Song retrieval based on similarity

Iraklis Evangelinos and Vangelis Lambrou
Final project for the Course Multimodal Machine Learning
University of Piraeus - NCSR Democritos
Athens, Greece

***Abstract*— Song retrieval is a task with many real world applications, it is widely used in everyday technologies on smartphones and other devices. The main aim of this project is to implement a system for song retrieval, using a number of variations of a model described below, in order to see the effect each hyperparameter has on the final model's performance. The features for the dataset are drawn from two different sources and then various proximity metrics are tested in order to retrieve the most relevant songs from the database. This specific task has been the subject of extensive research, as evidenced by the volume of scientific literature, engineering reports and various implementations readily available on the subject. The aim of this project is to provide a well documented implementation of the song retrieval task and also to show that simple, industry-standard methods are more than adequate to produce a fairly capable model. The scope of this work is narrow, but paying attention to both the dataset and the tuning of the model leads to interesting insights about both the distinguishing features of the songs in the database, as well as the hyperparameters effect on the model's performance.**

I. INTRODUCTION

In song retrieval various methods are used, with one of the most common being applying various metrics on feature vectors extracted from each song.

II. PROPOSED METHOD AND APPROACH

In this section the process to create the dataset is explained, along with notes on choices made about the features and other aspects. Following that, a subsection detailing the features used, as well as the sources and methods used to extract those features. In the subsection titled **The method used**, the model architecture is explained, along with the reasoning behind choices made during its implementation. Lastly, in the section titled **The metrics tested**, the various metrics chosen to implement and compare are explained, as well as some remarks regarding the author's choices and the reasoning behind them.

A. Dataset

The dataset consists of 170 songs sampled from the famous bands/artists ABBA, Bonnie Tyler, Iron Maiden, John Coltrane, Massive Attack, Metallica, Miles Davis and Portishead. The eight bands chosen to comprise the dataset belong to 4 different genres(Pop, Jazz, Metal, Alternative), with each genre having 2 bands representing it in our dataset. The specific artists were chosen to create the dataset because of their popularity, ensuring that the results from the model evaluation refer to plausible, real-world scenarios about song retrieval. Another factor taken into consideration is the range in genre and specific characteristics of the songs comprising the database offer. The dataset was created by obtaining the songs and converting the format into a .wav file for each song, while also retaining information about the song title, band etc. This choice was made in order to apply the library PyAudioAnalysis on the dataset to extract features, as detailed below.

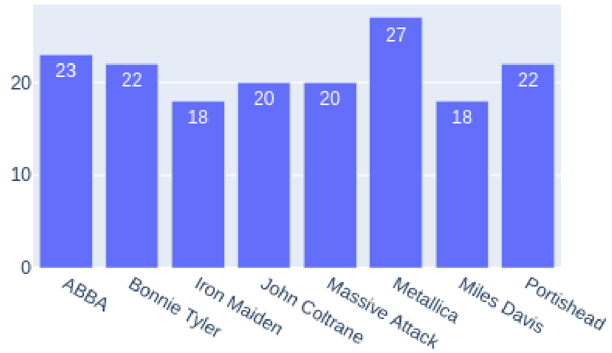


Fig. 1. *The number of songs per band/artist in the dataset*

B. Feature extraction

Features regarding each song were extracted using the PyAudioAnalysis library, as well as from Spotify's own API, which provides both data about each song, as well as extra features which prove to be very useful as demonstrated in a later section. Spotify's API provides us with 10 high-level features per song and 136 features from the midterm features pyAudioAnalysis provides are kept in order to create the final set of features per song. The features extracted from Spotify's API come ready and no parameters need to be chosen, as for the PyAudioAnalysis library, the following parameters are used to extract mid-term features:
$mid\_window = mid\_step = 5s$
$short\_window = short\_step = 1s$

The two vectors extracted for each song (i.e the one extracted from the python library and the one extracted from Spotify) are concatenated, creating a single feature vector for every song.

