# Social Tags and Emotions as main Features for the Next Song To Play in Automatic Playlist Continuation

Marco Polignano
marco.polignano@uniba.it
University of Bari "Aldo Moro", Dept. of Computer Science

Pierpaolo Basile
pierpaolo.basile@uniba.it
University of Bari "Aldo Moro", Dept. of Computer Science

Marco de Gemmis
marco.degemmis@uniba.it
University of Bari "Aldo Moro", Dept. of Computer Science

Giovanni Semeraro
giovanni.semeraro@uniba.it
University of Bari "Aldo Moro", Dept. of Computer Science

## ABSTRACT

The broad diffusion over the Internet of songs streaming services points out the need for implementing efficient and personalized strategies for incrementing the fidelity of the customers. This scenario can collect enough information about the user and the items for successfully design a Recommender System for the automatic continuation of playlists of digital contents. In particular, in this work we proposed a strategy for suggesting a set of tracks, starting from a list of songs played by the user, candidate as next to play. The list contains songs that are coherent with the main characteristics of songs already played. In order to collect enough information and for applying a recommendation strategy, we used third-party external sources of information. They provide data about the song, including its popularity, the emotion evoked by its lyrics, low and high-level audio features, lyrics and more. The system highlights the importance to use user-generated tags and emotional features for successfully predicts user next played songs.

## CCS CONCEPTS

• **Information systems → Recommender systems**; **Personalization**; Social networks.

## KEYWORDS

automatic playlist continuation, music, emotions, tags, recommender systems

**ACM Reference Format:**
Marco Polignano, Pierpaolo Basile, Marco de Gemmis, and Giovanni Semeraro. 2019. Social Tags and Emotions as main Features for the Next Song To Play in Automatic Playlist Continuation. In *27th Conference on User Modeling, Adaptation and Personalization Adjunct (UMAP'19 Adjunct), June 9–12, 2019, Larnaca, Cyprus.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3314183.3323455

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

*UMAP'19 Adjunct, June 9–12, 2019, Larnaca, Cyprus*
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6711-0/19/06…$15.00
https://doi.org/10.1145/3314183.3323455

## 1 INTRODUCTION AND BACKGROUND

Over the Internet, it is common to find a considerable quantity of streaming services that provide users with songs to listen for free everywhere they are. Each of them has based its own business offering personalized and adaptive services "tailored" to the user preferences and listening behaviors (E.g., Spotify, Pandora, Deezer, iTunes, Amazon Music). Using a general classification of the most common services available [8], we can identify:

- **Not personalized playlists**: the tracks are grouped in a static playlist using some specific descriptors as the discriminant feature. Examples of this strategy are the weekly charts about the top-100 most listened songs over the world.
- **Context-Aware but Not personalized playlists**: they use a single track or author for selecting a set of songs correlated with it. A common example is the service of a web radio station.
- **Not context-Aware but personalized playlists**: the tracks are selecting considering the music preferences of the user.
- **Context-Aware and personalized playlists**: the system uses the user preferences and her history of listening for selecting a set of tracks that can be relevant for her. When these playlists are used for automatically continuing a playlist created by the user, we are facing a task of Automatic Playlist Continuation (APC) that is the problem faced in this work.

