Computing and Informatics, Vol. 41, 2022, 834–860, doi: 10.31577/cai_2022_3_834

CONTEXT-AWARE MUSIC RECOMMENDATION WITH METADATA AWARENESS AND RECURRENT NEURAL NETWORKS

Igor André Pegoraro Santana, Marcos Aurélio Domingues

Departament of Informatics
State University of Maringá
Avenida Colombo, 5790
87020-900, Maringá, PR, Brazil
e-mail: {pg400816, madomingues}@uem.br

Abstract. Day by day, music streaming services grow the volume of data on the internet. To help the users to find songs that fit their interests, music recommender systems can be used to filter a large number of songs according to the preference of the user. However, the context in which the users listen to songs must be taken into account, which justifies the usage of context-aware recommender systems. Although there are some works about context-aware music recommender systems, there is a lack of automatic techniques for extracting contextual information for these systems. Thus, the goal of this work is to propose two methods to acquire contextual information (represented by embeddings) for each song, given the sequence of songs that each user has listened to. The first method, called Metadata-Aware, uses tags and genres to enrich the embeddings with additional information. The second method, called Dual Recurrent Neural Network, uses such a network to improve the embeddings generated from long sequences of songs. The embeddings generated by both methods were evaluated with four context-aware music recommender systems in two datasets. The results showed that the embeddings, obtained by our proposals, present better results than the state-of-the-art method proposed in the literature (in some cases with gains of more than 100 %). Finally, the experiments also showed that our second method provides better results than the first one.

Keywords: Metadata, recurrent neural networks, context-aware recommender systems, music recommendation, embeddings, context acquisition

Mathematics Subject Classification 2010: 68T99

1 INTRODUCTION

With the growing of music streaming services nowadays, the abundance of available songs for the users grows as well. Spotify, as an example, has 50 million songs available in its directory. Users cannot handle so much data, making it necessary for the system to implement a tool that assists users in finding songs that are fit for their preferences.

The usage of smartphones with music streaming services changed how people listen to music. A user can be texting a friend, browsing through its social networks or answering e-mails, while listening to music in the background. Thus, a user is inserted in a broader context while listening to a song. Also, as seen in [1], people look for songs based on occasions, events and emotions, which suggest that listening to songs is not an isolated event.

A useful tool to deal with this information overload is a recommender system, which can recommend songs to users based on their preferences. Several works propose and review music recommender systems. For example, the work of [2] uses pitch, duration and loudness of a music as descriptors for a content-based recommender. In [3], the authors propose a hybrid music recommendation algorithm that uses different kinds of social media information (e.g. tags, albums, artists, tracks, groups, etc.) and music acoustic-based content (i.e. Mel-Frequency Cepstral Coefficients (MFCCs)). In [4], the author describes how music recommenders work, explores some of the limitations seen in current recommenders, offers techniques for evaluating the effectiveness of music recommendations and demonstrates how to build effective recommenders by offering two real-world recommender examples. However, knowing that users usually listen to songs given a context, the traditional recommender systems can be replaced by context-aware recommender systems, which can include this kind of information in their model. The work of [5] reviews some context-aware music recommender systems.

As can be seen, although there are some works about context-aware music recommender systems, there is a lack of automatic techniques for extracting contextual information. To obtain such information, [6] proposed a method to obtain embeddings from songs with different goals: one goal was to obtain the general preference from the user (Music2Vec) and another one was to obtain the contextual preference from the user (Session-Music2Vec). The user's general preference can be inferred by its complete listening history and refers to the user's specific preferences for music. The contextual preference for songs indicates the recent preferences of the user in the current session/context. The method was based on the Skip-Gram architecture [7], a state-of-the-art embedding model.

Although the results obtained by [6] are promising, they can be improved. This work proposes two methods to obtain general and contextual preferences (i.e. embeddings) for context-aware music recommender systems. The first method combines the embeddings provided by [6] with Metadata-Aware embeddings. The second method consists of a Dual Recurrent Neural Network to provide general and contextual embeddings.

