is useful since this parameterized distribution, by virtue of being a mathematical object, is much easier to deal with (and store) than thousands of unordered feature vectors. In particular, we assume that these unordered vectors are generated by an underlying Mixtures of Gaussians distribution.

The reasoning behind the Mixtures of Gaussians assumption is as follows. A song typically consists of multiple sections: an intro, a chorus, etc... and each of these sections will have feature vectors corresponding to it. Further, vectors from a particular section will likely contain features unique to that section. For example, vectors corresponding to the intro of a Metallica song might all contain features unique to guitar solos, while vectors from the chorus might all contain features unique to the lead singer's voice. We thus consider songs as generated by an underlying Mixtures of Gaussians distribution where each mixture corresponds to a kind of part of a song. Songs themselves, then, under the bag of windows assumption, can be seen as generated by choosing a mixture from a particular song's distribution according to its prior, sampling a vector from that mixture, and repeating this process. Figure II illustrates the generative model:

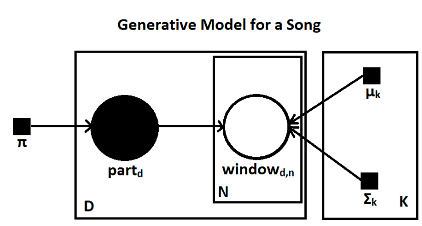


Figure II: The generative model for a song.

Having justified both the bag of windows assumption and the generative model, we compute the latent parameters for each song's distribution using standard Mixtures of Gaussians Expectation Maximization. Computing the cluster priors, means, and covariances, then, gives us a full parameterization of each song and allows us to consider songs as probability distributions over feature space, instead of simply as bags of unordered feature vectors. The former characterization is much easier to deal with and opens up a range of possibilities, affording us the flexibility of comparing songs by comparing their probability distributions.

III. A Probabilistic Similarity Measure

We compute a Mixtures of Gaussians probability distribution for each song using EM as discussed above. Now, having extracted a parameterized probability distribution over feature space for each song, we discuss several ways to compare these distributions and, by proxy, compare the songs.

We first naturally gravitated towards a similarity measure very much inspired by Euclidean distance. We define $dist(A, B)$ as the square of the difference between the two distributions integrated over every point in feature space. To be clear, taking f and g to be probability distributions over n-dimensional feature space corresponding to songs A and B respectively, we would want to compute $dist(A, B)$ as:

$$\int_{-\infty}^{\infty} ... \int_{-\infty}^{\infty} (f(x_1, ..., x_n) - g(x_1, ..., x_n))^2 \mathrm{d}x_1 ... \mathrm{d}x_n. \tag{1}$$

In theory this provides us with the most intuitive measure of similarity or distance between arbitrary distributions. This similarity measure also has the added benefit of being a metric since $dist(A, A) = 0$, $dist(A, B) = dist(B, A)$, and it obeys the triangle inequality. However, in practice, even with only twenty-dimensional features, $n = 20$, and using MoG's as our distributions for f and g, this sum is difficult to compute using numerical methods and doesn't appear to have a closed form (though it could).

Since this distance metric seems to be difficult to compute when dealing with Mixtures of Gaussians, we resort to approximation. We consider the following unbiased estimator of (1) that consists of sampling points from each distribution and computing the normalized squared difference between the two distributions at each point. In the following equation, f and g are again probability distributions over n-dimensional feature space, and x_i and y_i are n-dimensional points sampled from f and g respectively, with N being the total number of samples:

$$\frac{1}{N}(\sum_{x_i} \frac{(f(x_i) - g(x_i))^2}{f(x_i)} + \sum_{y_i} \frac{(f(y_i) - g(y_i))^2}{g(y_i)}) \tag{2}$$

Given a large enough number of samples, this would theoretically approach the metric described above. However, in practice this metric is not only very slow to compute under our implementation (using $N = 2000$), but also the results of using this similarity metric to compare songs are sub-par compared to the results of using the method we describe next. It is possible that even though this estimator is unbiased, its variance is too large to make using only 2000 points in the sample reasonable. In future research we want to explore this metric in further detail using a larger number of samples, after speeding up our implementation.

