Improving Music Recommendation in Session-Based Collaborative Filtering by using Temporal Context

Ricardo Dias, Manuel J. Fonseca
Department of Computer Science and Engineering
INESC-ID/IST-Technical University of Lisbon
Lisbon, Portugal
ricardo.dias@ist.utl.pt, mjf@inesc-id.pt

***Abstract*—Music recommendation systems based on Collaborative Filtering methods have been extensively developed over the last years. Typically, they work by analyzing the past user-song relationships, and provide informed guesses based on the overall information collected from other users. Although the music listening behavior is a repetitive and time-dependent process, these methods have not taken this into account and only consider user-song interaction for recommendation. In this work, we explore the usage of temporal context and session diversity in Session-based Collaborative Filtering techniques for music recommendation. We compared two techniques to capture the users' listening patterns over time: one explicitly extracts temporal properties and session diversity, to group and compare the similarity of sessions; the other uses a generative topic modeling algorithm, which is able to implicitly model temporal patterns. We evaluated the developed algorithms by measuring the Hit Ratio, and the Mean Reciprocal Rank. Results reveal that the inclusion of temporal information, either explicitly or implicitly, increases significantly the accuracy of the recommendation, while compared to the traditional session-based CF.**

***Index Terms*—Music Recommendation, Collaborative Filtering, Temporal Context, Listening Sessions**

I. Introduction

Over the last years, the increasingly easy access to online services for listening to music, made it possible for users to have access to millions of songs on the Internet. However, when dealing with this overwhelming number of songs to choose from, they often become paralyzed and doubtful (*The Paradox of Choice* [1]). To reduce the burden of choosing among too many items, researchers have focused on creating recommender systems that can automatically generate recommendations that fit users' preferences.

According to Celma [2], recommender systems can be classified into five categories: *Demographic Filtering, Collaborative Filtering, Context-Aware Recommendation, Content-Based Filtering* and *Hybrid Methods*. Though the most thoroughly studied and successful algorithms have been based on Collaborative Filtering (CF) methods, recently, Context-Aware recommender systems (CARS) have become more popular as researchers agree that music listening patterns can be influenced by contextual factors, such as temporal properties, the activity a listener is engaged in or the emotion one is feeling [3]. These studies found that listeners tend to prefer to listen to certain artists and genres at specific moments of the day and at certain days of the week, showing that time-contextualized music recommendation can be successful. However, typical CF methods do not exploit contextual factors, such as temporal properties [4], [5], nor the fact that users tend to listen to songs that fit well together [6]. Nevertheless, in [7], Park presented a modified User-based CF method, called *Session-Based CF* (SSCF), that uses session information to capture sequence and repetitiveness in the listening process. In [8], Zheleva et al. developed a session-based hierarchical graphical model using *Latent Dirichlet Allocation* (LDA), and showed that their model could facilitate playlist completion based on previous listening sessions or songs that the user has just listened to. However, none of these solutions used information from the session itself nor from the sequence of played songs.

In this paper, we explore the usage of temporal context and session diversity in Session-based CF algorithms, by comparing two algorithms. One with explicit characterization of sessions (TSSCF), that models the listening patterns over time, by grouping similar sessions based on *temporal properties* (such as, the time of the day, the weekday, the day of the month and the month) and a *diversity feature* (ratio between number of different songs and total songs played in that session) using an unsupervised clustering algorithm (Expectation Maximization).

To evaluate these solutions, we compared them with typical CF techniques, a simple Session-Based CF algorithm (based on the work developed in [7]), and also to a random algorithm. The experimental evaluation consisted in predicting the next song to be played in the current active session, measuring the *Hit Ratio* (HR@n) [9] and the *Mean Reciprocal Rank* (MRR). The results show a good accuracy improvement for the two algorithms, proving that CF methods can take advantage of temporal properties and diversity to improve and personalize recommendations. The LDA-based technique outperformed the other solutions, suggesting that implicit characterization of sessions might provide better information for the recommendation algorithm than an explicit description.

II. Related Work

Typical CF methods work by looking into past user listening behavior, and recommending songs to a user based on the ratings provided by other users with similar taste [10] (*User-based*). They look into the set of songs that a user has rated, and compute the similarity among the target song to decide whether is worth to recommend it or not. Various social websites and online radios, including *Last.fm*, use this type of collaborative filtering. However, CF methods require many users and ratings to work, being unable to recommend songs without ratings, or recommend songs to users that have not rated anything yet. Another problem with these approaches is that only user interaction is used to generate recommendations.

