Master's Thesis (Academic Year 2019)

Study on Context-aware Music Recommender System Using Negative Sampling

（ネガティブサンプリングを用いた コンテキスト情報に基づく 楽曲推薦システムに関する研究）

Tokyo Metropolitan University
Graduate School of System Design
Department of Computer Science

18860638 Jin-cheng Zhang
Supervisor: Yasufumi Takama

January 2020

Abstract

This thesis proposes a method for recommending music items considering listeners' context information without explicit feedback.

Recently, with the development of communication technology and portable electronic devices, the way people consume music has changed from hard devices to online music streaming services. Specifically, users need not buy albums from traditional brick-and-mortar shops, and play it by CD players or extract song from CD to a digital player. Instead, they can enjoy music easily regardless of time and a place by accessing online music streaming services such as Spotify, Apple Music and Amazon Music. These up-to-date music streaming services offer thousands of tracks in wide variety of music genres via Internet-accessible smartphones or tablets.

As a result, a gap of users' ability to access music items and the volume of accessible items has been large. In other words, finding appropriate music items from enormous resources is difficult for users. To solve this problem, recommender systems on music domain have been studied. Except two most common approaches content-based filtering and collaborative filtering, context information such as location, time, weather, demographic information and emotional state of a user plays an important role in the music recommendation domain. On the other hand, music recommendation domain has another challenge: because of the way of listening to music and characteristic of music items, explicit feedback such as rating is not usually available.

Regarding how to use implicit feedback for recommending music items more effectively, playing count has been used by existing music recommender systems. On the other hand, the context information has been introduced for music recommendation by a tensor factorization method. It is regarded as the extended version of a conventional matrix factorization approach, and can consider context axis in addition to user and item axes. However, it is difficult to optimize the use of contexts due to high time complexity. Therefore, the proposed method in this thesis employs FMs (Factorization Machines), in which the context information is treated as factors. It is shown that FMs can flexibly consider the interaction of users, items and context with relatively low time complexity. The result of preliminary experiment shows that using playing count can not improve performance of FMs. Therefore, proposed method does not use it in the process of matrix factorization, but utilizes Kullback–Leibler divergence of playing count distribution under each context to select relatively effective and appropriate context before a music item is to be recommended.

As FMs needs both positive and negative samples, a straightforward approach uses LEs (listening events), a record of context information when a user listened a music item, as positive samples, combinations of context information and music items a user has yet to be experienced are used as negative samples. The problem of such an approach is the number of negative samples are much larger than that of positive ones, which causes a problem of time complexity especially when

training a model by FMs. To reduce the number of negative samples, this thesis proposes three types of negative sampling methods: Random Sampling, Priority Popularity Sampling and Top Discount Popularity Sampling.

The effectiveness of the proposed method and the effect of negative sampling methods are evaluated with an offline experiment. The experimental result on #nowplaying-rs and LFM-1b dataset shows that (1) the proposed method outperforms wALS (weighted Alternating Least Squares), which is one of popular music recommendation algorithms based on matrix factorization, (2) it is also confirmed that performance of recommendation could be improved further by using appropriate context information, (3) when dataset has lower density, popularity-based sampling methods (Top Discount Popularity Sampling, Priority Popularity Sampling) have positive effect on prediction accuracy.

This thesis is organized as follows. Chapter 1 describes background and purpose of research. Chapter 2 summarizes related work, from basis of recommender systems to use of implicit feedback and context information on music recommendation. The details of the proposed method including negative sampling methods and how to utilize context information are described in chapter 3. Chapter 4 evaluates proposed context-aware music recommendation method by using two different datasets. Finally, chapter 5 wraps up the research results and discusses future direction of the research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, with the development of communication technology and portable electronic devices, the way people consume music has changed from hard devices to online music streaming services. Specifically, users need not buy albums from traditional brick-and-mortar shops, and play it by CD players or extract song from CD to a digital devices. Instead, they can enjoy music easily regardless of time and place by accessing online music streaming services such as Spotify[1], Apple Music[2] and Amazon Music[3]. These up-to-date music streaming services offer thousands of tracks in wide variety of music genres via Internet-accessible smartphones or tablets. The library size of Apple Music, Spotify, and Amazon Music have exceeded 45, 35, 16 millions tracks on 2018, respectively[4].

As a result, a gap between users' ability to access music items and the volume of accessible items has been large. In other words, finding appropriate music items from enormous resources is difficult for users. To help user discover their favorite music items, recommender systems on music domain have been studied [44] [33] [35] [43]. As in the case with other domains such as movies, e-commerce and documents, CF (collaborative filtering) [13] and CBF (content-based filtering) [15] are the two most common approaches. CF finds users having similar interest with a target user based on the interaction of users and items to predict ratings. CBF recommends items to a user based on the description of items and a profile of the user's preferences.

Moreover, previous research [34] has shown that context information such as location, time, weather, demographic information and emotional state of the user plays an important role in the music recommendation domain. On the other hand, music domain has different challenges from movie and e-commerce domain: explicit feedback such as rating is not usually available due to the way of listening to music and characteristic of music items. Instead, the interaction between users and items is recorded as a LE (listening event), a record of context information when a user listened a music item. User feedback is usually derived from LEs

[1] https://www.spotify.com/jp/
[2] https://www.apple.com/jp/apple-music/
[3] https://www.amazon.co.jp/gp/dmusic/promotions/PrimeMusic
[4] https://www.macobserver.com/news/songs-in-streaming-music-service-libraries/

implicitly such as total playing count. Therefore, in addition to cold-start [1] and data sparsity [22] problems which are common regardless of domains, the above-mentioned two challenges should also be tackled in music domain.

Regarding how to use limited implicit feedback for recommending music items more effectively, playing count has been used by existing music recommender systems [26]. On the other hand, the context information has been introduced for music recommendation by a tensor factorization approach [33]. It is regarded as the extended version of a conventional matrix factorization approach, and can consider context axis in addition to user and item axes. However, it is difficult to optimize the use of contexts due to high time complexity. Therefore, this thesis employs FMs (Factorization Machines) [46], in which the context information is treated as factors. It is shown that FMs can flexibly consider the interaction of users, items, and context with relatively low time complexity. The result of preliminary experiment conducted by myself showed that using playing count can not improve performance of FMs. Therefore, this thesis proposes a method that does not use it in the process of matrix factorization, but utilizes Kullback-Leibler divergence [53] of playing count distribution under each context to select relatively effective and appropriate context before a music item is to be recommend.

As FMs needs both positive and negative samples, a straightforward approach is to use LEs as positive samples, and the combination of context and music items a user has yet to be experienced are used as negative samples. The problem of such an approach is that the number of negative samples are much larger than that of positive ones, which causes a problem of time complexity especially when training a model by FMs. To reduce the number of negative samples, this thesis employs three types of negative sampling methods: (1) Random Sampling, (2) Priority Popularity Sampling (3) and Top Discount Popularity Sampling. Random sampling randomly selects music items which are not played by a user under the same context as actual LEs as negative samples. Furthermore, popularity of music items is also considered when determining negative samples. This thesis investigates two approaches: reducing the probability of selecting a part of unpopular items as negative samples (Top Discount Popularity Sampling), and increasing the probability of selecting a part of popular items as negative samples (Priority Popularity Sampling).

The effectiveness of the proposed method and the effect of negative sampling methods are evaluated with an offline experiment. The experimental result on dataset #nowplaying-rs [48] that is generated from all tweets with hashtag #nowplaying posted in 2014 and LFM-1b dataset [49] shows that (1) the proposed method outperforms wALS (weighted Alternating Least Squares), which is one of popular music recommendation algorithms based on matrix factorization. (2) it is also confirmed that performance of recommendation could be improved further by using appropriate context information, (3) when dataset has lower density, popularity-based sampling methods (Top Discount Popularity Sampling, Priority Popularity Sampling) have positive effect on prediction accuracy.

Chapter 2

Related work

2.1 Recommender system

2.1.1 Overview

With development of communication technology and computer-chip industry during these 30 years, people can access network more easily and quickly than ever before. Available devices are not only personal computers, but also wireless and mobile devices. According to the 2017-2022 white paper of Cisco Systems, Inc.[1], monthly global mobile data traffic will be 77 exabytes(=2^{30}gigabytes) by 2022, and annual traffic will reach almost one zettabyte(=2^{40}gigabytes), which is 7 times higher than that in 2017. Unprecedented enormous amount of data has already caused a problem of information overload [5]. However, people's information processing ability have not changed greatly. Therefore, a gap between huge volume of data and information processing ability of people has been large. In other words, users are confused and unable to make a decision when facing on enormous volume of data. To reduce information overload when searching the most relevant information from enormous data, as a subset of information filtering systems, recommender systems have been studied from 1990s [4].

Recommender systems are software tools which recommend suitable item to a user who is making decision on the basis of information about items or him/her self [2]. A meaning of items in this context does not only include real products such as books, music albums, and daily necessities, but also web services such as travel, online news and social networking services. Moreover, explicitly gathering the rating given by users in the past and implicitly monitoring users' behavior such as songs heard and web sites visited are two methods to obtain information required for recommendation. Regarding the type of information, demographic features of users (age, gender and nationality), item content (genre, release date and other description of item in those content), social (followers, followed and tweets) and miscellaneous context information (GPS location, time, weather)

[1] https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

could be used in recommender systems [1].

This section describes filtering algorithms used by recommender systems, its application, and existing problems.

2.1.2 Filtering algorithms

An essential part of recommender systems is how to use aforementioned information to filter information (items) by algorithms suitable for target application. The two most common filter algorithms are CBF (content-based filtering) [15] [43] and CF (collaborative filtering) [13] [26] [27].

CBF makes recommendation on the basis of content features of items that a user has bought, visited, watched, listened, or rated higher in the past. Items similar to those items will be recommended to this user. Figure 2.1 illustrates CBF, in which user A consumed item 1 in the past, and a recommender system will recommend item 2 to user A, because item 2 has similar content features to item 1. For example, in a music streaming service, if a user usually listens jazz songs, a recommender system will probably recommend jazz songs which is not discovered by this user yet to him/her.

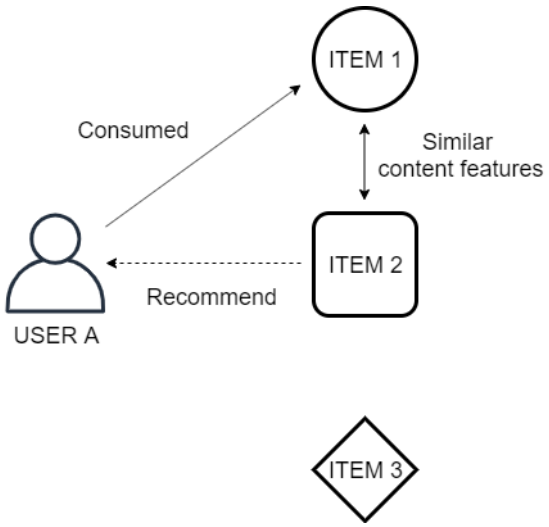

Figure 2.1: Content-based filtering

CF helps a user to make choices on the basis of the opinions of other users who have similar interests to this user in the past [13]. Depending on target objects for calculating similarity, CF can be divided into user-based and item-based one. The most significant difference of CF from CBF is that domain knowledge and content features are unnecessary when using CF. Because CF is an opinion-based filtering algorithm, the similarity between users/items is calculated on the basis of rating history of users. In the case of user-based CF, as shown in Figure 2.2, a recommender system judged user A is similar to user B because both of them consumed item 1 in the past. Therefore, item 2 that was consumed by user A

will be recommended to user B. Any item can not be recommended to user C because s/he does not share any items with other users. In the case of item-based CF, as shown in Figure 2.3, a recommender system judged item 1 is similar to item 2 because both of them were consumed by user A. Therefore, item 2 will be recommended to user B who consumed item 1. Item 3 can not be recommended to any user because it was only consumed by user C.

A rating matrix stores ratings given by users to items. Depending on whether constructing a model from a rating matrix or not, CF can be classified into memory-based (nearest-neighbor) and model-based (matrix factorization) one. Nearest-neighbor technique calculates similar users/items directly from a rating matrix. Pearson correlation, cosine, adjusted cosine, constrained correlation, mean squared differences, and Euclidean are the most commonly used similarity metrics [1].

Matrix factorization models [14] recommend an item to a user on the basis of latent factors extracted from a rating matrix. These models are known to be superior to nearest-neighbor methods in many cases [14] [26]. For example, Koren et al. achieved 10.05% improvement over nearest-neighbor algorithms and won the Netflix Prize competition[1]. Matrix factorization models factorize a rating matrix: both users and items are transformed to a joint latent factor space. User-item interactions are modeled as inner products of a user and an item in the transformed space. Details of matrix factorization are described in Sec. 2.2.2.

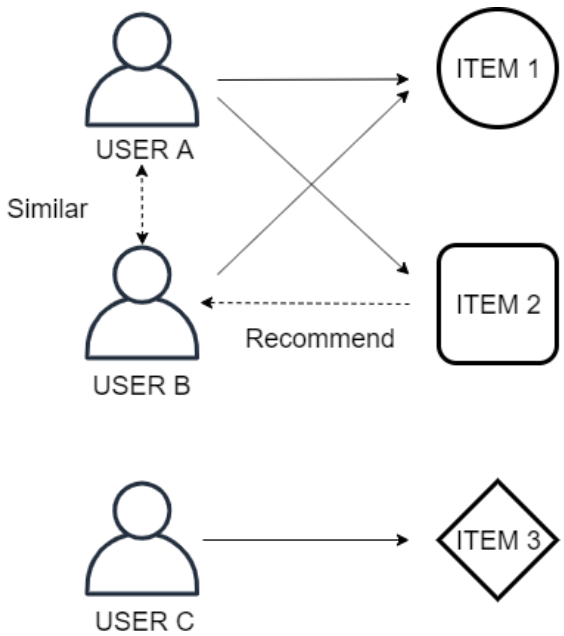

Figure 2.2: User-based collaborative filtering

In marketing literature, a demographic profile of a user is used by marketers since the late 1900s [21]. In the field of recommendation, DF (Demographic

[1]https://www.netflixprize.com/

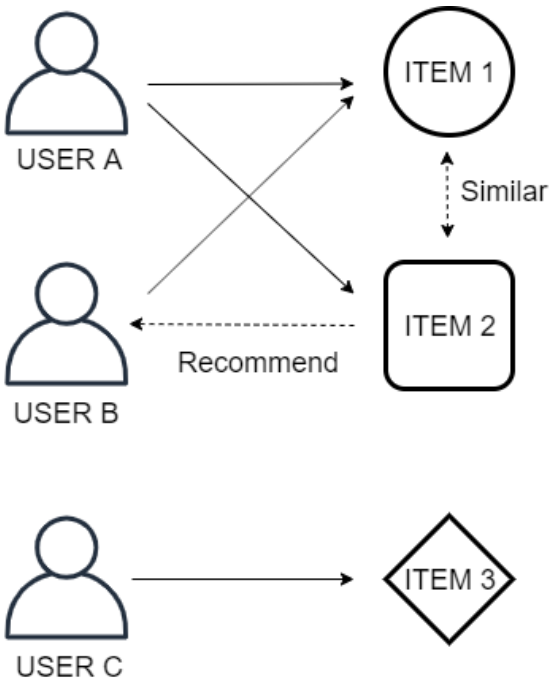

Figure 2.3: Item-based collaborative filtering

filtering) [16] refers to the technique for recommending an item to a user on the basis of the demographic profile of this user (e.g. age, country, nationality, gender, language, marital status, family size, income, religion, race, occupation, etc). It assumes that those who have similar personal demographic attributes will have similar preference. For example, females have more interest in cosmetics than males, users listen music according to the their age and/or country in general. As shown in Figure 2.4, user A and B have similar demographic profile. Therefore, item 1 that was only consumed by user B will be recommended to user A, and item 2 that was only consumed by user A will be recommended to user B.