The similarity approximation actually used was a simple Monte Carlo approximation to Kullback-Leibler divergence, a non-symmetric measure of the difference between two probability distributions widely used in many aspects of speech and image recognition. We approximate the exact Kullback-Leibler measure for continuous densities f and g:

$$D(f||g) = \int_{-\infty}^{\infty} f(x) \ln \frac{f(x)}{g(x)} \mathrm{d}x \tag{3}$$

using Monte Carlo sampling as suggested in [2]. Taking f and g to again be the densities of two MoG distributions corresponding to songs, we sample N iid points, $\{x_i\}_{i=1}^{N}$, from f and compute:

$$D_{MC}(f||g) = \frac{1}{N} \sum_{i=1}^{N} \log \frac{f(x_i)}{g(x_i)}. \tag{4}$$

This additionally turns out to be an unbiased estimator of KL divergence since:

$$E_f[\frac{1}{N} \sum_{i=1}^{N} \log \frac{f(x_i)}{g(x_i)}] = \int_{-\infty}^{\infty} f(x) \ln \frac{f(x)}{g(x)} \mathrm{d}x = D(F||G). \tag{5}$$

Further, since we are simply taking the average over iid samples, the Central Limit Theorem applies and we see that $Var[D_{MC}] =$

$\frac{1}{N} Var_f[log f/g]$, meaning the variance of our estimator decreases as N increases, as discussed in [2]. This measure also captures the intuition that songs should be more likely to generate points that came from songs similar to themselves.

We use this approximation without the $f(x_i)$ term when doing music retrieval, since f doesn't change when searching for a specific song, to give us the following expression:

$$-\frac{1}{N}\sum_{i=1}^{N}\log g(x_i). \quad (6)$$

We do the same when using KL divergence to cluster songs.

IV. Probabilistic Music Retrieval

For this section, we define $dist(A, B) = -\frac{1}{N}\sum_{i=1}^{N}\log g(x_i)$, where g is the probability distribution corresponding to song B, and the x_i are points sampled from f, the distribution corresponding to song A. We test this measure by doing music retrieval searches, returning a list of the songs in our collection ordered by their distance from the query song, using a library of approximately 160 songs. We then evaluate the relevance of the results using standard information retrieval metrics. We acknowledge that the library of songs we use is relatively small, containing songs from only about thirty artists, but we emphasize the accuracy of the results even on this small dataset, as well as the scalability of our retrieval computation (It is "embarassingly parallelizable.").

Figure III at the end of this paper illustrates the results of queries for selected songs using our measure and our 160-song library. The results returned when querying for a song A are ordered by their similarity to A based on the measure discussed.

Note how the top ten songs returned for a given query always consist mostly of songs with the same artist as the query song, and that, overall, songs returned that are not by the same artist are songs that are perceptually similar to the query. For example, the top ten results in a query for a Steely Dan song are all Steely Dan, and the eleventh result is a song by Eric Clapton, another prolific rock artist. This pattern is prevalent in all of the searches performed, even across different genres, showing that perceptually significant features from all genres are being captured by our system. Further, songs from similar genres and bands are confused in a way consistent with how humans would confuse them. For example, in a query for "I Want It That Way" by the Backstreet Boys, "Tearin' Up My Heart", a song by 'N Sync, a very similar boyband, is returned above "The Perfect Fan", another song by the Backstreet Boys.

We also include an evaluation of how this system performs using standard information retrieval metrics, shown in the figure above. We compute the reciprocal rank, precision at rank 15, discounted cumulative gain at Rank 15, and the average precision at each point a song in the same genre appears for each of the queries above. As can clearly be seen, this system's precision is very high with almost 90% of songs returned in the same genre. Further, the classical music query reaches the maximum DCG value of 194.7, with the average DCG not too far below that.

V. A Technique for Clustering Songs

Clustering data provides insight into what features of the data are being captured by a particular model. Hence, we now try clustering our data using the full (symmetrized) KL divergence approximation as our distance measure:

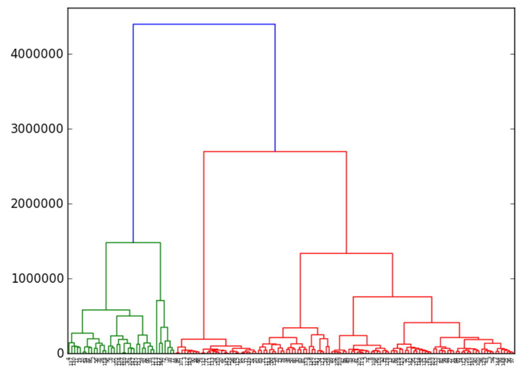
$$dist(A, B) = \frac{1}{N}\sum_{i=1}^{N}\log\frac{f(x_i)}{g(x_i)} + \frac{1}{N}\sum_{i=1}^{N}\log\frac{g(x_i)}{f(x_i)} \quad (7)$$

where f and g are the MoG distributions for songs A and B respectively. Note that we symmetrize the measure so that $dist(A, B) = dist(B, A)$.

