The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

CAARS: A Context-Aware Artist Recommender System for Twitter Users

Linh Tang, Fahmida Hamid
Department of Computer Science
Grinnell College
Grinnell, IA 50112, USA
tanglinh@grinnell.edu
fahmida.hamid@gmail.com

Abstract

In this work, we introduce a context-aware hybrid artist recommender system (CAARS) that uses Twitter users' tweet-time patterns as context and users' bias about gender and types of musicians to recommend artists. Our model offers a novel approach to improve a personalized music recommender system (MRS) as it extracts implicit information from the users' past tweet-behavior and combines that with related content. The proposed model performs significantly better than collaborative and hybrid recommender systems and encourages further exploration.

1 Introduction

With the aid of technology, personal music collection has grown dramatically in the last decade. But the idea of automatically recommending music genuinely did not begin until 2001 (Celma 2010). As evidence of people's interest in this domain, we notice a consistent rise in the number of research articles related to music recommendation in the International Society for Music Information Retrieval (ISMIR) every year. The sales record of the American music industry, on the other hand, shows that music consumption is biased towards a few popular artists. The scenario might be similar to the rest of the world as well. We need a remedy for this situation. That said, we propose a recommender system (RS) for musical performers. Our goal is to personalize the system by looking into the users' preferences on some content and expand the horizon by considering dominant contexts; hence we call it the Context-Aware Artist Recommender System (CAARS).

In building *CAARS*, we use Million Musical Tweets (Hauger et al. 2013) and MusicBrainz (Foundation 2017) datasets. We consider the available information about the listeners and their tweet-histories to find some association with the musicians and the types of music. We then use context information (users' tweet-time pattern) and personalize the contents (their biases about the gender of the artist and the type of the artist) to refine the recommendation list. This composite system helps identify users who are similar in terms of the same tweet-time patterns and produces interesting results. To the best of our knowledge, we are the first group to offer such a model using these datasets.

2 Related Work

In this section, we give the rationales for choosing the hybrid system over collaborative and content-based systems and we describe the importance of context and implicit feedbacks for designing a recommender system.

2.1 Hybrid Systems (HY)

In Collaborative Filtering (CF) systems, people collaborate to help each other perform filtering by recording their reactions to items that they use (Goldberg et al. 1992). CF recommender systems record users' preferences as ratings (numerical values). The more ratings the system can draw out from the users, the more effective the recommendations are (Elahi, Ricci, and Rubens 2016); which directs us towards the cold-start problem, the major drawback of a basic CF approach. In Content-Based (CB) recommender systems, the descriptions of the attributes of items are used to make recommendations. CB methods are effective at providing recommendations for new items (Aggarwal 2016), mitigating the cold-start problem. CB methods are not effective at providing recommendations for new users and sometimes good descriptions of the item-features are not available to the system. On the other hand, the incorporation of contextual information about the user in the recommendation process has recently attracted major interest (Verbert et al. 2012). The CB systems need well-defined content information to work reasonably well. As the traditional CF and CB systems only consider similar users or similar items, users have fewer chances to be exposed to new items of different types that may become new favorites. Hybrid systems have been designed to overcome these stated limitations and explore more possibilities. Feature combinations and parallel/sequential module designs are popular techniques for building hybrid systems.

2.2 Context

Research in behavioral studies shows that decision-making is contingent upon the relevant context consumers are in(Adomavicius et al., Verbert et al.). Time, location, social interaction, the user's mood, etc. are considered as context. Collecting appropriate context directly is difficult due to issues of privacy and stability. Context is not always readily available; it can be learned explicitly by gathering information through surveys or other means.

2.3 Implicit User Feedback

For modeling users, RS depends on different types of input which we can broadly categorize as *explicit-feedback* and *implicit-feedback*. Explicit-feedback includes users' explicit opinions (e.g. ratings) regarding their interest in products. They are relatively easy to interpret but time-consuming and expensive to collect (Hu, Koren, and Volinsky 2008). Implicit-feedback techniques seek to avoid this bottleneck by inferring something similar to the ratings that a user would assign from observations that are available to the system (Oard and Kim 1998). Implicit information is inherently ambiguous but a good source of representing complicated and high-dimensional information.

3 Dataset

We use two large datasets: the Million Musical Tweets Dataset (MMTD) which provides time, location and songs from the tweets of users from 2011 to 2013, and MusicBrainz which contains information of songs and artists' features such as artist type, artist area, and artist gender.

