KV Multimedia Search and Retrieval

Exercise 2 Group E

Ali Ayadi
Johannes Kepler University
K12042692@students.jku.at

Luca Della Mura
Johannes Kepler University
k12241884@students.jku.at

Ruo Li
Johannes Kepler University
k12247854@students.jku.at

Sara Scheucher
Johannes Kepler University
k1648069@students.jku.at

ABSTRACT

In this paper, a rudimentary text-based retrieval system is being developed. The goal of this paper is to test the effect of different combinations of similarity measures and word embeddings in a retrieval system on its results. For calculating this similarity measure only text features are used. This paper compares the results of tf-idf, word2vec as well as the transformer-based BERT. The similarity measure is in all cases the cos-sim-similarity.

1 Introduction

The increasing availability of digital libraries has paved the way for a new generation of music recommender systems. Music recommendation systems play a pivotal role in helping users discover new tracks, artists, and genres and thus drive customer satisfaction in a significant way.
The purpose of this paper is to evaluate and compare 4 different approaches to music retrieval systems. Within the scope of this paper all features are going to be text-based.
Concretely, the cos-sim-similarity is used as a similarity measure and tf-idf, word2vec and BERT as word embeddings.
The resulting recommendations will be evaluated qualitatively according to the similarity to the queried song.
The data set used for testing the 4 retrieval systems is a subset of the Music4All-Onion dataset which was kindly provided by the university.

2 Methodology

For implementing the assignment, the programming language Python is used as it is most suitable for data analysis and data science purposes. The coding environment used is Jupyter Notebook as it supports the programming language Python. We also used the libraries NumPy and pandas as well as Scikit-learn which provide us with different similarity functions. The code repository is hosted on the platform GitHub. The coordination and integration of code contributions of each team member is therefore ensured using Git.

To ensure that new functionality as well as new algorithms can easily be added to the text-based music retrieval system in the future, a large focus is set on making the code modular.

The input of the system is the string which contains the name and the artist of the query track. The recommender system should output a list of songs with the title and the artist-

To be able to better analyze the results of the recommender system this output of the script is saved in a Python dictionary.

To keep the code modular and make it reusable we first implemented some basic functions in a separate Python file. First, we defined a function to get information about the artist and song name from the IDs. The function takes a list of ids as input and returns a list of tuples of strings containing the name and artist of tracks as the output. Then we implemented a function, which can be used for all text-based analysis. The function itself is called “text-based” and takes as input parameters “id”, the id of the query song, “repr” the string which represents the word embedding used in the data, for example, the tf-idf, “N”, the number of tracks retrieved as well as the similarity function for example Cosine similarity or Euclidean similarity. These measures help increase the flexibility and reusability of the code. Then we imported the functions into the main file where they can be called with different input parameters.

2.1 The dataset

Music4All-Onion is a large-scale, multi-modal music data set, which expands the Music4All-dataset with additional features and meta-data. For this task, only text-based features are considered (i.e.: title, lyrics, artist, album name and ID). [1]

The provided data is presented in four TSV files (Tab-Separated Values file) this type of file is similar to the CSV (Comma-Separated Values) data format, but it uses tabs to separate the values. To read these files we use the Pandas library which interprets the files as data frames.

The four files are:

- id_information_mmsr.tsv: contains the track IDs in the first column, the artist, song, and album name are contained in the remaining columns.
- id_lyrics_BERT_mmsr.tsv: presents the data using the BERT feature vectors with one column for the IDs.
- id_lyrics_tf-idf_mmsr.tsv: presents the data using the Term Frequency-Inverse Document Frequency feature vectors with one column for the IDs.

- id_lyrics_word2vec_mmsr.tsv: presents the data using the word2vec feature vectors with one column containing the IDs.

BERT: is a text-based word-embedding model trained on a collection of words that is capable of capturing rich contextual information in natural language. This allows BERT to understand the meanings of words in a given phrase. This method is widely used in NLP tasks.

TF-IDF: this method is the mix of two components TF (Term Frequency) and IDF (Inverse Document Frequency), this combination yields a vector capturing the relevance of words in a document relative to a corpus.
TF calculates the occurrence of a word for a given document on the other hand IDF calculates a word's importance for a given collection of documents the product of IDF by TF yields the relative importance of a word to a document within the overall corpus.

Word2vec: is an embedding technique that uses vector space. Words with similar context or meaning have similar vector representations (close vectors in the vector space). Through this, semantic relationships between words can be captured. This method is based on representing each word as a high-dimensional feature vector and training a neural network to learn their continuous vector space representation.

2.2 Random baseline

For the random baseline, we first shuffled the songs in a random order, so we get a different result each time the function is called. Then we excluded the query song from the data frame, so it does not appear in the result list. Afterwards, we retrieve the top N random songs and stored them in the result list.