Since we have a clearly defined similarity measure, the first thing we try is heirarchical clustering. We perform hierarchical agglomerative clustering, using the median similarity criterion, on the songs using a similarity matrix with each value computed according to the approximation above. Figure IV is the dendrogram obtained. Looking at this dendrogram, we see noticeable decreases in height for about ten to fifteen splits. Thus, we generate a flat clustering using $K = 15$ based on this hierarchy. Some of the more significant clusters are shown in figure V at the end of this paper.

From the clusters shown in figure V, we can clearly see that hierarchical clustering is creating groups based on perceptual similarities. It finds and groups all of the classical music

Figure IV: Dendrogram found using median criterion.



in the collection, as well as all of the R&B. Classic rock and pop (namely boy band songs) are grouped as well. What isn't shown is a small set of singleton clusters and one large absorbing cluster, containing a hodge-podge of songs from different genres, that was too large to include. In an effort to improve these flaws, we propose an alternative clustering method based on Expectation Maximization.

The K-Means algorithm is a widely accepted way to cluster related pieces of data. However, a direct application of the K-Means algorithm doesn't quite make sense in our framework since it is not clear what the "mean" of a collection of MoG distributions is. However, if we take the bag of windows assumption seriously, that is, assuming songs can really be seen as unordered bags of windows represented by feature vectors, then we can come up with a natural Expectation Maximization algorithm that emulates K-Means.

The fundamental idea is this: since clusters contain songs, and songs are bags of windows, we can consider a cluster to be a bag of windows as well (essentially a bag of bags of windows). A collection of windows in a cluster, then, can be seen as generated by choosing one of its MoG components uniformly at random, sampling from that component, and then repeating this process. So, we define a cluster's distribution to be the uniform mixture of its MoG components. Further, by considering a cluster to be a probability distribution over feature space, we can then approximate the similarity between a song and a cluster using KL divergence.

We now describe an EM algorithm to cluster songs where K is the number of initial clusters, I is the number of iterations, and N is the number of points to sample for each song:

Clustering Algorithm(K, I, N)**:**
  Presample N **points for each song**
  Precompute $dist(A, B)$ **for all songs**
  Assign each song to a cluster randomly
  for $i \leftarrow 1$ **to** I
    for each song A
      for each cluster k
        (1) **Compute dist(A, k)**
      (2) **Assign song** A **to closest cluster**
return clustering

In (1) above, we compute the asymmetric approximation to KL Divergence. That is, we compute:

$$dist(A, k) = -\frac{1}{N} \sum_{i=1}^{N} \log g(x_i) \qquad (8)$$

where g is the distribution for cluster k and the x_i are sampled from song A's distribution. This is essentially the probability that cluster k generated song A, and it can be computed very quickly using the log-sum-exp trick, to sum log likelihoods over the MoG components of each cluster, and the precomputed values for song similarity. Here we again leave out the $f(x_i)$ term since f doesn't change when comparing different clusters to the same song. In (2) we assign each song to the cluster with the lowest distance value; that is, we assign each song to the cluster that was most likely to generate it.

Since each cluster is essentially a uniform mixture of the songs contained within it, it appears reasonable to consider this mixture to be the "mean" distribution for the cluster of songs. Considering the cluster distribution in this way, it should be clear that this approach is nothing more than a generalization of the K-Means algorithm to handle probability distributions as input data.

Figure VI at the end of this paper shows a subset of the results of running this algorithm on our collection of songs with $K = 10$. It is worth noting that the clusters generated by this algorithm are significantly better than the clusters generated using hierarchical clustering. Here, not only are there no singleton clusters, but every cluster clearly represents a genre and, a lot of the time, a specific artist. While this method is more impacted by initialization (a different clustering is obtained depending on the initial clusters used to seed the algorithm), by starting each cluster off with a single random song from the collection, we found the results almost always converge to the same clustering.

VI. Future Work

While we covered a lot of ground in this paper, we still feel as though there is much work to be done. For one, the feature extraction mechanism we implemented was only the bare minimum of what we actually wanted to do. While the MFCC feature extraction technique appeared to work very well, in the future we want to adapt MFCC's, which were developed specifically for speech recognition, to make them better at representing musical data. For example, possibly because speech recognition doesn't rely heavily on pitch, MFCC's appear not to take changes in perceived pitch into account, even though humans perceive some pitches much more strongly than others. We also wanted to experiment with MFCC flux, that is the difference between MFCC vectors at each point, to see how adding first-order derivative vectors affects our results. We wanted to experiment with using distributions other than Mixtures of Gaussians to cluster the MFCC vectors and see how this affects the similarity measure. We used mixtures of five Gaussians to cluster our vectors; however, it is not terribly clear that this number is optimal and being able to figure out the best number of mixtures each song should be represented with would also be an important goal. KL Divergence worked very well as a similarity measure; however, the Monte Carlo approach to estimating it forces one to make a trade-off between computation time (number of samples) and accuracy. In the future we want to experiment with more efficient and more accurate approximations to KL Divergence, as well as try completely new measures (and metrics) to see how they perform. Finally, while the bag of windows assumption simplifies a lot of things, in the future we would like to experiment with temporal models to seriously take into account how songs evolve over time, and include this information in our analysis.

