The Importance of Song Context in Music Playlists

Enabling Recommendations in the Long Tail

Andreu Vall, Markus Schedl, Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz, Austria
andreu.vall@jku.at

Massimo Quadrana, Paolo Cremonesi
Politecnico di Milano, Italy
massimo.quadrana@polimi.it

ABSTRACT

Music recommender systems often operate in sequential mode by suggesting a collection of songs that constitute a listening session. This task is usually called *automated music playlist generation* and it has been previously studied in the literature with different successful approaches based on, e.g., variations of collaborative filtering or content-based similarity. Some of the proposed playlist models take into consideration the current song and a number of previous songs, i.e., the *song context*, in order to predict the next song. However, it is not yet clear to what extent knowing this song context improves next-song predictions. To shed light on this question, we conduct a numerical experiment on two datasets of hand-curated music playlists, where we compare playlist models that account for different song context lengths. Our results indicate that knowing the song context seems, at first, uninformative. However, we explain this effect by a strong bias in the data towards very popular songs and observe that, in fact, songs in the long tail are more accurately predicted when the song context is considered.

CCS CONCEPTS

•Information systems →Data mining; Recommender systems;

KEYWORDS

music playlist generation, music recommender systems, recurrent neural networks

ACM Reference format:
Andreu Vall, Markus Schedl, Gerhard Widmer and Massimo Quadrana, Paolo Cremonesi. 2017. The Importance of Song Context in Music Playlists. In *RecSys 2017 Poster Proceedings, Como, Italy, August 27–31.*

1 MODELING MUSIC PLAYLISTS

We describe the playlist models considered in our experiment. We generally assume that: two sets of playlists are available, one for training and one for test; given the current song and a number of previous songs from a test playlist, a playlist model has to be able to rank all next-song candidates according to how likely they are to be the next song in that playlist. Note that we refer to the current and the previous songs in the playlist as the *song context* as it is commonly done in language models, but this should not be confused with the incorporation of general contextual information into recommendation systems.

1.1 Song Popularity

This is a unigram model that computes the popularity of a song according to its relative frequency in the training playlists. At test time, the next-song candidates are ranked by their popularity, regardless of the current and previous songs in the playlist.

1.2 Song-based Collaborative Filtering

This is an item-based Collaborative Filtering (CF) model. A song s is represented by the binary vector $\mathbf{p}_s$ indicating the playlists to which it belongs. The similarity of each pair of songs s_i, s_j in the training set is computed as the cosine between $\mathbf{p}_{s_i}$ and $\mathbf{p}_{s_j}$. At test time, the next-song candidates are ranked according to their similarity to the current song, but previous songs in the playlist are ignored.

1.3 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of neural network models particularly suited to processing sequential data. They have a hidden state that accounts for the input at each time step while recurrently incorporating information from previous hidden states.

We adopt the approach proposed in [2], where an RNN model with one layer of gated recurrent units is combined with a loss function designed to optimize the ranking of next-item recommendations. At test time, given the current and all the previous songs in the playlist, the RNN outputs a vector of song scores that is used to rank the next-song candidates.

2 DATASETS

The "AotM-2011" dataset [3] is a playlists collection derived from the Art of the Mix[1] database. The "8tracks" dataset is a private playlists collection compiled from 8tracks,[2] an on-line platform where users can share playlists and listen to other users' playlists.

We keep only the playlists with at least 3 unique artists and with a maximum of 2 songs per artist. This is to discard artist- or album-themed playlists, which may correspond to a less careful compilation process. We also keep only the playlists with at least 5 songs. Songs occurring in less than 10 playlists are removed to ensure that the models have sufficient observations for each song.

We randomly assign 80% of the playlists to the training set and the remaining 20% to the test set. As in any recommendation task blind to item content, the songs that occur only in test playlists need to be removed because they can not be modeled at training time. This affects the final playlist length and song frequency.

The filtered AotM-2011 dataset has 17,178 playlists with 7,032 songs by 2,208 artists. The filtered 8tracks dataset has 76,759 playlists

[1] www.artofthemix.org
[2] https://8tracks.com

Table 1: Descriptive statistics for the AotM-2011 and the 8tracks datasets. "Song popularity" corresponds to the song frequency in the dataset, i.e., the number of playlists in which each song occurs.

dataset	statistic	min	1q	med	3q	max
AotM-2011	Songs per playlist	5	6	7	8	34
	Artists per playlist	3	5	7	8	34
	Song popularity	1	8	12	20	249
8tracks	Songs per playlist	5	5	6	7	46
	Artists per playlist	3	5	6	7	41
	Song popularity	1	9	15	30	2,320

with 15,649 songs by 4,290 artists. Table 1 reports the distribution of unique songs per playlist, unique artists per playlist and song popularity in the datasets.