The methods were evaluated by using four context-aware recommender systems, and two music datasets that include the listening history of thousands of users. The own method proposed by [6] was used as baseline to compare with our proposals. The results showed that, in both datasets, our proposed methods outperformed the baseline, indicating that it can capture better general and contextual information through the embeddings.

Thus, the main contributions of this paper can be summarized as follows:

- Proposal of a Metadata-Aware method that uses tags and genre as metadata to improve the embeddings provided by [6];
- Proposal of a Dual Recurrent Neural Network method that uses Long Short-Term Memory networks to analyze the sequence of songs that the users listened to, and to produce better general and contextual embeddings;
- Extensive experiments to evaluate the two methods on two real world datasets, and the results show that our proposals outperform the method proposed by [6].

The remaining of this paper is organized as follows: The related work is described in Section 2. The motivation for this work is presented in Section 3. In Section 4, the two proposed methods are described in details. Section 5 describes the empirical evaluation, i.e. the datasets, the recommender systems and the results. Finally, conclusions and future directions are presented in Section 6.

2 RELATED WORK

This section describes some related work in the context-aware music recommendation, as well as embedding and recurrent neural network models that also motivated our work.

2.1 Context-Aware Music Recommendation

As the number of people using mobile devices to listen to songs grows due to the number of applications and the quality of connection to stream those songs, the amount of research conducted to study how to recommend songs to people in mobile devices grows as well.

There are lots of works in the context-aware music recommendation area that focus on mobile devices, since there is a lot of contextual information that can be used by the recommenders. For example, the work of [8] proposes a probabilistic model to integrate contextual information with music content analysis to offer music recommendation for daily activities. In [9], the authors propose a novel approach for recommending music pieces by reflecting the user's current context captured from the mobile device.

Different types of information can be used as contextual information for context-aware music recommender systems. Location, for example, can be used as contextual information, as shown by [10]. The emotion of the user when they are listening to

the song is also a valuable information for a context-aware recommender system and can be used to improve recommendations, as seen in [11].

2.2 Embedding

Some types of data, such as images and songs, can be modeled through analogic signals to be used by neural networks. However, there are types of data, i.e. texts, that cannot be processed in its original shape, requiring an adequate representation to be processed.

Representing data as a vector of real numbers has its origins in the area of distributed representations [12]. In distributed representations, an item is represented by a pattern of activities in a set of computational elements, e.g. neurons in neural networks, and each element can represent multiple items.

However, there are many methods to obtain those vectors, known as embedding vectors. One of the most prominent methods to obtain those vectors was proposed by [7], and is called Word2Vec. It is composed of two models that are shallow neural networks that are trained on a corpus of words and sentences.

There are two works proposed in the literature that are related to our work and that used an adaptation of the Word2Vec method, as it can be extended to a lot of domains. In the work of [13], and in its extended version [6], the authors used Skip-Gram, one of the models in Word2Vec, for the song domain, intending to obtain embeddings considering the songs that are around to a target song.

Another work proposed by [14] adapted an AutoEncoder model to the next song recommendation task. Instead of using the AutoEncoder to transform the songs in embeddings, the playlists, in which are the songs, were transformed in embeddings. The next songs to be recommended are then computed based on the embeddings of the playlists.

2.3 Recurrent Neural Network

To the best of our knowledge, the first work that used Recurrent Neural Networks to obtain embeddings for songs in order to be used in context-aware recommender systems was [15], which used a Gated Recurrent Unit. Similar to the [6], they proposed a model to obtain the embeddings based on the sequence of songs that a user has listened to. However, [15] not only used Gated Recurrent Unit to obtain the embeddings but also to recommend the songs for the next song recommendation task. A more recent work proposed a Recurrent Neural Network embedding model that learns the sequential listening behaviour of users, and adapts it to the current context [16].

