Mood-Aware Music Recommendation via Adaptive Song Embedding

Chaima DHAHRI, Kazunori MATSUMOTO and Keiichiro HOASHI
KDDI Research, Inc 2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502 Japan
Email: {ch-dhahri, matsu, hoashi}@kddi-research.jp

***Abstract*—In this paper, we propose an *autonomous* and *adaptive* recommendation system that relies on the user's mood and implicit feedback to recommend songs without any prior knowledge about the user preferences. Our method builds autonomously a latent factor model from the available online data of many users (generic song map per mood) based on the associations extracted between user, song, user mood and song emotion. It uses a combination of the Reinforcement Learning (RL) framework and Page-Hinkley (PH) test to personalize the general song map for each mood according to user implicit reward. We conduct a series of tests using LiveJournal two-million (LJ2M) dataset to show the effect of mood in music recommendation and how the proposed solution can improve the performance of music recommendation over time compared to other conventional solutions in terms of hit rate and F1 score.**

I. Introduction and Previous Works

Recently, researchers have been focused on exploring user emotion in music recommendation and have shown the importance of user's mood in improving the performance of context aware recommendation systems (CARS). Many mood-based music recommendation services have emerged in the past years; Moodplay [1], GEMRec [2], [3] and [4], to name a few. However, these previous methods usually rely on user's input about their mood or their preferred song/artist to generate personalized recommendations. The authors in [3], require a rating matrix issued by users for items. This type of recommendation system suffers from the amount of noise introduced by the user. Since users are inconsistent in giving their feedback, the effectiveness of such systems is questionable. Besides, rating songs poses additional burden for users which makes them unsatisfied with the system. The model of Moodplay [1] enables the change of user's preference by enabling the user to move his Avatar within the generated graph manually. The authors in [2] proposed an emotion-aware graph to model the association between the user, music and the emotion using acoustic features. They use a random-walk to recommend songs based on real-time emotion of the user. The proposal in [2] is not personalized; the recommendations are done based on a general graph for all users. Autonomous context-aware recommendation systems were also studied in the literature where user's mood is extracted from social networks [5]. In their paper, S. Deng et al. propose a collaborative filtering approach considering user mood to recommend to user a top N songs. This method does not work without historical association data and its performance is dependent of the amount of history collected about each user. Recently, RL framework has been used by researcher to model the song recommendation problem [4] when no historical data is available about the user. However, it is known that the trade-off achieved is not appropriate in changing environment because it needs a long time to adjust the reward when changes occur.

IEEE/ACM ASONAM 2018, August 28-31, 2018, Barcelona, Spain
978-1-5386-6051-5/18/$31.00

We propose a mood-based music recommendation system which recommends songs that match user's mood without any prior knowledge about user's preferences. We first predict the user's mood from their public social accounts, using the same method described in [6]. The user's mood is classified into three classes, namely positive, negative and neutral. Then, we embed user's mood and song's emotion in a latent space to learn a general function using the data of many users collected from online blogging websites. To cope with the limitation of [4], we rely on the combination of a reinforcement learning (RL) framework and a statistical test called Page-Hinkley (PH) test in which the recommendation process is iterative and adaptive using a change point detection method. At each trial, the algorithm adapts the song's selection function based on implicit feedback and allows a change in the songs position over the song map when a change is detected in user's implicit feedback. Over time the algorithm builds a personalized map per mood for each user from a general song map using the user implicit feedback.

The main contributions of this work are two-fold: (1) We propose an autonomous approach that is independent of any kind of input from the user (music preference, mood, etc.) and of any extra devices (sensors, cameras, etc.) and build a generic song map that associates user's mood and song's mood in a latent space from the online available data of many users.(2) We build a personalized song map using a reinforcement learning framework and PH test to update the song's selection method based of implicit feedback.