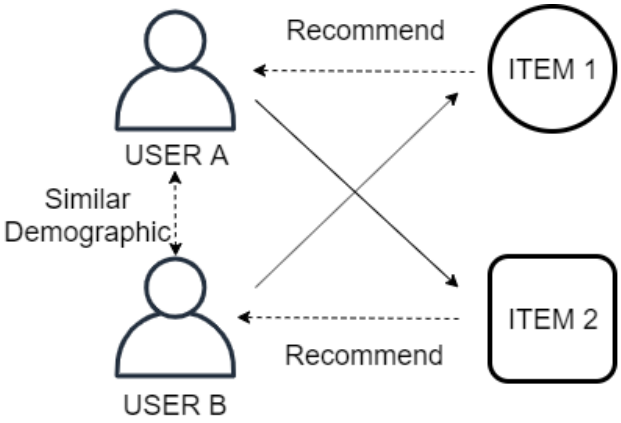

Figure 2.4: Demographic filtering

Knowledge of human/computer interaction also can be utilized in recommender systems. KBF (Knowledge-based filtering) [17] makes recommendation on the basis of specific queries that were made by a user. KBF collects domain knowledge as the form of relevant rules about interaction between users and items. When user A submits a query such as "I want to find an item that looks like a square," a recommender system finds an appropriate (i.e. square) item for this user, as shown in Figure 2.5. Results of recommendation can be improved by a user feedback given to the recommended items.

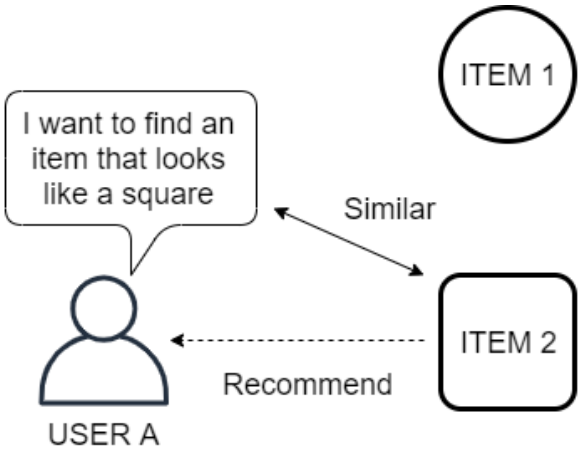

Figure 2.5: Knowledge-based filtering

It is claimed that combining different filtering algorithms to constructing a hybrid recomender system is an effective approach to improve performance of recommender systems [18]. CF is often combined with CBF [19] and DF [20].

It was shown that using only one of filtering algorithms can not cover all situations and domains in practice [3]. In other words, appropriate filtering algorithms need to be considered depending on target applications. Joeran et al. summarized paradigms of news recommendation [8]. It was reported that CBF is more popular than CF in news recommendation domain. The same result was reported in research-paper recommendation, which is also document-based recommendation [7]. In contrast, implicit CF is usually used in music recommendation domain, because it is difficult to utilize content features of music items directly due to the semantic gap between low-level descriptors and the concepts of music listeners have [44] [33] [35]. In e-commerce domain, Castro et al. [11] separated recommendation tasks into LIP (low involvement product, such as books, albums and films) and HIP (high involvement product, such as appliance, cameras and musical instrument) recommendation task. In the case of LIP, CF is one of the most widely used algorithms. On the other hand, in the case of HIP, KBF is used to provide appropriate recommendations because HIP's click-to-buy rate is usually lower than that of LIP. On the other hand, CF and CBF may make duplicated and outdated recommendations in e-commerce domain due to the lack of purchasing cycles (e.g. weekly, monthly and seasonal), which is also

called commodity purchase cycle [12]. They proposed a modified CF by considering both users behaviors and commodity characteristics to make weekly and seasonal recommendations. Chen et al. claimed that reciprocal social or impression management issues should be considered in social recommendation domain [10]: for example, before adding a new friend, a user often considers how the new friend will be perceived by other friends. They reported that content matching (similar to CBF) and friend-of-friend (similar to CF) algorithms are often used in this domain. In tourism recommendation domain, mobility of users is essential, and recommendation should be done in different moments and places. Therefore, recommender systems should employ context-aware technique [9].

2.1.3 Problem

Recommender systems have been already applied in multiple domains. However, some problems still exist and need to be solved.

As noted in Sec. 2.1.2, CF could recommend an item to a user even without any domain knowledge and content features. However, data sparsity [22] and cold start problem [1] are unavoidable. Data sparsity problem means that most of items could not be recommended on the basis of the interaction of users and items because most of users experienced a limited number or genre of items. Cold start problem means that when starting up a recommender system, new users and items do not have interaction and it is difficult to obtain a sufficient amount of ratings for making reliable recommendation. As shown in figure 2.6, both of user A and user B consumed item 1. However, user C does not share any item with others. Therefore, it is difficult to recommend any item to user C. Item 3 is not consumed by any user. Therefore, it is difficult to recommend it to any users. Bobadilla et al. [1] classified cold start problem into the new item problem and the new user problem. Motivating users to rate new items could release the new item problem. In the case of the new user problem, using additional information such as demographic and social information of a user, and latent features that are extracted from items may be effective.

Accuracy-based metric such as precision, recall, F1 and MRR (Mean Reciprocal Rank) are most common methods to evaluate performance of a recommender system. Kaminskas et al. [23] claimed that other aspects of recommender systems such as diversity, serendipity, novelty and coverage should also be evaluated. Diversity is defined as dissimilarity of items or the number of item genres in a recommend list. Serendipity means to find valuable or pleasant items that are not discovered by oneself. Novelty is similar to serendipity, but it emphasizes recommending items which are different from what the user has seen before. On the other hand, high coverage means many available items (i.e. those stored in DB) can be recommended: higher coverage may benefit both satisfaction of users and profit of service providers [24].

With success of deep learning in computer vision and natural language processing, it has also been employed to recommender systems. However, repro-

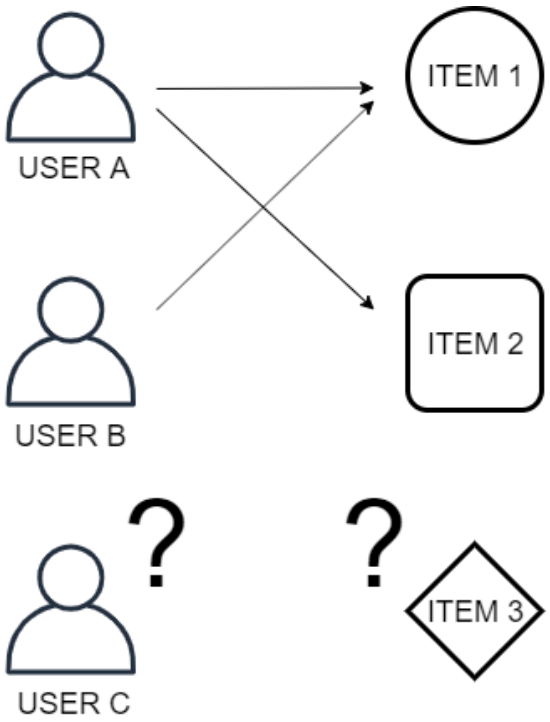

Figure 2.6: Cold start problem

ducibility is known to be a significant problem [6]. Maurizio et al. [6] collected 18 recommendation-related papers, which employed deep learning, from KDD, SIGIR, TheWebConf and RecSys. It was found that only 7 papers could be reproduced. Reminding 11 papers did not open their source code and dataset. Even though a source code is published, it dose not include data pre-processing, hyper-parameter tuning and exact evaluation procedures. They executed the experiment of above-mentioned reproducible 7 papers, and compared the results with user-based and item-based nearest neighbors CF with hyper-parameter tuning. As a result, 6 of 7 reproducible papers could not outperform the nearest neighbors CF.

2.2 Music recommendation

2.2.1 Overview

In music recommendation, explicit feedback such as rating is not usually available due to the way of listening to music and characteristic of music items. Instead, the interaction between users and items is recorded as a LE. Moreover, previous research [34] has shown that context information such as location, time, weather, demographic information and emotional state of the user plays an important role in the music recommendation domain.

On the other hand, the content such as key and temp and signal features of

music could be described at a lower level with the development of signal processing. However, the semantic gap between such low-level descriptors and the concepts of music listeners still exist [44]. In other words, content objects (e.g. key, tempo, melody, and harmony of a song) and signal features (e.g. duration, mode, energy and valence of a song) can not reflect the preference of a user precisely. For example, a user may not like a song only because the key of this song is major A or its acousticness is too high. Instead, s/he would be more inclined to listen a song that makes him/her feel pleasure. It is difficult to precisely represent these kinds of user preference with signal features. Table 2.1 shows the relation between signal features, content objects, and human knowledge.

This subsection introduces some related works of music recommendation, including implicit feedback for recommendation, and recommendation methods leveraging additional information such as context, social information and lower-level audio features of music.

Table 2.1: Semantic gap [44]

Human knowledge	personal identity, expectations, emotions, memories, understanding, opinion
Semantic gap	
Content object	key, tempo, rhythm, genre, melody, harmony, dyanamics, tags, motion, groove
Signal feature	duration, time, specturm, instrumentalness, liveness, speechiness, danceability, valence, loudness, acousticness, mode, energy

2.2.2 Implicit feedback

How to use implicit feedback from users effectively has been a crucial theme for recommendation regardless of domain [25]. In many cases, the preference of a user can be expressed as implicit feedback. Therefore, CF approaches, including memory-based (nearest-neighbor) and model-based (matrix factorization) ones, are usually used in most cases because implicit feedback usually denotes the existence/absence of an event, which is typically represented by a densely filled matrix [14]. This subsection introduces the mathematical definition of related work that are used implicit feedback, which include nearest-neighbor, wALS (weighted alternating least square), logistic matrix factorization, sampling-based ALS, and BPR (Bayesian personalized ranking). Table 2.2 shows list of symbols used in this thesis. Suppose that m users play n songs (music items) and the play counts are stored in a matrix $\boldsymbol{R}$.

Regarding nearest-neighbor approaches, similarity metric is essential to find

Table 2.2: List of symbols used in this thesis

Symbol	Description
m	Number of users
n	Number of items
$\boldsymbol{R}$	Rating matrix, $\boldsymbol{R} = (r_{i,j})_{m \times n} \in \mathbb{R}_{+}^{m \times n}$
$\boldsymbol{P}$	Preference matrix, $\boldsymbol{P} = (p_{i,j})_{m \times n} \in \{0,1\}^{m \times n}$
$\boldsymbol{C}$	Confidence matrix, $\boldsymbol{C} = (c_{i,j})_{m \times n} \in \mathbb{R}_{+}^{m \times n}$
α	hyper-parameter of wALS
f	Number of latent factors
k_1	Hyper-prameter of Okapi BM25
b	Hyper-prameter of Okapi BM25
$\boldsymbol{X}$	User-factor matrix, $\boldsymbol{X} \in \mathbb{R}^{m \times f}$
$\boldsymbol{Y}$	Item-factor matrix, $\boldsymbol{Y} \in \mathbb{R}^{n \times f}$
β_i	bias of user i
β_j	bias of item j
s	Number of records in a listening events matrix
t	Number of features in a listening events matrix (processed by one-hot encoder)
$\boldsymbol{E}$	Listening events matrix $\boldsymbol{E} \in \{0,1\}^{s \times t}$
Pr	Probability of a song being selected as a negative sample
q_i	Priority of a sample i
γ	Hyper-parameter of top discount popularity sampling, $\gamma \in [0,1]$
δ	Hyper-parameter of priority popularity sampling, $\delta \in [0,1]$
ϵ	Ratio of negative samples

similar user/item. Cosine, TF-IDF [30] and Okapi BM25 [31] are common similarity metrics in memory-based implicit CF. Depending on the activity of users, different people provide different number of implicit feedback from a user is different. Cosine metric could ignore the influence of these individual differences in terms of the volume of feedback information on inter-user similarity. Cosine similarity between pairs of users (u_1 and u_2) can be calculated as follows.

$$\mathrm{sim}_{\cos}(u_1, u_2) = \frac{\sum_j r_{1,j} r_{2,j}}{\sqrt{\sum_j r_{1,j}^2}\sqrt{\sum_j r_{2,j}^2}}, \tag{2.1}$$

where $r_{1,j}$ is the play history of user u_1 for j-th music item. Similarly, cosine similarity between pairs of items (i_i and i_2) is calculated as follows.

$$\mathrm{sim}_{\cos}(i_1, i_2) = \frac{\sum_i r_{i,1} r_{i,2}}{\sqrt{\sum_i r_{i,1}^2}\sqrt{\sum_i r_{i,2}^2}}. \tag{2.2}$$

On the other hand, TF-IDF [30] and Okapi BM25 [31], which are methods for measuring the importance of a word in a document. It can also be used to reflect preference of a user. TF-IDF similarity is calculated as follows.

$$\mathrm{sim}_{\mathrm{tfidf}}(i_1, i_2) = \frac{\sum_y \mathrm{tfidf}(i_1, u_y)\mathrm{tfidf}(i_2, u_y)}{\sqrt{\sum_y \mathrm{tfidf}(i_1, u_y)^2}\sqrt{\sum_y \mathrm{tfidf}(i_2, u_y)^2}}, \tag{2.3}$$

where, $\mathrm{sim}_{\mathrm{tfidf}}(i_x, u_y)$ represents the importance of an item i_x for a user u_y:

$$\mathrm{tfidf}(i_x, u_y) = \mathrm{tf}(i_x, u_y) \cdot \mathrm{idf}(i_x). \tag{2.4}$$

In this context, $\mathrm{tf}(i_x, u_y)$ represents the normalized frequency of appearance of i_x in the listening history of u_y. $\mathrm{df}(i_x)$ represents the number of users who played item i_x. They are calculated as:

$$\mathrm{tf}(i_x, u_y) = \frac{r_{y,x}}{\sum_j r_{y,j}}, \tag{2.5}$$

$$\mathrm{idf}(i_x) = \log\frac{m}{\mathrm{df}(i_x)}. \tag{2.6}$$

The problem of tf is that, active users would have higher value than others. It may cause a problem when using TF-IDF similarity. In order to mitigate this problem, Okapi BM25 [31] can be used to normalize the play counts of each user. Okapi BM25 similarity is calculated as follows.

$$\mathrm{sim}_{\mathrm{bm25}}(i_1, i_2) = \frac{\sum_y \mathrm{bm25}(i_1, u_y)\mathrm{bm25}(i_2, u_y)}{\sqrt{\sum_y \mathrm{bm25}(i_1, u_y)^2}\sqrt{\sum_y \mathrm{bm25}(i_2, u_y)^2}}, \tag{2.7}$$

$$\mathrm{bm25}(i_x, u_y) = \mathrm{idf}(i_x) \cdot \frac{\mathrm{tf}(i_x, u_y) \cdot (k_1 + 1)}{\mathrm{tf}(i_x, u_y) + k_1 \cdot (1 - b + b \cdot \frac{m \sum_j r_{y,j}}{\sum_{i,j} r_{i,j}})}, \tag{2.8}$$

where $\frac{\sum_j r_{y,j}}{\sum_{i,j} r_{i,j}}$ denotes normalized play counts of user u_y over all users. Difference of tf$(\cdot)$ among users can be adjusted by this term. k_1 and b are hyper-parameters of Okapi BM25.

Regarding matrix factorization-based approaches, Koren et al. proposed a matrix factorization based CF method called wALS (weighted alternating least square) for implicit feedback datasets [26]. They consider the decomposition $\boldsymbol{P} = \boldsymbol{X}\boldsymbol{Y}^T$, where $\boldsymbol{P} = (p_{i,j})_{m\times n} \in \{0,1\}^{m\times n}$, $\boldsymbol{X} \in \mathbb{R}^{m\times f}$, and $\boldsymbol{Y} \in \mathbb{R}^{n\times f}$, as shown in Figure 2.7. f denotes the number of latent factors. As an implicit feedback such as play count is not as reliable as explicit rating, it is difficult to make recommendations by solving the same loss function as methods based on matrix factorization. They introduced a confidence computation to a loss function based on playing count and considered all missing LEs as negative samples as shown in Equation (2.9).