3 MOTIVATION

The main task of a music recommender system is to propose interesting music to the users based on their musical preference, as defined by [4]. However, the preference

of a user for music changes over time, as the user listens to music while is texting a friend, browsing through its social networks, or answering e-mails. Thus, it is possible to say that the user is inserted in a broader context while listening to a song.

There are some techniques to collect the context in which the user is inserted, and to use it in a context-aware recommender system to improve the recommendations. In [17], the authors described that contextual information can be obtained explicitly, i.e., when the user explicitly provides the information about its listening record, or implicitly, i.e., from a sequence of songs that a user has listened to.

This work exploits both ideas described by [17] to collect contextual information, explicitly and implicitly, in the music domain. Thus, this work is based on the following two motivations.

Motivation 1. The explicitly contextual information in the format of metadata attributes is used to improve the embedding vectors.

As described in [4], there are different types of music recommender systems that use contextual information to recommend songs to a user. One of those methods uses the tags of the song to produce better results than simply using traditional recommender systems [18]. One of our proposals aims at using not only tags but also metadata attributes of songs with embeddings to improve context-aware music recommender systems.

Motivation 2. The implicitly contextual information is obtained by using Recurrent Neural Networks to analyze the sequence of songs that a user has listened to.

Recurrent Neural Networks are a family of neural networks specialized in processing sequences of data, and can scale to much longer sequences than other networks without sequence-based specialization, as defined by [19].

Knowing that Recurrent Neural Networks are specialized in processing sequences, our second proposal aims at processing the sequence of songs on each user's listening history to obtain implicit contextual information to be used in context-aware music recommender systems.

4 PROPOSED WORK

This section describes the two methods proposed in this work. The notation used in this work is formalized in Subsection 4.1. Subsection 4.2 presents the model proposed by [6], and shows how it has been extended in this work to obtain Metadata-Aware song embeddings (i.e. our first method). Subsection 4.3 describes our second method, a Dual Recurrent Neural Network proposed to obtain song embeddings.

4.1 Notation

Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$ be the set of users and $M = \{m_1, m_2, \ldots, m_{|M|}\}$ be the set of songs, in which $|U|$ and $|M|$ are the total number of unique users and songs,

respectively. For each user u, its listening history are the songs that were listened to by the user with its respective date and time, defined as $H^u = \left\{m_1^u, m_2^u, \ldots, m_{|H^u|}^u\right\}$. The user's listening history can be divided into sessions $S^u = \left\{S_1^u, S_2^u, \ldots, S_{|S^u|}^u\right\}$ according to how much time has passed between two songs. A session n from user u is defined as $S_n^u = \left\{m_{n,1}^u, m_{n,2}^u, \ldots, m_{n,|S_n^u|}^u\right\}$, in which $m_{n,j}^u \in M$. In this work, sessions are created after 30 minutes passed between two songs. Additionally, let us assume that $A_{H_u} = \left\{a_{m_1^u}, a_{m_2^u}, \ldots, a_{m_{|H^u|}^u}\right\}$, where A_{H_u} consists of metadata attribute for each song in the listening history of the user u. Similar to the songs, the metadata attribute will also be split into sessions. Thus, for the metadata attribute, a session n from user u is defined as $A_{S_n^u} = \left\{a_{m_{n,1}^u}, a_{m_{n,2}^u}, \ldots, a_{m_{n,|S_n^u|}^u}\right\}$.

4.2 Metadata-Aware Method

As seen in [20], embeddings are a kind of distributed representation that can be learned by different feature learning techniques. An embedding of an item is a continuous vector in a low dimensional space that was mapped from a space with one dimension per item by a feature learning technique. As an example, words in a text are not easy to represent given the amount of words that exist in a vocabulary. Word embeddings, that are embeddings that represent a word, were proposed first by [21] and popularized by [7].

Inspired by the Word2Vec model proposed in [7], [6] introduced a method to obtain the embedding vectors from songs. The method is based on the Skip-Gram model and consists of two models: Music2Vec and Session-Music2Vec.