Context aware recommendation systems (CARS) use *context information* [11] to describe and characterize the songs or artists we listen. For example, Su et al. [12] improved CF methods combining user grouping by location, motion, calendar, environment conditions and health conditions, while using content analysis to assist the system to select appropriate songs. On the other hand, Park [7] developed a Session-based CF, where session profiles are used for recommendation, adding a temporal dimension to CF recommendations. In [4], Lee presented an approach that combined temporal features, such as, the *Season, Weekday, Month, Weather and the Temperature*, with *case-based reasoning* to infer the *mood* of the user and recommend music. In [5], Liu et al. took the change in the interests of users over time into account by adding time scheduling to the music playlist. Baltrunas et al. [13] introduced a new context-aware approach called user micro-profiling, where the user profile is split into several sub-profiles, each one representing the user in a particular context. These sub-profiles model the choice of songs based on contextual conditions, such as, the time of day, mood or the current activity listeners perform. In [8], the authors presented a novel and improved statistical model for characterizing user preferences in consuming social media content. By taking into account information about listening sessions of individual users, they have arrived at a new session-based hierarchical graphical model that enhanced individual user experience, for example, by recommending songs for the user's playlists that would be relevant for the current music listening session.

In short, although there have been some effort in developing techniques that capture temporal context, few work has been done to study its usage combined with Session-based CF techniques, requiring further research on this subject. Moreover, despite the works developed so far take advantage of the songs listened in sessions and of the implicit temporal context they provide, none of them used temporal and diversity features extracted from the sessions to personalize recommendations.

III. Applying Temporal Context on Session-based Collaborative Filtering Algorithms

Music recommendation systems relying on *content-based* methods or *collaborative filtering*, rarely take *time* into account, something that should not be neglected when dealing with personal preferences [5]. Recently, in CF techniques, latent factor models have gained more attention over neighborhood methods because they offer high expressive ability to describe various aspects of the data [14] (including time), and therefore provide more accurate results than neighborhood models. However, neighborhood methods prevail mainly because of their relative simplicity: they provide intuitive explanations of the reasoning behind recommendations, and can provide immediate recommendations based on newly entered user feedback. Based on these facts, we decided to explore neighborhood methods as base algorithm for recommendation.

A. Session-based Collaborative Filtering

Traditional CF approaches do not incorporate temporal context, and use the full listening history of a user to perform recommendations. Park et. al overcame this by creating a Session-based CF (SSCF) [7], where the authors added temporal context to CF techniques by using *sessions* profiles instead of user profiles for generating recommendations

To formalize, a user listening history H can be defined as a sequence of tuples H_k, where each tuple is composed of a *user identifier*, a *song played* and a *timestamp*. Each session is composed by a set of tuples from the listening history within a given *time frame*. Sessions are non-overlapping subsets of tuples. All songs played at the same session are grouped together, and the songs and their playing frequencies become the *profile* of the session. Assuming that there are *n* songs and *m* sessions, we can define a set of songs as $I = \{i_1, i_2, ..., i_n\}$ and a set of sessions $S = \{s_1, s_2, ..., s_m\}$. We view each session profile *s* as pairs of songs and their listening frequencies. Based on this data, in the CF *recommendation matrix* rows represent *session profiles*, and the columns *songs*. Each cell contains the *frequency* a song was played in that session.

Finally, predicted *ratings* for new songs i in the *active session* s are determined as shown in Equation 1 [7] (similar to the predicted rating formula for User-based CF [2]).

$$\widehat{r}_{s,i} = \bar{r}_s + \frac{\sum_{v \in S(s)^k} sim(s,v)(r_{v,i} - \bar{r_v})}{\sum_{v \in S(s)^k} sim(s,v)} \quad (1)$$

$S(s)^k$ represent the k session neighbors for the active session s, $\bar{r}_s$ and $\bar{r}_v$ represent the average rating for sessions *s* and *v* under comparison. $r_{v,i}$ represents the frequency a song i was played in session v, and $sim(s,v)$ the similarity between sessions s and v (applying Cosine Similarity [15]).