VI.References

[1] B. Logan, *Mel Frequency Cepstral Coefficients for Music Modelling*, Cambridge Research Laboratory.

[2] J. Hershey and P. Olsen, *Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models*, IBM T.J. Watson Center.

[3] G. Tzanetakis, *Musical Genre Classification of Audio Signals*, IEEE Transactions on Speech and Audio Processing Vol. 10 No. 5, 2002.

[4] D. Blei, M. Hoffman, and P. Cook, *Content-Based Music Similarity Computation Using the Hierarchical Dirichlet Process*.

[5] J. Aucouturier and F. Pachet, *Music Similarity Measures: What's the Use?*.

Figure III: Music Retrieval Using Approximation to KL Divergence (Without Normalization)

a) Classical Music (Rachmaninov)

```
Song being searched for: 12_-_Prelude_in_C_major,_Op.32_No.1

Songs closest to this song:
1: 12_-_Prelude_in_C_major,_Op.32_No.1: -68591.204978
2: 24_-_Prelude_in_D_flat_major,_Op.32_No.13: -73033.827607
3: 15_-_Prelude_in_E_minor,_Op.32_No.4: -73506.562205
4: 06_-_Prelude_in_G_minor,_Op.23_No.5: -73638.926928
5: 14_-_Prelude_in_E_major,_Op.32_No.3: -74374.963012
6: 07_-_Prelude_in_E_flat_major,_Op.23_No.6: -74412.354153
7: 19_-_Prelude_in_A_minor,_Op.32_No.8: -74686.723507
8: 04_-_Prelude_in_D_minor,_Op.23_No.3: -74721.104207
9: 20_-_Prelude_in_A_major,_Op.32_No.9: -74985.791112
10: 08_-_Prelude_in_C_minor,_Op.23_No.7: -75102.159756
11: 01_-_Prelude_in_C_sharp_minor,_Op.3_No.2: -75110.30196
12: Vladimir_Ashkenazy_-_Prelude_in_C_minor,_Op.23_No.7: -75367.916979
13: 21_-_Prelude_in_B_minor,_Op.32_No.10: -75547.950154
14: 23_-_Prelude_in_G_sharp_minor,_Op.32_No.12: -75665.925955
15: Prelude2: -76398.211258
```

b) Classic Rock (Steely Dan)

```
Song being searched for: 05_Steely_Dan_-_Home_At_Last

Songs closest to this song:
1: 05_Steely_Dan_-_Home_At_Last: -78377.776749
2: Steely_Dan_-_Home_At_Last: -79040.130178
3: SteelyDan1: -81844.197219
4: 01_Steely_Dan_-_Black_Cow: -81908.941216
5: 02_Steely_Dan_-_Aja: -82005.614359
6: 07_Steely_Dan_-_Josie: -82102.040561
7: 06_Steely_Dan_-_I_Got_The_News: -82716.637673
8: SteelyDan2: -82755.512876
9: 04_Steely_Dan_-_Peg: -82781.06428
10: 03_Steely_Dan_-_Deacon_Blues: -83755.671851
11: 07-Eric_Clapton-Layla: -84162.728819
12: 06-Eric_Clapton-Nobody_Knows_You_When_You're_Down_&_Out: -84598.616681
13: 04_-_Yes_-_Roundabout: -84967.870495
14: 07_-_Dr._John_-_Right_Place_Wrong_Time: -84979.619995
15: 13-Eric_Clapton-Old_Love: -85097.85801
```

c) Classic Rock (Eric Clapton)

```
Song being searched for: 01-Eric_Clapton-Signe

Songs closest to this song:
1: 01-Eric_Clapton-Signe: -75922.412186
2: 13-Eric_Clapton-Old_Love: -80044.87701
3: 05-Eric_Clapton-Lonely_Stranger: -80456.36072
4: 06-Eric_Clapton-Nobody_Knows_You_When_You're_Down_&_Out: -81552.032228
5: 02-Eric_Clapton-Before_You_Accuse_Me: -81905.231136
6: 07-Eric_Clapton-Layla: -82309.644601
7: 03-Eric_Clapton-Hey_Hey: -82578.647622
8: 12-Eric_Clapton-Malted_Milk: -83142.276544
9: 08-Eric_Clapton-Running_On_Faith: -83332.194951
10: 04-Eric_Clapton-Tears_In_Heaven: -83920.588263
11: 10-Eric_Clapton-Alberta: -84274.192797
12: 04_-_It's_Gotta_Be_You: -85153.731215
13: SteelyDan2: -85573.667245
14: 11-Eric_Clapton-San_Francisco_Bay_Blues: -85626.624126
15: 06_Steely_Dan_-_I_Got_The_News: -85732.5736
```