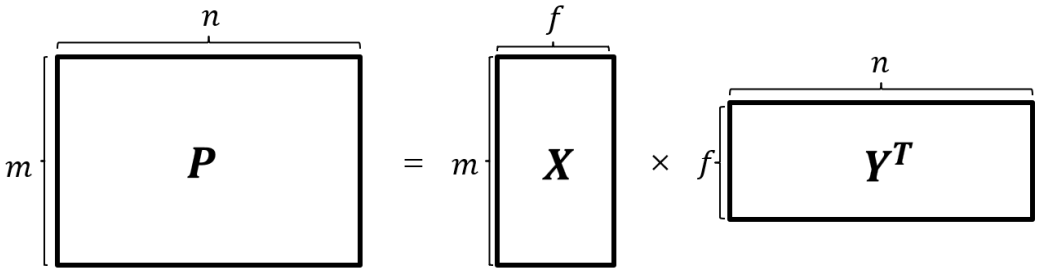

Figure 2.7: Matrix Factorization

$$L(x_i, y_j) = \sum_{i,j} c_{i,j}(p_{i,j} - x_i^T y_j)^2 + \lambda(\sum_i ||x_i||^2 + \sum_j ||y_j||^2), \tag{2.9}$$

where x_i, y_j is the latent factors of users and items, $p_{i,j}$ represents the element of user-item binary matrix representing whether or not a user i played a music item j.

$$p_{i,j} = \begin{cases} 1, & r_{i,j} > 1 \\ 0, & r_{i,j} = 0 \end{cases}, \tag{2.10}$$

where $r_{i,j}$ denotes the play counts of user i to music item j. $c_{i,j}$ represents the element of confidence matrix.

$$c_{ij} = 1 + \alpha r_{ij}, \tag{2.11}$$

where α is a hyper-parameter.

In order to mitigate the impact of user and item biases and improve performance of wALS, Johnson et al. proposed logistic matrix factorization, a probabilistic model for matrix factorization with implicit feedback [32]. They let $l_{i,j}$ denote the event that user i prefers item j. Probability of this event occurring is

defined as a logistic function.

$$p(l_{i,j}|x_i, y_j, \beta_i, \beta_j) = \frac{\exp(x_i y_j^T + \beta_i + \beta_j)}{1 + \exp(x_i y_j^T + \beta_i + \beta_j)}, \tag{2.12}$$

where β_i, β_j are user bias and item bias, respectively. The loss function can be rewrote as a likelihood function of rating matrix $\boldsymbol{R}$ given bias β and factor matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$. α is a hyper-parameter that is the same as α in Equation (2.11).

$$L(R|\boldsymbol{X}, \boldsymbol{Y}, \beta) = \prod_{i,j} p(l_{i,j}|x_i, y_j, \beta_i, \beta_j)^{\alpha r_{i,j}} \left\{1 - p(l_{i,j}|x_i, y_j, \beta_i, \beta_j)\right\}. \tag{2.13}$$

Instead of using all missing ratings as negative samples, various methods that reduce the number of negative samples have been proposed. Rong et al. [27] proposed sampling-based ALS for implicit feedback recommendation. Their method considers only a part of missing ratings on the basis of negative sampling. Three types of sampling approach are proposed: uniform random sampling (all missing data share the same probability to be sampled as negative), user-oriented sampling (if a user has played more items, item that s/he has not viewed could be negative), item-oriented sampling (if an item is viewed by fewer users, it may be a negative sample). These sampling-based approaches approximate the exact solution with much lower computational costs for large scale sparse datasets than original wALS. Furthermore, a method that dynamically chooses negative samples from the ranked list of Top-N recommendation produced has been proposed [28]. As another approach to solve negative samples problem, Steffen et al. [29] have introduced a generic optimization criterion called Bayesian personalized ranking optimization (BPR-OPT), which is derived from maximum estimator for optimal personalized ranking. Their proposed method involved pairs of items to generate a more personalized ranking for each user instead of using all of the missing ratings as negative samples.

2.2.3 Context information on music recommendation

Marius et al. [34] have classified context information on music domain into user-related ones (emotional state, demographic information, and activity of user), environment-related ones (location, current time, weather and noise level) and multi-media (image of album, lyrics, and review of a song).

Deng et al. [35] have explored user's emotions in microblogs service for music recommendation. In accordance with the hypothesis that users have higher similarity each other when they played similar music in the same emotional state, this method utilizes the text of microblogs service to extract users' emotions at different granularity levels during different sizes of a time window. Users' emotion when playing a song is utilized to calculate similarity among users and items. Wang et. al [36] have enhanced emotion-aware approach by modeling the relations among user, music, and emotion as an emotion aware graph. Marius et al.

[37] proposed a method to recommend music items suited for places of interest on the basis of the auto-tagging and the knowledge of the semantic relations among items.

Regarding the utilization of multi-media information, Martin et al. [38] proposed a pre-filtering method, which divides users into clusters by the name of playlists given by user and recommends music items for each user cluster with nearest-neighbor based CF. However, recommendation accuracy substantially varies among clusters because size of each cluster is different. To improve the performance of this approach, they have used the cluster label as one of the features of FMs to overcome the drawbacks of dividing user groups in advance by pre-filtering. This method does not sample any negative LE: instead, they let a rating equal to 1 when a certain user listened to a certain track in a certain cluster. For each <user, item, cluster> combination that dose not exist, they let rating equal to -1.

Schedl and Schnitzer [40] have introduced multiple types of context information into hybrid music recommendation. The process of similarity computation considers genres, styles, instruments, moods from "virtual artist document (i.e. web pages retrieved for an artist)," rhythmic features of music item, and user contexts such as timestamp and location.

Regarding different information from context, Alexandros et al. proposed a high-order SVD using social tagging [41] [42]. Markus et al. [45] introduced the concept of user mainstreamness to judge whether users prefer music items that are currently popular or not. When using audio signals for a recommendation, CNN (convolutional neural network) has been used to extract characteristics affecting user preference directly from audio signals [43]. This approach tries to reduce the semantic gap as noted in Section 2.2.1 between low-level descriptors and the concepts music listeners have.

Chapter 3

Proposed Method

The details of the proposed context-aware music recommendation method are described in this Chapter. First, the reason why this thesis employs FMs as well as how to use it, and an approach for selecting appropriate context information are described in Sec. 3.1. Three types of negative sampling methods: (1) random sampling, (2) priority popularity sampling, and (3) top discount popularity are introduced in Sec. 3.2. Sec. 3.3 describes a method that uses content features of music items to reduce the semantic gap. Figure 3.1 shows the flowchart of the proposed method.

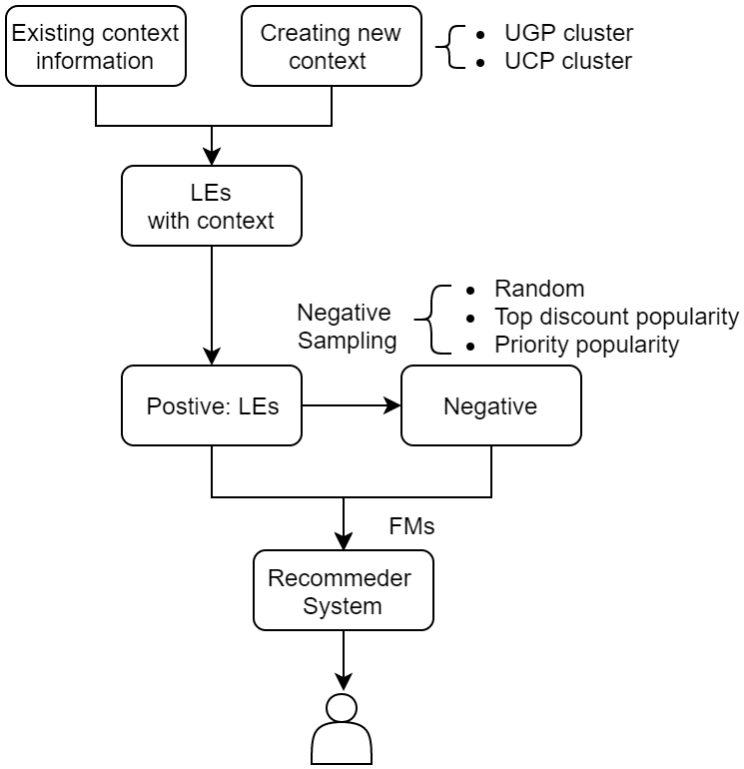

Figure 3.1: Flowchart of propose method

3.1 Context-aware recommendation

This thesis employs FMs [46] to recommend music items, in which users, music items and users' context are used as feature variables. FMs can not only learn the latent factors of users and items like matrix factorization methods, but also flexibly consider the interaction of users, items, and any other features with relatively low time complexity.

Table 3.1 shows an example of LEs, records of context information (language of, period of day, and played date when a user listened a music item. After applying the processes of label encoding (transform categorical context information to numerical variable) and one-hot encoding, these LEs can be described by a design matrix $\boldsymbol{E} \in \{0,1\}^{s \times t}$, where s, t are the number of LEs and feature variables of each LE, respectively. Figure 3.2 shows a design matrix. Given $\boldsymbol{e} \in \{0,1\}^t$ as a LE, which corresponds to a row of Figure 3.2, the model equation for 2-way FMs (degree $d = 2$) is defined as Equation (3.1).

Table 3.1: An example of listening events

LE	User	Item	Language	Period of a day	Date
0	1001	1001	en	Afternoon	2014-01-01
1	1001	1004	en	Morning	2014-01-01
2	1001	1002	jp	Evening	2014-01-01
3	1002	1003	en	Evening	2014-01-02
4	1002	1001	es	Afternoon	2014-01-02
5	1003	1001	en	Morning	2014-01-03
6	1003	1004	es	Evening	2014-01-03

$$\hat{r}(\boldsymbol{e}) := \omega_0 + \sum_{i=1}^{t} \omega_i e_i + \sum_{i=1}^{t-1} \sum_{j=i+1}^{t} \hat{\omega}_{i,j} e_i e_j, \tag{3.1}$$

where e_i is ith feature variable of LE $\boldsymbol{e}$, $\hat{r}(\boldsymbol{e})$ is the prediction value for $\boldsymbol{e}$. ω_0 is the global bias, ω_i is the weight of e_i. $\hat{\omega}_{i,j}$ shows the interaction of e_i and e_j with f factors. It can capture all single and pairwise interaction between these variables. $v_{i,k}, v_{j,k}$ are respectively the latent factors of feature i and j.

$$\hat{\omega}_{i,j} := \sum_{k=1}^{f} v_{i,k} v_{j,k}. \tag{3.2}$$

It is supposed that a part of context information may not have a positive effect on recommendations. To optimize the use of contexts, instead of using all of the existing context information, this thesis proposes a method to select context information. This method selects context information in according with the hypothesis that if the difference of users' preference among each possible value in a specific context were significant, this context would have a positive

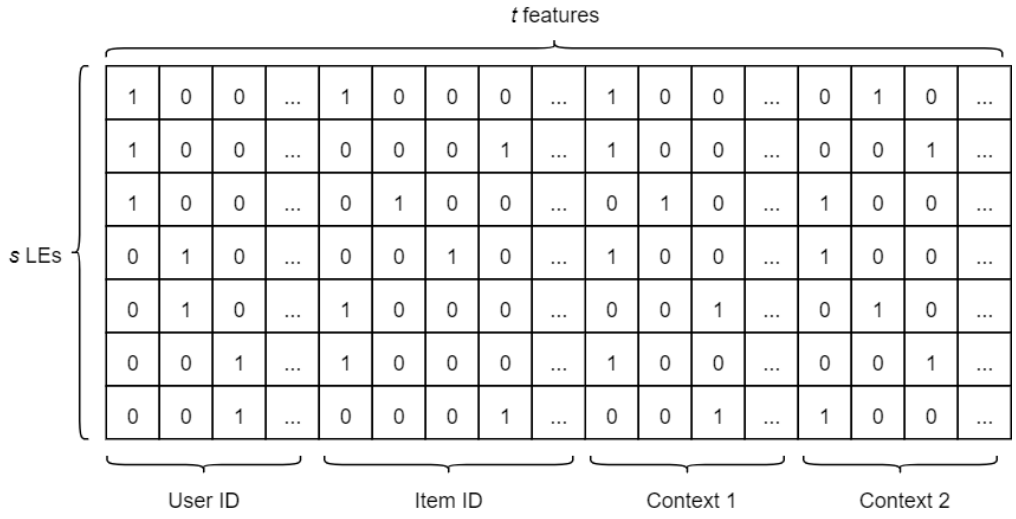

Figure 3.2: design matrix $\boldsymbol{E}$

effect on recommendations. The differences are represented as the distance table of a context feature. Figure 3.3 shows an example of a distance table of the native language of users. A value of a cell in this table represents the distance between two popularity distributions of different feature values (languages in the case of Figure 3.3). Let D_i and D_j denote two different popularity distributions of different feature values x_i and x_j. The distance of D_i and D_j can be calculated as follows.

$$\text{Dist}(D_i, D_j) = \frac{1}{2}\left\{\text{KL}(D_i||D_j) + \text{KL}(D_j||D_i)\right\}, \tag{3.3}$$

where $\text{KL}(D_i||D_j)$ denotes KL-divergence between distribution D_i and D_j. It is calculated as:

$$\text{KL}(D_i||D_j) = \sum_{x_k \in M_{i,j}} D_i(k) \log \frac{D_i(k)}{D_j(k)}, \tag{3.4}$$

where $D_i(k)$ and $D_j(k)$ are normalized popularity of k-th music item in distribution D_i and D_j respectively. $M_{i,j}$ represents a set of music items that relate with both feature values x_i and x_j. A music item that relates with only one of the feature variables will be removed when calculating the distance. For example, it is observed from Figure 3.3 that users who speak Japanese, French, Dutch and Indonesian have relatively different preference to other users. From this result, language could be selected as a context feature.

This thesis employs alternating least squares [47] to learn parameters ω_0, ω_i and $\hat{\omega}_{i,j}$ of FMs. In the learning process, both positive and negative samples are necessary. However, only positive samples (LEs) could be obtained directly when users play music items. Therefore, this thesis considers utilizing negative sampling to find negative samples for solving this problem.

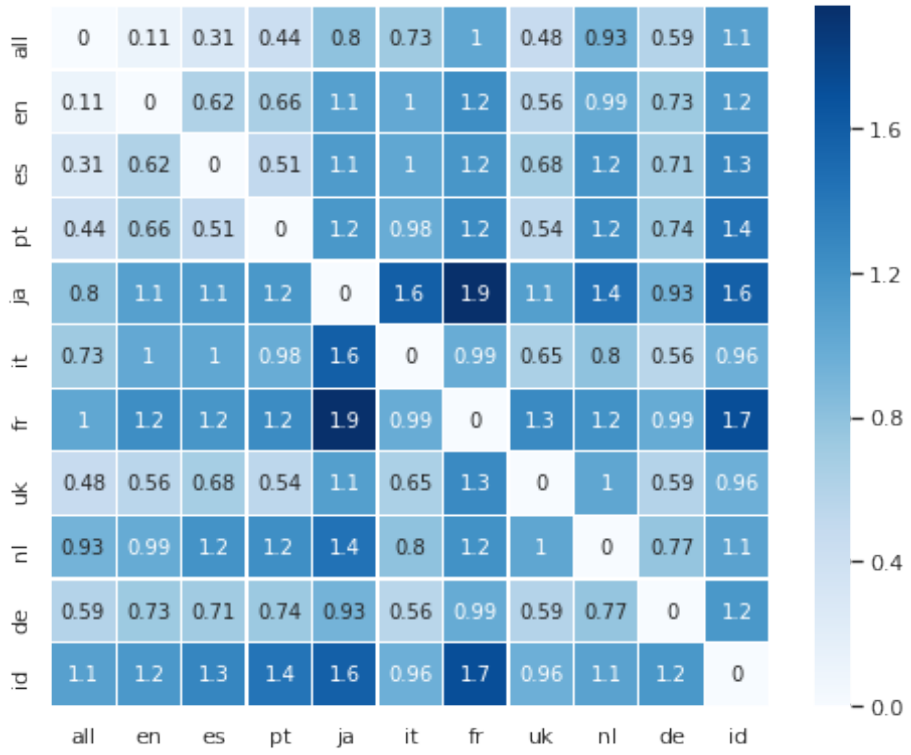

Figure 3.3: an example of a distance table of a context

3.2 Negative sampling

Regarding all music items a user has yet to be listened as negative is a naive method. Although it could be effective in matrix factorization methods such as wALS [26], the number of negative samples becomes huge, which causes a problem of time complexity especially when training a model by FMs. To solve this problem, this thesis selects a part of music items that a user has not yet listened to as negative samples. This thesis proposes three types of negative sampling methods, which are compared in the experiment.

Random Sampling this method selects negative samples in according with the hypothesis that **if a music item is not played by a user in some context, s/he would not be interested in it in that context.** When training FMs model, for each LE, this method selects music item(s) on random from music items that are not played in this LE and uses it (those) as negative sample(s). The same values of context information as the LE is added to the context information of the negative sample(s). The more active users will obtain more negative samples.

