CMPE-256: Million Songs Dataset

Carlos Hernandez
Computer Engineering Dept.
San Jose State University
carlos.hernandez@sjsu.edu

Hardy Leung
Computer Engineering Dept.
San Jose State University
kwok-shing.leung@sjsu.edu

John Lu
Computer Engineering Dept.
San Jose State University
john.lu@sjsu.edu

***Abstract*—We propose to investigate the Million Songs Dataset (MSD) [3], a collection of user data and metadata for popular contemporary songs. In particular, we will look at the user taste subset of datasets which consists of users, songs, and their respective play counts. The training data is approximately 500 MB (compressed) and has 1 million users, 380,000 unique songs, and 48 million user-song play count triplets. The test data has 100K users, and the validation data has 10K users. Note that users don't rate the songs; instead, we are given how many times a user played a certain song.**

***Index Terms*—recommender system, implicit feedback**

I. Dataset

We propose to investigate the Million Songs Dataset (MSD) [3], a collection of user data and metadata for popular contemporary songs. In particular, we will look at the user taste subset of datasets which consists of users, songs, and their respective play counts. The training data is approximately 500 MB (compressed) and has 1 million users, 380,000 unique songs, and 48 million user-song play count triplets. The test data has 100K users, and the validation data has 10K users. Note that users don't rate the songs; instead, we are given how many times a user played a certain song.

We will be using the truncated mAP (mean average precision) as the evaluation metric. For users in only the validation test sets, half of the songs in each player's playlist are kept hidden, and the recommendations for each user will be compared to the hidden half.

II. Metadata

We intend to use additional metadata provided as part of the Million Songs Dataset. We believe the following datasets would be helpful to our research.

A. Last.fm dataset [3]

`http://millionsongdataset.com/lastfm/`

This dataset contains annotations for 943,347 tracks. An annotation may include tags (such as POP, CLASSIC ROCK, etc), as well as a list of similar songs. We will be mindful of the uneven quality of the annotation, as some songs were heavily annotated while others are clearly not.

B. Tagtraum Genre Labels

`https://www.tagtraum.com/msd_genre_datasets.html`

This dataset, curated by Henrik Schreiber [8] contains genre annotations for up to 280,831 tracks. The annotations are of the form ROCK, BLUES, etc.

C. The MusiXMatch Dataset

`http://millionsongdataset.com/musixmatch/`

This dataset contains the lyrics of 237,662 tracks in the form of stemmed bag-of-words. There are, however, a variety of factors that limit the number of tracks with lyrical information, including copyright considerations, songs without vocal tracks, etc. It is not clear whether the high degree of missing features would hamper the value of this dataset for our project, but we could assume popular songs should have the lyrical information available.

D. MSD Allmusic Genre Dataset (MAGD)

`http://www.ifs.tuwien.ac.at/mir/msd/`

This dataset provides another single-word genre annotation for up to 422,714 tracks.

E. MSD Acoustic and Spectrogram Dataset

`http://www.ifs.tuwien.ac.at/mir/msd/`

This dataset is actually generated by the same group and in fact is the basis for the MAGD genre classification. In particular, an audio sample of each song is analyzed into a 2D power spectrum that reflects human loudness sensation [7] of dimension $24 \times 60 = 1440$ features (24 bands and 60 modulation frequencies per band). The power spectrum is also further analyzed into 7 features per band, detailing mean, median, variance, skewness, kurtosis, min- and max-value per band. It would be up to us to experiment with this information to perform similarity analysis between songs.

Overall, the metadata that we will consider include: (a) similar tracks, (b) genre classification, (c) lyrics, and (d) spectral information. It would allow us to perform similarity analysis based on metadata.

III. Proposed Methods

We will consider applying the following methods, carefully curated based on our specific problem formulation. That said, we may not know how well each technique would work until we go further into the experiments.

A. Hybrid Neighborhood-based Collaborative Filtering

We will consider both item-based and user-based methods, with adjustments to both the similarity metrics as well as how they should be used as weights, including variants such as Jaccard similarity with an exponentiated normalization that can be tuned as hyperparameters, and another non-linear weighting factor similar to the work done by Aiolli [1]. For example, using Jaccard similarity (1 means having listen to the song at least once, and 0 means never), the weights between items i and j can be calculated as

$$w(i,j) = \left(\frac{|\text{common}(i,j)|}{(|\text{item}(i)|^{\alpha} \times |\text{item}(j)|^{(1-\alpha)}} \right)^{\gamma}$$

where α and γ are hyperparameters to be tuned. The datasets are significantly larger than what many Python recommender systems can handle (e.g. Surprise will fail, since it constructs either MxM or NxN similarity matrices). We will need custom C++ implementation to handle our datasets, which must take advantage of the sparsity of the matrix. For example, if we provide two separate views of the utility matrix, one sorted by U and then I, and one sorted by I and then U, we believe we can calculate the similarity scores with the optimal complexity.