d) R&B (T-Pain)

```
Song being searched for: 09_Mix'd_Girl

Songs closest to this song:
1: 09_Mix'd_Girl: -75667.434432
2: 05_5_O'clock_(feat._Lily_Allen_and_Wiz_Khalifa): -79630.147104
3: 12_When_I_Come_Home: -79669.205613
4: 14_Turn_All_the_Lights_On_(feat._Ne-Yo): -79801.020262
5: 02_-_I_Just_Wanna_Be_With_You: -79856.656869
6: 13_Best_Love_Song_(feat._Chris_Brown): -79864.081645
7: 15_Center_of_the_Stage_(feat._R._Kelly_and_Bei_Maejor): -79870.480037
8: 13_-_Giddy_Up: -79888.555924
9: 04_Default_Picture: -80048.809021
10: 16_Regular_Girl: -80106.410729
11: 06_Sho-Time_(Pleasure_Thang): -80129.588706
12: 01_-_Tearin'_Up_My_Heart: -80602.286814
13: 02_-_Circus_[Main_Version]: -80836.814016
14: 08_-_I_Want_You_Back: -80854.505095
15: 08_Look_at_Her_Go_(feat._Chris_Brown): -80899.829419
```

e) Boy Bands (Backstreet Boys)

```
Song being searched for: 02_-_I_Want_It_That_Way

Songs closest to this song:
1: 02_-_I_Want_It_That_Way: -76024.091635
2: 03_-_Show_Me_The_Meaning_Of_Being_Lonely: -78433.01848
3: 07_-_Don't_Wanna_Lose_You_Now: -78754.466319
4: 06_-_Don't_Want_You_Back: -78808.847484
5: 08_-_The_One: -78965.104408
6: 09_-_Back_To_Your_Heart: -79464.563236
7: 05_5_O'clock_(feat._Lily_Allen_and_Wiz_Khalifa): -79802.186367
8: 01_-_Tearin'_Up_My_Heart: -79937.715927
9: 12_-_The_Perfect_Fan: -80108.859516
10: 04_-_For_The_Girl_Who_Has_Everything: -80276.630152
11: 13_-_Giddy_Up: -80319.443261
12: 08_-_I_Want_You_Back: -80398.577942
13: 09_-_Everything_I_Own: -80518.221767
14: 04_-_It's_Gotta_Be_You: -80549.390305
15: 06_-_You_Got_It: -80550.72322
```

f) Rap (Eminem)

```
Song being searched for: 07_-_Soldier

Songs closest to this song:
1: 07_-_Soldier: -74971.368866
2: 14_-_Hailie's_Song: -77266.52674
3: 05_-_Square_Dance: -77381.720516
4: 09_-_Drips_(feat._Obie_Trice): -77501.958539
5: 13_-_Superman_(feat._Dina_Rae): -77623.34898
6: 04_-_Cleanin_Out_My_Closet: -77652.86317
7: 17_-_Say_What_You_Say_(feat._Dr._Dre): -77823.99845
8: 03_-_Business: -77893.344758
9: 08_-_Say_Goodbye_Hollywood: -77901.609986
10: 10_-_Without_Me: -78133.000954
11: 11_-_In_Pieces: -78158.332246
12: 16_-_When_The_Music_Stops_(feat._D-12): -78162.271219
13: 05_-_God_Must_Have_Spent_A_Little_More_Time_On_You: -78307.462052
14: 02_-_White_America: -78330.330947
15: 19_-_My_Dad's_Gone_Crazy_(feat._Hailie_Jade): -78339.535255
```

g) Pop (Britney Spears)

```
Song being searched for: 06_-_If_U_Seek_Amy_[Main_Version]

Songs closest to this song:
1: 06_-_If_U_Seek_Amy_[Main_Version]: -75929.337897
2: 02_-_Circus_[Main_Version]: -79277.894309
3: 11_-_Lace_And_Leather_[Main_Version]: -79455.920435
4: 13_-_Radar_[Main_Version]: -79711.211247
5: 05_-_Shattered_Glass_[Main_Version]: -79711.83541
6: 19_-_My_Dad's_Gone_Crazy_(feat._Hailie_Jade): -79755.683138
7: 09_-_Drips_(feat._Obie_Trice): -79800.468216
8: 09_-_Mmm_Papi_[Main_Version]: -79891.923333
9: 06_-_Don't_Want_You_Back: -79933.860554
10: 02_-_White_America: -80188.871534
11: 13_-_Giddy_Up: -80210.094828
12: 10_-_Without_Me: -80229.60496
13: 15_-_Phonography_[Main_Version]: -80428.051516
14: 17_-_Say_What_You_Say_(feat._Dr._Dre): -80493.740907
15: 03_-_Business: -80604.567461
```