2.3 Cos-sim based on tf-idf

To calculate the cosine-similarity of the tf-idf representation of the lyrics, we created a wrapper function called "cos_sim" that takes two Numpy-arrays as input and reshapes them to 2d arrays so they can be used in the cosine similarity function which is provided by the Scikit-learn library. The result of the cos_sim function is the similarity score of the two arrays. The similarity function cos_sim is then passed to the "text_based" function, as well as the query song-id and the dataset containing the tf-idf values. The first thing the test_based function does is search for the query song in the tf-idf dataset and extract the row vector representing the queried song. Then we create an array which is called similarities to store the similarity scores.
Afterwards, the "text_based" function iterates through all the rows in the tf-idf dataset. The similarities between the query-vector and track-vector are then calculated using the cos_sim function. The song-id as well as the similarity-score are then saved in the similarities list. Afterwards, we sort the list in decreasing order of the similarity score and retrieve the ids of the 10 most similar songs.

2.4 Cos-sim based on word2vec

The next retrieval system uses the word2vec-embedding with the cos-sim-similarity measure. The flow of the code is the same as tf-idf. So only one parameter has to be changed when calling the function.

2.5 Cos-sim based on BERT

As the last retriever that we have implemented for this project, we have chosen cosine similarity as its similarity function and lyric representation generated by BERT. As mentioned above, we have adopted a modularized scheme when implementing the text-based function. Thus, for this step, we just set the repr parameter of the text_base function to BERT, and the sim_func parameter to "cos-sim." The flow of execution is the same as when using the other two representations.

3 Qualitative Analysis

For the qualitative analysis, we selected 3 tracks for each retrieval system and retrieved 10 tracks for each query track:

3.1 Random baseline

The random baseline function will generate a random set of tracks every time it is called, irrespective of the query track. It often returns songs of different genres that are unrelated to the query track.

3.2 RS<Cos-sim, tf-idf>

Query Track 1: Love me by the 1975:
Result list:

Song	Oh Yeah	Artist	Big Time Rush
Song	The Gospel	Artist	Alicia Keys
Song	Fire Starter	Artist	Demi Lovato
Song	Rat Fink	Artist	Misfits
Song	How Bad Do You Want It (Oh Yeah)	Artist	Sevyn Streeter
Song	Yeah! (feat. Lil Jon & Ludacris	Artist	Usher
Song	Regarde-moi	Artist	Céline Dion
Song	Miss Independent	Artist	Ne-Yo
Song	Euphoria	Artist	BTS
Song	Let There Be Love	Artist	Simple Minds

In the results, there are songs of different genres like R&B, K-Pop, Pop, Indie-Rock, and Punk. The lyrics of the query song "Love me"

contain many occurrences of the tokens “yeah” and “love”. By examining the lyrics of the retrieved songs, we also noticed a high number of occurrences of the token “Yeah”. The tracks “Let There Be Love” and “Miss Independent” also exhibit a high occurrence of the word “Love” respectively. Other than that, the query song and the retrieved songs do not have so much in common and come from different genres.

Query Track 2: One by U2
Result list:

Song	One	Artist	Mary J. Blige
Song	One Love (feat. Estelle)	Artist	David Guetta
Song	Love the One You're With	Artist	Stephen Stills
Song	One	Artist	Alanis Morissette
Song	No One	Artist	Alicia Keys
Song	One Tribe (Defqon.1 2019 Anthem)	Artist	Phuture Noize
Song	You Can Be the One	Artist	Late Night Alumni
Song	Rape Me	Artist	Nirvana
Song	Palavras No Corpo	Artist	Gal Costa
Song	No One in the World	Artist	Anita Baker

For the second query track, we picked “One” by U2. The data also includes a cover version of this song by Mary J. Blige, which appears first in the result set because of the identical lyrics. The results also display a diversity in genre. The genre of the query track best described as a rock ballad, whereas the genre of the retrieved songs ranges from the genres R&B, Grunge, EDM and Pop-Rock. There is also one Spanish song in the results.

Query Track 3: Every Christmas by Kelly Clarkson
Result list:

Song	Christmas Conga	Artist	Cyndi Lauper
Song	Three Ships	Artist	Cyndi Lauper
Song	Hellhound On My Trail	Artist	Robert Johnson
Song	St. Patrick's Day	Artist	John Mayer
Song	Last Christmas	Artist	Carly Rae Jepsen
Song	My Only Wish (This Year)	Artist	Britney Spears
Song	Christmas Vacation	Artist	Descendents
Song	Last Christmas - Studio Version	Artist	Jimmy Eat World
Song	The Christmas Song (Merry Christmas To You)	Artist	Nat King Cole
Song	I Shut Doors and Windows	Artist	September Malevolence

For the third query, we analyzed a Christmas song because Christmas songs usually contain recurring tokens in the lyrics, for example, the tokens “year”, “wish” or “mistletoe”. As we can see in the results, we retrieved 9 Christmas songs and one song with a different theme. The song “I Shut Doors and Windows” by September Malevolence could not be described as a Christmas song but also contains one occurrence of the token “Christmas” in its lyrics. The results include two tracks from the same artist “Cyndi Lauper”, which also happens to be from the same Christmas-themed album.