Top Discount Popularity Sampling although selecting negative samples on random is a comparatively fair method, it is supposed that the popularity of a music item could influence the result of learning FMs. Therefore, in addition to the hypothesis of Random sampling, this paper adopts another hypothesis that **if a music item is popular, but it not played by a user in some context,**

Table 3.2: An example of negative sample

LE	User	Item	Language	Period of a day	Date	Rating
0	1001	1001	en	Afternoon	2014-01-01	1
7	1001	1004	en	Afternoon	2014-01-01	0
1	1001	1004	en	Morning	2014-01-01	1
8	1001	1002	en	Morning	2014-01-01	0
2	1001	1002	jp	Evening	2014-01-01	1
9	1001	1001	jp	Evening	2014-01-01	0
3	1002	1003	en	Evening	2014-01-02	1
10	1002	1004	en	Evening	2014-01-02	0
4	1002	1001	es	Afternoon	2014-01-02	1
11	1002	1003	es	Afternoon	2014-01-02	0
5	1003	1001	en	Morning	2014-01-03	1
12	1003	1004	en	Morning	2014-01-03	0
6	1003	1004	es	Evening	2014-01-03	1
13	1003	1002	es	Evening	2014-01-03	0

it would be a negative sample for this users. A sampling method based on this hypothesis reduces the probability of selecting a part of unpopular items as negative samples. Probability of a music item i being selected as a negative sample is defined as:

$$\Pr(i) = \begin{cases} \frac{2}{(2-\gamma)n}, & \text{popular} \\ \frac{1}{(2-\gamma)n}, & \text{unpopular} \end{cases}, \tag{3.5}$$

where, $\gamma \in [0, 1]$ is a hyper-parameter to determine whether or not a music item is popular. Music items in top $(1 - \gamma) * 100$ percentage of popularity ranking list are defined as popular ones. After selecting negative sample(s) for each LE in accordance with the popularity, the context information is given in the same manner as Random sampling.

Priority Popularity Sampling Top Discount Popularity Sampling only focuses on a part of music items and uniformly increases the probability of selecting them as negative samples. It may influence the result of a recommendation. That is, a method that can tune the probability of a music item being selected as a negative sample more flexibly might be preferable. To realize this idea, this thesis employs a priority sampling method. It has already been utilized in reinforcement learning to sample appropriate experience replay according to TD-error in the buffer [50]. In the context of this thesis, TD-error is replaced with the popularity of a music item. Probability of a music item i being selected as a negative sample is defined as:

$$\Pr(i) = \frac{q_i^\delta}{\sum_k q_k^\delta}, \tag{3.6}$$

where δ is a hyper-parameter. q_i denotes priority of this music item, which can be calculated as:

$$q_i = 1/\text{rank}(i), \tag{3.7}$$

where rank(i) denotes the popularity ranking of a music item i. In other words, a music item that is popular has higher priority to be considered as a negative sample.

As shown in Table 3.2, we set rating $r(e_i) = 1$ for positive samples $r(e_i) = 0$ for negative samples. The ratio of negative samples are determined by the parameter $\epsilon = \frac{\#\text{Negative samples}}{\#\text{Positive samples}}$. In other words, the proposed method selects ϵ negative sample(s) for each LE. On the other hand, it is supposed that the attention to a music item would decrease since its release date. Based on this idea, this paper changes the size of time window to reduce the sample space of negative samples.

Furthermore, this paper considers both top-N recommendation and rating prediction task. In top-N recommendation task, a recommendation list that contains N music items that have higher predicted scores than others is generated for each LE. The purpose of this task is to recommend appropriate music items to users. On the other hand, rating prediction task predicts whether or not a specific music item is played by a user in a specific context.

3.3 Using content features of music items

The preliminary experiment by myself showed the same results as the previous work described in Sec. 2.2.1: content features of music items are difficult to be used directly to improve the performance of recommendation. Therefore, these content features need to be integrated more flexibly. This thesis proposes to use labels of clusters that are created from user profiling as a feature variable of FMs. User profiling could be obtained by this user's play counts of each genre (UGP, user genre profiling) [51] or mean of each content feature of all music items (UCP, user content profiling) in playing history of this user. UGP of a genre j in a user i's play history is defined as:

$$\text{UGP}_{i,j} = \frac{g_{i,j}}{|M_i|}, \tag{3.8}$$

where M_i is a set of all music items in i's play history. $g_{i,j}$ represents counts of genre j in M_i. UCP of a content feature j in a user i's play history is defined as:

$$\text{UCP}_{i,j} = \frac{s_i(j)}{|M_i|}, \tag{3.9}$$

where $s_i(j)$ is the summation of a content feature j's values of all music items in M_i.

Chapter 4

Experiment

This Chapter evaluates proposed context-aware music recommendation method by using two different datasets. Sec. 4.1 describes the details of these two datasets, which includes an overview of two datasets, the method of pre-processing, and an analysis of two datasets. Sec. 4.2 introduces details of experimental setup. Four evaluation metrics: Mean Percentage Ranking (MPR), Mean Reciprocal Ranking (MRR), Root Mean Squared Error, and Accuracy are described in Sec. 4.3. Sec 4.4 introduces result of experiments: (1) comparing proposed method with wALS, (2) influence of ratio of negative samples ϵ, (3) performance of each sampling method, (4) performance of proposed method when adding different context information, and (5) comparing proposed method with using signal features directly.

4.1 Dataset

4.1.1 Overview

This thesis uses #nowplaying-RS dataset[1] and LFM-1b[2] to evaluate proposed method. #nowplaying-RS is extracted from all tweets with hashtag #nowplaying posted in 2014. Poddar et al. [48] extracted from tweets the name of music items, artists, and the 6 context features (timestamp, tweet language, user language, time zone, hashtags, and sentiment) of a user when playing music items. 11 content features (instrumentalness, liveness, speechiness, danceability, valence, loudness, tempo, acousticness, energy, mode, and key) of these music items are also obtained from Spotify API[3]. Table 4.1 shows the description of these contents features. The number of users, music items, and LEs are 138,781, 346,273, and 11,639,541, respectively.

LFM-1b dataset is presented by Markus [49]. They collected LEs from more than 120,000 users of Last.fm from 2012 to 2013. Each LE includes user ID,

[1] http://dbis-nowplaying.uibk.ac.at/#nowplayingrs
[2] http://www.cp.jku.at/datasets/LFM-1b/
[3] https://developer.spotify.com/documentation/web-api/

Table 4.1: Description of content features

Feature	Value type	Description
key	int	The overall key of a track
mode	int	The modality (major or minor) of a track
acousticness	float	Whether a track is acoustic (range in [0,1])
danceability	float	How suitable a track is for dancing (range in [0,1])
energy	float	A perceptual measure of intensity and activity (range in [0,1])
instrumentalness	float	Whether a track contains no vocals (range in [0,1])
liveness	float	The presence of an audience in the recording (range in [0,1])
loudness	float	The overall loudness of a track in decibels (dB)
speechiness	float	The presence of spoken words in a track (range in [0,1])
valence	float	The musical positiveness conveyed by a track (range in [0,1])
tempo	float	The overall estimated tempo of a track in beats per minute (BPM)

music ID, artist ID, demographic and context features of a user (timestamp, nationality, age and gender). In addition to that, Markus and Bruce created UGP as introduced in Sec. 3.3, which was calculated on the basis of preference genre statistics of a user according to artists' genre (rnb ,rap, electronic, rock, new age, classical, reggae, blues, country, world, folk, easy listening, jazz, vocal, children's, punk, alternative, spoken word, pop, heavy metal) in his/her play history. Figure 4.1 shows a part of UGP, in which a row represents UGP of each user. The number of users, music items, and LEs in LFM-1b are 120,332, 32,291,134, and 1,088,161,692, respectively.

4.1.2 Pre-processing

#nowplaying-rs for avoiding the influence of tweet bots and extremely not active users, I removed the users who listened less than 10 and larger than 5000 music items from the #nowplaying-rs. After pre-processing, the number of users, music items, and LEs in #nowplaying-rs become to 18,946, 22,023, and 1,835,993, respectively. Timestamp is difficult to be used as a context feature. Therefore, I generated 4 features from each timestamp: DoW (day of week), Weekday/Weekend, Hour, PoD (period of day). I did not use hashtags and sentiment this time because of only a few LEs including them. In addition to those context features, 8 content features of music items: instrumentalness, liveness, speechiness, danceability, valence, tempo, acousticness, and energy, are integrated to

user_id	UGP_rnb	UGP_rap	UGP_electronic	UGP_rock	UGP_new age	UGP_classical	UGP_reggae
48795663	0.041353	0.019847	0.120831	0.165263	0.000981	0.001253	0.006958
10813538	0.014630	0.005148	0.022469	0.223237	0.002269	0.007112	0.004930
13925623	0.105057	0.096165	0.124403	0.122305	0.001849	0.001954	0.022271
11036381	0.017206	0.018738	0.162799	0.203882	0.001468	0.003288	0.007342
47461038	0.009068	0.013279	0.134668	0.180293	0.009762	0.013815	0.008721
45689780	0.024006	0.021234	0.049201	0.134485	0.001782	0.009058	0.010345
31379650	0.109677	0.013840	0.139874	0.177301	0.000967	0.002418	0.009723
2409301	0.001365	0.009143	0.106888	0.230408	0.009805	0.015486	0.004785
7227974	0.050336	0.057158	0.111206	0.121042	0.000072	0.000579	0.031629

Figure 4.1: User genre profiling

calculate UCP. To simplify the calculation of UCP, I do not use categorical features key and mode. Loudness is not used as well because most of the values are none. On the other hand, UGP can not be calculated in this dataset because of lack of genre information.

LFM-1b for LFM-1b, I modified the lower limit of play count of a user from 10 to 20 because the density of LFM-1b is higher than #nowplaying-rs significantly. After pre-processing, the number of users, music items, and LEs in LFM-1b become to 13,658, 139,303, and 40,761,277, respectively. Four features are generated from timestamp in the same way as #nowplaying-rs. UCP can not be calculated because signal features of music items are not included in this dataset. Summary of these two datasets are shown in 4.2.

4.1.3 Data analysis

Figure 4.2 and 4.3 show popularity distribution of music items in #nowplaying-rs and LFM-1b, respectively. X-axis denotes the Logarithm of played counts and Y-axis denotes the number of music items. It is observed that popularity distribution in #nowplaying-rs is similar to LFM-1b. Popular music items account for only a small part of the whole.

On the other hand, activity distribution of users in two datasets are respectively shown in Figure 4.4 and 4.5. X-axis denotes the Logarithm of play counts of users and Y-axis denotes the number of users. It shows that LFM-1b dataset contains much more active users than #nowplaying-rs. Whereas most of users in #nowplaying are not quite active, most of users in LFM-1b have middle activity.

For selecting the appropriate context, I used the method that is noted in Section 3.1. Figure 4.6 to 4.9 show results in #nowplaying-rs. 'all' denotes overall

Table 4.2: Summary of datasets after pre-processing

	#Nowplaying-rs	LFM-1b
#LEs	1,835,993	40,761,277
#user	18,946	13,658
#tracks	22,023	139,303
Timestap context	PoD, DoW, Hour, Weekday/Weekend	PoD, DoW, Hour, Weekday/Weekend
User context	Language and UCP cluster	Age, Nationality, and UGP cluster
Signal features	Instrumentalness, Liveness, Speechiness, Danceability, Valence, Tempo, Acousticness, Energy	-