B. Temporal Session-based Collaborative Filtering

Although the SSCF technique takes advantage of temporal context by using session profiles, it does not use temporal properties about the sessions. These properties could be useful for modeling implicit listening patterns and habits based on time. Therefore, besides using information about which songs were played in sessions (and their frequencies), we went further by making sense of temporal properties of sessions for determining session similarity. In particular, to characterize each session, we created a *feature vector* consisting of 5 properties. These features encode *temporal properties* of a session and also the *diversity* of the songs played in that

session. Although these features were mainly chosen based on the work presented in [4], [13], [3], [16], none of these features were used on previous works to determine *session similarity* for music recommendation. Here is a description of each feature, and its rationale:

- **Time of day**: captures the fact that users listen to different songs (and styles), at different periods of the day (e. g., at morning, afternoon, or at night)
- **Weekday**: users listen to different songs at different week days, for example, during a working day or at the weekend, while relaxing
- **Day of Month**: users listen to different songs regarding the day of the month, for example listen to more happy music in the beginning of the month, and sad at the end
- **Month**: captures the fact that users listen to different songs in different months, perhaps associated with seasons or work phases
- **Song Diversity**: this is a new feature, developed by us, that measures the ratio of different songs played in a session and the total songs played. It tries to capture the tendency of song repetition in a session (1 - all the songs where different; approximately 0 - played repeatedly the same songs) [17]

After extracting these features from sessions, we needed a new similarity metric to compare them. Based on our assumption that these features could be used to model listening patterns over time, we decided to use an unsupervised *clustering* mechanism to group similar sessions [18]. By using clustering we are grouping sessions according to the temporal variance in listening habits. Due to the nature of the data, we opted to apply *fuzzy clustering*, where objects can belong to more than one cluster, having a degree of membership for each cluster, as in fuzzy logic [19]. Clustering is performed by training a *Gaussian Mixture Model* using *Expectation Maximization algorithm* [20], with default parameters, and the number of clusters as detailed in Section IV. As a result, to each session is assigned a probability distribution vector, with as many values as the number of clusters, indicating the degree of membership to each cluster.

Since sessions are now described as *probability distribution vectors*, we can infer similarity between them by comparing the vectors. Sessions with *higher* probabilities for the *same clusters* will be more similar than those with lower values. We use *Kullback-Leibler* divergence [21] to compare the sessions probability distributions.

Finally, this similarity metric can be used in the predicted rating formula (see Equation 1) to determine how likely a song will fit in the current active session.

C. LDA-based Collaborative Filtering

LDA is a generative model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. In LDA, each document is considered as a collection of words, and it is assumed that each document is represented as a mixture of *latent topics*, and each topic has probabilities of generating various words [22]. It has been widely used in topic modeling domain, and more recently in music recommendation [23], [24], [8].

In this work, we use LDA with the previously defined session data model, to infer session similarity and use it in the predicted rating formula (see Equation 1) to determine how likely a song will fit in the current active session. Moreover, our objective is to use LDA to implicitly extract temporal properties from sessions, by infering topic distributions for the sessions. To this extent we modeled *sessions* as *documents* and *songs* as *words*. Similarly to Temporal SSCF, as each session is now described as a topic distribution, we can infer similarity between them, by comparing the topic distribution vectors. Sessions with *higher* values for the *same topics* will be more similar than those with lower values. We also use the *Kullback-Leibler* divergence [21] to compare the sessions topics distributions.

IV. Experimental Evaluation

To evaluate the two algorithms, we compared them with typical CF approaches (Item, User and Session-based [7]), and also with a random recommender, by measuring *HR@n* and *MRR*.

In the following sections we describe the conducted evaluation, presenting the objectives, the dataset used and the procedure for evaluating the algorithms.

A. Dataset

We used the *Last.fm dataset*[2] as input data to evaluate all the techniques. This log contains *19,150,868* entries for 992 users, each entry containing a user, artist and song id, an artist and song name, and a timestamp indicating when the song was listened to. This data encodes information from *176,948* unique artists and *1,500,661* unique songs.

B. Session Generation

Sessions have been widely used and explored in web mining research [25], and recently applied to the analysis of music listening histories [8], [7], because they capture the sequence and order of users' listening preferences over time.