h) Alternative (Linkin Park)

```
Song being searched for: 02_-_Don't_Stay

Songs closest to this song:
1: 02_-_Don't_Stay: -66422.870663
2: 04_-_Bleed_It_Out: -68579.848654
3: 07_-_Faint: -68755.398423
4: 08_-_Figure.09: -68792.602796
5: 05_-_Hit_The_Floor: -68884.471254
6: 04_-_Lying_From_You: -69091.109983
7: 01_-_Larger_Than_Life: -69263.897448
8: 13_-_Numb: -69367.477098
9: 08_-_No_More_Sorrow: -69395.665779
10: 18_-_'Till_I_Collapse_(feat._Nate_Dogg): -69501.140352
11: 05_-_Square_Dance: -69752.969882
12: 04_-_It's_Gotta_Be_You: -69880.247539
13: 19_-_My_Dad's_Gone_Crazy_(feat._Hailie_Jade): -70029.777173
14: 14_Turn_All_the_Lights_On_(feat._Ne-Yo): -70144.337005
15: 06_-_You_Got_It: -70146.37201
```

	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	Average
Reciprocal Rank (1/rank of highest song in same genre)	1	1	1	1	1	1	1	1	1
Precision at Rank 15 (# songs in same genre / # songs retreived)	1	.93	.93	.8	1	.93	.73	.73	.88
Discounted Cum. Gain (same artist = 10, same genre = 8, other = 0)	194.7	180.8	182.3	194.7	186.0	152.6	154.2	176.3	176.3
Average Precision At Each Point a Song in Same Genre Appears	1	.93	.92	.75	1	.92	.58	.69	.85

Figure V: Clusters found using hierarchical clustering with median similarity criterion.

Classical Music

```
Cluster 8:
Prelude3
03_-_Prelude_in_B_flat_major,_Op_23_No.2
14_-_Prelude_in_E_major,_Op.32_No.3
22_-_Prelude_in_B_major,_Op.32_No.11
Prelude1
21_-_Prelude_in_B_minor,_Op.32_No.10
Vladimir_Ashkenazy_-_Prelude_in_C_minor,_Op.23_No.7
08_-_Prelude_in_C_minor,_Op.23_No.7
06_-_Prelude_in_G_minor,_Op.23_No.5
24_-_Prelude_in_D_flat_major,_Op.32_No.13
12_-_Prelude_in_C_major,_Op.32_No.1
16_-_Prelude_in_G_major,_Op.32_No.5
05_-_Prelude_in_D_major,_Op.23_No.4
13_-_Prelude_in_B_flat_minor,_Op.32_No.2
04_-_Prelude_in_D_minor,_Op.23_No.3
15_-_Prelude_in_E_minor,_Op.32_No.4
01_-_Prelude_in_C_sharp_minor,_Op.3_No.2
Prelude2
10_-_Prelude_in_E_flat_minor,_Op.23_No.9
09_-_Prelude_in_A_flat_major,_Op.23_No.8
07_-_Prelude_in_E_flat_major,_Op.23_No.6
20_-_Prelude_in_A_major,_Op.32_No.9
23_-_Prelude_in_G_sharp_minor,_Op.32_No.12
17_-_Prelude_in_F_minor,_Op.32_No.6
11_-_Prelude_in_G_flat_major,_Op.23_No.10
02_-_Prelude_in_F_sharp_minor,_Op._23_No.1
Vladimir_Ashkenazy_-_Prelude_in_F_major,_Op.32_No.7
18_-_Prelude_in_F_major,_Op.32_No.7
19_-_Prelude_in_A_minor,_Op.32_No.8
```

R&B

```
Cluster 12:
04_Default_Picture
09_Mix'd_Girl
12_When_I_Come_Home
06_-_Deep_Purple_-_Smoke_On_The_Water
13_Best_Love_Song_(feat._Chris_Brown)
10_I_Don't_Give_a_Fuck
14_Turn_All_the_Lights_On_(feat._Ne-Yo)
08_Look_at_Her_Go_(feat._Chris_Brown)
15_Center_of_the_Stage_(feat._R._Kelly_and_Bei_Maejor)
16_Regular_Girl
11_Drowning_Again_(feat._One_Chance)
13_-_Superman_(feat._Dina_Rae)
10_-_Without_Me
02_-_I_Just_Wanna_Be_With_You
07_Rock_Bottom
05_5_O'clock_(feat._Lily_Allen_and_Wiz_Khalifa)
06 Sho-Time (Pleasure Thang)
```