II. Proposed Approach

We build a multi-stage system which consists of four stages: (1) Embed all users' songs in a latent 2D-space (songs map) using mood-aware features to learn a general function of users preferences per mood. (2) Predict each target user's mood from their activities on online social networks by extracting linguistic and non-linguistic features over social, cyber and physical spaces, as described in [6]. (3) Recommend songs

using Softmax selection and songs map following an RL framework. (4) Update each user's song map per mood based on their implicit feedback and songs position within the map when a change in user's reward is detected.

A. Songs Maps Generation

Our system starts with an initialization phase. It consists of collecting data about users' preferred songs for each mood (positive, negative and neutral) from music-sharing microblogs where users mention the music they listen to with metadata including posting-time, song name, artist name, self-report (location, mood), etc. This phase is done offline as follows: (1) Collect real-life music listening contexts. Associate, for each user, a set of songs per mood. (2) Build a map of songs per mood using features related to user's mood and song's emotion:

- User mood features: a vector of 3 values representing the number of users in the dataset who listened to a specific song in a specific mood; positive, negative and neutral.
- Song emotion features: a vector of 43 values representing song's emotion where each dimension is a numerical value indicating the 'affinity' (i.e., degree of association, in $[0, 1]$) between the song and a music emotion [7].

(3)Embed the songs in 2D space using t-SNE algorithm [8].

B. Reinforcement Learning Framework with Softmax Selection

Song selection process is done through trial-and-error iteration where the system represents the agent who selects songs to user based on his mood and his feedback. The list of possible songs per mood in this framework is found based on data collected from many users on microblogs.

Implicit User Feedback: Our system does not need any explicit input from the user (song rating, for example). It rather relies on implicit feedback (song click, duration of listening, listening count, etc). User preference toward a song s is obtained by observing his behavior on the recommended song in mood m. It consists of several factors: duration of listening (l_t), whether the song is added to the playlist of user (f_v), listening count (n_l). For each user u, the total score of implicit feedback (Reward) after listening to song s at time t is computed as the sum of two rewards:

$$R_{total,t}(u, s, m) = R_{pref,t}(u, s, m) + R_{d,t}(u, s, m) \quad (1)$$

where $R_{pref,t}(u, s, m)$ represents user u preference to the song s in mood m. It is defined as follows:

$$R_{pref,t}(u, s, m) = \sum_{k \in K} c_k r_{k,t}(u, s, m) \quad (2)$$

where c_k is the weight of each factor of the implicit feedback, $r_{k,t}(u, s, m)$ is the value of factor k for song s and $K = l_t, f_v, n_l$.

$R_{d,t}(u, s, m)$ represents the reward received from the position of the song on the map. It is defined as follows:

$$R_{d,t}(u, s, m) = \begin{cases} +d(s_t, s_{t-1}), \\ \quad \text{if } d > d_{avg} \text{ in } T_1 \text{ and if } d < d_{avg} \text{ in } T_2 \\ -d(s_t, s_{t-1}), \\ \quad \text{if } d < d_{avg} \text{ in } T_1 \text{ and if } d > d_{avg} \text{ in } T_2 \end{cases} \quad (3)$$

where T is the total observation time and d_{avg} is the average distance between all songs in the map. The time T is divided into two epochs: The first epoch is exploration T_1 where the system explores new songs to learn about user's preference by favoring different songs from the current song. The distance $d(s_t, s_{t-1})$ between the current song and the previous one should be greater than d_{avg} in this epoch. The second epoch is the exploitation time T_2 where the system has collected enough information about the user and decided to exploit his learned policy by choosing similar song to the current one (songs with $d(s_t, s_{t-1})$ smaller than d_{avg}).