3.3 RS<Cos-sim, word2vec>

Query song 1: Love me by the 1975:
Result list:

Song	Miss Independent	Artist	Ne-Yo
Song	If Our Love Is Wrong	Artist	Calum Scott
Song	Looking For Clues	Artist	Robert Palmer
Song	Out on the Tiles	Artist	Led Zeppelin
Song	So Much Love	Artist	The Rocket Summer
Song	Let There Be Love	Artist	Simple Minds
Song	In the Evening	Artist	Led Zeppelin
Song	All You Got	Artist	Tegan and Sara
Song	Rosalyn	Artist	David Bowie
Song	How Bad Do You Want It (Oh Yeah)	Artist	Sevyn Streeter

The results contains two different tracks by the same artist “Led Zeppelin”. It is noticed that the result set contains several tracks by British artists. The artist who made the query track is also a British Band. which leads us to the speculation that the cluster might be

attributed to different language use between British English and American English. The genres of the retrieved tracks are Rock, Pop, R&B and Rap, when ranked in decreasing order.

Query Track 2: One by U2
Result list:

Song	One	Artist	Mary J. Blige
Song	One Love (feat. Estelle)	Artist	David Guetta
Song	Quien Eres Tu (Feat. Trey Songz)	Artist	María José
Song	Dance With The One That Brought You	Artist	Shania Twain
Song	Apocalyptic	Artist	Halestorm
Song	I Will Survive - Extended Version	Artist	Gloria
Song	King For A Day	Artist	Thompson Twins
Song	Never Let Me Down	Artist	Kanye West
Song	Fica Mais um Pouco Amor	Artist	Emicida
Song	I Will Survive	Artist	Gloria Gaynor

For the query track “One” by “U2”, the cover version by Mary J. Blige ranks the first in the results, just like we did with the tf-idf representation. We also retrieved one Spanish and one Portuguese song. The Genres of the retrieved track are Rock, Pop, Electronica, Country, Hip-Hop, Samba and Pagode, when rank in decreasing order of similarity.

Query Track 3: Every Christmas by Kelly Clarkson
Result list:

Song	St. Patrick's Day	Artist	John Mayer
Song	Junesong Provision	Artist	Coheed and Cambria
Song	My Only Wish (This Year)	Artist	Britney Spears
Song	PERFECT!	Artist	WJSN
Song	Si Tu Novio Te Deja Sola	Artist	J Balvin
Song	So Doggone Lonesome	Artist	Johnny Cash
Song	The Best Day	Artist	Taylor Swift
Song	Wait For You	Artist	Elliott Yamin
Song	Jesus Christ	Artist	Brand New
Song	Alone (feat. Big Sean & Stefflon Don)	Artist	Halsey

For the third track, the result includes one Korean song and one song in Spanish. We retrieved all kinds of different genres like country, Rock, and Pop. The entropy of the genre is thus high. Not as many Christmas themed songs are included in the results compared to the results obtained from other data.

3.4 Cos-sim based on BERT

Query song 1: “Love me” by “The 1975”
Result list:

Song	Dance Gavin Dance	Artist	Thug City
Song	Shine	Artist	Take That
Song	One, Two, Three, GO!	Artist	Belanova
Song	Right There	Artist	Ariana Grande
Song	Bing Bing	Artist	Crayon Pop
Song	Come Get It Bae	Artist	Pharrell Williams
Song	We Made You	Artist	Eminem
Song	Here I Am	Artist	Monica
Song	Wannabe	Artist	why mona
Song	Edge of the World	Artist	Faith No More

For the results generated by the first query track with the BERT data, none of the ten retrieved tracks appears in the result from the other two datasets. In terms of genre, the results show a similar pattern result to the other two datasets. The genres that appeared in the results including Pop, Indie Rock, R&B, Funk, Hip Hop and K-pop, showed no effects on the result.

Query song 2: “One” by “U2”
Result list:

Song	One	Artist	Mary J. Blige
Song	What About Love	Artist	Austin Mahone
Song	All of Your Glory	Artist	Broods
Song	La Tortura	Artist	Shakira
Song	Love One Another	Artist	Cher
Song	Black Lake	Artist	Björk
Song	El Triste	Artist	José José
Song	Love Makes the World Go Round	Artist	Ashlee Simpson
Song	Keep It Together	Artist	Madonna
Song	U Want Me 2	Artist	Sarah McLachlan

The results generated with the second query track and the BERT data, again, do not show similarity with the results obtained from the other two data with the sole exception being “One” by Mary J.