Rock

```
Cluster 10:
09-Eric_Clapton-Walkin'_Blues
03-Eric_Clapton-Hey_Hey
02-Eric_Clapton-Before_You_Accuse_Me
12-Eric_Clapton-Malted_Milk
11-Eric_Clapton-San_Francisco_Bay_Blues
10-Eric_Clapton-Alberta
07-Eric_Clapton-Layla
06-Eric_Clapton-Nobody_Knows_You_When_You're_Down_&_Out
04-Eric_Clapton-Tears_In_Heaven
05-Eric_Clapton-Lonely_Stranger
14-Eric_Clapton-Rollin'_&_Tumblin'
08-Eric_Clapton-Running_On_Faith
01-Eric_Clapton-Signe
13-Eric_Clapton-Old_Love
04_Steely_Dan_-_Peg
10_-_The_Bellamy_Brothers_-_Let_Your_Love_Flow
05_Steely_Dan_-_Home_At_Last
Steely_Dan_-_Home_At_Last
SteelyDan1
01_Steely_Dan_-_Black_Cow

Cluster 6:
03_Steely_Dan_-_Deacon_Blues
02_Steely_Dan_-_Aja
SteelyDan2
06_Steely_Dan_-_I_Got_The_News
07_Steely_Dan_-_Josie

Cluster 8:
02_-_Bad_Company_-_Feel_Like_Makin'_Love
08_-_Lynyrd_Skynrd_-_Sweet_Home_Alabama
01_-_Chicago_-_Saturday_In_The_Park
09_-_America_-_Sister_Golden_Hair
```

Pop

```
Cluster 2:
07_-_Hands_Held_High
03_-_Show_Me_The_Meaning_Of_Being_Lonely
07_-_Don't_Wanna_Lose_You_Now
12_-_The_Perfect_Fan
04_-_For_The_Girl_Who_Has_Everything
12_-_The_Little_Things_Give_You_Away
09_-_Back_To_Your_Heart
05_-_I_Need_You_Tonight

Cluster 14:
10_-_Spanish_Eyes
11_-_No_One_Else_Comes_Close
```

Figure VI: Clusters found using algorithm described.

Alternative (Linkin' Park)

```
Cluster 6:
09_-_Valentine's_Day
05_-_Hit_The_Floor
04_-_Bleed_It_Out
08_-_Figure.09
12_-_Session
04_-_Lying_From_You
03_-_Leave_Out_All_The_Rest
02_-_Don't_Stay
06_-_Easier_To_Run
06_-_What_I've_Done
18_-_'Till_I_Collapse_(feat._Nate_Dogg)
03_-_Somewhere_I_Belong
13_-_Numb
07_-_Faint
05_-_Square_Dance
```

Classic Rock

```
Cluster 4:
07_-_Dr._John_-_Right_Place_Wrong_Time
Steely_Dan_-_Home_At_Last
08_-_Lynyrd_Skynrd_-_Sweet_Home_Alabama
05_Steely_Dan_-_Home_At_Last
01_-_Chicago_-_Saturday_In_The_Park
09_-_America_-_Sister_Golden_Hair
10_-_In_Between
03_Steely_Dan_-_Deacon_Blues
10_-_The_Bellamy_Brothers_-_Let_Your_Love_Flow
07_Steely_Dan_-_Josie
SteelyDan1
01_Steely_Dan_-_Black_Cow
04_-_Yes_-_Roundabout
11_-_Warren_Zevon_-_Poor_Poor_Pitiful_Me
04_Steely_Dan_-_Peg
```