An interesting endeavour would be to keep the 10 features Spotify provides along with the result of the PCA algorithm applied on the other feature vector, in order to reduce the dimensionality, while also preserving the 10 high level

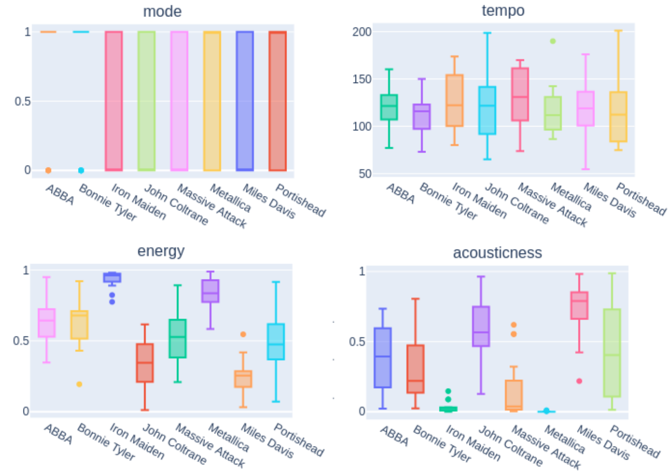


Fig. 2. *The features mode and tempo are not as useful for the task at hand, as opposed to the features energy and acousticness, which can help differentiate between bands and songs.*

features Spotify provides and then compare the results with the other models.

C. The method used

The extracted features constitute a single feature vector per song which is then used to create a reference database. When a new song is given to the model in order to make recomendations based on it, the feature vector is calculated in the same way, and the Spotify features are queried based on the name and band given along with it.

The resulting feature vector is then used to calculate proximity between the input-song and the songs on the reference database, in order to determine which are the ”closest” ones, therefore the most similar. The model ranks every song and returns the 5 closest ones in descending order as recommendations. During the testing section, the following metrics are compared as to their effectiveness and accuracy: *Euclidian, Manhattan, Cosine similarity, Chebyshev, as well as two more metrics which will be explained in detail bellow*. The aim is to chose the metric that performs the best in recommending similar songs. This is tested through the number of songs by the same band and same genre that end up in the top 5 recommendations.

D. The metrics tested

1) Manhattan, Euclidean, Chebyschev and Cosine metrics: The metrics are defined by the following formulas:
$Euclideandistance : \sqrt{|x_1 - y_1|^2 + ... + |x_{146} - y_{146}|^2}$
$Manhattandistance : |x_1 - y_1| + ... + |x_{146} - y_{146}|$
$Cosinedistance : \frac{\sum x_i y_i}{\sqrt{|x_1|^2+...+|x_{146}|^2}\sqrt{|y_1|^2+...+|y_{146}|^2}}$
$Chebyshevdistance : \max |x_i - y_i|$
Where x_i is the i-th value of a song’s feature vector and y_i is the corresponding value of another song’s feature vector. The formulas are applied to every vector pair, thus creating a 170x170 matrix containing all the distances between songs.

2) Hand-crafted metric based on Spotify’s features: The metric takes into account only the features provided by Spotify in order to determine the genre and suggest other similar songs. Utilising the determining characteristics for each genre and information for each song’s genre which is available via Spotify’s API specific features are used to optimise the suggestion depending on the input song. The features per genre are as follows:
Metal : Energy, Acousticness
Jazz : Energy, Loudness, Acousticness
Alternative : Danceability, Instrumentalness, Valence
Pop : Energy, Loudness, Instrumentalness

For each song name the metric finds a list of similar songs per feature (absolute distance) and then the list intersection is taken, in order to return the most ’similar’ songs.

3) Cosine PCA metric: This metric uses the feature vector and applies the PCA algorithm to the feature vector in order to reduce the dimensionality of the vector. This is especially significant due to the fact that the feature vector is almost the same size as the song database. The first two principal components are used for visualisation purposes. The first 25 principal components are retained and then cosine distances between each song are considered to determine which is to be recommended. This model was implemented after fine tuning and testing the number of principal components and best metric to use. By using 25 features the ’curse of dimensionality’ is avoided, while at the same time maximizing the effectiveness of the model. Other values for the principal components were experimented with and the value 25 was found to yield the maximum performance.The results are presented in detail in the following section.