The main idea of the method is that the sequence of songs listened by a user reflects its song preferences during that period, and that co-occurrence of songs in a sequence indicates that those songs are similar. Embeddings of songs that are close in a sequence must appear close in a low dimensional space.

The Music2Vec learns the embedding vectors of a song m_i^u from its neighbor songs $\left\{m_{i-c}^u : m_{i+c}^u\right\} \setminus m_i^u$ on the complete listening history H^u from a user u. In a formal way, as described by [6], the objective function of the Music2Vec model is defined as:

$$L = \sum_{u \in U} \sum_{m_i^u \in H^u} \sum_{-c \leq j \leq c} \log p\left(m_{i+j}^u | m_i^u\right) \tag{1}$$

where c correlates to the size of the contextual window. As in the Skip-Gram model, the contextual window slides through the whole listening history of the user. In Equation (1), $p\left(m_{i+j}^u | m_i^u\right)$ represents the conditional probability of a song m_{i+j}^u being in the contextual window of a song m_i^u in H^u, which formally is defined by using the softmax function:

$$p\left(m_{i+j}^u | m_i^u\right) = \exp\left(\mathbf{v}_{m_i^u}^T \cdot \mathbf{v}'_{m_{i+j}^u}\right) / \sum_{m \in M} \exp\left(\mathbf{v}_{m_i^u}^T \cdot \mathbf{v}'_m\right) \tag{2}$$

where $\mathbf{v}_m$ and $\mathbf{v}'_m$ are input and output vectors of the song m, respectively. As seen in Equations (1) and (2), the contextual window of the model Music2Vec slides through the whole listening history of the user, obtaining the general embedding vector from the user. However, the musical tastes of a user can vary a lot during the listening history, indicating that an approach based on sessions is more viable to capture those variations.

The Session-Music2Vec tries to solve the problem of the variation of the musical taste on the listening history of the user. In the Session-Music2Vec model, instead of learning the embedding vectors from the whole learning history, the listening history is split into smaller sessions to learn the embeddings. Then, we have the contextual window slides through the sessions instead of the whole listening history, which allows us to obtain the contextual embedding vector from the user.

Formally, the Session-Music2Vec tries to learn the embedding vectors of a song $m^u_{n,i}$ from its neighbor songs $\left\{m^u_{n,i-c} : m^u_{n,i+c}\right\} \setminus m^u_{n,i}$, in the session n of the user u. The objective function for the Session-Music2Vec is defined by:

$$L = \sum_{u \in U} \sum_{S^u_n \in S^u} \sum_{m^u_{n,i} \in S^u_n} \sum_{-c \leq j \leq c} \log p\left(m^u_{n,i+j} | m^u_{n,i}\right), \tag{3}$$

which is similar to the objective function defined in Equation (1), with the difference that it iterates through the user sessions. In a similar way, $p\left(m^u_{n,i+j} | m^u_{n,i}\right)$ represents the conditional probability of a neighbor song $m^u_{n,i+j}$ given a song $m^u_{n,i}$ in a session S^u_n, which is defined using the softmax function:

$$p\left(m^u_{n,i+j} | m^u_{n,i}\right) = \exp\left(\mathbf{v}^T_{m^u_{n,i}} \cdot \mathbf{v}'_{m^u_{n,i+j}}\right) / \sum_{m \in M} \exp\left(\mathbf{v}^T_{m^u_{n,i}} \cdot \mathbf{v}'_m\right). \tag{4}$$

Our first proposal consists of using both models defined by [6] not only to obtain embedding vectors for the songs, but also to obtain embedding vectors from another metadata attribute of the sequence. Following the notation from Subsection 4.1, our proposal uses the metadata attribute for each song in the listening history A_{H_u} as input for the Music2Vec, and the metadata attribute for each song in the listening session $A_{S^u_n}$ as input for the Session-Music2vec.