Blige. As explained in the section above. It is a cover version of the query song. Therefore, its lyrics is identical to that of the query song. In terms of genre, most tracks appeared in the results belong to the Pop genre, different from Rock, the genre of the query song.

Query song 3: “Every Christmas” by “Kelly Clarkson”
Result list:

Song	My Only Wish (This Year)	Artist	Britney Spears
Song	Christmas Conga	Artist	Cyndi Lauper
Song	Merry Christmas, Kiss My Ass	Artist	All Time Low
Song	St. Patrick's Day	Artist	John Mayer
Song	The Christmas Song (Merry Christmas To You)	Artist	Nat King Col e
Song	Last Christmas	Artist	Carly Rae Jepsen
Song	Next Year	Artist	Foo Fighters
Song	December's Boudoir	Artist	Laura Nyro
Song	Last Xmas	Artist	Allie X
Song	Santa Claus Is Coming To Town	Artist	The Jackson 5

In the results generated by the third query track with the BERT data, again, five of the retrieved tracks appear in the result from other datasets. Also, in terms of genre, most tracks belong to the Pop genre, which could be attributed to the theme of the song, Christmas. Christmas music is known to be associated with instrumentals, Carol and Pop genre. It is also worth mentioning that the song “St. Patricks Day”, which appears in the results obtained from other datasets, is also included in the results. An examination of the lyrics reveals that, despite the title of the song being St Patricks Day, there are repeated references to words such as "cold", "snow", "December" and other words that might be found in other Christmas songs, as well as the phrase "Christmas times" itself appears three times.

4. Audio-based retrieval systems

For the audio-based retrieval systems, we used the same 3 query Songs as we used for Task 1. We used the following representations: MFCC stats, blf correlation, ivec256 and musicnn. For all these representations we calculated the cosine similarity. The results for all 3 query tracks are displayed in the main.ipynb file.

5. Methodology & Implementation

5.1 Precision & Recall

Recall and Precision are measures that show how well a retrieval system retrieves relevant information. The precision is the ratio of true positives and total retrieved results. The recall is the ratio of true positives and the actual number of positives. [2]

For the calculation of precision and recall we first obtained the genres of our retrieved results and put them into a list which consists of ids and genres of retrieved songs. This list is one of the parameters for the precision_at_k function. We also need k as a parameter as well as the id and genre of the query track. Afterwards, we store the top k results into a variable and compare the genres of the top k results with the query genre and count how many of the retrieved results are relevant (a result is relevant if it has at least one common genre with the query track). Finally, we divide the relevant retrieved results by k.

For the calculation of the recall, we need one more parameter which is the whole genre dataset. The method calculates the number of retrieved relevant songs as well as the number of relevant songs in the whole genre dataset and then divides the relevant retrieved songs by all relevant songs.

5.2 Genre diversity

Diversity:
Diversity is the opposite of similarity, in music recommendation systems (RS), diverse genres are needed to give the user a better and broader recommendation outcome so he can choose flexibly according to his preferences. To ensure large diversity a wide range of genres have to be presented in the retrieved list by the RS. Different methods have been proposed by researchers to calculate the diversity such as calculating the distance between two elements i and j in the recommended list. Cosine similarity can be also used as a distance function to calculate the diversity.[3]
Diversity as an evaluation metric:
This method measures the diversity of genres of a top k retrieved tracks given a query. It calculates how the genres are evenly distributed over k retrieved tracks.
We can break down the formula into two parts:
Genre distribution:
We initialize a vector of zeros with a length equal to the number of all genres existing in the dataset, then for every genre found in each retrieved track, we add one to the corresponding genre position in the zeros vector divided by the number of genres of the retrieved track.
So, this could be considered as the normalized attribution of each genre within the retrieved track genres to the overall genres in the dataset.

Normalize of the distribution:
We divide the resulting vector by the number of the retrieved tracks. To get the genre diversity we calculate Shannon's entropy of the resulting vector.
This is calculated by taking the negative sum of all the items of the resulting vector multiplied by its logarithm (base 2).