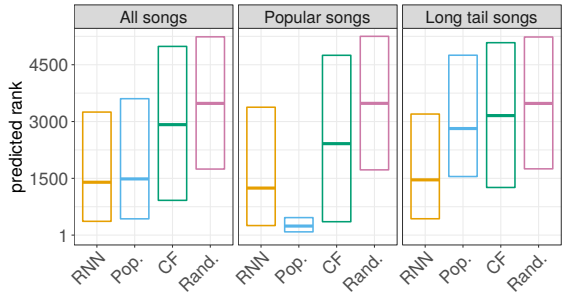
3 EVALUATION

The evaluation is based on the prediction of next songs, considering all the songs in the dataset as next-song candidates. Given a trained model, the following procedure is repeated over all the test playlists. We show the model the first song in a playlist. The model then has to rank all the next-song candidates according to their likelihood to be the second song in that playlist. We only keep track of the rank assigned to the actual second song. We then show the model the first and the second actual songs. The model has to rank all the next-song candidates for the third position, having now a longer song context. In this way, we progress until the end of the playlist, always keeping track of the rank assigned to the actual next song. Finally, the performance of a playlist model is characterized by the distribution of its predicted ranks for the actual next songs.

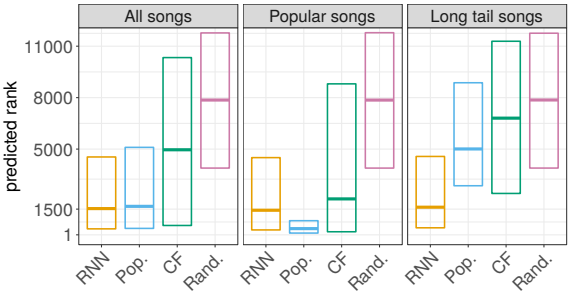
Hyperparameter tuning is performed on a validation split. Along with the described models, we also consider a random model that assigns scores to songs uniformly at random, yielding random ranks.

4 RESULTS

Considering the ranks achieved for all next-song predictions (left panels in Figure 1), the RNN and the popularity-based model perform comparably well, and significantly better than the song-based CF model. This is a surprising result given the simplicity and null song context of the popularity-based model, but might be explained by the impact of a small number of very popular songs in both datasets (Table 1). To investigate this effect, we consider the ranks achieved when the actual next songs belong to the 10% most popular songs in the dataset (central panels in Figure 1), and when the actual next songs belong to the long tail of 90% least popular songs in the dataset (right panels in Figure 1). As expected, the popularity-based model performs well on the most popular songs, but is close to the random model for songs in the long tail. Thus, the overall performance of the popularity-based model is similar to that of the RNN model only because the datasets are biased towards popular songs. The song-based CF model is not competitive in this experiment and, especially in the 8tracks dataset, is also affected by the bias towards popular songs. In contrast, the performance of the RNN model, which keeps track of the full song context, is robust to the popularity of the actual next song.



(a) AotM-2011 dataset



(b) 8tracks dataset

Figure 1: Distribution of predicted ranks for the actual next songs (lower is better). The boxplots indicate the first quartile, median and third quartile ranks. Left: All songs are considered. Center: Only the 10% most popular songs in the dataset are considered. Right: Only the 90% least popular (long tail) songs in the dataset are considered. "Pop." "CF" and "Rand." correspond to the popularity-based, the song-based CF and the random models, respectively. The scale of the y-axis relates to the number of songs in each dataset.

In our view, this is a remarkable finding given that the bias towards popular songs is a characteristic feature of playlist datasets in the music recommendation domain [1]. Playlist models that account for a longer song context are therefore better suited to next-song prediction for songs in the long tail.

5 CONCLUSION

In this work we investigated the importance of the song context for next-song recommendations in music playlists. Our results indicate that the bias towards very popular songs masks the importance of the song context. However, we observe that a playlist model based on an RNN, which considers the full song context of a playlist, clearly outperforms simpler models when recommending songs belonging to the long tail of non popular songs.

6 ACKNOWLEDGMENTS

We thank Matthias Dorfer, Bruce Ferwerda, Rainer Kelz, Filip Korzeniowski, Rocío del Río and David Sears for helpful discussions.

REFERENCES

[1] Òscar Celma. 2010. *Music recommendation and discovery.* Springer.
[2] Balázs Hidasi et al. 2016. Session-based recommendations with recurrent neural networks. In *Proc. ICLR.*
[3] Brian McFee and Gert Lanckriet. 2012. Hypergraph models of playlist dialects. In *Proc. ISMIR.* 343–348.