Dataset	Basic Statistics
# of unique artists	24673
# of unique users	214741
# of unique tweets	1074713
# of unique tracks	133228

Table 1: Basic Statistics of MMTD dataset

3.1 Million Musical Tweets Dataset

The dataset (Hauger et al. 2013) contains listening histories inferred from Twitter, each of which identified by an unique Tweet ID and linked to a user ID. Each tweet is annotated with temporal (date, time, weekday, timezone), spatial (longitude, latitude, continent, country, county, state, city), and demographic information of the country. Most importantly, each tweet contains a song with the corresponding artist(s).

3.2 MusicBrainz

MusicBrainz (Foundation 2017) is a music metadata project that includes information about artists, release groups, releases dates, recordings, works, and labels, as well as the many other relationships between them. For this work, we consider two features:{*artist's type, artist's gender*}, as the content of artists.

The type attribute is used to state whether an artist is a person, a group, or something else. Musicbrainz dataset identifies six possible artist types: {*Person, Group, Orchestra, Choir, Character, Other*}. The gender attribute identifies a person as either *male, female* or *neither*. Groups do not have genders.

4 Our Hypothesis

Our hypothesis is that **the temporal pattern of tweets can provide helpful information to recommender systems**. If two users like to tweet about songs usually at late night (regardless of their geographical location), it is possible that their choice of songs are similar to some extent. We only need to determine how to adjust this context to other features while building a recommender system. In the next section, we explain how we implement this hypothesis to build our model.

5 Our Methodology

5.1 Preprocessing Data

MusicBrainz provides a permanent id (a 36 character universally unique identifier) to each artist in the database. We use this id to associate the entries between the MMTD and MusicBrainz dataset and extract artist's type, gender, area as features for the CF model.

5.2 Our Model

Figure 1 shows the basic components of our model. We detail the steps as follows:

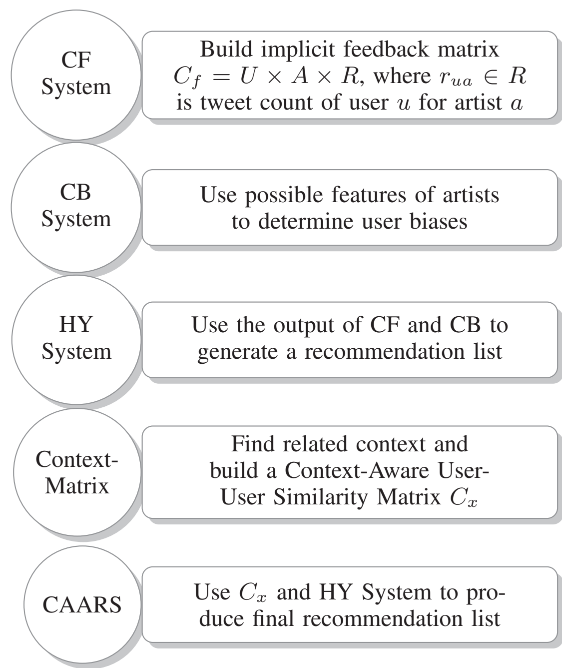

Figure 1: Basic Architecture

CF System: The collaborative filtering stage builds a neighborhood matrix by counting the total tweets of each user for each artist. As the tweet count implicitly provides some feedback, we use the Alternating Least Square (Felsenstein 1997) method to factor our matrix into small neighborhoods and find similar users based on their tweet counts. In order to calculate the final recommendation score of each artist for each user, we use the method discussed by Hu, Koren, and Volinsky (2008).

We use the following notations in describing our model: Let U be the set of all users, A be the set of all artists, and R be our implicit feedback matrix with dimensions $|U| \times |A|$. Each element $r_{ua} \in R$ represents the total number of tweets posted by user u for artist a. By using the alternating least-square method, we calculate the recommendation (collaborative filtering) score, $cf\text{-}score(u, a)$, of each test user for each artist.

CB System: We consider two features (artist's gender and artist's type) as content, each with its own implicit vectors. We compute the bias, $f\text{-}bias(u, f_i)$, of a user u toward a feature f_i with the formula:

$$f\text{-}bias(u, f_i) = \frac{twf(u, f_i)}{tw(u)} \tag{1}$$

where, $twf(u, f_i)$ is the total tweets made by u for a specific feature category f_i and $tw(u)$ is the total number of tweets made by u.