Though there are different techniques for session generation [25], we decided to use the time gap approach, because in our dataset we only have the timestamp indicating when a song was listened to. We set up different thresholds based on related work [25], [7] and studied their suitability for session identification. Sessions were generated using threshold values between 10 and 30 minutes, with intervals of 5 minutes. Overall, each resulting dataset contained more than one million sessions, with an average number of 15 songs per session.

C. Procedure and Objectives

1) ***Evaluation Measures****:* To evaluate the accuracy of the different recommendation algorithms, we measured the Hit Ratio (*HR@n* [9]). *HR@n* indicates whether if the desired items appear on the generated recommendation lists (*presence*), and how many times they appear (*frequency*).

Additionally, we also measured the *Mean Reciprocal Rank* (MRR) to evaluate the generated recommendation lists, in

terms of the *rank* where the intended songs appear. Notice that this metric is quite sensitive to the rank values and as a sharp decay. For example, in a set of 1000 queries, with a list size of 100, if on average the desired items were found in the list at position 10, MRR would be 0.1. However, if the items were found at position 50, MRR would be 0.02.

2) ***Parameters****:* In the neighborhood algorithms, the number of similar neighbors k is essential, and it represents the constraint number of similar users/items/sessions that are found using a similarity function. We adopted the values presented in [7] for the parameter k, namely, $k \in \{30, 50, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000\}$.

Another parameter for the experiments is the number of clusters c used for training the *EM* algorithm. We used cross-validation to find out the best suitable number of clusters c for the session datasets, identifying *8* as the best value.

Finally, the last parameter is the number of topics t to be used in the *LDA-based* recommender. We used Gibbs sampling for determining the most suitable value for the number of topics. Here we also found *8* to be the best value.

3) ***Experiment****:* In this evaluation, we intended to measure if we could predict the next song to be played in the current *active session*, based on what users have previously listened in that session. To measure the *HR@n* and *MRR*, we randomly selected 1000 sessions from the overall dataset of sessions to be used as *queries* (active sessions). Sessions with less than two songs were discarded. Next, we removed the last song from each session and executed all the algorithms to obtain the *top-100* recommendations (*n*). For these recommendations, we registered the *presence* and the *position* (rank) given by the algorithms for the last items removed from sessions. These results were then used to compute the two metrics.

To be able to collect enough data for performing statistical comparisons, we executed this experiment 20 times, randomly choosing another set of 1000 queries each time.

V. Results and Discussion

In this section we summarize and discuss the results obtained from the experimental evaluation. Notice that we omit results for User and Item-based CF recommenders, and also for the Random technique, because their results were not relevant for the evaluation. Moreover, because we could not assume data normality in all the cases, we applied non-parametric statistical tests when required. The Bonferroni approach was applied in pairwise comparisons for controlling Type I error across tests when necessary.

A. Overall Results

In short, overall results regarding HR (see Figure 1-top), evidence that temporal context improved recommendation accuracy, with both algorithms (TSSCF and LDA-based technique) outperforming the SSCF technique. Despite that, the LDA-based algorithm outperformed the TSSCF solution. However, in terms of MRR (see Figure 1-bottom), the base SSCF algorithm performed better than the other two.

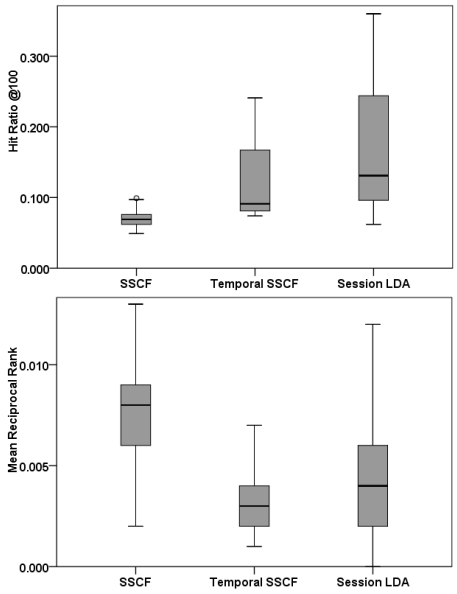

Fig. 1. Overall HR (top) and MRR (bottom) values.