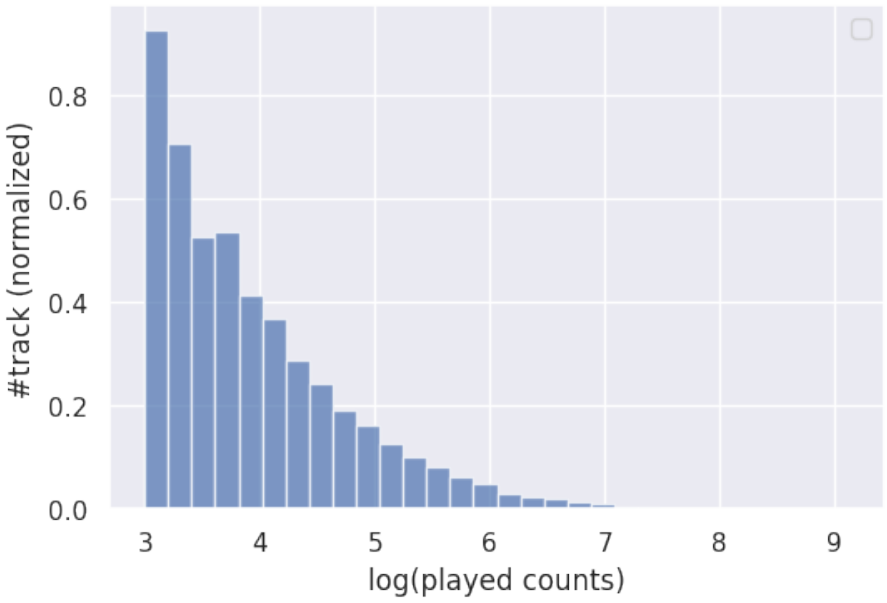

Figure 4.2: Popularity distribution of music items in #nowplaying-rs

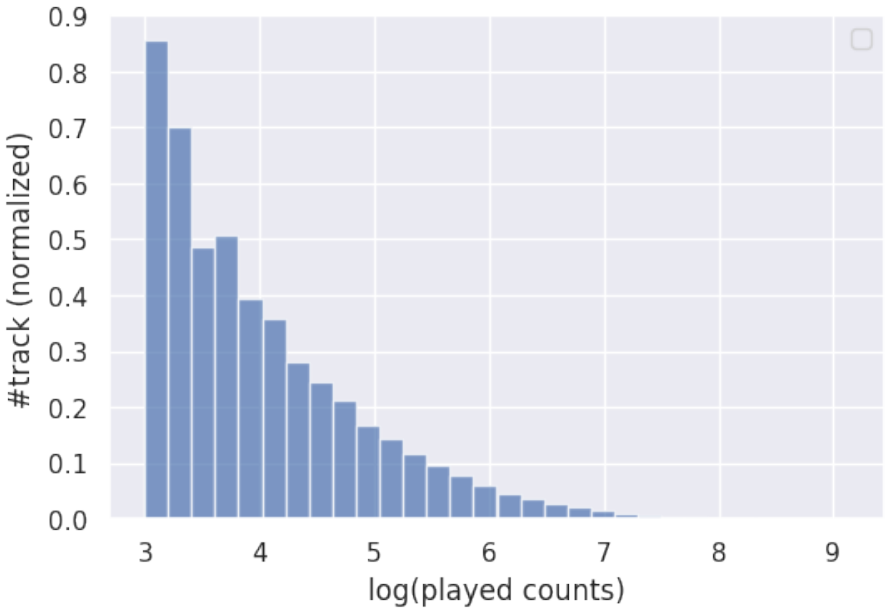

Figure 4.3: Popularity distribution of music items in LFM-1b

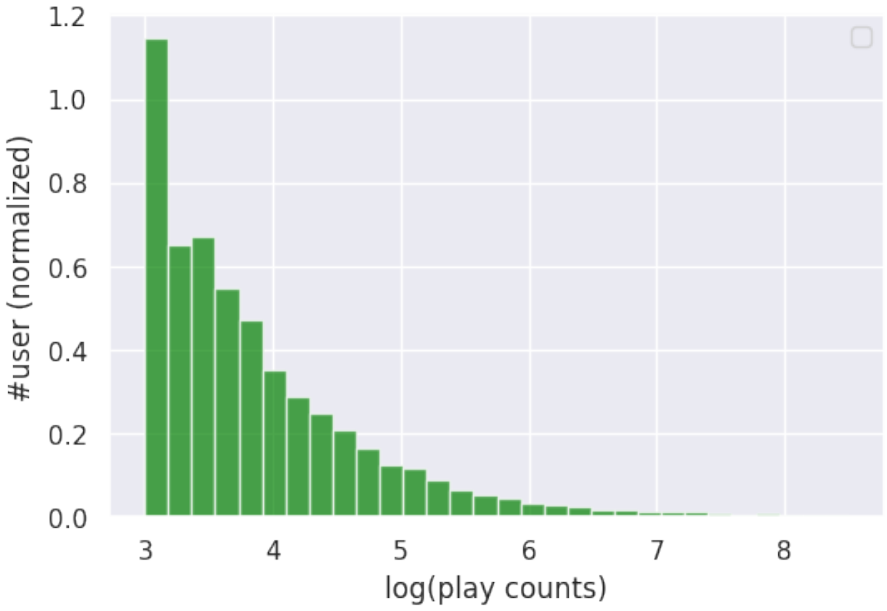

Figure 4.4: Distribution of users' activity in #nowplaying-rs

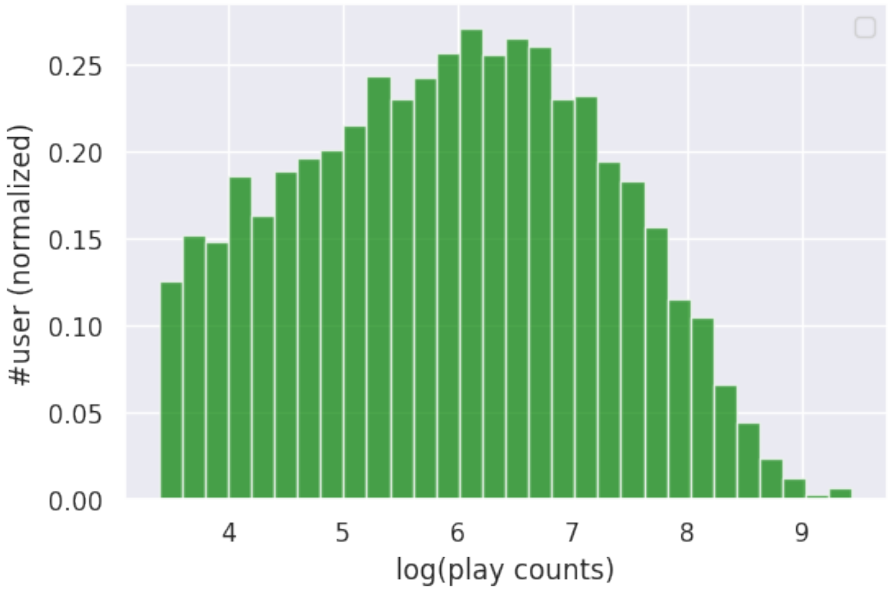

Figure 4.5: Distribution of users' activity in LFM-1b

popularity distribution. It is observed in Figure 4.6 that the users who speak Japanese (ja), French (fr), Italian (it), Dutch (nl) and Indonesian (id) have different preference for other users. As shown in Figure 4.7, Monday to Sunday are encoded to 0-6. The preference of users in the weekend (5, 6) is slightly different from weekday (0-4). Figure 4.8 shows that users have different preferences in the morning and evening. It is observed in Figure 4.9, which column/row corresponds to a cluster. Based on the result of preliminary experiment, the number of clusters is set to 5. The differences of preference among UCP clusters are larger than other contexts such as shown in Figure 4.6 to 4.8.

For further observation, Figure 4.10 shows a radar chart of UCP clusters. Each content feature is normalized to $[0, 1]$. It is observed that users in cluster 0 prefer to play music items that are more emotional (i.e. higher valence and danceability) and have more voice (i.e. speechiness). Users in cluster 1 prefer music items that have relatively higher tempo and energy. Users in cluster 2 do not represent significant preference. (i.e. all of 8 content features are moderate) Users in cluster 3 are interested in music items that have higher tempo and energy than users in cluster 1. Moreover, these music items have higher instrumentalness and liveness. Users in cluster 4 prefer to play music items that are emotional (i.e. higher valence and danceability) and have relatively higher acousticness.

Figure 4.11 to 4.15 show results in LFM-1b. It is observed that in Figure 4.11 users who are from Indonesia (id), Ukraine (ua), and Netherlands (nl), have different preferences for other users. Figure 4.12 shows that the difference among DoW is fewer than that in #nowplaying-rs. As shown in Figure 4.13, users have different preferences in the morning and evening, which is also observed in #nowplaying-rs dataset. Figure 4.14 shows that the preference between young

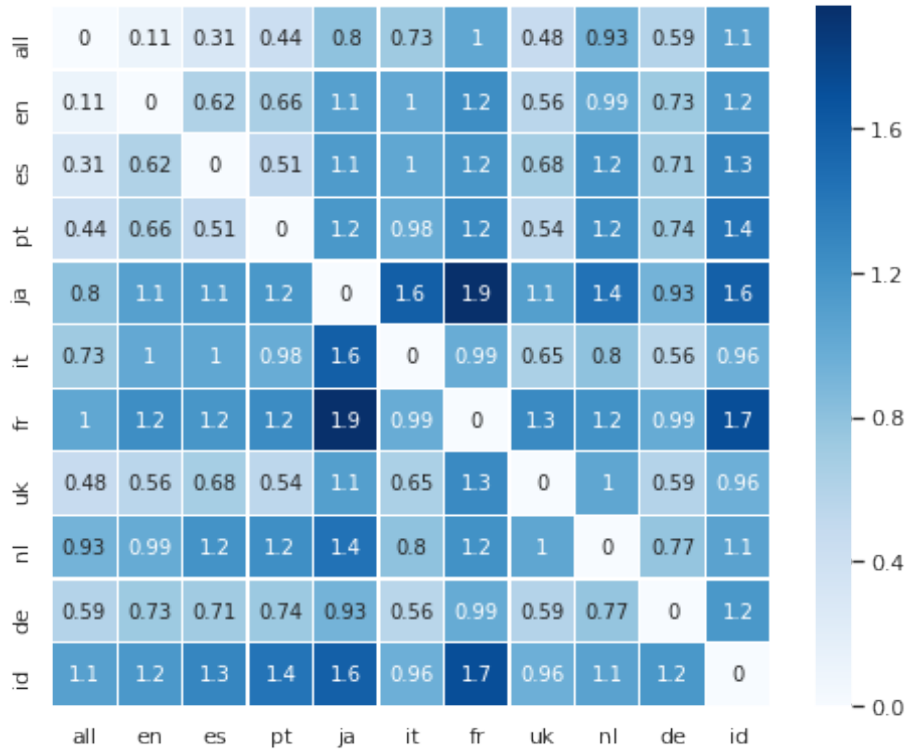

Figure 4.6: Distance table of 'language' in #nowplaying-rs (en: English, es: Spanish, pt: Portuguese, ja: Japanese, it: Italian, fr: French, uk:British English, nl: Dutcdh, de: German, and id : Indonesian)

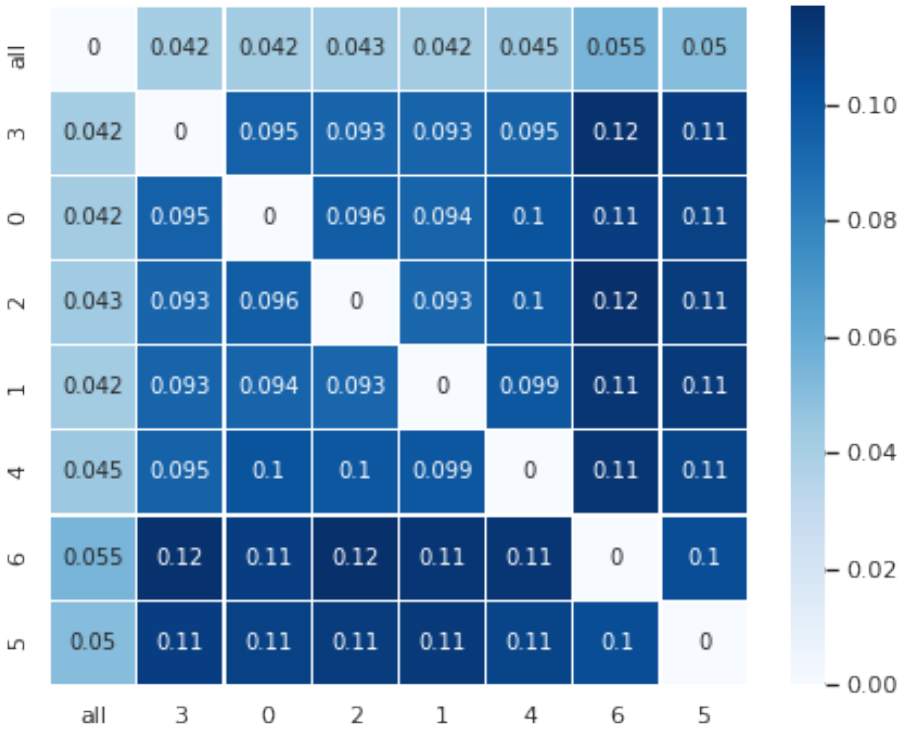

Figure 4.7: Distance table of 'day of week' in #nowplaying-rs

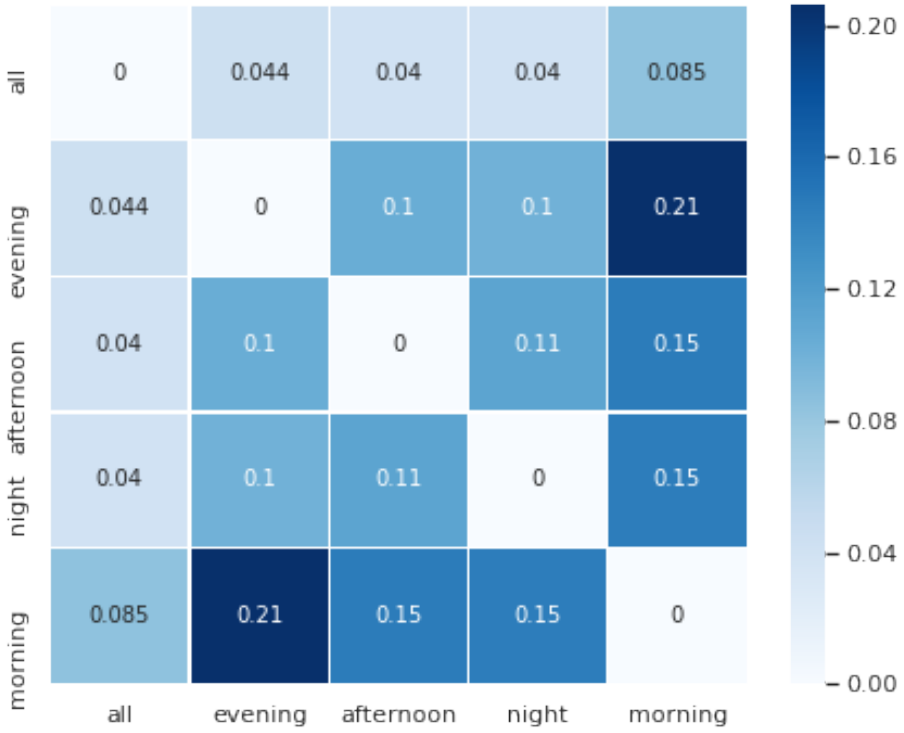

Figure 4.8: Distance table of 'period of day' in #nowplaying-rs

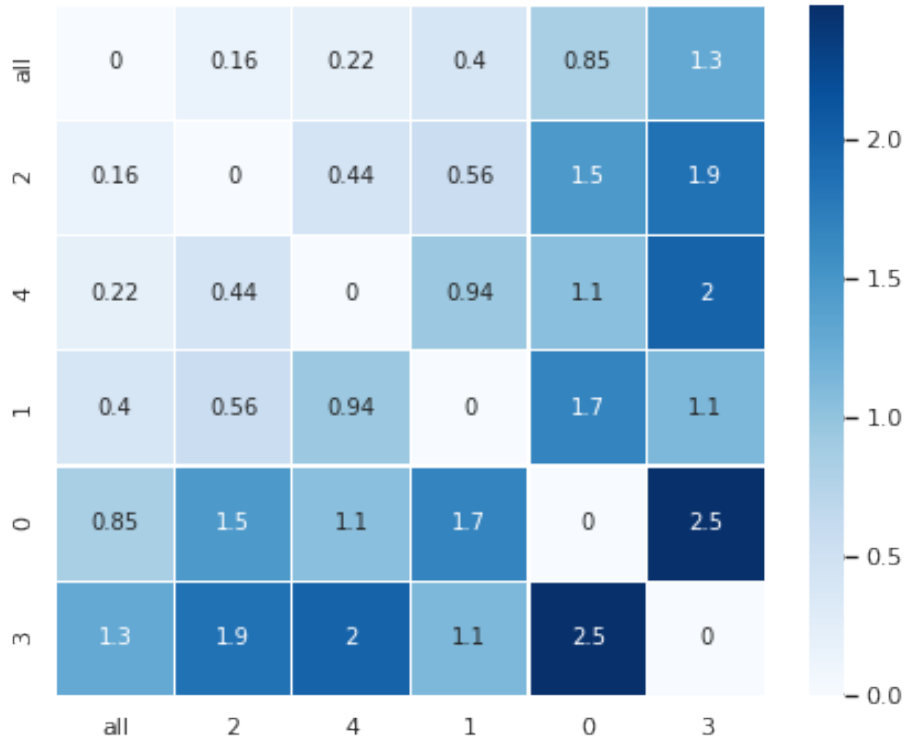

Figure 4.9: Distance table of 'UCP cluster' in #nowplaying-rs

users and the elderly is slightly different. As shown in Figure 4.15, which column/rows corresponds to a cluster. The number of clusters is set to 5, which is equal to the number of UCP clusters. The differences of preference among UGP clusters are larger than other contexts such as shown in Figure 4.11 to 4.14. Summarizing the above, users in LFM-1b do not represent a more significant difference than users in #nowplaying-rs.

For further observation, Figure 4.16 shows a radar chart of difference among each UGP clusters. It is observed that preference genre of users in cluster 0 is easy listening, folk, country, blues, and pop. Users in cluster 1 prefer "gentle" songs such as jazz, world, reggae, rap and RnB. Users in cluster 2 prefer "heavy" songs such as rock and heavy metal. Users in cluster 3 more inclined to listen to a song that has complicated melody such as electronic, classical and spoken word. Users in cluster 4 are interested in punk, alternative, and pop.

4.2 Experimental setup

As shown in Figure 4.17, this thesis uses a 5-fold real-life split strategy [52] based cross-validation to evaluate the performance of baseline method wALS and proposed methods because it is closer to real situation than usual cross-validation. The 5-fold real-life split strategy based cross-validation is repeated 5 times, and