Formula in python:
For this formula, we defined a function called diversity which takes genres_retrived, all_genres and N as input parameters.
Genre_retrived: it is a list of sets containing the genres of the retrieved tracks.
All_genres: list of all unique genres in the whole dataset
N: is the number of retrieved tracks. The formula should return the genre diversity@k. We first define the zeros vector (zeros_vec) with the length of all_genres, and then we run through all the retrieved genres of every retrieved track.
Position: takes the index of the retrieved genre in the whole dataset's unique genres.
We calculate each retrieved genre contribution by dividing one by the length of the retrieved set of genres and we store it in a variable called g_i_contribution. Afterwards, we accumulate the attribution to the zeros_vec in the giving position. After running through all the genres_retrieved we divide the zeros_vec by N and we assign the result to the variable: result_vec. Next, we move on to the second part of the formula (Shannon's Entropy) that returns the genre diversity@k. We initialize the diversity variable to zero.

We run through all the items in the result_vec and if it is different than zero then multiply it by its logarithm (base 2) and the result should be accumulated into the diversity variable.
The function will then return the negative diversity which represents the genre diversity@k.

5.3 Genre coverage

This metric is similar to the Genre Diversity described in the last chapter a way to measure the quality of the audio-based retrieval system beyond using accuracy. For this research project the Genre Coverage is defined as the proportion of the number of unique genres present within at least one the top k retrieved tracks and the number of unique genres within the dataset itself.

5.4 nDCG

Normalized discounted cumulative gain (later referred in the current study as nDCG) evaluates the results based on graded relevance, i.e. nDCG assumes the users prefer the elements in the list of retrieved results to be presented in the descending order of their degree of relevance. Its calculation can be summarized in the following four steps:
1) Calculate the degree of relevance of each element (later referred in the current study as gain) in the list of retrieved results.
2) Assign the weight to the gain obtained from each element respectively according to its position in the list of retrieved results. Calculate the weighted sum of the gains. (later referred in the current study as DCG, i.e. Discounted Cumulative Gain)
3) Generate an ideal list of the retrieved results by reordering the elements in the list in the descending order of their gains. Calculated the DCG score for the ideal list. (later referred in the current study as iDCG, i.e. ideal Discounted Cumulative Gain)
4) Divide the DCG score of the list of the retrieved results by the iDCG score. [4]
The setting of the current study is as follows:
The function written to calculate the nDCG score for current study only considers the top k elements in the list of retrieved results. When evaluating the results of this particular study, k is set to 10. (The metric is thus referred to in the result section as nDCG@10).
The Sørensen–Dice coefficient of the genres is used to compute the gain, which adopts the following formula:

$$\text{gain}_i = 2 * |G_{query} \cap G_i| / (|G_{query}| + |G_i|)$$

G_{query} refers to the set of genres of the query track. G_i refers to the set of genres of the track used to calculate the gain. The genre information is obtained from the id_genres_mmsr.tsv dataset.
The weight to the gain was calculated with an inverse logarithm of 2. The formulae adopted by the current study to calculate the nDCG are the following:

$$\text{DCG@10} = \text{gain}_1 + \sum_{i=2}^{k} \frac{gain_i}{\log_2 i}$$

$$\text{NDCG@10} = \frac{\text{DCG@10}}{\text{iDCG@10}}$$

6. Evaluation

We evaluated 5 different measurements over our 3 query tracks. In this chapter, we are going to present our findings regarding the Precision and Recall of our retrieval systems as well as the Genre Diversity, Genre Coverage and nDCG. Detailed results can be found in Tables 1 – 3 located at the end of the report.

6.1 Precision@10 & Recall@10

The first query track which is "Love Me" by "The 1975" is assigned to the genres: pop, rock, indie pop, electro-pop, indie rock, funk, and funk rock. We achieved a high precision of 90 % with all the audio-based representations except for the ivec256 where we only achieved a precision of 50 %. We achieved the highest recall of 0.125 % with the MFCC Stats, the blf correlation and the musicnn features. The recall is relatively low for all retrieval systems because the dataset consists of a large number of tracks, and we only retrieve 10 tracks with each system. Also, the genres pop and rock are very common genres so there might be many songs in the dataset assigned to these genres, therefore we have many relevant documents in the dataset which are not retrieved with our systems.

The second query track "One" by "U2" is assigned to the genres rock, classic rock, pop, alternative rock, soft rock, easy listening and Irish rock. We achieved a precision of 80 % with the blf-correlation, ivec256 and musicnn representations. With these three we also obtained the highest recall of 0.114 % With the mfcc stats we only get a precision of 60 %.

The third query track "Every Christmas" by "Kelly Clarkson" is only assigned to one genre which is pop. With the audio-based retrieval system, we achieved the highest precision of 60 % with the blf-correlation representation. With the text-based retrieval system, we were able to achieve a precision of 90 % using the BERT representation. The reason for this overall lower precision for query track 3 could be that it is only assigned to one genre, whereas the other two query tracks belong to several genres. With

the audio-based retrieval systems, we obtained the highest recall for query track 3 with the blf-correlation feature with 0.143 %.