Suppose a user makes 10 tweets, 4 for female artist, 3 for male artist, 2 for groups, and 1 for others. Our system will calculate the user's bias toward gender feature as table 2. Similar computation can be done for the type feature.

category:	male	female	group	other
gender-bias:	0.4	0.3	0.2	0.1

Table 2: Gender-bias of a Tweet user

Since we use only two features, we call them $gender\text{-}bias$ and $type\text{-}bias$.

HY System: Our hybrid model combines $cf\text{-}score(u, a)$ with $gender\text{-}bias$, and $type\text{-}bias$ using the following formula:

$$\begin{aligned} hy\text{-}score(u, a) = cf\text{-}score(u, a) \times \\ \{1 + gender\text{-}bias(u, gender(a)) \\ + type\text{-}bias(u, type(a))\} \end{aligned} \tag{2}$$

If the $gender\text{-}bias$ and $type\text{-}bias$ of a user is zero or insignificant, the HY system generally predicts the same list as the CF system.

Modeling Context Metrix: We divide the twenty-four hours of a day into four major time-slots:

- Morning (5:00 AM - 12:00 PM)
- Afternoon (12:00 PM - 5:00 PM)
- Evening (5:00 PM - 9:00 PM)
- Night (9:00 PM - 5:00 AM)

We count total tweets of each user in each time-slot based on his/her/their **local time-zone**. Then we apply the *Chi-Square Test of Independence* between time-slots (normalized, as we do not divide the durations evenly) and *total tweet count*. The $p\text{-}value$ is less than 0.001 with a confidence of 99%, which indicates that time-slots and tweet-counts are *dependent* on each other.

Context-aware Similarity Matrix: We build a context-aware, $|U| \times 4$ dimensional, user-user similarity matrix C_x. In C_x, each row represents a distinct user and each column represents the normalized tweet count of the user based on our chosen context: ***time-slot=(morning, afternoon, evening, night)***.

$$C_x = \begin{pmatrix} 0.28 & 0.43 & 0.0 & 0.29 \\ 0.50 & 0.50 & 0.0 & 0.00 \\ 0.67 & 0.33 & 0.0 & 0.00 \\ 0.00 & 0.0 & 0.00 & 1.00 \\ \dots & & & \\ \dots & & & \end{pmatrix} \tag{3}$$

As we have a multi-dimensional vector to define each user, we find neighbors of each of them from C_x using KD-Tree (Bentley 1975). The tree returns a distance score, $dist(u_i, u_j)$, between each pair of users u_i and u_j. The range of the $dist$ scores is $[0, 1]$. We change it to similarity score by using the following formula:

$$sim\text{-}score(u_i, u_j) = 1 - dist(u_i, u_j) \tag{4}$$

In our model, we use the KD-Tree with neighborhood size $N = 20$. For each test user u_t, we get a list of 20 nearest neighbors $N_{u_t} = (u_1, u_2, \dots)$ with corresponding $sim\text{-}score$ vector $S_{ut} = (s_{u_1}, s_{u_2}, \dots)$ where $s_{u_i} = sim\text{-}score(u_t, u_i)$.

Context-aware Hybrid Model: Our system performs the steps discussed in Algorithm 1 to find the final ranked list. The first block (line no #1 to #9) of the algorithm finds a collection of (size = $|N_{u_t}| * k$) preferred artists by considering the neighborhood, N_{u_t}, of the test user u_t. k is some positive integer. The second block (line no #12 to #18) prepares the final rank of the preferred artists by considering the number of times the artists appear in the neighborhood list and by taking the maximum preference score for each artist. Function $Top_k(D)$ returns top k keys from a given dictionary D after sorting the items in decreasing order of their values.

6 Performance Analysis

In order to evaluate the performance of our design, we split the dataset into different train-test ratio (90-10, 80-20, 70-30, 60-40) and calculate *precision, recall,* and *F-measure* on the test users with fifteen or more tweets. The detailed reports are presented in table 3 and 4. Test no $1, 2, 3, 4$ use test users only found on the test set and test no 5 uses all users from the dataset for performance evaluation. In all of these test cases, CAARS shows significant improvement over the basic CF and HY models.

7 Conclusion & Future Directions

In this work, we demonstrate that implicit feedback (e.g., tweet count) and context-sensitive information are important features for modeling artist recommender systems. Our main contributions are:

- Combine MMTD and MusicBrainz dataset for extracting artists' information that we use for finding user biases.