To validate these observations we performed two Kruskal-Wallis tests. Regarding HR values we report that there was a statistically significant difference ($H(2) = 1560.115, P < 0.01$), with a mean rank of 598.58 for SSCF, 1599.34 for Temporal SSCF, and 2026.28 for the LDA-based technique. In terms of MRR, significant differences were also found ($H(2) = 1064.659, P < 0.01$) among the techniques, with mean rank of 2070.78 for SSCF, 912.08 for Temporal SSCF and 1188.72 for the LDA-based technique.

B. Session Threshold Comparison

To observe the impact of the session generation mechanism, we evaluated the algorithms with the 5 session datasets obtained using different threshold values (see Section IV-B).

Overall results for each dataset are depicted in Figure 2. These plots confirm the previous results, that the LDA-based recommender outperforms both the TSSCF and SSCF techniques in terms of HR across all datasets, but that SSCF performs better in terms of MRR. We observe that the TSSCF algorithm is not sensitive to the dataset used regarding HR, and that MRR is also constant, with better values for small and larger threshold values. SSCF reports increasingly higher values of HR and MRR when the threshold increases. LDA-based algorithm performs better for smaller values of the threshold used, decreasing with higher values, both for HR and MRR (these with some oscillations).

Moreover, Figure 2 also indicates that the threshold value that leads to better HR and MRR results is between 15 and 20. To be able to select the best number of neighbors, we conducted *Mann-Whitney U Tests* to choose a dataset to be used as comparison basis. We did not find any significant differences between using $th = 15$ or $th = 20$ in terms of HR ($z = -0.293, p > 0.770$), but for MRR values we found a significant difference between using $th = 15$ or

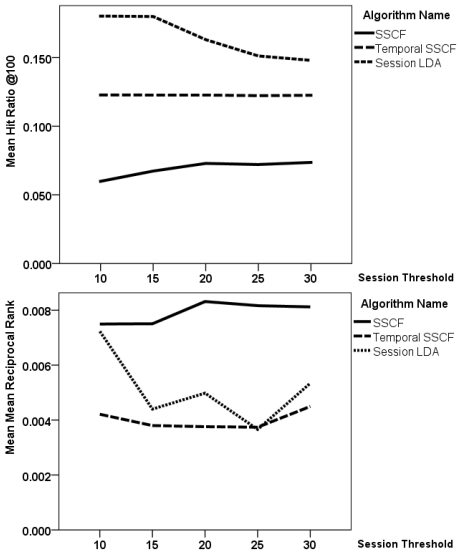

Fig. 2. Session threshold impact in HR (top) and MRR (bottom).

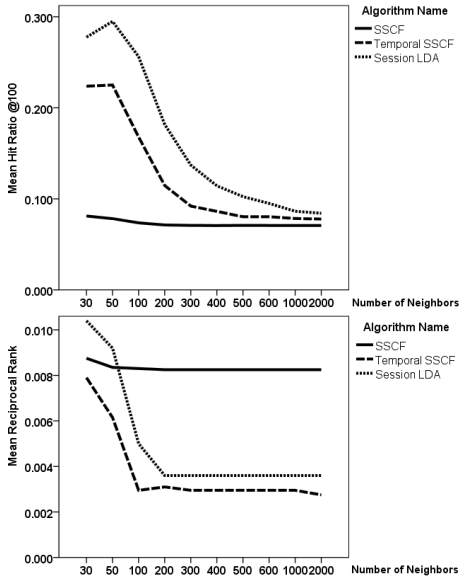

Fig. 3. Impact of the number of neighbors in HR (top) and MRR (bottom) ($th = 20$).

$th = 20$ ($z = -2.356, p < 0.05$). Based on this result, all the following comparisons were performed using the session dataset generated with threshold $th = 20$.

C. Neighborhood Comparison

Figure 3 depicts the values of HR and MRR of each algorithm evaluated with the dataset generated with session threshold *20*, for selecting the best number of neighbors (*k*). We can observe that small values of *k* lead to better values of HR and MRR. Indeed, *30* and *50* are the best values for *k*, within the three algorithms. Increasing *k* leads to smaller values of HR and MRR for all the algorithms, although this decay is much more evident in context aware algorithms.

Moreover, notice that although SSCF outperforms both the TSSCF and the LDA-based technique in terms of MRR for $k > 100$, MRR values are small (less than 0.01) for any value of *k*. We discuss this later in Section V-E.