In the precision-recall curves we plotted for all three query tracks we can see how the number of k affects the precision and recall. We note that, as the value of k increases, also the recall increases.

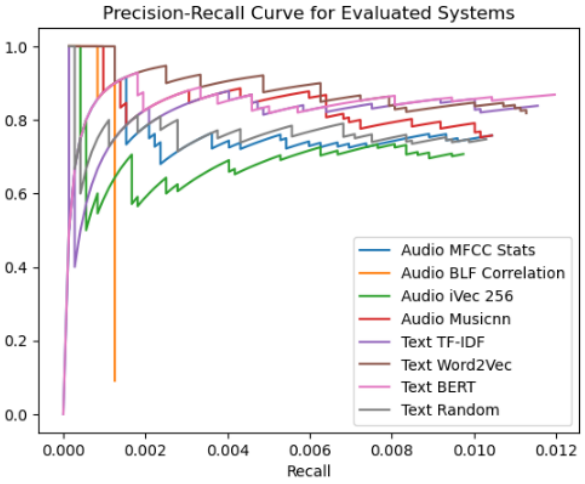

Figure 1: Precision-Recall Curve Track1

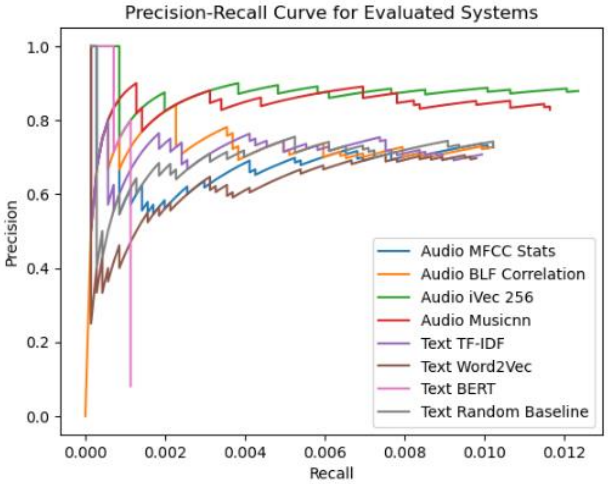

Figure 2: Precision-Recall Curve Track2

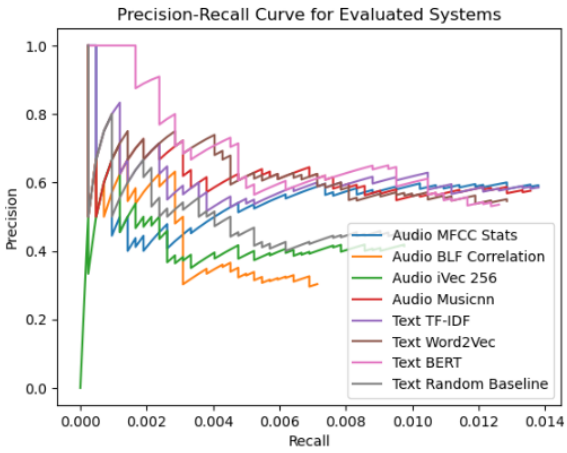

Figure 3: Precision-Recall Curve Track3

6.2 Genre diversity@10

The diversity formula is based on Shannon's entropy which we would like to maximize this is equivalent to minimizing its negative. Consequently, a smaller diversity score indicates a more diverse retrieved list. For query track 1 the audio-based retrieval system using the ivec 256 features embedding shows a bigger score around 4.69 compared to the other three. On the other hand, text-based retrieval systems using BERT and word2vec as feature embedding got bigger diversity scores of 4.989 and 5.269 while the other text-based systems showed a smaller score of about 4.3 this could be interpreted that BERT and word2vec are powerful embedding systems that lead to retrieving tracks more similar to the query track and therefore very small genre diversity has resulted and this can be seen as overfitting. however, the baseline showed a big diversity score which means that it has a small range of genre diversity and is incapable of suggesting a large variety of genres. for the first track, the audio-based retrieval systems showed better scores than the random baseline and the text-based retrieval system excluding the one using tf-idf embeddings. Concerning query track 2 we can observe from the table the text-based retrieval system had an improved score (i.e. smaller score) especially word2vec which had the biggest score for query track 1 while tf-idf became worse. For the audio-based retrieval system the scores did not change too much only for ivec 256 that reached a score of 5.02 (become worse) and still has the biggest score among all the audio-based retrieval system scores. The baseline showed better performance compared to track 1 and since its random the result may very every time. Overall, the scores were of all retrieval system were close for this query track. For the query track 3 the audio-based retrieval system has shown bad scores this time (i.e. big score) as most of them exceed the score 5. This shift also happened with the text-based systems but inversely as they had better scores (i.e. smaller scores) only the word2vec system is still overfitting with a score of 5.33. the random baseline behaved slightly better this time as it achieved the best score of 4.529 for all three query tracks. We can conclude that text-based systems outperformed the audio-based systems for this query track.