This topic is a common domain of application for Recommender Systems [6, 9, 10, 13, 14]. It is not surprising that one of the first Music Recommender System, *Ringo*, was proposed already in 1994 [15] and it allowed users to leave a feedback to a set of artists and albums for consequently receive personalized suggestion in accord with the preferences expressed. The explicit feedbacks in the music domain are often rare, and for this reason, content-based approaches for recommendation are more common than in other domain of application [14]. Among the collaborative recommendation strategies Model-Based, which use a model for representing user preferences, it is relevant to cite clustering strategies, such as Approximate Nearest Neighbors [7] Used in the "Approximate Nearest Neighbors Oh Yeah" algorithm (Annoy) implemented into Spotify for computing items similarity for the algorithm of radio continuation. The algorithm is based on the recursive binary subdivision of the items space using hyperplanes until the ending condition has been satisfied. Collaborative approaches for recommender systems, are commonly implemented in many huge music streaming platforms that can guarantee a large number of ratings left by users. Other commercial systems such as Pandora.com are mostly based on content-based approaches. Songs are cataloged by their characteristic features including low-level meta-data such as the melody but also high-level meta-data as the composer, year of publication, genre and more. The features are used as filters for searching new songs to listen that are similar to those previously liked by the user. The content-based strategies move the problem of APC to a classification task where for each user an item can be classified as liked or disliked. In literature, there are many works based on K-nn, Support Vector Machine or Gaussian Mixture Model for solving the problem [5, 12]. In our proposed model we moved towards content-based approaches. This decision is supported by the frequent absence of information about the context of listening and about the explicit feedback of the users about specific songs. Moreover, the song characteristics are, naturally thinking, the main elements used by the final user for selecting songs to play. As an example, some days a user decides to listen to 80' pop music another the most popular in the week and other music of only a specific music genre. For this reason, we decide to follow this natural approach to simulating the possible choice that the user could do for the task of playlist continuation using constraints over song features derived by the characteristics of tracks already played in the same playlist.

## 2 ARCHITECTURE

The content-based recommender system proposed for the task aims to collect many features as possible about each song played by the user. These data are entirely derived from the characterization of each track over the Internet. The proposed system is called PLACeBo (PLAylist Constraint-Based continuation system), and its whole architecture is subdivided into three macro-layers that communicate each other using internal calls:

- **Items enchantment**: the first layer has the responsibility to collect as much as possible information about the songs in the catalog from external sources.
- **Items selection**: it is the module that detects the set of constraints valid for the playlist which is used for the selection of candidate songs for the recommendation phase.
- **Songs Recommendation**: the candidate items are re-ranked and proposed to the final user as the set of recommended items for the task of APC.

When a playlist is provided as input for the APC task, PLACeBo finds, from the song catalog, the set of candidates that better fit the common characteristics shared among the tracks of it. As described by Endsley [4] the human decision-making task is guided by conceptual models that are "situation-aware". Following this idea, we are aware of the importance of a formalization of user goals (short or long-term), objectives, expectations, the mood for guiding the reasoning process and take a decision. This information is strictly dependent by the user and in the case of an APC task without information about the creator of the playlist, these elements are deduced directly by the composition of the playlist. This concept guided the design of PLACeBo. Using the layer of "Item Selection", the system can detect common features among the tracks in order to model the constraints used by the user while selecting what listening. Consequently, the tracks that satisfy the constraints identified are select from the catalog and a set of songs for the specific situation is retrieved. For adopting a solution for the decisional task, the subject commonly choice elements that looks more familiar [4]. Following this idea, we use the popularity of the items as the re-ranking strategy of the designed recommender system.

## 3 ITEM ENCHANTMENT LAYER

The *Item Enchantment* layer it the first slice of the system architecture involved in the recommending process. Every single playlist available as ground knowledge of the domain of application is analyzed and subdivided in songs. The information about the title of the track and its position in the specific playlist is the first piece of meta-data stored in our song catalog. It is important to note that this information has been considered the starting point for the phase of feature collection; for this reason, we consider them as valid and not misspelled or inaccurate. This consideration allows us to do not consider any strategy for mapping the title in the dataset with the one which could be available over the Internet. Any further analysis merely excludes the songs not found on the web. Aware of this decision, for each song the *"Features Finder module"* communicate with external sources for enriching the song description. In particular, the module can query:

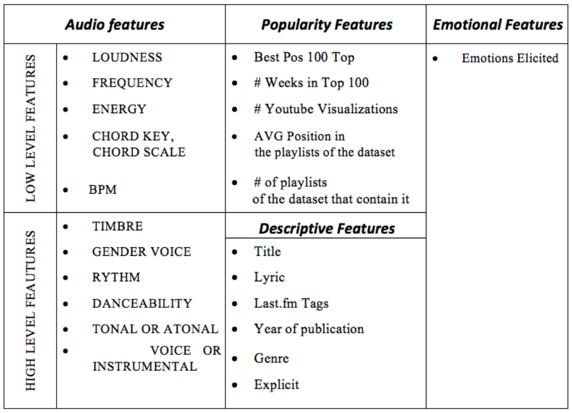
- **Acoustic Brainz**: the online service [1] provides free to use API for obtaining a set of low-level (tonal features) and high-level descriptors of the track.
- **Billboard.com**: it allows to identify the *best position reached* by the song, and for how many *weeks it has been in top 100 played tracks.*
- **Youtube.com**: the popular platform can provide valuable information about the popularity of the song over time using the *number of visualizations of the song video.*
- **Lyrics.wikia.com**: the music lyrics database allow to collect the textual description of the song through its *lyric.*
- **Indico.io**: the platform is composed of a complete set of modules for the analysis of textual sources. Among them, it has been used the API for the detection of emotions over the lyrics based on the Ekman model [2].
- **Last.fm**: the API offers much information about song tracks, but we focused on the list of *social tags* left by listeners about the track.
- **iTunes.com**: the famous streaming provider provides a set of free API for collecting *the music genre*, *the presence of explicit words*, *the year of publication.*

The final list of the groups of features extracted from the different online services is reported in Fig. 1.

## 4 ITEM SELECTION LAYER

For performing the task of item selection two main modules have been implemented: *the Constraints Identifier module* and, *the Candidate Track Selector.* The first has the responsibility to find a subset of the features collected in the previous layer, with the correspondent thresholds, which allows maximizing the probability to select tracks coherent with the songs of the playlist. The second performs

[1] https://acousticbrainz.org/data



| | *Audio features* | *Popularity Features* | *Emotional Features* |
|---|---|---|---|
| LOW LEVEL FEATURES | • LOUDNESS<br>• FREQUENCY<br>• ENERGY<br>• CHORD KEY, CHORD SCALE<br>• BPM | • Best Pos 100 Top<br>• # Weeks in Top 100<br>• # Youtube Visualizations<br>• AVG Position in the playlists of the dataset<br>• # of playlists of the dataset that contain it | • Emotions Elicited |
| HIGH LEVEL FEAUTURES | • TIMBRE<br>• GENDER VOICE<br>• RYTHM<br>• DANCEABILITY<br>• TONAL OR ATONAL<br>• VOICE OR INSTRUMENTAL | *Descriptive Features*<br>• Title<br>• Lyric<br>• Last.fm Tags<br>• Year of publication<br>• Genre<br>• Explicit | |

**Figure 1: Characterization of the features extracted from the Web Services for each song**

the selections of the candidates directly querying the catalog. The selection model is constructed evaluating each property of songs:

- *Features with numeric values*: We plotted the value of the features using the probabilistic frequency distribution, and then we calculated its variance and means. The two values allow calculating the *coefficient of variation* $C_v = \frac{\sigma}{\mu}$ that provides a significant number that represents the sparsity of the data. When data are too sparse, $C_v < 0.5$ the features are excluded by the model because considered to be not statistically relevant for the selection of candidate next songs to play. On the contrary, when $C_v >= 0.5$ the feature is used as discriminant characteristics and its threshold is set as the interval $[\mu - \sigma; \mu + \sigma]$.
- *Features with boolean values*: the features with boolean values are used in the model as discriminant features when its presence or absence (value 1 or 0) is observed in at least the 70% of the songs in the playlist. This threshold has been chosen because with a lower value the presence of the feature loses its discriminant power. On the contrary, a higher level can be too much restrictive. In any case, it needs to be optimized depending on the dataset used for the system execution.
- *Features with nominal values*: the features that can accept $n$ nominal values are used are managed as an extended case of the binary value. In particular, the feature with the associated nominal value, are used as discriminant characteristic if they are observed in more than the $((\frac{100}{n}) + 10)\%$ of the tracks in the playlist. Also in this case, the decision of this threshold has been taken for guarantee as possible to each value of the nominal feature its importance in the play-list
- *Textual features* (lyrics): it is semantically important to know if the tracks are close in a distributional semantic space. To make them computable, we use word-embeddings using is word2vec [11] introduced by Mikolov for considering them as *numeral features of our model.*