Song Selection: We refer to Softmax method where the probability distribution of each song is updated after receiving a feedback on the current recommendation.
Let's define an update function as $Q_t(u, s, m)$:

$$Q_0 = \frac{\text{number of times the song has been listened to}}{\text{total number of times all songs has been listened to}} \quad (4)$$

$$Q_t(u, s, m) = Q_{t-1}(u, s, m) + \alpha_t(R_{total,t}(u, s, m) - Q_{t-1}(u, s, m)) \quad (5)$$

Where α_t determines to what extent the newly acquired information will override the old information.
At each iteration, the next song is selected using Softmax method. For each song, we calculate the probability $P(u, s, m)$ which is considered as an update for the song's popularity (Q_0). The system selects the top-k songs with the highest probability as the recommended song list:

$$P_t(u, s, m) = \frac{\exp^{Q_t(u,s,m)}}{\sum_{t=1}^{T} \exp^{Q_t(u,s,m)}} \quad (6)$$

C. Song Map Update Strategy

The position of songs inside each user's map can also be updated based on the user's feedback ($R_{total,t}$) about the current song. We are inspired by the change point detection process in [9]. The idea is to re-embed the songs in a latent space when the system detects a change in the user's reward function ($R_{total,t}$) . Using an extra feature that represents the user's feedback allows the position of songs to change over time. To re-build the map, we use the same algorithm t-SNE. However, we add an extra feature representing the user's feedback (Equation (1)). The change is manifested by a change in the reward in Equation (1). This change-point test is known as Page-Hinkley test (PH). Let's consider the series of rewards gathered by recommending the current best song in the last T steps/trials: $R_{total,1}(u, s, m), R_{total,2}(u, s, m), ..., R_{total,T}(u, s, m)$. The question, now, is whether this series can be attributed to

a single statistical law; otherwise the series demonstrates a change in the statistical law underlying the rewards (change-point detection). The strategy after change detection is as follows: (1) Re-build the map (2) If the change was detected during T_2 (exploitation phase), then the algorithm will force the system to go back to T_1 (exploration phase), because the already learned policy for user's preference is no more effective after a change is detected and the system should learn again about user's preference.

III. Experimental Evaluations

To evaluate the effectiveness of our method in a large-scale dataset before deploying it to the real world, we have run a simulation. We refer to the large LiveJournal two-million post (LJ2M) dataset [7] harvested from the social blogging service LiveJournal. The LJ2M dataset is used to build the generic song maps per mood. The features of song emotions and users mood are annotated in this dataset. We clean the data by discarding users who have less than five records in the dataset. We select 800 users and 4296 songs, per mood. We kept the full listening history of 90% of the users as a training set and the remaining 10% of users are considered as a test set. We run the simulation for 1000 trials. In each trial, the system recommends the top-10 songs with the highest probabilities, gets a reward then, updates P_t for next trial.

A. Simulate User's Reward

Since there are no real users in the simulation, we need to simulate the reward $R_{pref,t}$ to evaluate the performance of our proposal. For each user, we give a vector of 4296 reward-values that represent the reward of user after listening to each song. Many works in the literature assumed a linear Gaussian distribution to model users' preferences. Let θ be the utility function of each song and r_s is the expected reward for each song s. Assuming a linear Gaussian distribution, the posterior $p(\theta|r)$ can be computed as in [10]:

$$p(\theta|r) = \mathcal{N}(\mu_{\theta|r}, \Sigma_{\theta|r}) \tag{7}$$

$$\Sigma_{\theta|r}^{-1} = \Sigma_{\theta}^{-1} + F^T \Sigma_r^{-1} F \tag{8}$$

$$\mu_{\theta|r} = \Sigma_{\theta|r}[F^T \Sigma_r^{-1}(r) + \Sigma_{\theta}^{-1}\mu_{\theta}] \tag{9}$$

Also, we assume Σ_θ and Σ_r are diagonal matrices with diagonal values of σ_θ and σ_r, respectively. The utility function θ of each song is drawn as a sample from the posterior distribution. The expected reward for each song s, is then estimated as $r_s = f_s \times \theta$ where f_s represents the expected features for song s. F represents a matrix containing the features of the recommended song.