B. Matrix Factorization-based Collaborative Filtering for Implicit Feedback

Unlike collaborative filtering with explicit feedback, implicit feedback such as the number of plays is best modeled as a measure of preference [5]. Songs that have not been played will be assigned a low preference, and as such, technically the utility matrix would be fully populated. This is different from MF with explicit feedback where only explicitly rated user-item pairs show up in the loss function. We will adopt a modified version of the MF formulation to address this problem to avoid long runtime, along the line of Hu et al. [6]. We will also look into the use of logistic matrix factorization for implicit feedback a la Johnson [7] to better model the probabilistic nature of the preference.

Clustering based on Computing Embeddings from Utility Matrix Barkan, Oren, and Koenigstein [2] showed that the word2vec algorithm for NLP can be adapted to extract the latent embedding of songs based on the utility matrix alone, and demonstrated its superiority over standard MF formulation. We believe in the merit of their argument, and would like to consider implementing such an approach. However, we are mindful of the potential pitfall in applying an item-based approach with a similarity metric that measures the distance between embeddings, as a naive implementation would perform O(NxN) comparisons at a runtime complexity of O(NxNxd), where N is close to 1 million, and d (the dimension of the embeddings) is between 5 and 100. This is clearly impractical, and necessitates a clustering-based nearest neighbor approach such as k-mean clustering. For an item-based method we'll only consider items that are in the same cluster, thereby reducing the complexity by a factor of k because each item will be compared against only a fraction of the itemset.

C. Clustering based on Computing Embeddings from Metadata

Instead of computing similarity based on ratings, we can do so based on features extracted from the metadata. We can concatenate the metadata from different analysis – genre (one-hot encoded), sentiment classification from lyrics, and acoustic- and spectral-based based metrics as described earlier. However, we cannot use the vectorized features as-is to compute cosine similarity, because different features may have different meanings. One possibility is to use the vectors but normalize each source. For example, let's say we have three sources – genre from Tagtraum (1 dimension), lyrics from MusiXmatch (10 dimensions), spectral from MAGD (168 dimensions), we'll concatenate the normalized features into a vector of 1+10+168 = 179 dimensions, but weighing the dimensions with three hyperparameters that control the importance of individual dimensions.

D. Neural Ensemble on top of CF or MF recommendations

We are considering the possibility of analyzing under what circumstances would any one of the techniques out-perform others, to better cater to different scenarios (e.g. recommendation to a user who listens to many obscure songs, vs to a user who listens to a few popular songs). One possibility is to train a neural network, or a gradient-boosting decision tree (GBDT) to classify the user based on its embedding and the number of songs played by the user. We would need to sort out how we should approach this problem, either via hyper-parameter tuning or a simple neural network-based regression or classification. Another possibility is a simple ensemble method aggregating the recommendations from different techniques separately, and make a meta-recommendation prioritized by adding up the normalized score from each recommendation.

E. More Sentiment Analysis on Lyrics and Acoustic and Genre Metadata

We believe good embedding may hold the key to superior recommendations because of the sparsity of the utility matrix. It may be hard to concoct similarity information from user neighborhoods due to almost non-existent common songs outside of the popular ones. Perhaps it would be more promising to look at songs the user already likes, and find songs that share characteristics with his/her own songs. For example, if a user likes a certain sappy love song about breakup, she may like another sappy love song about breakup. Therefore, we plan to look deeper into sentiment analysis based on lyrics, tempo, and genre information. While we did mention clustering based on computing embeddings from metadata already, we feel this may fall under the "deeper analysis" category [4].

REFERENCES

[1] AIOLLI, F. Efficient top-n recommendation for very large scale binary rated datasets. In *Proceedings of the 7th ACM conference on Recommender systems* (2013), pp. 273–280.

[2] BARKAN, O., AND KOENIGSTEIN, N. Item2vec: neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)* (2016), IEEE, pp. 1–6.

[3] BERTIN-MAHIEUX, T., ELLIS, D. P., WHITMAN, B., AND LAMERE, P. The million song dataset.

[4] CHEN, X., AND TANG, T. Y. Combining content and sentiment analysis on lyrics for a lightweight emotion-aware chinese song recommendation system. In *Proceedings of the 2018 10th International Conference on Machine Learning and Computing* (2018), pp. 85–89.

[5] HU, Y., KOREN, Y., AND VOLINSKY, C. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining* (2008), Ieee, pp. 263–272.

[6] JOHNSON, C. C. Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems 27*, 78 (2014), 1–9.

[7] LIDY, T., AND RAUBER, A. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *ISMIR* (2005), pp. 34–41.

[8] SCHREIBER, H. Improving genre annotations for the million song dataset. In *ISMIR* (2015), pp. 241–247.