  In particular, the transformation of the plaintext into unigrams and bigrams have been implemented through the Apache Lucene Standard Analyzer class [2] with English set of stop-words and a grammar-based tokenizer with a max length of tokens set to 2. The formalization through a vector of word embeddings has been approached averaging the singular embedding vectors of the words encountered in the document into only one per record. In particular, the word embedding procedure used is word2vec introduced by Mikolov [11]. We decided to train our models to obtain a representation of words which is suitable to the nature of the sentences to annotate. In particular, we used a corpus of 7.500 lyrics collected fromLyrics.wikia.com randomly, through CBOW and ten epochs of learning and, 100 dimensionality vectors of words with a minimum number of occurrences in the collection equal to 5.

The constraints identified have been formalized into a selection model using them as filters, over the catalog of songs, combined using the AND logical operation. When the number of retrieved elements is less than 50, the constraints are relaxed. We removed one group of constraints at a time considering their impact on the global performances of the system as observed in the results of the ablation test detailed in the further section (Sec. 6).

## 5 SONGS RECOMMENDATION

The candidate tracks selected by the *Items Selection* layer need to be re-ranked using a strategy able to associate to each song a score that summarizes her probability to have been chosen. In accord with the mental model of decision-making proposed by Endsley [4], already described, and the common strategy of item re-ranking in recommender systems and retrieval systems [16] we decided to use the popularity of the item as relevance score. This approach is a consequence of the intuition that, among a set of songs that are close enough to them already listened, an excellent approach to propose something to listen is to follow the collaborative approach. That means in the absence of explicit feedback and information about other users of the platform, a re-ranking score could be an external score of the popularity of the song. A good source of this score is the worldwide song charts. We decided to use the following popularity index:

$$\begin{aligned} PI_i = AVG(&Youtube\_Views\_Ratio_i, \\ &Best\_Position\_Billboard\_Ratio_i, \\ &Num\_Weeks\_Top100\_Ratio_i, \\ &Frequency\_In\_Dataset\_Ratio_i) \end{aligned} \tag{1}$$

Where each ratio has been defined as a value between 0 and 1 proportional to the min and max value of the feature in the dataset.

## 6 EXPERIMENTAL SESSION

The evaluation aims to establish what groups of features are the most relevant for the recommendation task and to compare the system with a collaborative approach (Coll-RS) which uses the matrix PLAYLIST x SONG → POSITION instead of the standard user-rating representation. Finally, we also evaluate the performances of the system comparing the results with them obtained by two static lists

[2] https://lucene.apache.org/core/

of recommendation based on randomity (Rand-RS) and frequency of the song in the dataset (Freq-RS).
More precisely, we formulated two research questions:

- **RQ1:** Does PLACeBo overcome the results of traditional recommender systems used for the task of APC?
- **RQ2:** What is the influence of each set of descriptive features on the results obtained by PLACeBo?

**Dataset and data representation.**
We performed our experiments over the dataset provided by Spotify for the challenge organized for the RecSys 2018 conference [3]: *"Million Playlist Dataset"* [1]. In total, we process 4,347 playlists and 50,579 song tracks. For each playlist, 80% of randomly chosen songs are used as a training set, the 20% left has been considered as a test set and then hidden at the system. The final dataset is smaller than the one used for the challenge and for this reason the final scores are not comparable also if the F1 score obtained by PLACeBo is in line with them obtained in the challenge [1]. In any case, the research questions are not focused on obtaining the state of the art F1 score. On the contrary, they are focused on the way to deduct the influence of each feature set considered for obtaining a final recommendation that is transparent, human comprehensible and which follow a reasoning strategy human-like.

