Using Global Vectors in Social Interaction Network for Song Recommendation

1 Introduction

Much like speech, or written text, music is one of the cohesive forces that binds society together, and is arguably the oldest form of communication, even before language. Today music has become more globalized than ever, and social networks have a lot to contribute to it. The music industry today relies heavily on streaming services, like Apple Music, Spotify, Google Play Music, Pandora, and the like. A defining characteristic of these services is that they have either curated playlists, where a team of music experts compile lists of songs of the same genre, or recommended playlists, using the user's listening history. However, a particular user can be the member of different streaming services, and can listen to various songs on different platforms at different points of time. Therefore, the auto-generated lists based on history for that particular user will vary across streaming services. Also, a limitation of history based recommendation strategies is that it depends a lot on the user's history and globally trending songs. The first runs the risk of getting stuck in a filter bubble of the same songs again and again, while the second runs the risk of over-generalization, as in globally trending songs may not cater to the tastes of each user. Our proposed recommender is the best of both worlds, as it harnesses both the trend component and personalized reaction components in a social network, and is agnostic to the music service preferred by the user.

2 Related Work

Recommendation systems are one of the widely researched areas in computer science. The main tasks involved are either tag recommendation for items or item recommendation on the basis of the tags or both by using user data. In our case we consider item prediction (here item is song) on the basis of tags and user data. The authors of [1] have approached the problem of karaoke recommendation by developing distinct embeddings of the songs. The paper [3] demonstrates a hybrid song recommendation procedure involving playlist proximity information and song similarity. The authors use matrix factorization technique in order to meet the objective.

3 Dataset

3.1 Method of creating dataset

We used a Facebook group where users share music from various streaming sites to validate this idea. The users, subsequently react ('like', 'love', 'haha', 'angry', 'wow' and

'sad'), and/or comment on the song links shared by each other. All posts in the Facebook group between 15th August 2015 and 16th February 2018 were scraped using the Facebook graph api. This dump consisted of 4618 Facebook posts. Each post contained the the link of the song shared (from some music streaming site), the timestamp of posting, the id of the user posting, the number of reactions and comments ,along with the id of the users who reacted or commented.

The song urls shared were from a variety of different streaming sites (Youtube, Apple Music, Spotify, Soundcloud, etc.). To disambiguate these links and obtain the song details, we used songwhip [1] to obtain the name and the artist of the track. Additionally the use of last.fm [2] crawler helped us to get the tags related to the song. Along with each genre, each of the artists involved in the song was a tag for that song. All of this information about the songs were merged into the crawl dump to create the final database. Figure 1 shows a wordcloud of the tags in the dataset.

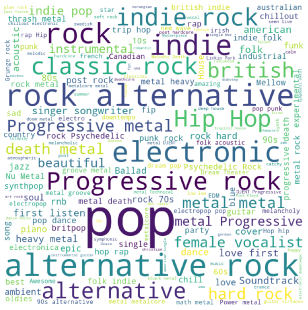

Fig. 1. Tags in our dataset

4 Procedure

4.1 Forming communities

Using the user-user graph (formed using Networkx[2] python package), we first cluster them using the Girvan-Newman algorithm[4] to discover communities successively. We stop at the 29th iteration, to find 46 communities. Communities having less than 5 users are considered invalid, leaving us with a total of 30 distinct communities. Standard KNN is applied to merge the users that were not classified into any community to the nearest cluster by Euclidean distance.

4.2 Vector representation of songs and tags

We convert the song tags (genre tags and artist tags) to vectors for easier representation using Glove . The co-occurence dictionary of the tags in various songs is calculated to train the glove vector of dimension 50. Every tag is represented as a vector of size 50 and subsequently, each song is represented as a weighted average of these tags.

4.3 Vector representation of users and communities

For user, we assign the weights 6, 3, 2, 1 if the song was posted, reacted 'love' , reacted 'wow', reacted 'like' respectively by the user. After this, the weighted average of all the songs the user has interacted with is calculated using the above weights.

[1] Songwhip: https://songwhip.com/

[2] Last.fm Website. https://www.last.fm/

For a community, the arithmetic mean of the vectors for all users is taken as the vector representing the centroid of the community.

5 Recommendation Algorithm

Result: List of recommended songs in descending order of relevance
User vector (u)
 ← weighted average of tag vectors for each tag associated with this user
Community vector (c) ←
 weighted average of tag vectors for tags associated by community conyaining this user
Recommendation_score ← $dict()$
for *each song in dataset* **do**
  song vector (s)
   ← weighted average of tag vectors for tags associated with this song
  Recommendation_score[song] ←$\alpha_1 \times u \cdot s + \alpha_2 \times c \cdot s$
end
Sort Recommendation_score by value, descending order Show top k songs

α_1 and α_2 are hyperparameters to weight the importance of user and community respectively.

6 Results and Future Work

Our recommendation algorithm as expected, generates a list of top 10 songs for every user, and the results subjectively match the artist and genre preferences of the user upon manual testing.

However, we need to create standardized train-test sets to accurately calculate the performance of our algorithm and fine tune our weight hyperparameters. Another direction that we can explore is to use a personalized pagerank algorithm as discussed on this blog[3] to improve the vector representation of the songs and users by taking into account that users represent songs and vice versa (analogous to hub-authority relationship). Thus, we have created an early prototype of a system that can be used as a lightweight and reliable recommender for music discovery by harnessing the feedback from users in a closely knit group inside a popular social network.

References

1. Wu, Xiang and Liu, Qi and Chen, Enhong and He, Liang and Lv, Jingsong and Cao, Can and Hu, Guoping,"Personalized Next-song Recommendation in Online Karaokes" in Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13) pages = 137–140
2. Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, Exploring network structure, dynamics, and function using NetworkX, in Proceedings of the 7th Python in Science Conference (SciPy2008), Gel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 1115, Aug 2008

[3] https://blogs.cornell.edu/info2040/2017/10/25/personalized-pagerank-in-recommendation-system/

3. Kirell Benzi, Vassilis Kalofolias, Xavier Bresson, Pierre Vandergheynst, ”Song Recommendation with Non-Negative Matrix Factorization and Graph Total Variation” https://arxiv.org/abs/1601.01892
4. Girvan, M. and Newman, M. E. J.,”Community structure in social and biological networks”, pages = 7821–7826, year = 2002, http://www.pnas.org/content/99/12/7821, in Proceedings of the National Academy of Sciences