6.3 Genre Coverage@10

The next result section was concerned with the genre coverage@10 score. As mentioned in the methodology section, genre coverage assesses the proportion of unique genres covered in the retrieved list. Therefore, a higher genre coverage@10 score indicates a more diverse retrieved list. As can be seen from table 1, genre coverage@10 score obtained with query track 1 spread out within the range between 0.02878 and 0.06205. The results from the four audio-based retrieval systems show two clusters. The one using ivec 256 feature embedding with a higher genre coverage@10 score of 0.3957 forms a cluster of its own. The rest three with genre coverage@10 scores around 0.3 forms another cluster. The results from the three text-based retrieval system also show two clusters. The two using BERT and word2vec feature embeddings achieved a higher genre coverage@10 score around 0.6, whereas the one using tf-idf achieved a lower genre coverage@10 score of 0.02878. When comparing the results between those obtained from audio-based and text-based retrieval systems, genre coverage@10 scores of text-based retrieval systems are high, indicating for the results generated with query track 1, text-based retrieval systems tend to return lists that are more diverse in genre.

Moving on to the results obtained with query track 2, as shown in table 2, genre coverage@10 score obtained with query track 2 spread out within the range between 0.02518 and 0.04856. The results from the four audio-based retrieval systems show three clusters. The retrieval system using ivec256 feature embedding returns the highest genre coverage@10 score of 0.04856. The one using blf-correlation feature embedding returns the second highest genre coverage@10 score of 0.03957. The rest two return the genre coverage@10 scores around 0.03. The results from the three text-based retrieval systems show two clusters. The retrieval systems using tf-idf and BERT feature embeddings form one cluster with higher genre coverage@10 scores around 0.04. The one using word2vec feature embedding achieved a lower score of 0.02518. When comparing the results between those obtained from audio-based and text-based retrieval systems, the genre coverage@10 scores are distributed evenly, indicating for the results generated with query track 2, neither text-based retrieval systems nor audio-based retrieval system tends to return lists that are more diverse in genre.

Finally, let us examine the results obtained with query track 3. A closer inspection of table 3 reveals that genre coverage@10 score obtained with query track 3 spread out within the range between 0.03507 and 0.05845. The results from the four audio-based retrieval systems are evenly distributed around 0.05. The results from the three text-based retrieval systems show two clusters. The one using word2vec feature embedding achieved a slightly higher genre coverage@10 score than the other two at 0.04676. The other two achieved scores around 0.035. When comparing the results between those obtained from audio-based and text-based retrieval systems, genre coverage@10 scores of text-based retrieval systems are high, indicating for the results generated with query track 3, audio-based retrieval systems tend to return lists that are more diverse in genre overall than text-based retrieval systems.

6.4 nDCG@10

Now let us shift our focus to the results concerning the nDCG@10 score. As mentioned in the methodology section, a larger nDCG@10 score indicates a better performance of the retrieval system as the lists obtained from the results are more similar to their ideal counterparts when ranking is taken into account. Closer inspection of table 1 shows that all seven retrieval systems achieved better performance than the random baseline with query track 1. Also, all seven retrieval systems managed to achieve an nDCG@10 score larger than 0.7. Among all four audio-based retrieval systems, the results show two clusters, with the two retrieval systems using musicnn and ivec 256 feature embeddings forming one cluster which achieved an nDCG@10 score larger than 0.9, and the rest two retrieval systems forming a slightly underperformed cluster. Among all three text-based retrieval systems, the one using tf-idf feature embedding achieved the best performance. No apparent cluster can be observed from the results. When comparing the results between those obtained from audio-based and text-based retrieval systems, audio-based retrieval systems achieved a better performance regarding query track 1.

Let us then move on to discuss the results concerning the nDCG@10 scores obtained with query track 2. As shown in table 2, same as the results obtained with query track 1, all seven retrieval systems achieved an nDCG@10 score larger than 0.7 and better performance than the random baseline. Among all four audio-based retrieval systems, the results again show two cluster, with the retrieval system using ivec 256 feature embeddings in one cluster which achieved an nDCG@10 larger than 0.8 and the rest three in another cluster with an nDCG@10 larger than 0.7. The nDCG@10 from the text-based retrieval systems all exceed the threshold of 0.8 and exhibit a close interval from each other. When comparing the results between those obtained from audio-based and text-based retrieval systems, text-based retrieval systems achieved a better performance regarding query track 2.