Algorithm 1: Context-aware Model

Input: u_t, U, A, (N_{u_t}, S_{u_t})
Output: $(a_1, a_2, \ldots, a_k)$
Initialize an empty list, L_{u_t}
for $u_i \in N_{u_t}$ **do**
  Initialize an empty dictionary, E_{u_i}
  for $a \in A$ **do**
    $v_{a_{u_i}} = hy\text{-}score(u_i, a) * s_{u_i}$
    $E_{u_i} = E_{u_i} \cup (a, v_{a_{u_i}})$
  end
  $L_{u_t} = L_{u_t} \cup Top_k(E_{u_i})$
end
L_{u_t} contains $|N_{u_t}| * k$ entries possibly with some duplicate artist entries as some artists may appear in the favorite lists of multiple neighbors of u_t
We convert the list L_{u_t} to a dictionary of lists where the artist id is the key to combine the multiple entries for some artists
Initialize an empty dictionary, RA
for $(a, \{v_{a_1}, v_{a_2}, \ldots\}) \in L_{u_t}$ **do**
  $s_a = max(v_{a_1}, v_{a_2}, \ldots)$
  $c_a = \log(count(v_{a_1}, v_{a_2}, \ldots))$
  $rank_a = s_a + c_a$
  $RA = RA \cup (a, rank_a)$
end
return $Top_k(RA)$

#	Train-Test	# of tweets	# of users	# of users with tweets ≥ 15
1	**90-10**	107472	47689	250
2	**80-20**	214943	78547	812
3	**70-30**	322414	103785	1621
4	**60-40**	429886	125008	2463
5	**60-40**	1074713	214741	6840

Table 3: Statistics of Performance Analysis

- Determine a relevant and dominant context (users tweet-time patterns) that we use successfully along with the hybrid model.

We plan to expand the research in several other dimensions. The user's current location, cultural background, and native language may be the determining factors to improve our system. We may use the lyrics and genre of the songs as features for the next phase of our work.

#	Train-Test	Scale	CF	HY	CAARS
1	90-10	Precision	0.005371	0.005371	0.182933
		Recall	0.001512	0.001512	0.041596
		F-measure	0.002234	0.002234	0.062847
2	80-20	Precision	0.004961	0.004961	0.147209
		Recall	0.00194	0.00194	0.048172
		F-measure	0.002628	0.002628	0.068338
3	70-30	Precision	0.003717	0.003717	0.126918
		Recall	0.001818	0.001818	0.05374
		F-measure	0.002303	0.002303	0.070773
4	60-40	Precision	0.003581	0.003581	0.117932
		Recall	0.001922	0.001922	0.057593
		F-measure	0.002382	0.002382	0.072448
5	60-40	Precision	0.003163	0.003163	0.106881
		Recall	0.001887	0.001887	0.06040
		F-measure	0.002242	0.002242	0.071867

Table 4: Detailed Results

References

Adomavicius, G.; Mobasher, B.; Ricci, F.; and Tuzhilin, A. 2011. Context-aware recommender systems. *AI Magazine* 32:67–80.

Aggarwal, C. C. 2016. *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st edition.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM* 18(9):509–517.

Celma, O. 2010. Music recommendation. In *Music recommendation and discovery*. Springer. 43–85.

Elahi, M.; Ricci, F.; and Rubens, N. 2016. A survey of active learning in collaborative filtering recommender systems. *Comput. Sci. Rev.* 20(C):29–50.

Felsenstein, J. 1997. An alternating least squares approach to inferring phylogenies from pairwise distances. *Systematic biology* 46(1):101–111.

Foundation, M. 2017. Musicbrainz - the open music encyclopedia [online]. *Metabrainz Foundation*.

Goldberg, D.; Nichols, D.; Oki, B. M.; and Terry, D. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35(12):61–70.

Hauger, D.; Schedl, M.; Košir, A.; and Tkalčič, M. 2013. The million musical tweet dataset: What we can learn from microblogs. *International Society for Music Information Retrieval*.

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. *2008 Eighth IEEE International Conference on Data Mining*.

Oard, D., and Kim, J. 1998. Implicit feedback for recommender systems. In *in Proceedings of the AAAI Workshop on Recommender Systems*, 81–83.

Verbert, K.; Manouselis, N.; Ochoa, X.; Wolpers, M.; Drachsler, H.; Bosnic, I.; Member, S.; and Duval, E. 2012. Context-aware recommender systems for learning: A survey and future challenges. *IEEE Trans. Learn.*