Figure 1 illustrates how our proposed method uses the Music2Vec model with a song and a metadata attribute from the listening history. The embedding vectors for the song and for the metadata attribute related to the song are obtained by using the Music2Vec. Then, those embeddings are used to perform an embedding operation that will result in a song embedding that is Metadata-Aware. The final embedding will be used by the context-aware recommender systems. Our proposed method also carry out a process similar to the one presented in Figure 1 using the Session-Music2Vec model to obtain the contextual information that is also Metadata-Aware.

With respect to the embedding operation in Figure 1, three simple vector operations are used to combine the song and metadata embeddings: vector addition

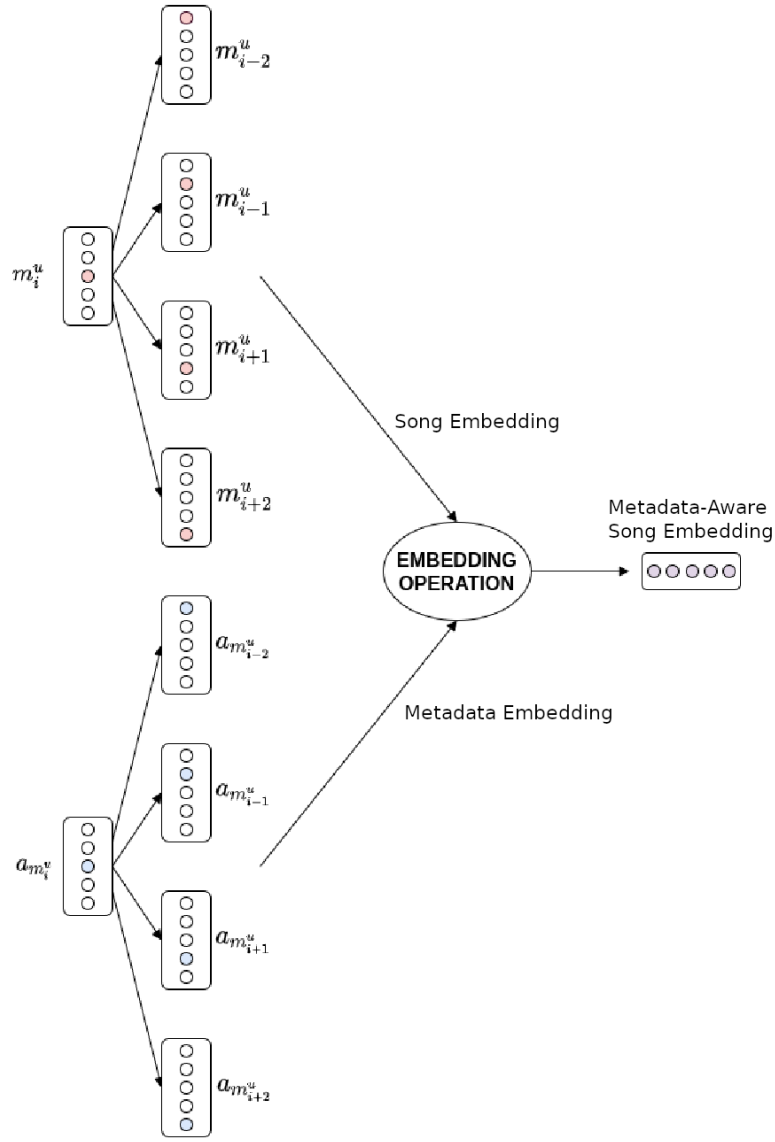

Figure 1. Process for obtaining the metadata-aware song embeddings using the Music2Vec model

(ADD), vector product (MUL) and vector average (AVG). Assuming that $\mathbf{v}'_m$ is the embedding for the song m, and $\mathbf{v}'_{a_m}$ is the embedding for the metadata attribute of the song m, those operations can be defined as:

$$\text{ADD} = \mathbf{v}'_m + \mathbf{v}'_{a_m}, \tag{5}$$

$$\text{MUL} = \mathbf{v}'_m \times \mathbf{v}'_{a_m}, \tag{6}$$

$$\text{AVG} = \frac{\mathbf{v}'_m + \mathbf{v}'_{a_m}}{2}. \tag{7}$$

4.3 Dual Recurrent Neural Network Method

Recurrent Neural Networks are neural networks designed to process a sequence of information, which can scale sequences of various lengths, as seen in [19]. This is possible because they share the network parameters across different parts of a model, which makes it possible to generalize what was learned across the model.