We also tested with smaller values of neighbors ($k < 30$), however, both HR and MRR results were worse than using $k > 30$ and therefore they were not considered for the evaluation.

D. Best case configuration

The best configuration for each algorithm is achieved by using a session threshold $th = 20$, and the number of neighbors $k = 30$. Although the LDA-based algorithm achieves better HR values with $k = 50$, we also used $k = 30$, because it represents a trade-off between HR and MRR values for this algorithm. To observe if the differences between the best setup for each algorithm were statistically relevant (see Figure 4), we conducted Kruskal-Wallis Tests between the three algorithms using $k = 30$. For MRR, we report that there was no statistically significant difference between the values given by the three algorithms ($H(2) = 2.294, P > 0.318$), but there was a statistically significant difference between the HR values of the algorithms ($H(2) = 52.52, P < 0.01$) with

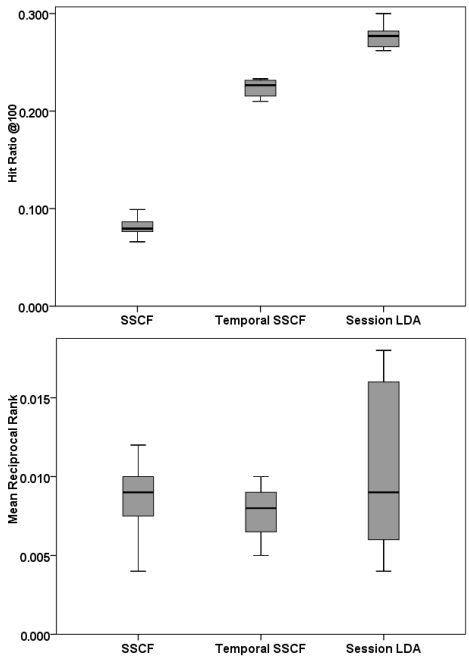

Fig. 4. Comparing the three algorithms using $th = 20$ and $k = 30$.

a mean rank of 10.50 for SSCF, 30.5 for TSSCF and 50.5 for the LDA-based technique.

E. Discussion

Results presented in the previous section allow us to conclude that temporal context improves recommendation accuracy in Session-based CF. Despite that, the LDA-based algorithm outperformed TSSCF in both metrics. Regarding MRR, although we did not find a statistical difference, the results evidence that the context aware algorithms generate

recommendation lists with the most probable songs in the bottom of the lists instead of the top.

Different threshold values for generating sessions (using the time gap approach) allowed us to analyze the impact of this component in the recommendation process. This evaluation end up showing that explicit characterization of temporal context leads to a stable algorithm (TSSCF), which outperforms the simple SSCF approach. Nevertheless, a threshold value of *20* lead the remaining approaches to better results, and therefore was the best value for evaluating other solutions.

The number of neighbors (k) used in CF algorithms usually have an high impact on the accuracy of recommendation systems. Results obtained for all metrics indicate that smaller values of k (30) can improve recommendation accuracy. In fact, regarding HR values, TSSCF and LDA-based algorithms obtained the double / triple of the SSCF algorithm (0.2 and 0.3 compared to 0.1). However, MRR values are small (less than 0.02), showing that all the algorithms generate recommendation lists, where the "desired" items are in the bottom instead of the top. Notice the particular behavior of MRR with its sharp decay, and that the accuracy results also influence the MRR, because missing hits are used in calculations. For example, lets take into consideration the best configuration for all the algorithms (see Figure 3, with $k = 30$). TSSCF achieved an average value of 0.220 in HR and 0.008 in MRR, and although the MRR value is less than 0.1, the average rank position of the required items in the recommendation lists is *28* out of 100 ($0.008 = ((1/28) * 220)/1000$). For the LDA-based algorithm the average rank is *26* and for SSCF is *11*. It is clear that SSCF has less hits than temporal context aware algorithms, however, it outputs the "desired" items with higher ranking values. This behavior is due to the variability of the ranking positions obtained during the experiments. Further evaluation is required to analyze this behavior.