**Procedure and runs.**
The evaluation measure used is the *precision*, *recall* and *F1-measure*. For each run, one target item has been extracted from the test set, then the system computed the recommendation list, and we checked if the target items (hidden from the playlist) are in the top recommendation list (@1), among the top-5 , 10, 20, 50 recommendations (@5,@10,@20,@50).

The first run of the experimental session involves the comparison of PLACeBo with baselines. We decided to evaluate the system with collaborative recommender systems (Coll-RS) implemented through Lenskit framework [3] and two different variants: Playlist-Playlist and Item-Item; Content-based Recommender system (CB-RS) implemented as a $k$ nearest neighborhoods among items using as $k$ the values 3 (3-nn), 5 (5-nn), and as similarity function the cosine similarity. Moreover, the two static lists generated random (PlACeBo-Rand) and with the popularity of items (PlACeBo-Freq) has been used. Table 1 shows the results. It is possible to note that the PLACeBo system proposed in this work outperforms the baselines with a difference statistically valid using the Wilcoxon statistical test at $p < 0.05$. The results, showed in Fig. 1, allow us to answer positively at the **RQ1**.

To answering the **RQ2**, we performed an ablation test removing one group of song characteristic features at a time from those available as item characteristics. During the test, the re-ranking score function has been kept fixed as the Popularity Index ($PI_i$). When observing the results of the ablation test in Tab. 2, it is possible to note that the F1 score improves while removing the set of *Low-level Features*. This result supports our decision to remove this set of features from the item representation used in the final configuration of the system. Moreover, considering the @10 results, as a standard length of recommendation, we ordered the group of features descending using the differences in F1@10. The final ordered list is used for relaxing system constraints when not enough elements have been retrieved in the *Item Selection Phase*. The results show that social tags and emotional features are the most important for an accurate selection of candidate entries to recommend, and consequently, they emerged as the most relevant set of features for describing the song for the APC task. These considerations allow us to provide an answer for the **RQ2**. The results in Tab. 2 provide us a guideline for understanding the importance of some features instead of others to use while recommending songs. In particular, it is possible to deduce that a human-like reasoning process can detect relevant aspects for the final user and consequently it can use them for making recommendations. Social and psychologic aspects have been demonstrated to be central while human decision-making tasks are fired, like the one about the next song to play.

| | **F1@1** | **F1@5** | **F1@10** | **F1@20** | **F1@50** |
|---|---|---|---|---|---|
| ***PLACeBo*** | ***0.0629*** | ***0.1406*** | ***0.1848*** | ***0.1793*** | ***0.1410*** |
| ***PLACeBo-Rand*** | 0.0000 | 0.0000 | 0.0005 | 0.0042 | 0.0025 |
| ***PLACeBo-Freq*** | 0.0000 | 0.0246 | 0.0359 | 0.0173 | 0.0124 |
| ***Coll-RS (Playlist-Playlist)*** | 0.0000 | 0.0028 | 0.0032 | 0.0044 | 0.0059 |
| ***Coll-RS (Item-Item)*** | 0.0062 | 0.0115 | 0.0144 | 0.0193 | 0.0243 |
| ***CB-RS (3-NN)*** | 0.0000 | 0.0020 | 0.0031 | 0.0027 | 0.0036 |
| ***CB-RS (5-NN)*** | 0.0000 | 0.0014 | 0.0025 | 0.0030 | 0.0032 |