B. Baseline Algorithms

The "random", "most popular", "k-NN" and "UCFE" methods are considered as baselines in our evaluation. In each trial, "random" method selects a list of songs randomly. The "most popular" method selects a list of songs based on equation (4Song Selectionequation.2.4). The songs with the highest value are selected. User-based k-NN was considered as baseline for this simulation ($k = 15$). Cosine metric was used to calculate the similarity between users. At each trial, random songs from the neighbor/similar users' songs were recommended. In user-based collaborative filtering with emotion (UCFE) , a factor of emotional context was added when computing the similarity between users and score for songs (for details see [5]).

C. Evaluation Tests

Let U be the set of users in the test set, $R(u)$ be the recommended songs for user u and $T(u)$ be the set of user's songs in the test set. We adopt three metrics to evaluate the recommendation results:

1) Hit rate: A recommended list is considered as hit if it contains a song that the user is interested in his current mood. The hit rate is the ratio of the number of hits over the times of recommendations:

$$HitRate = \frac{\text{Number of hits}}{\text{times of recommendations}} \tag{10}$$

2) F1 score: F1 score is the harmonic mean of precision and recall metrics. It is defined as:

$$F1score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

where: Precision is the percentage of the recommended items that are relevant:

$$Precision = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \tag{12}$$

and Recall is the percentage of relevant items that are recommended:

$$Recall = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \tag{13}$$

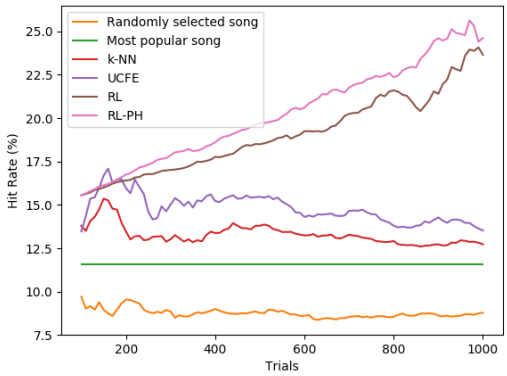
We conduct two evaluation tests:
TEST1: We compare the performance achieved after 1000 trials in terms of hit rate and F1 score between the proposal and baselines. The objective of this test is to measure the overall performance of our proposal.
TEST2: We compare the performance of the proposal in terms of hit rate and F1 score when running the simulation in different number of trials. This test evaluates the improvement of performance over time.

D. Results and Discussions

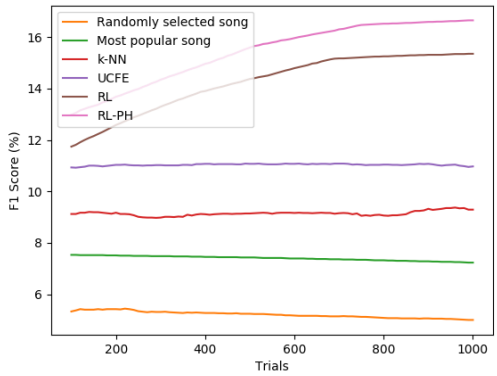
The result of TEST1 is summarized in Table **??**. It shows that the proposal outperforms the conventional methods in all metrics; hit rate and F1 score. As we have mentioned, in our dataset, the average number of preferred songs per user is five per mood. This explains why the values in Table **??** are not high. Moreover, if we compare the performance of the proposals in different moods, we can say that the improvement is more important in positive and negative moods than that of neutral mood. In all metrics, the improvement in neutral mood is not as significant as that in the other moods, suggesting that our proposal works effectively when the user's mood is explicit (i.e. positive or negative). The result of TEST2 is shown in Fig.