Classical

```
Cluster 3:
12_-_Prelude_in_C_major,_Op.32_No.1
06_-_Prelude_in_G_minor,_Op.23_No.5
05_-_Prelude_in_D_major,_Op.23_No.4
01_-_Prelude_in_C_sharp_minor,_Op.3_No.2
07_-_Prelude_in_E_flat_major,_Op.23_No.6
21_-_Prelude_in_B_minor,_Op.32_No.10
03_-_Prelude_in_B_flat_major,_Op_23_No.2
02_-_Prelude_in_F_sharp_minor,_Op._23_No.1
16_-_Prelude_in_G_major,_Op.32_No.5
Vladimir_Ashkenazy_-_Prelude_in_F_major,_Op.32_No.7
18_-_Prelude_in_F_major,_Op.32_No.7
22_-_Prelude_in_B_major,_Op.32_No.11
09_-_Prelude_in_A_flat_major,_Op.23_No.8
08_-_Prelude_in_C_minor,_Op.23_No.7
Prelude3
24_-_Prelude_in_D_flat_major,_Op.32_No.13
14_-_Prelude_in_E_major,_Op.32_No.3
10_-_Prelude_in_E_flat_minor,_Op.23_No.9
19_-_Prelude_in_A_minor,_Op.32_No.8
04_-_Prelude_in_D_minor,_Op.23_No.3
Prelude1
17_-_Prelude_in_F_minor,_Op.32_No.6
11_-_Prelude_in_G_flat_major,_Op.23_No.10
Vladimir_Ashkenazy_-_Prelude_in_C_minor,_Op.23_No.7
23_-_Prelude_in_G_sharp_minor,_Op.32_No.12
13_-_Prelude_in_B_flat_minor,_Op.32_No.2
Prelude2
15_-_Prelude_in_E_minor,_Op.32_No.4
20_-_Prelude_in_A_major,_Op.32_No.9
```

Rock (Clapton)

```
Cluster 5:
14-Eric_Clapton-Rollin'_&_Tumblin'
04-Eric_Clapton-Tears_In_Heaven
11-Eric_Clapton-San_Francisco_Bay_Blues
10-Eric_Clapton-Alberta
01-Eric_Clapton-Signe
05-Eric_Clapton-Lonely_Stranger
05_-_God_Must_Have_Spent_A_Little_More_Time_On_You
08-Eric_Clapton-Running_On_Faith
07_-_Unusual_You_[Main_Version]
11_Drowning_Again_(feat._One_Chance)
07-Eric_Clapton-Layla
12_-_The_Doobie_Brothers_-_Rockin'_Down_The_Highway
06-Eric_Clapton-Nobody_Knows_You_When_You're_Down_&_Out
13-Eric_Clapton-Old_Love
03-Eric_Clapton-Hey_Hey
09-Eric_Clapton-Walkin'_Blues
02-Eric_Clapton-Before_You_Accuse_Me
12-Eric_Clapton-Malted_Milk
```

Pop (Boy Bands)

```
Cluster 1:
11_-_No_One_Else_Comes_Close
07_-_Don't_Wanna_Lose_You_Now
09_-_Everything_I_Own
12_-_Sailing
04_-_For_The_Girl_Who_Has_Everything
05_-_I_Need_You_Tonight
02_-_I_Want_It_That_Way
07_-_Hands_Held_High
12_-_The_Perfect_Fan
03_-_Out_From_Under_[Main_Version]
09_-_Back_To_Your_Heart
05_-_Shadow_Of_The_Day
04_-_Cleanin_Out_My_Closet
12_-_The_Little_Things_Give_You_Away
12_-_My_Baby_[Main_Version]
03_-_Show_Me_The_Meaning_Of_Being_Lonely
10_-_Spanish_Eyes
```

Pop (Britney Spears)

```
Cluster 8:
09_-_Drips_(feat._Obie_Trice)
15_-_Phonography_[Main_Version]
14_-_Rock_Me_In_[Main_Version]
17_-_Say_What_You_Say_(feat._Dr._Dre)
08_-_Blur_[Main_Version]
02_-_Circus_[Main_Version]
04_-_Kill_The_Lights_[Main_Version]
05_-_Shattered_Glass_[Main_Version]
09_-_Mmm_Papi_[Main_Version]
10_-_Mannequin_[Main_Version]
03_-_Business
03_-_Foreigner_-_Feels_Like_The_First_Time
02_-_White_America
06_-_If_U_Seek_Amy_[Main_Version]
11_-_Lace_And_Leather_[Main_Version]
13_-_Radar_[Main_Version]
```

R&B (T-Pain)

```
Cluster 0:
09_Mix'd_Girl
12_When_I_Come_Home
15_Center_of_the_Stage_(feat._R._Kelly_and_Bei_Maejor)
03_It's_Not_You_(It's_Me)_(T-Pain_vs._Chuckie)_(feat._Pitbull)
07_Rock_Bottom
05_5_O'clock_(feat._Lily_Allen_and_Wiz_Khalifa)
10_I_Don't_Give_a_Fuck
16_Regular_Girl
05_-_Argent_-_Hold_Your_Head_Up
17_Nothin'_(feat._E-40_and_Detail)
13_Best_Love_Song_(feat._Chris_Brown)
06_Sho-Time_(Pleasure_Thang)
04_Default_Picture
01_Bang_Bang_Pow_Pow_(feat._Lil_Wayne)
08_Look_at_Her_Go_(feat._Chris_Brown)
```