Table 1: Comparison of PLACeBo with considered baselines

| | ***F1@1*** | ***F1@5*** | ***F1@10*** | ***F1@20*** | ***F1@50*** |
|---|---|---|---|---|---|
| all features | *0.0629* | *0.1406* | *0.1848* | *0.1793* | *0.1410* |
| | ***Diff. F1@1*** | ***Diff. F1@5*** | ***Diff. F1@10*** | ***Diff. F1@20*** | ***Diff. F1@50*** |
| - HighLevel f. | -0.0038 | -0.0016 | -0.0027 | 0.0110 | 0.0052 |
| - LowLevel f. | 0.0069 | 0.0568 | 0.0612 | 0.0590 | 0.0515 |
| **- Emotional f.** | **-0.0127** | **-0.0136** | **-0.0157** | **-0.0148** | **-0.0139** |
| **- Last.fm Tags** | **-0.0133** | **-0.0138** | **-0.0163** | **-0.0155** | **-0.0143** |
| - Descriptive f. | -0.0072 | -0.0021 | -0.0055 | -0.0086 | -0.0132 |
| - Lyrics w2vec | 0.0127 | -0.0081 | -0.0042 | 0.0051 | 0.0053 |

Table 2: Ablation test on groups of descriptive features

The results obtained by PLACeBo are not the final demonstration of the importance of psychological features for prediction task but they are encouraging, and consequently we are sure about the importance of their use in the APC task.

## 7 CONCLUSION

In this work, we proposed PLACeBo, a framework content-based that follows the idea about the importance of the characteristics of the songs as discriminant elements used in the mind of the final user while approaching the decisional task of playlist making. The internal strategy of constraints generation for the selection of candidate items has allowed identifying a subset of the catalog about the user could be interested in due to the high coherence with many of the songs in each playlist. The system has been evaluated with traditional strategies of recommendation based on collaborative approaches and content-based approaches showing encouraging results. The code used in this work can be found at: https://github.com/marcopoli/PLACeBo

## 8 ACKNOWLEDGMENT

This research has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement N. 691071.

[3] http://www.recsyschallenge.com/2018/

# REFERENCES

[1] Ching-Wei Chen, Paul Lamere, Markus Schedl, and Hamed Zamani. 2018. Recsys challenge 2018: Automatic music playlist continuation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 527–528.

[2] Paul Ekman, Wallace V Friesen, and Phoebe Ellsworth. 2013. *Emotion in the human face: Guidelines for research and an integration of findings*. Elsevier.

[3] Michael D Ekstrand, Michael Ludwig, Joseph A Konstan, and John T Riedl. 2011. Rethinking the Recommender Research Ecosystem: Reproducibility, Openness, and {LensKit}. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. ACM, 133–140. https://doi.org/10.1145/2043932.2043958

[4] Mica R Endsley. 1995. Toward a theory of situation awareness in dynamic systems. *Human factors* 37, 1 (1995), 32–64.

[5] Marco Grimaldi and Pádraig Cunningham. 2004. Experimenting with music taste prediction by user profiling. In *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*. ACM, 173–180.

[6] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2012. Context-aware music recommendation based on latenttopic sequential patterns. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 131–138.

[7] Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 604–613.

[8] Iman Kamehkhosh and Dietmar Jannach. 2017. User perception of next-track music recommendations. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. ACM, 113–121.

[9] Marius Kaminskas and Francesco Ricci. 2012. Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review* 6, 2-3 (2012), 89–119.

[10] Brian McFee, Thierry Bertin-Mahieux, Daniel PW Ellis, and Gert RG Lanckriet. 2012. The million song dataset challenge. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 909–916.

[11] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 746–751.

[12] Yvonne Moh, Peter Orbanz, and Joachim M Buhmann. 2008. Music preference learning with partial information. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021–2024.

[13] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[14] Markus Schedl, Peter Knees, Brian McFee, Dmitry Bogdanov, and Marius Kaminskas. 2015. Music recommender systems. In *Recommender systems handbook*. Springer, 453–492.

[15] Upendra Shardanand and Pattie Maes. 1995. Social information filtering: algorithms for automating âĂIJword of mouthâĂİ. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 210–217.

[16] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 125–132.