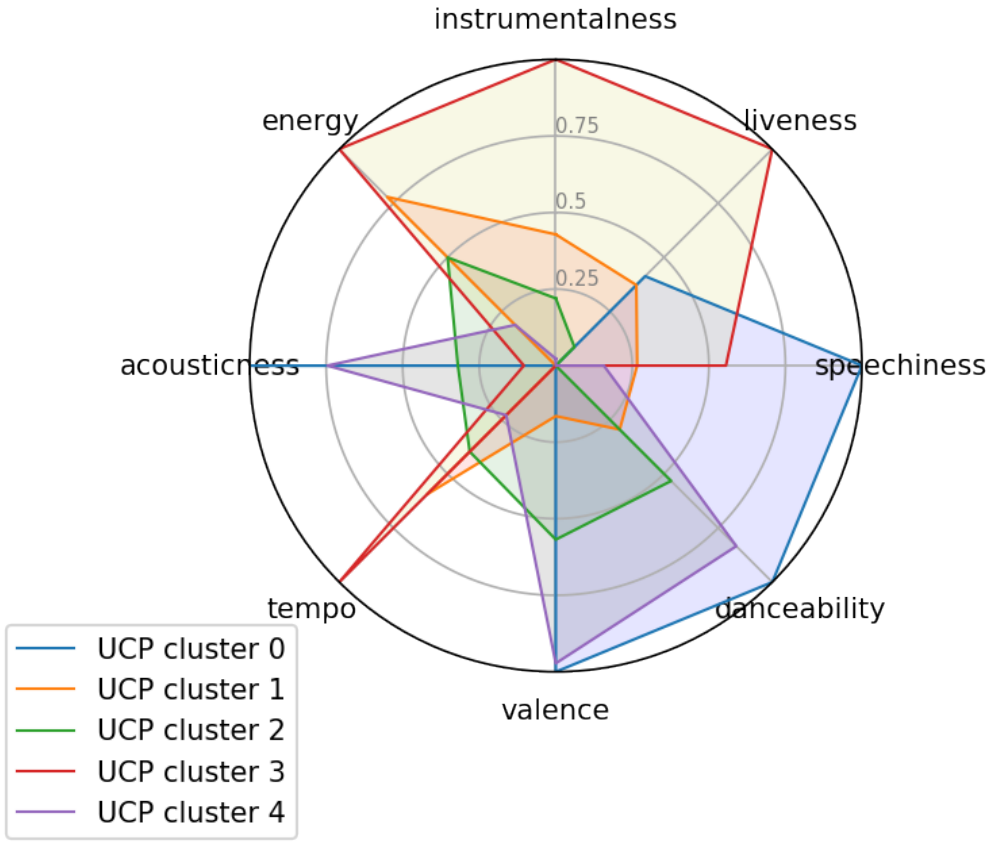

Figure 4.10: Difference among each UCP cluster

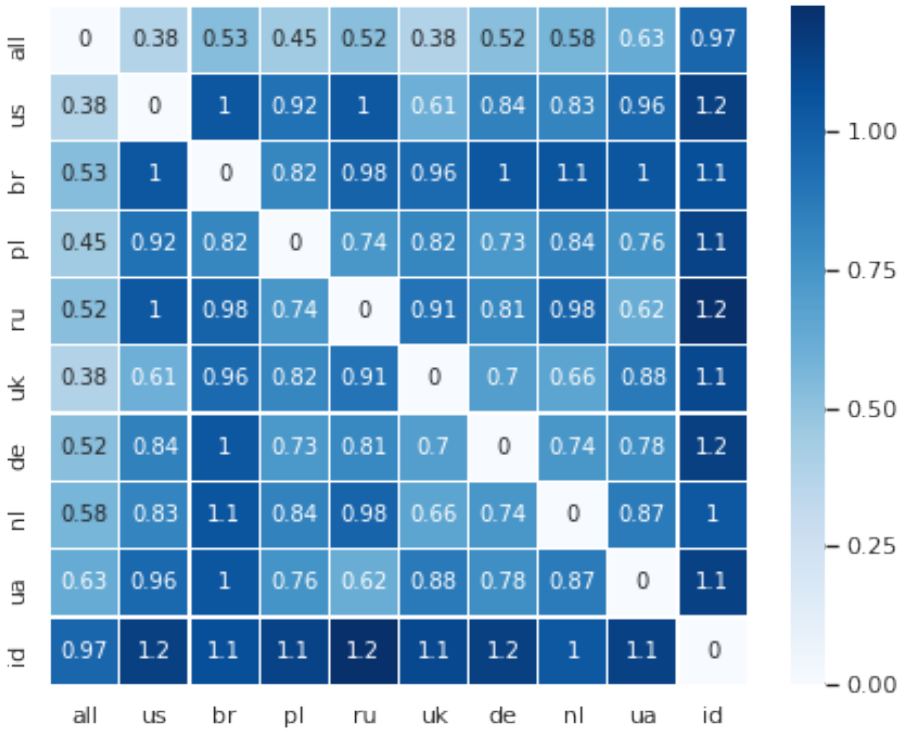

Figure 4.11: Distance table of 'nationality' in LFM-1b (us: America, br: Brazil, pl: Poland, ru: Russia, uk:British, de: German, nl: Netherlands, ua: Ukraine, and id : Indonesian)

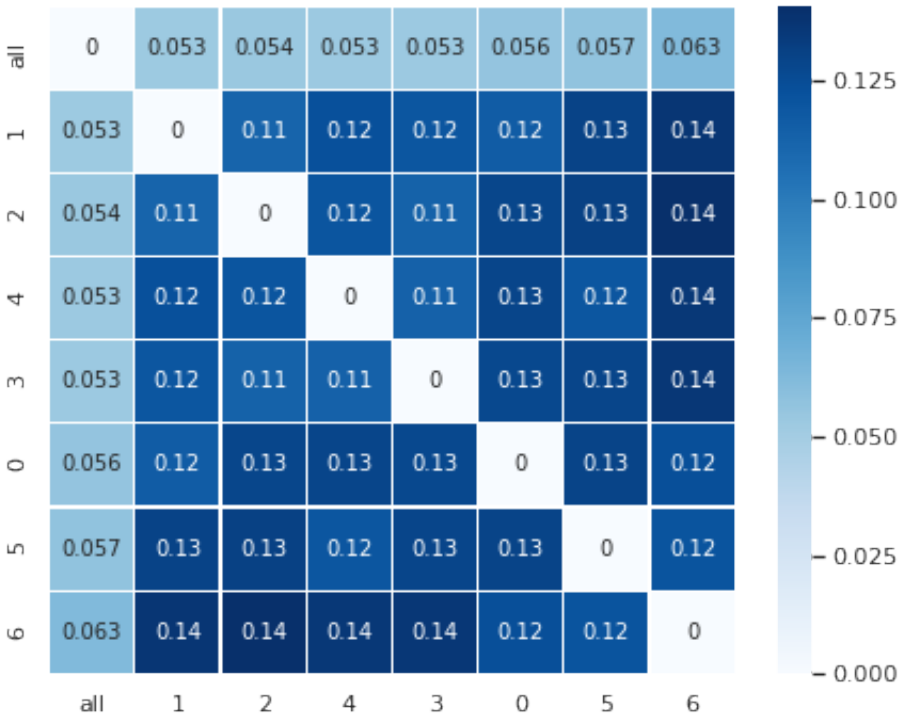

Figure 4.12: Distance table of 'day of week' in LFM-1b

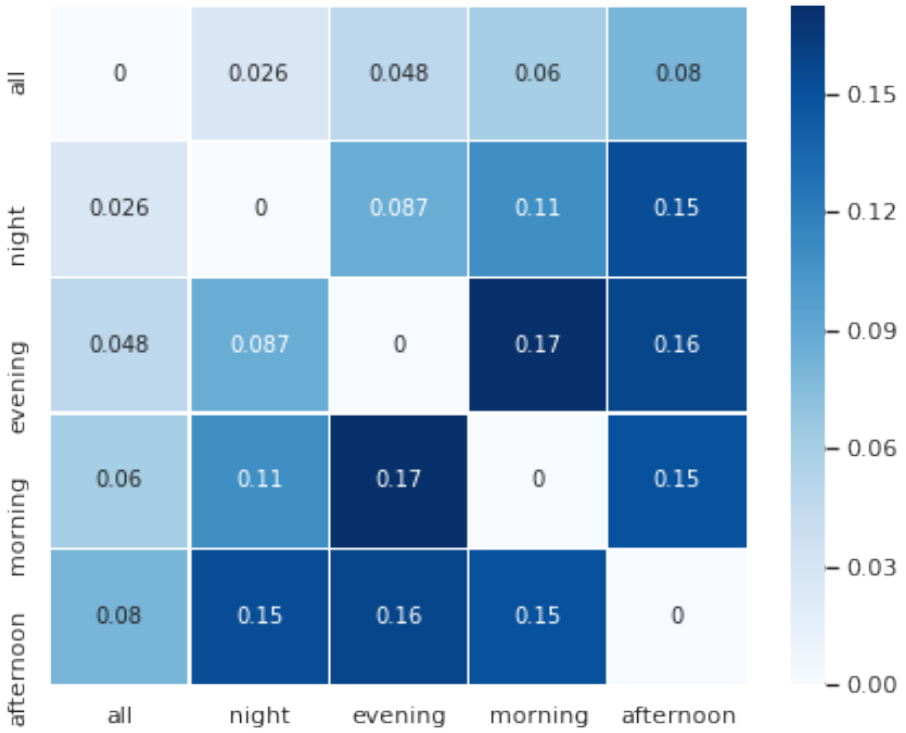

Figure 4.13: Distance table of 'period of day' in LFM-1b

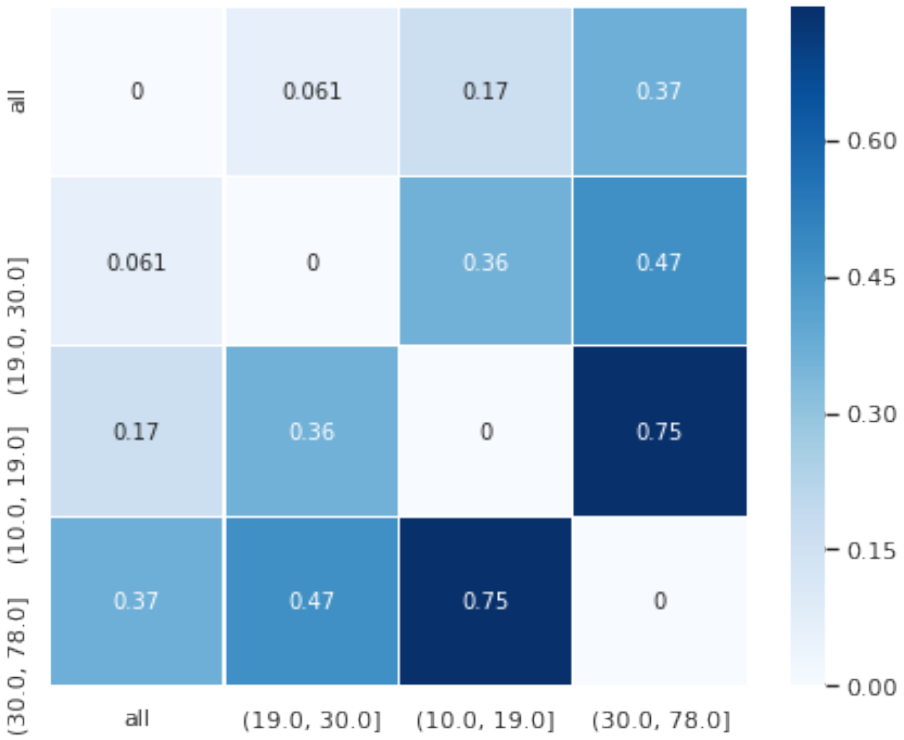

Figure 4.14: Distance table of 'age' in LFM-1b

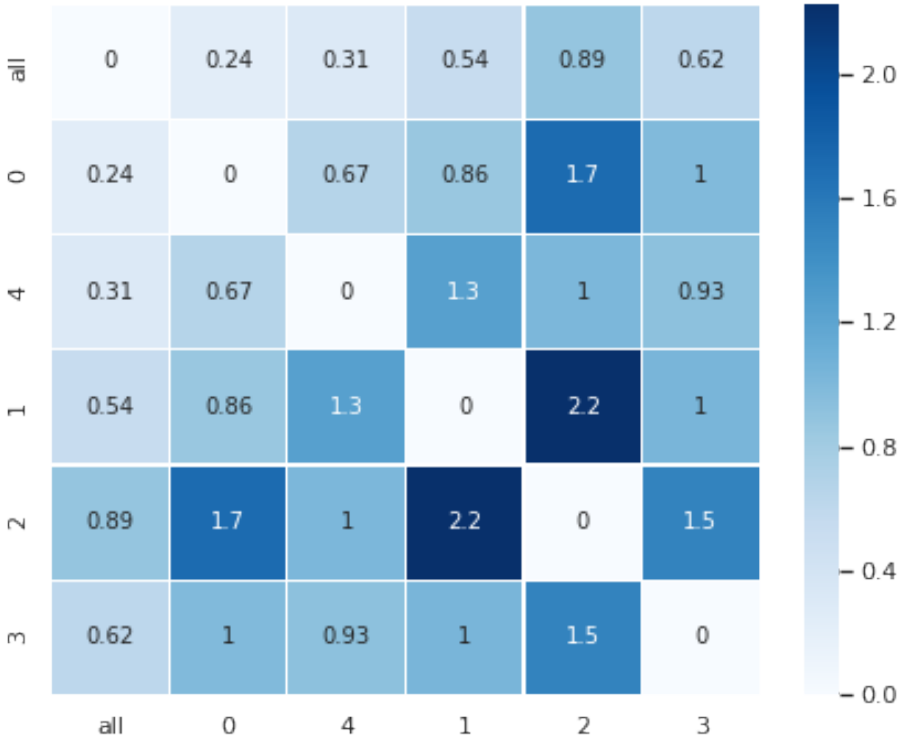

Figure 4.15: Distance table of 'UGP cluster' in LFM-1b

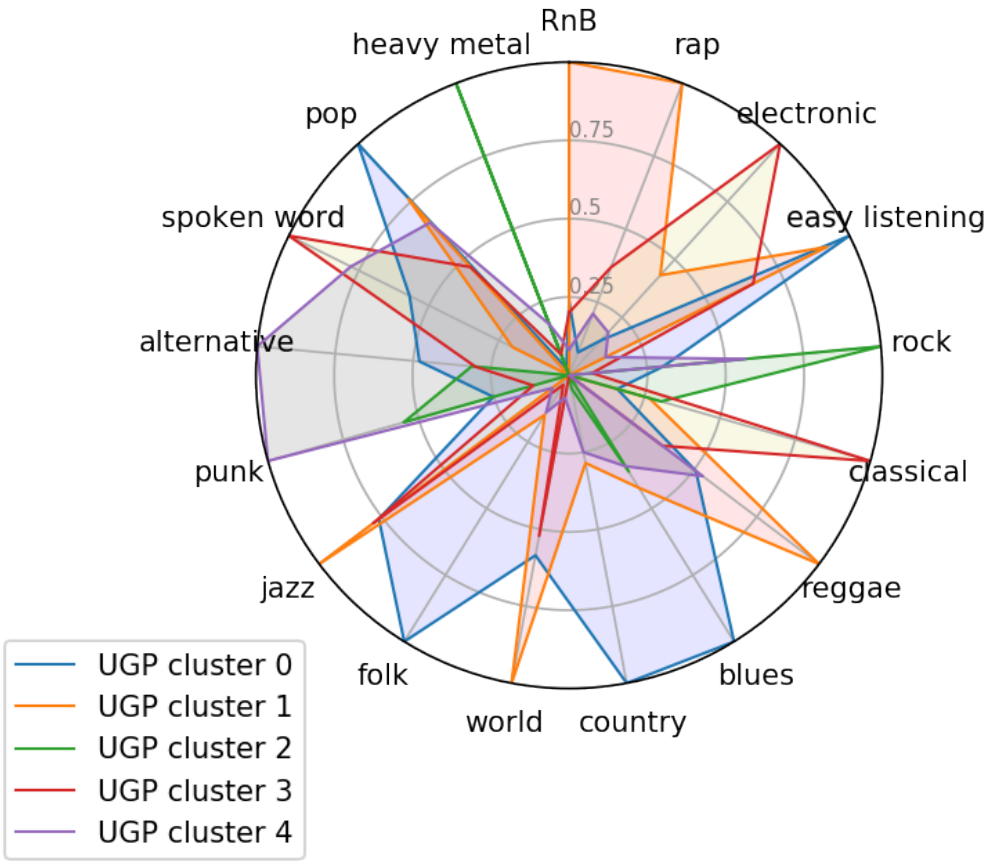

Figure 4.16: Difference among each UGP cluster

t-test is applied to confirm whether or not a significant difference to wALS is observed. Used parameters are $\alpha = 250$, $f = 15$, $\lambda = 0.05$ in #nowplaying-rs and $\alpha = 250$, $f = 35$, $\lambda = 0.05$ in LFM-1b for wALS, which are defined with hyper-parameter tuning.

The sample space of negative items for a LE only includes all music items that were played before the date of this LE. Selecting unreleased music items as negative sample is not appropriate. Therefore, release date of music items should be considered. However, the real release date does not exist in these two datasets. As alternative for the real release date, I considered the day that a music item was played as release day of this music item because. For the proposed method, the ratio of negative samples is set to $\delta = 3$. 8 signal features are used to generate UCP cluster in #nowplaying-rs as described in Sec. 4.1.2. The number of cluster of both UCP and UGP is set to 5 as noted in Sec. 4.1.3. The number of positive (LEs) and negative samples are balanced by bootstrap method [54] to avoid the imbalance problem. Because negative samples are a kind of virtual LE, they are not used in evaluation but only used in the process of training models (Figure 4.18) for ensuring the fairness of evaluation.

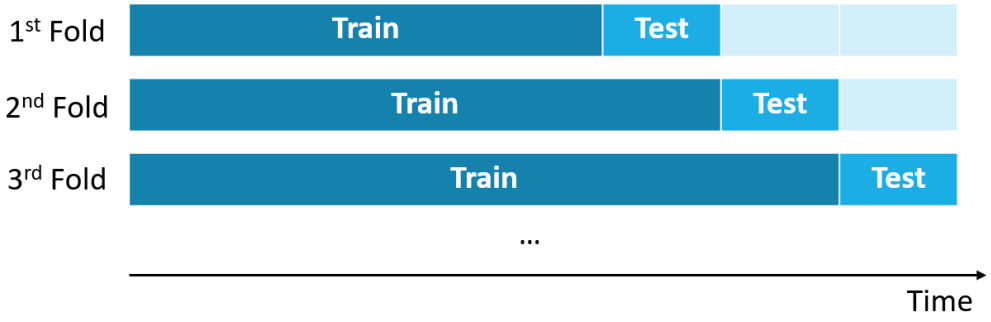

Figure 4.17: Real-life split strategy based cross-validation

Figure 4.18: Negative samples are not used in evaluation but only used in the process of training models for ensuring the fairness of evaluation.