Finally, turning now to the results concerning the nDCG@10 scores obtained with query track 3. As can be seen in table 3, the retrieval systems achieved overall a worse performance than with the other two query tracks with the exception of the text-based retrieval system using the tf-idf feature embedding. No apparent cluster can be observed from the results obtained from the audio-based retrieval systems. The nDCG@10 scores are evenly distributed around 0.6. As mentioned above, the results obtained from the text-based retrieval systems show two clusters. The retrieval system employing tf-idf feature embedding achieved an nDCG@10 score larger than 0.9. The rest two form another cluster with nDCG@10 scores around 0.7. When comparing the results between those obtained from audio-based and text-based retrieval systems, text-based retrieval systems achieved a better performance regarding query track 3.

Results (rounded to 5 digits)

Query Track 1: Love Me by The 1975

Table 1: Evaluation Results Track 1

	Precision@10	Recall@10	nDCG@10	Coverage@10	Diversity@10
Audio-based(cosine, mfcc_stats)	0.9	0.00125	0.76718	0.02878	4.51273
Audio-based(cosine, Blf-correlation)	0.9	0.00125	0.82250	0.02878	4.43499
Audio-based(cosine, ivec 256)	0.5	0.00070	0.93006	0.03957	4.69441
Audio-based(cosine, musicnn)	0.9	0.00125	0.96787	0.03058	4.32781
Text-based(cosine, tf-idf)	0.7	0.00097	0.85336	0.02878	4.30428
Text-based(cosine, word2vec)	0.9	0.00125	0.73128	0.05486	5.26926
Text-based(cosine, BERT)	0.9	0.00125	0.77874	0.06205	4.98983
Random-Baseline	0.6	0.00083	0.67518	0.04317	5.24167

Query Track 2: One by U2

Table 2: Evaluation Results Track 2

	Precision@10	Recall@10	nDCG@10	Coverage@10	Diversity@10
Audio-based(cosine, mfcc_stats)	0.6	0.00085	0.75426	0.02968	4.46326
Audio-based(cosine, Blf-correlation)	0.8	0.00114	0.70208	0.03957	4.94064
Audio-based(cosine, ivec 256)	0.8	0.00114	0.83261	0.04856	5.02604
Audio-based(cosine, musicnn)	0.8	0.00114	0.74983	0.03237	4.48012
Text-based(cosine, tf-idf)	0.6	0.00085	0.85755	0.03956	4.98059
Text-based(cosine, word2vec)	0.4	0.00057	0.84163	0.02518	4.41128
Text-based(cosine, BERT)	0.8	0.00114	0.88648	0.03866	4.73847
Random-Baseline	0.5	0.00071	0.65161	0.03327	4.88552

Query Track 3 : Every Christmas by Kelly Clarkson

Table 3: Evaluation Results Track 3

	Precision@10	Recall@10	nDCG@10	Coverage@10	Diversity@10
Audio-based(cosine, mfcc_stats)	0.5	0.00119	0.57058	0.05845	5.33913
Audio-based(cosine, Blf-correlation)	0.6	0.00143	0.61340	0.04946	4.84336
Audio-based(cosine, ivec 256)	0.5	0.00119	0.58630	0.05486	5.37249
Audio-based(cosine, musicnn)	0.4	0.00167	0.59099	0.04856	5.14491
Text-based(cosine, tf-idf)	0.6	0.00143	0.95267	0.03507	4.30369
Text-based(cosine, word2vec)	0.7	0.00167	0.73712	0.04676	5.33174
Text-based(cosine, BERT)	0.9	0.00214	0.76511	0.03776	4.24776
Random-Baseline	0.3	0.00071	0.49639	0.03417	4.52973

7. Tables

Table 1: Evaluation Results Track 1 1
Table 2: Evaluation Results Track 2 2
Table 3: Evaluation Results Track 3 3

8. Figures

Figure 1: Precision-Recall Curve Track1 8
Figure 2: Precision-Recall Curve Track2 8
Figure 3: Precision-Recall Curve Track3 8

9. REFERENCES

[1] Moscati, M., Parada-Cabaleiro, E., Deldjoo, Y., Zangerle, E., & Schedl, M., „Music4All-Onion“, doi: 10.5281/zenodo.6609677.

[2] K. M. Ting, „Precision and Recall“, in *Encyclopedia of Machine Learning*, C. Sammut und G. I. Webb, Hrsg., Boston, MA: Springer US, 2010, S. 781–781. doi: 10.1007/978-0-387-30164-8_652.

[3] M. Kuanr, P. Mohapatra, "Assessment Methods for Evaluation of Recommender Systems: A Survey" *Foundations of Computing and Decision Sciences*, vol 46, pp. 394-42, 2021.

[4] S. Büttcher, C. L. A. Clarke, G. V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engines*. MIT Press, Cambridge, USA; London, UK, 2010.