VI. Conclusions and Future Work

In this paper we explored the usage of temporal context and session diversity in Session-based CF recommendation algorithms, by comparing two algorithms that used temporal features: one with explicit description of the features and the other with implicit extraction from the data. Results showed an improvement of 200% in accuracy over typical Session-based CF of both algorithms developed, and revealed that the explicit characterization of temporal features led to an algorithm independent of the session generation mechanism. Regarding the number of neighbors, results also indicated that with small values of neighbors ($k = 30$), Session-based CF techniques can achieve better accuracy values. However, results show that more research must be conducted in order to determine the factors that affect the ranking in the recommendation list, that lead to small values of MRR (< 0.1).

As for future work, we intend to follow two paths: the first one is to combine the temporal context with other context features (such as, the location of the users, or the activities they are performing, etc.) to better describe the variability and diversity of sessions; the second is to conduct a user study to evaluate the performance of temporal context aware algorithms in a real-world application and use feedback from users to improve recommendation.

VII. Acknowledgements

This work was supported by national funds through FCT –Fundação para a Ciência e a Tecnologia, under project PEst-OE/EEI/LA0021/2013. Ricardo Dias was supported by FCT, grant reference SFRH/BD/70939/2010.

References

[1] B. Schwartz, *The Paradox of Choice: Why More Is Less*. Harper Perennial, 2005.
[2] O. Celma, *Music Recommendation and Discovery*. Springer, 2010.
[3] P. Herrera, "Rocking around the clock eight days a week: an exploration of temporal patterns of music listening," in *WOMRAD*, 2010.
[4] J. Lee, "Music for my mood: A music recommendation system based on context reasoning," *Smart Sensing and Context*, 2006.
[5] N. Liu, S. Lai, and C. Chen, "Adaptive music recommendation based on user behavior in time slot," *JCSNS*, 2009.
[6] D. Hansen and J. Golbeck, "Mixing it up: recommending collections of items," in *Proc. of CHI*, 2009.
[7] S. E. Park, S. Lee, and S.-g. Lee, "Session-Based Collaborative Filtering for Predicting the Next Song," *Proc. of CNSI*, 2011.
[8] E. Zheleva, J. Guiver, E. Mendes Rodrigues, and N. Milić-Frayling, "Statistical models of music-listening sessions in social media," *Proc. of World Wide Web*, 2010.
[9] D. Lee, S. E. Park, M. Kahng, S. Lee, and S.-g. Lee, "Exploiting Contextual Information from Event Logs for Personalized Recommendation," *Computer and Information Science*, pp. 121–139, 2010.
[10] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, J. Riedl, and H. Volume, "Grouplens: Applying collaborative filtering to usenet news," *Communications of the ACM*, vol. 40, pp. 77–87, 1997.
[11] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," ser. HUC '99. Springer-Verlag, 1999.
[12] J. Su, H. Yeh, and P. Yu, "Music recommendation using content and context information mining," *Intelligent Systems, IEEE*, vol. 25, no. 1, 2010.
[13] L. Baltrunas, X. Amatriain, and V. Augusta, "Towards Time-Dependant Recommendation based on Implicit Feedback," in *CARS*, 2009.
[14] S. Rendle, "Factorization machines with libfm," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, pp. 57:1–57:22, May 2012.
[15] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.
[16] Y. Hu and M. Ogihara, "Nextone Player: A Music Recommendation System Based on User Behavior," in *Proc. of ISMIR*, 2011.
[17] C. A. Russell and S. J. Levy, "The temporal and focal dynamics of volitional reconsumption: A phenomenological investigation of repeated hedonic experiences," *Journal of Consumer Research*, vol. 39, no. 2, pp. 341–359, 2012.
[18] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. MIT Press, 2010.
[19] D. Man and M. Toa-Abrudan, "Fuzzy clustering algorithms: A survey," in *7th Joint Conference on Mathematics and Computer Science*, 2008.
[20] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, vol. 39, 1977.
[21] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 49–86, 1951.
[22] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
[23] R. Cai, C. Zhang, C. Wang, L. Zhang, and W.-Y. Ma, "Musicsense: contextual music recommendation using emotional allocation modeling," in *Proc. of ACM MM*, 2007.
[24] G. Sharma and M. N. Murty, "Mining Sentiments from Songs Using Latent Dirichlet Allocation," *LNCS 7014*, pp. 328–339, 2011.
[25] Z. Pabarskaite and A. Raudys, "A process of knowledge discovery from web log data: Systematization and critical review," *Journal of Intelligent Information Systems*, vol. 28, 2007.