4.3 Evaluation metric

This paper employs MPR and MRR to evaluate the performance of each method for top-N recommendation task.

$$\mathrm{MPR} = \frac{\sum_{i,j} h_{i,j} \mathrm{rank}_{i,j}}{\sum_{i,j} h_{i,j}}, \tag{4.1}$$

$$\mathrm{MRR} = \frac{\sum_{i,j} h_{i,j} (1/\mathrm{rank}_{i,j})}{\sum_{i,j} h_{i,j}}, \tag{4.2}$$

where $\mathrm{rank}_{i,j}$ represents the percentile-ranking of a music item j in the recommend list for a user i, $h_{i,j}$ corresponds to $r(e_i)$ in Equation (3.1): $h_{i,j} = 1$ if a user i actually listened a music item j, otherwise 0. This thesis compares MPR among each method firstly: smaller MPR indicates better performance of Top-N recommendation. MRR emphasizes importance of music items that are on top of a recommendation list. This thesis supposes that larger MRR indicates better performance when two methods do not have significant difference on MPR.

For rating prediction task, this thesis computes RMSE (Root Mean Squared Error) and Accuracy between predicted rating $\hat{r}_i$ and actual rating r_i for a i-th LE in the test data (TestSet). Testset only includes positive samples as noted in Section 4.2.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i \in \mathrm{TestSet}} (r_i - \hat{r}_i)^2}{|\mathrm{TestSet}|}}, \tag{4.3}$$

$$\mathrm{Accuracy} = \frac{\sum_{i \in \mathrm{TestSet}} \mathrm{Sign}(\hat{r}_i > 0.5)}{|\mathrm{TestSet}|}, \tag{4.4}$$

where $\mathrm{Sign}(x)$ is a sign function to count the number of correct answers: $\mathrm{Sign}(x) = 1$ if $x > 0$, otherwise 0. This thesis supposes when predicted rating of a music item is larger than 0.5, this music item would be recommended. That is, Equation (4.4) means the percentage of items that are recommended correctly. Smaller RMSE or larger Accuracy indicate more accurate rating prediction.

4.4 Experimental result

4.4.1 Comparison with wALS

Table 4.3 shows MPR with the p-value among wALS, proposed method (Random Sampling) and proposed method with contexts (UCP cluster, Language, DoW) in dataset #nowplaying-rs. The last row of this table shows MPR of each method. The upper part represents the p-value on MPR between different methods. The result shows that the proposed methods are more accurate than wALS, and its performance is improved by introducing context information. Table 4.4 shows MPR with the p-value among wALS, proposed method (Random Sampling) and proposed with contexts (UGP cluster, Country) in LFM-1b. It observed that the result is the same as dataset #nowplaying-rs.

Table 4.3: MPR and p-values for comparison of proposed and baseline methods on #nowplaying-rs

	wALS	Proposed method	Proposed method with Contexts
wALS	-	0.000251	1.61505e-12
Proposed method	0.000251	-	0.001160
Proposed method with Contexts	1.61505e-12	0.001160	-
MPR	0.109136	**0.104574**	**0.100718**

Table 4.4: MPR and p-values for comparison of proposed and baseline methods on LFM-1b

	wALS	Proposed method	Proposed method with Contexts
wALS	-	0.002427	2.38048e-07
Proposed method	0.002427	-	0.016985
Proposed method with Contexts	2.38048e-07	0.016985	-
MPR	0.093303	**0.091478**	**0.090412**

4.4.2 Influence of ϵ

Figure 4.19 and Figure 4.20 show MPR against the ratio ϵ of negative samples to positive samples in dataset #nowplaying-rs and LFM-1b, respectively. In #nowplaying, it is observed that setting ϵ larger than 1 can obtain more accurate results than wALS. Although MPR decreases as ϵ increases, the effect is gradual. The computation time when $\epsilon = 2$ and 6 are 688(s) and 2142(s) per fold in cross-validation, respectively. A similar trend was observed in LFM-1b except $\epsilon = 1$. This means that ϵ should be set by considering the balance between time complexity and accuracy of recommendation.

4.4.3 Sampling method

Table 4.5 shows Top Discount Popularity Sampling by setting different hyper-parameter γ from 0.1 to 0.5 in #nowplaying-rs (context information is not used here). Results show that Top Discount Popularity Sampling could not improve the performance on RMS. However, MPR (p-value=0.006318) and Accuracy (p-value=0.046119) could be reduced in the case of $\gamma = 0.1$. This means that considering the popularity of music items when selecting negative samples could improve the performance of both top-N recommendation and rating prediction in #nowplaying-rs. Contrary to this, Table 4.6 shows the result in LFM-1b. It is observed that changing value of γ can not effect any metric. Experiments

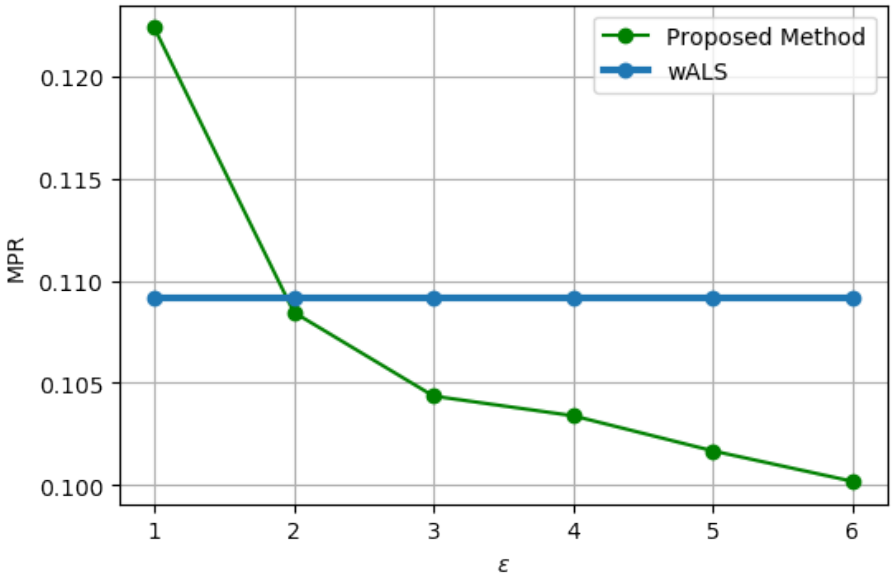

Figure 4.19: Effect ϵ on MPR in #nowplaying-rs

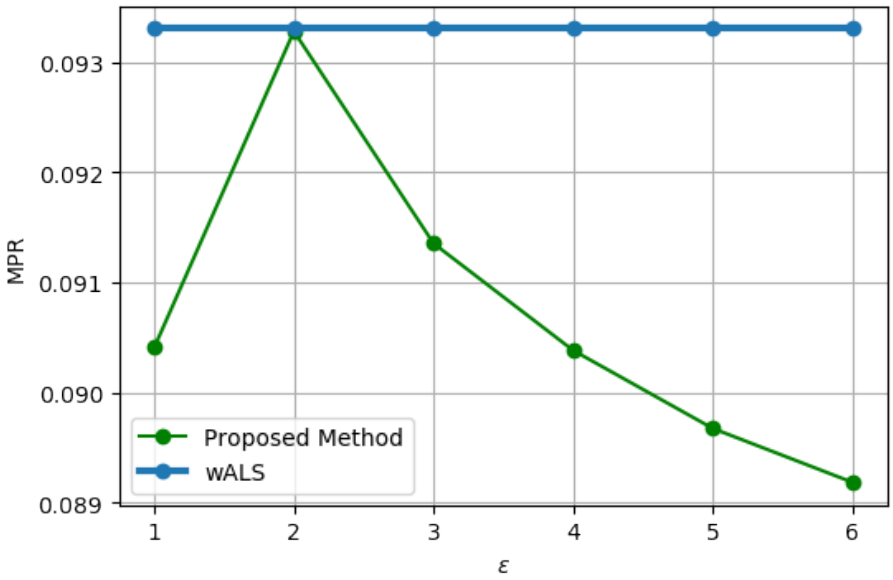

Figure 4.20: Effect ϵ on MPR in LFM-1b

are not done with γ larger than 0.2 because less difference was observed with $\gamma = 0, 0.1, 0.2$. This means that Top Discount Popularity Sampling has no effect on a relatively dense dataset such as LFM-1b.

Table 4.5: Comparison of Random Sampling and Top Discount Popularity Sampling in #nowplaying

	MPR	MRR	RMSE	Accuracy
Random(γ=0)	0.104362	0.018871	0.148141	0.803160
γ=0.1	**0.102920**	0.019811	0.148332	**0.803342**
γ=0.2	0.103830	0.021694	0.149531	0.804147
γ=0.3	0.105374	0.020511	0.149552	0.803634
γ=0.4	0.104366	0.022012	0.150619	0.800669
γ=0.5	0.104818	0.021604	0.152607	0.799405

Table 4.6: Comparison of Random Sampling and Top Discount Popularity Sampling in LFM-1b

	MPR	MRR	RMSE	Accuracy
Random(γ=0)	0.091355	0.004466	0.159674	0.805466
γ=0.1	0.090973	0.004471	0.159738	0.806025
γ=0.2	0.091372	0.004504	0.160677	0.804989

For investigating performance of Priority Popularity Sampling, Hyper-parameter δ is changed from 0.1 to 0.5 (context information is not used here). Results of #nowplaying-rs and LFM-1b are respectively shown in Table 4.7 and Table 4.8. It is observed that MPR, RMSE, and Accuracy could not be improved by tuning hyper-parameter δ in #nowplaying-rs. However, MRR (p-value=0.019364) could be increased when δ=0.1. This means that using Priority Popularity Sampling to select negative samples could improve the performance of top-N recommendation partially in #nowplaying-rs. On the contrary, similar to results on Top Discount Popularity Sampling, all metrics can not be improved in LFM-1b.

Summarizing the above, when a dataset has lower density, proposed popularity-based sampling methods (Top Discount Popularity Sampling and Priority Popularity Sampling) have positive effect on top-N recommendation task. Top Discount Popularity Sampling can also outperform Random Sampling on rating prediction task.

4.4.4 Context information

This subsection investigates the performance of proposed method when adding different context information. Firstly, MPR when using context information is compared with the method that does not use any context information (None). For each context information that can outperform None on MPR, t-test is applied to

Table 4.7: Comparison of Random Sampling and Priority Popularity Sampling in #nowplaying

	MPR	MRR	RMSE	Accuracy
random(δ=0)	0.104362	0.018871	0.148141	0.80316
δ=0.1	0.103891	**0.020854**	0.148726	0.803371
δ=0.2	0.104105	0.022304	0.150208	0.803075
δ=0.3	0.104891	0.022180	0.151292	0.800238
δ=0.4	0.104534	0.023440	0.152382	0.798227
δ=0.5	0.105671	0.023558	0.152510	0.799101

Table 4.8: Comparison of Random Sampling and Priority Popularity Sampling in LFM-1b

	MPR	MRR	RMSE	Accuracy
random(δ=0)	0.091355	0.004466	0.159674	0.805466
δ=0.1	0.091398	0.004350	0.160110	0.804711
δ=0.2	0.091319	0.004503	0.160501	0.804410

confirm whether or not significant differences of evaluation metrics are observed. Furthermore, combinations of context information are also evaluated.

Table 4.9 shows the result of recommendations after adding a single context in #nowplaying-rs. It is observed that MPR of DoW, PoD, Language, and UCP cluster outperform None. The significant difference in RMSE and Accuracy are observed in these four contexts. The significant difference in MPR is only observed when using Language and UCP cluster. Form this result, the performance of each context can be ranked as: UCP > Lang > PoD > DoW. The same tendency is observed in the distance table described in Sec. 4.1.3. The hypothesis noted in Sec. 3.1: if the difference of users' preference among each possible value in a specific context were significant, this context would have a positive effect on recommendations, is verified in dataset #nowplaying-rs. This means that context information that has larger differences among possible values tends to have more positive effect on both top-N recommendation and rating prediction in #nowplaying-rs.

To improve the effect of UCP clusters further, Lang, PoD, and DoW are combined with UCP. Table 4.10 shows that MPR can be reduced when combining UCP with other context. Significant difference of MPR is observed in UCP + Lang + PoD and UCP + Lang + PoD + DoW. However, difference between of UCP + Lang + PoD and UCP + Lang + PoD + DoW is not significant on all metrics. It means that UCP + Lang + PoD is a more efficient combination.

In the case of LFM-1b, Table 4.11 shows that using UGP clusters can improve performance on MPR. However, significant difference in MPR are not observed when using any context information. The performance of each context can be

Table 4.9: Performance of a single context in #nowplaying

	MPR	MRR	RMSE	Accuracy
None	0.104362	0.018871	0.148141	0.803160
DoW	0.103748	0.019976	**0.136077**	**0.824848**
Is weekday	0.105202	0.018840	0.137105	0.824624
Hour	0.106610	0.018068	0.141158	0.815884
PoD	0.104370	0.020581	**0.135286**	**0.825979**
Language	**0.104246**	0.017990	**0.137477**	**0.821102**
UCP	**0.102236**	0.018978	**0.133388**	**0.828536**

Table 4.10: Performance of multiple context in #nowplaying

	MPR	MRR	RMSE	Accuracy
UCP	0.102236	0.018978	0.133388	0.828536
UCP + Lang	0.101648	0.018852	0.135147	0.823150
UCP + Lang + PoD	**0.100212**	0.019355	0.136219	0.822895
UCP + Lang + DoW	0.100984	0.018925	0.133922	0.826638
UCP + Lang + PoD + DoW	**0.099829**	0.019011	0.135616	0.822533

ranked as: UGP > Country > PoD > DoW > Age. The same tendency is observed in the distance table described in Sec. 4.1.3 except for Age.

Table 4.11: Performance of single context in LFM-1b

	MPR	MRR	RMSE	Accuracy
None	0.091355	0.004466	0.159674	0.805466
Age	0.092033	0.004317	0.158702	0.806177
Country	0.091560	0.004421	0.157233	0.808695
PoD	0.091977	0.004391	0.158082	0.807534
DoW	0.092011	0.004452	0.158475	0.807032
UGP	0.091140	0.004353	0.157800	0.808116

Table 4.12 shows the result when using multiple contexts. It is observed that combining Country with UGP can improve performance further on MPR. Significant difference on MPR is observed (p-value=0.016985) between None and UGP + Country.

These results mean that (1) performance of a single context has the same tendency as the distance table, (2) combining multiple contexts can improve performance of recommendation, and (3) using context can only bring limited improvements in the case of a dense dataset.

Table 4.12: Performance of multiple context in LFM-1b

	MPR	MRR	RMSE	Accuracy
None	0.091355	0.004466	0.159674	0.805466
UGP + Country	**0.090364**	0.004370	0.156623	0.809781
Country + Age	0.091402	0.004359	0.157609	0.807607
UGP + Country + Age	0.090574	0.004372	0.157034	0.809520

4.4.5 Content features of music items

Table 4.13 shows the comparison between UCP and signal features of music directly. "overall" represents that combining all signal features that are used in generating UCP. It is observed that using some signal features of music items (instrumentalness, liveness, danceability, valence, acousticness, mode, and energy) could improve RMSE and Accuracy. However, none of them could improve MPR. Combining all signal features is not effective as well: it can not outperform None on any metric. On the other hand, UCP could improve MPR, RMSE and Accuracy. This result indicates that considering UCP of a user as a context could improve the performance of both top-N recommendation and rating prediction.

Table 4.13: Performance of UCP

	MPR	MRR	RMSE	Accuracy
None	0.104362	0.018871	0.148141	0.803160
instrumentalness	0.109320	0.018943	0.142889	0.810220
liveness	0.104577	0.019000	0.138641	0.819866
speechiness	0.104671	0.019884	0.144179	0.808337
danceability	0.104026	0.020009	0.135582	0.825495
valence	0.104342	0.018399	0.135105	0.825640
tempo	0.193850	0.005062	0.209721	0.694052
acousticness	0.104289	0.019034	0.137100	0.819904
energy	0.106400	0.018468	0.139753	0.821302
overall	0.223682	0.002967	0.212018	0.669973
UCP	**0.102236**	0.018978	**0.131457**	**0.828536**

To find an appropriate number of UCP clusters, I changed the number of clusters to 3, 5, 8, 10, 15, and 20. The result is shown in Figure 4.21, in which the shadow indicates standard deviation. Although the standard deviation is not small, this result indicates that the number of UCP clusters should be set to 5 for dataset #nowplaying.