	Positive Mood						Negative Mood						Neutral Mood					
	Random	Popular	k-NN	UCFE	RL	RL-PH	Random	Popular	k-NN	UCFE	RL	RL-PH	Random	Popular	k-NN	UCFE	RL	RL-PH
Hit Rate	8.63	11.31	12.43	13.02	23.51	**24.19**	8.78	11.57	12.73	13.53	23.65	**24.62**	7.02	10.93	11.34	12.22	15.11	**16.10**
F1 score	5.42	6.95	9.04	10.95	15.26	**17.06**	4.99	7.23	9.29	10.96	15.35	**16.64**	5.12	6.01	8.01	9,95	10.63	**11.54**

TABLE I: Comparison in Terms of Hit rate and F1 score per Mood



(a) Hit rate



(b) F1 score

Fig. 1: Effect of Number of Trials: Negative Mood

1Effect of Number of Trials: Negative Moodfigure.caption.3. Due to lack of space, we show the result of one mood, i.e. negative mood. It shows the performance in terms of hit rate and F1 score of the proposal versus conventional schemes over different number of trials. The results show that the proposal (RL-PH in pink line) has better performance than the baselines in the two metrics. Besides, when increasing the number of trials, the performance of the learning-based approaches (RL and RL-PH) is improving compared to the performance of other approaches. We obtained similar results for positive mood. The improvement over the baseline approaches shows that: (1) the emotional context plays an important role in user's songs preference when comparing between emotion based approaches (UCFE, RL, RL-PH) and non emotion based approaches (random, most popular, k-NN). (2) starting from many users' data collected from microblogs to embed songs in different mood maps then personalize the embedding based on user's preference is feasible and useful for emotion aware music recommendation.

IV. Conclusion

We proposed an autonomous and adaptive recommendation system that recommends songs to users based on their predicted mood from online social networks (OSNs) without requiring any input from the user. We combined Reinforcement Learning with Page-Hinkley test to adapt the generic song map per mood to a personalized map based on user's preference. We simulate our proposal using a large dataset (LJ2M) collected from blogging websites. The results show that the proposal outperforms the baselines in terms of hit rate and F1 score. As future work, we plan to develop an application and run experiment with real users to evaluate the performance of the proposal in practice.

References

[1] Ivana Andjelkovic, Denis Parra, and John O'Donovan. Moodplay: Interactive mood-based music discovery and recommendation. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, UMAP '16, pages 275–279, New York, NY, USA, 2016. ACM.

[2] Dongjing Wang, ShuiGuang Deng, and Guandong Xu. Gemrec: A graph-based emotion-aware music recommendation approach. In *Web Information Systems Engineering - WISE 2016 - 17th International Conference, Shanghai, China, November 8-10, 2016, Proceedings, Part I*, pages 92–106, 2016.

[3] Rana Forsati, Iman Barjasteh, Dennus Ross, and Abdol-Hossein Esfahanian. Learning the implicit perference of users for effective recommendation. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM'17)*, 2017.

[4] Elad Liebman, Maytal Saar-Tsechansky, and Peter Stone. Dj-mc: A reinforcement-learning agent for music playlist recommendation. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, pages 591–599, Richland, SC, 2015. International Foundation for Autonomous Agents and Multiagent Systems.

[5] Shuiguang Deng, Dongjing Wang, Xitong Li, and Guandong Xu. Exploring user emotion in microblogs for music recommendation. *Expert Syst. Appl.*, 42(23):9284–9293, December 2015.

[6] Chaima Dhahri, Kazunori Matsumoto, and Keiichiro Hoashi. Personalized mood prediction over online social networks: Data analysis on cyber-social-physical dimensions. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM'17)*, 2017.

[7] J. Y. Liu, S. Y. Liu, and Y. H. Yang. Lj2m dataset: Toward better understanding of music listening behavior and user mood. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2014.

[8] L.J.P van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9: 2579-2605, Nov 2008.

[9] Cédric Hartland, Nicolas Baskiotis, Sylvain Gelly, Michèle Sebag, and Olivier Teytaud. Change Point Detection and Meta-Bandits for Online Learning in Dynamic Environments. *CAp*, pages 237–250, 2007.

[10] Negar Hariri, Bamshad Mobasher, and Robin Burke. Context adaptation in interactive recommender systems. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 41–48, New York, NY, USA, 2014. ACM.