III. EXPERIMENTAL RESULTS

A. Evaluation

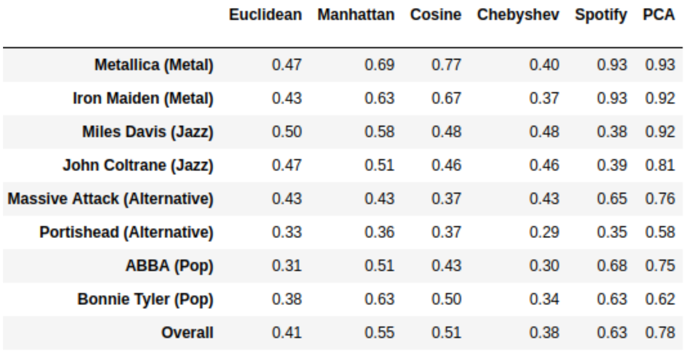
The evaluation of our model presents a difficulty, due to the fact that there is no top-5 recommendations per song in the dataset in order to use as ground-truth. Therefore the way the model is evaluated is such that each set of 5 recommendations is scored based on the number of songs that are the same genre as the song queried. Therefore the score is calculated by the following formula:

$$band_score = \frac{total\ no.\ of\ correct\ recommendations}{total\ no.\ of\ recommendations}$$

B. Results

The results are presented in figure 4, in the following page.

It is apparent that every one of the four ’standard’ metrics performs rather poorly in the task at hand, in the worst cases, some metrics are marginally better than random chance. As opposed to this phenomenon, the Spotify metric performs rather well, as was to be expected. The performance of the handcrafted metric is a testament to the quality of the features provided by Spotify’s API. It is apparent that the cosine metric utilising the PCA-reduced vector performs best, which is to be expected by the fact that the vectors are low dimension compared to the other metrics, as well as



	Euclidean	Manhattan	Cosine	Chebyshev	Spotify	PCA
Metallica (Metal)	0.47	0.69	0.77	0.40	0.93	0.93
Iron Maiden (Metal)	0.43	0.63	0.67	0.37	0.93	0.92
Miles Davis (Jazz)	0.50	0.58	0.48	0.48	0.38	0.92
John Coltrane (Jazz)	0.47	0.51	0.46	0.46	0.39	0.81
Massive Attack (Alternative)	0.43	0.43	0.37	0.43	0.65	0.76
Portishead (Alternative)	0.33	0.36	0.37	0.29	0.35	0.58
ABBA (Pop)	0.31	0.51	0.43	0.30	0.68	0.75
Bonnie Tyler (Pop)	0.38	0.63	0.50	0.34	0.63	0.62
Overall	0.41	0.55	0.51	0.38	0.63	0.78

Fig. 3. *The results of the evaluation per metric analysed into performance per band.*

the fact that the number of principal components as well as the final metric used to calculate distance were fine tuned by the authors.

IV. Notes for future study

The model bears some limitations, due to the nature of the dataset, as well as the choices made by the authors given the scope of this project. An interesting endeavour would be to use the Spotify metrics along with a cosine metric as a ground truth with which to test the predictions made by other models, thus overcoming the limitations of the evaluation mechanism, which tries to score song similarity based on genre classification. Another interesting endeavour would be to use a bigger dataset, with more similar songs and to apply the PCA algorithm only to the PyAudioAnalysis features, thus preserving the ones provided by Spotify, whose quality is apparent by the results of the project.

V. Conclusion

The dataset was created in such a way as to represent popular songs and genres in a small scale, due to the amount of manual work required to create it. The features were drawn from two sources commonly used in similar tasks and the feature vectors were used a means to gauge the similarity of a given song to the ones in the database. The various metrics tested, along with the parameters of the final metric were fine tuned in order to provide the optimal choice when judged by the performance score explained above. The final model is implemented in a demo which is provided along with this report and the project's presentation.