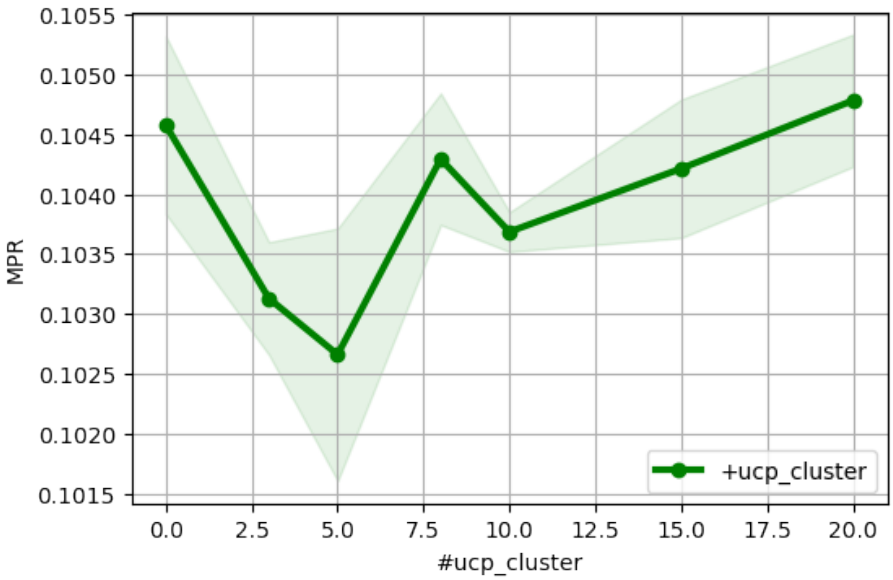

Figure 4.21: Effect number of UGP cluster on MPR

Chapter 5

Conclusion

This thesis proposed a method for recommending music items considering listeners' context information without explicit feedback.

This thesis employs FMs, in which the context information is treated as factors. Kullback-Leibler divergence of playing count distribution under each context was utilized to select relatively effective and appropriate context before a music item is recommended. UCP/UGP clusters were proposed to integrate signal features of music items.

FMs needs both positive and negative samples to learn their parameters. Using combination of context and music items a user has yet to be experienced as negative samples is a straightforward approach. However, large negative samples may cause a problem of time complexity. To reduce the number of negative samples, this thesis employs three types of negative sampling methods: (1) Random Sampling, (2) Top Discount Popularity Sampling, and (3) Priority Popularity Sampling. Random sampling randomly selects music items that are not played by a user under the same context as actual LEs as negative samples. Top Discount Popularity Sampling reduces the probability of selecting a part of unpopular items as negative samples. Priority Popularity Sampling can tune the probability of a music item being selected as a negative sample more flexibly depending on popularity.

The effectiveness of the proposed method and the effect of negative sampling methods were evaluated with dataset #nowplaying-rs and LFM-1b dataset. Evaluation metrics MPR and MRR were used to evaluated performance of proposed method on top-N recommendation. For rating prediction task, RMSE and Accuracy were used. Result of experiment showed that the proposed method outperformed wALS, which is one of popular music recommendation algorithms based on matrix factorization. When dataset has lower density, proposed popularity-based sampling methods have positive effect on prediction accuracy. It is found that negative sample ratio of proposed method should be set by considering the balance between time complexity and accuracy of recommendation. It was also confirmed that performance of recommendation could be improved further by using appropriate context information. Using UCP clusters was effective to uti-

lize signal features of music items. Moreover, the distance table was proposed to select appropriate context in advance.

In future work, an online experiment could be considered to investigate performance of the proposed method further. A negative sampling method could also be improved when a dataset has higher density.

Proposed methods can improve the accuracy of previous research wALS by considering context information, using negative sampling, and integrating signal features of music items. In the future, the proposed method as well as the finding about useful context information and negative sampling methods are expected to contribute to the improvement of existing online music streaming services by recommending more appropriate music items to users to reduce their information retrieval costs.

Acknowledgment

It would not have been possible to write this master thesis and finish master's course without the help and support of the kind people around me.

I would like to thank my supervisor, Professor Yasufumi Takama, for the kind guidance, comments, and engagement through the learning process of this master thesis. Furthermore, I would like to thank Professor Toru Yamaguchi and Associate professor Kaoru Katayama for useful advice. In addition, I would like to thank Assistant Professor Hiroki Shibata, who has supported me throughout the entire process and helped solve many problems. I would like to thank members of Takama Laboratory for sharing knowledge and information selflessly. Finally, I must express my gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of study in Japan.

List of Publications

1. Jin-cheng Zhang, Yasufumi Takama,“Proposal of Context-aware Music Recommender System Using Negative Sampling” 人工知能学会全国大会論文集 一般社団法人 人工知能学会, No. 2A3E502-2A3E502, 一般社団法人 人工知能学会, 2019

2. 張 錦程，柴田祐樹，高間康史，コンテキスト情報に基づく楽曲推薦システムにおけるネガティブサンプリングの効果検証 ARG 第15回Webインテリジェンスとインタラクション研究会 P2, 2019

3. Jin-cheng Zhang, Yasufumi Takama,“Proposal of Context-aware Music Recommender System Using Negative Sampling,” Advances in Intelligent Systems and Computing series, Vol.1128, pp.1-12, 2020.

Reference

[1] Bobadilla Jesús, Ortega Fernando, Hernando Antonio and Gutiérrez Abraham, "Recommender systems survey," Knowledge-based Systems, Vol.46, pp.109-132, 2013

[2] Ricci Francesco, Rokach Lior and Shapira Bracha, "Recommender Systems: Introduction and Challenges," Recommender systems handbook, pp.1-35, 2011.

[3] Lu Jie, Wu Dianshuang, Mao Mingsong, Wang Wei and Zhang Guangquan, "Recommender system application developments: a survey," Decision Support Systems, Vol.74, pp.12-32, 2015

[4] Resnick Paul and Varian Hal R, "Recommender systems", Communications of the ACM, Vol.40, No.3, pp. 56-59, 1997

[5] Gross Bertram M, "The Managing Organizations: The Administrative Struggle," Vol. 2, pp.856, 1964.

[6] Maurizio Ferrari Dacrema, Paolo Cremonesi, Dietmar Jannach, "Are we really making much progress? A worrying analysis of recent neural recommendation approaches," RecSys '19 Proceedings of the 13th ACM Conference on Recommender Systems, pp.101-109, 2019.

[7] Beel Joeran, Langer Stefan, Genzmehr Marcel, Gipp Bela, Breitinger Corinna and Nürnberger Andreas, "Research paper recommender system evaluation: a quantitative literature survey," Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation, pp.15-22, 2013.

[8] Joeran Beel, Bela Gipp, Stefan Langer and Corinna Breitinger, "News recommender systems – Survey and roads ahead," Information Processing and Management, Vol.54, pp.1203–1227, 2018.

[9] Borràs Joan, Moreno Antonio and Valls Aida "Intelligent tourism recommender systems: A survey," Expert Systems with Applications, Vol.41, No.16, pp.7370-7389, 2014.

[10] Chen Jilin, Geyer Werner, Dugan Casey, Muller Michael and Guy Ido, "Make new friends, but keep the old: recommending people on social networking

sites," Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.201-210, 2009.

[11] Castro-Schez Jose Jesus, Miguel Raul, Vallejo David and López-López Lorenzo Manuel, "A highly adaptive recommender system based on fuzzy logic for B2C e-commerce portals," Expert Systems with Applications, Vol.38, No.3, pp.2441-2454, 2011.

[12] Song Meina, Zhou Xue, Haihong E and Ou Zhonghong, "A Recommender System Model based on Commodity-Purchase-Cycle Classification," Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications, pp.48-53, 2016.

[13] Resnick Paul, Iacovou Neophytos, Suchak Mitesh, Bergstrom Peter and Riedl John, "an open architecture for collaborative filtering of netnews," ACM conference on Computer supported cooperative work, pp.175-186, 1994.

[14] Koren Yehuda, Robert Bell and Chris Volinsky, "Matrix factorization techniques for recommender system." Computer, Vol. 42, No. 8, pp. 30-37, 2009.

[15] Van Meteren Robin and Van Someren Maarten, "Using content-based filtering for recommendation," Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop, pp.47-56, 2000.

[16] Krulwich Bruce, "Lifestyle finder: Intelligent user profiling using large-scale demographic data," AI magazine, Vol.18, No.2, 1997.

[17] Burke Robin, "Knowledge-based recommender systems," Encyclopedia of library and information systems, Vol.69, No.32, pp.175-186, 2000.

[18] Burke Robin, "Hybrid recommender systems: Survey and experiments," User modeling and user-adapted interaction, Vol.12, No.4, pp.311-370, 2002.

[19] Barragáns-Martínez Ana Belén, Costa-Montenegro Enrique, Burguillo Juan C, Rey-López Marta, Mikic-Fonte Fernando A and Peleteiro Ana, "A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition," Information Science, Vol.880, No.22, pp.4290-4311, 2010.

[20] Vozalis Manolis G and Margaritis Konstantinos G, "Using SVD and demographic data for the enhancement of generalized collaborative filtering," Information Sciences, Vol.177, No.14, pp.3017-3037, 2007.

[21] Arieez Dutta, "Market Segmentation Definition, Bases, Types & Examples," Feedough, retrieved 2019/12/31, from https://wwwfeedoughcom/market-segmentation-definition-basis-types-examples/

[22] Lee Sanghack, Yang, Jihoon and Park Sung-Yong, "Discovery of hidden similarity on collaborative filtering to overcome sparsity problem," Proceedings of the 7th International Conference on Discovery Science, pp. 396-402, 2004.

[23] Kaminskas Marius and Bridge Derek, “Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems,” ACM Transactions on Interactive Intelligent Systems (TiiS), No.1, Vol.7, pp.2, Article 2, 2017.

[24] Adomavicius Gediminas and Kwon YoungOk, “Improving aggregate recommendation diversity using ranking-based techniques,” IEEE Transactions on Knowledge and Data Engineering, No. 5 Vol. 24, pp.896-911, 2011.

[25] Douglas W. Oard and Kim Jinmook, “Implicit feedback for recommender systems,” AAAI workshop on recommender systems, Vol.83, pp.81-83, 1998.

[26] Hu Yifan, Yehuda Koren, and Chris Volinsky, “Collaborative filtering for implicit feedback datasets,” 8th IEEE International Conference on Data Mining, pp.263-272, 2008.

[27] Pan Rong, Zhou Yunhong, Cao Bin, Nathan N. Liu, Lukose Rajan, Scholz Martin and Yang Qiang, “One-class collaborative filtering,” 8th IEEE International Conference on Data Mining, pp.502-511, 2008.

[28] Zhang Weinan, Chen Tianqi, Wang Jun and Yu Yong, “Optimizing top-n collaborative filtering via dynamic negative item sampling,” 36th international ACM SIGIR conference on Research and development in information retrieval, pp785-788, 2013.

[29] Rendle Steffen, Freudenthaler Christoph, Gantner Zeno and Schmidt-Thieme Lars, “BPR: Bayesian personalized ranking from implicit feedback,” 25th conference on uncertainty in artificial intelligence, pp.452-461, 2009.

[30] Salton Gerard, “Developments in automatic text retrieval,” Science, Vol. 253, No. 5023, pp. 974-980

[31] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu and Mike Gatford. “Okapi at TREC-3,” 3rd Text REtrival Conference, 1994.

[32] Johnson Christopher C, “Logistic matrix factorization for implicit feedback data,” Advances in Neural Information Processing Systems, Vol. 27, 2014.

[33] Adomavicius Gediminas and Alexander Tuzhilin, “Context-aware recommender systems,” Recommender systems handbook, Springer, pp.217-253, 2011.

[34] Kaminskas Marius and Ricci Francesco, “Contextual music information retrieval and recommendation: State of the art and challenges,” Computer Science Review, Vol.6, pp.89-119, 2012.

[35] Deng Shuiguang, Wang Dongjing, Li Xitong and Xu Guandong, “Exploring user emotion in microblogs for music recommendation,” Expert Systems with Applications, Vol.42, No.23, pp.9284-9293, 2015.

[36] Wang Dongjing, Deng Shuiguang and Xu Guandong, "Gemrec: A graph-based emotion-aware music recommendation approach," International Conference on Web Information Systems Engineering, pp.92-106, 2016.

[37] Kaminskas Marius, Ricci Francesco and Schedl Markus, "Location-aware music recommendation using auto-tagging and hybrid matching," 7th ACM conference on Recommender systems, pp.17-24, 2013.

[38] Pichl Martin, Zangerle Eva and Specht Günther, "Towards a context-aware music recommendation approach: What is hidden in the playlist name," IEEE International Conference on Data Mining Workshop, pp.1360-1365, 2015.

[39] Pichl Martin, Zangerle Eva and Specht Günther, "Improving context-aware music recommender systems: Beyond the pre-filtering approach," 2017 ACM on International Conference on Multimedia Retrieval, pp.201-208, 2017.

[40] Markus Schedl and Dominik Schnitzer, "Hybrid retrieval approaches to geospatial music recommendation," 36th international ACM SIGIR conference on Research and development in information retrieval, pp.793-796, 2013.

[41] Symeonidis Panagiotis, Ruxanda Maria, Nanopoulos Alexandros and Manolopoulos Yannis, "Ternary Semantic Analysis of Social Tags for Personalized Music Recommendation," International Society for Music Information Retrieval, Vol.8, pp.219-224, 2008.

[42] Nanopoulos Alexandros, Rafailidis Dimitrios, Symeonidis Panagiotis and Manolopoulos Yannis, "Musicbox: Personalized music recommendation based on cubic analysis of social tags," IEEE Transactions on Audio, Speech, and Language Processing, Vol.18, No.2, pp.407-412, 2009.

[43] Van den Oord Aaron, Dieleman Sander and Schrauwen Benjamin, "Deep content-based music recommendation," Advances in neural information processing systems, pp.2643-2651, 2013.

[44] Celma Oscar, "Music recommendation," Music recommendation and discovery, Springer, pp.43-85, 2010.

[45] Schedl Markus and Bauer Christine, "Distance-and Rank-based Music Mainstreaminess Measurement," 25th Conference on User Modeling, Adaptation and Personalization, pp.364-367, 2017.

[46] Rendle Steffen, "Factorization machines with libfm," ACM Transactions on Intelligent Systems and Technology, Vol.3, No.3, Article No.57, pp.1-22, 2012.

[47] Rendle Steffen, Gantner Zeno, Freudenthaler Christoph and Schmidt-Thieme Lars, "Fast context-aware recommendations with factorization machines," Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp.635-644, 2011.

[48] Asmita Poddar, Eva Zangerle and Yi-hsuan Yang, "#nowplaying-RS: A New Benchmark Dataset for Building Context-Aware Music Recommender Systems," 15th Sound and Music Computing Conference, 2018.

[49] Schedl Markus, "The lfm-1b dataset for music retrieval and recommendation," Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, pp.103-110, 2016.

[50] Schaul Tom, Quan John, Antonoglou Ioannis and Silver David, "Prioritized experience replay," arXiv preprint arXiv:1511.05952, 2015

[51] Schedl Markus and Ferwerda Bruce, "Large-scale analysis of group-specific music genre taste from collaborative tags", 2017 IEEE International Symposium on Multimedia (ISM), pp.479-482, 2017

[52] Chen-Shu Wang, Shiang-Lin Lin and Heng-Li Yang, "Evaluating music recommendation in a real-world setting: On data splitting and evaluation metrics," IEEE International Conference on Multimedia and Expo, pp.1-6, 2015.

[53] Kullback, Solomon and Leibler, Richard A, "On information and sufficiency," The annals of mathematical statistics, Vol.22, No.1, pp.79-86, 1951.

[54] Rushi Longadge, Snehlata S. Dongre and Latesh Malik, "Class Imbalance Problem in Data Mining: Review," International Journal of Computer Science and Network (IJCSN), Vol.2, Issue 1, 2013.