Our second method consists of a Dual Recurrent Neural Network that is able to learn the general and contextual preferences in a same model, generating embeddings for the songs that can be used in context-aware music recommenders. Similar to the Context Bag-of-Words (CBOW) model proposed by [22], the goal of our method is to predict the center song in the contextual window given its neighborhood songs. However, in contrast to the CBOW model, the Recurrent Neural Network is used to analyze the contextual windows, and the order in which the songs are in the window does matter. Figure 2 presents an overview of the proposed method and its most important layers.

Long Short-Term Memory (LSTM) was proposed by [23] and is a kind of Recurrent Neural Network that has the intent to solve problems with long term dependencies. It uses gated cells that are capable to forget information that will no longer be useful, and to keep information that can be used later on the sequence [19]. There are three gates in the LSTM cell: forget gate, input gate, and output gate. Through those gates, the LSTM cell learns which information is useful in a sequence and passes that information to make predictions through the output gate, and the cell state containing the relevant information is passed to the next timestamp.

The learning process of a LSTM is comparable to the learning process of a feedforward neural network, which consists of a forward phase and a backward phase. The forward phase is similar to the one from feedforward neural networks with a single hidden layer. However, there is a difference, the activation values from the hidden layer comes from the input layer and the last time step of the hidden layer, since LSTMs work with a sequence of data.

The BackPropagation Through Time algorithm, responsible for updating the weights of the LSTM, is also similar to the BackPropagation algorithm of the feedforward neural networks. It uses the gradient descent method to update the weights of the neurons, but because it goes back in the time steps, it receives the name

BackPropagation Through Time [19].

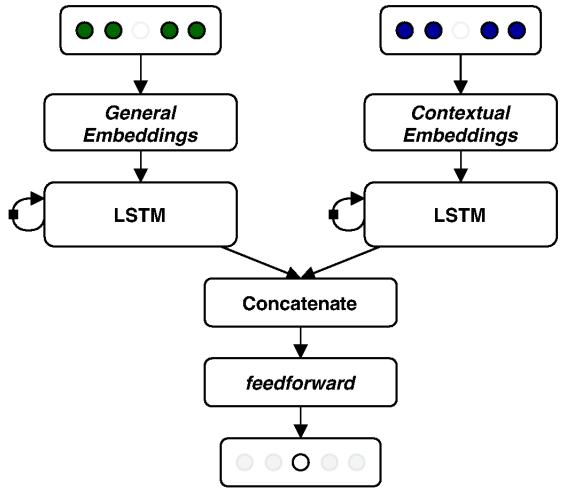

Figure 2. Overview of the Dual Recurrent Neural Network method

As seen in Figure 2, our method receives streams of data: the left one, which has as input the sliding window of a song m_i^u, taking into the complete listening history of a user H^u, for all users. The right stream has as input the sliding window of the same song $m_{n,i}^u$, taking into account the session in which the song is, instead of the whole listening history.

Those sliding windows are passed to the LSTM layers, that are responsible to analyze the hidden relationships between the sequences of songs in the windows of both streams of data. Then, the cell state of the last timestamp of both LSTMs are concatenated side by side into a single vector. Then, this vector is used as input by a fully connected feedforward layer, that tries to predict the song that is in the center of both sliding windows, using as input the concatenated vector.

Each part of the model has its own embedding matrix that is initialized with all the songs in the dataset and, as the model learns through the forward and back-propagation process, this matrix gets updated. Those embeddings will later be used in the context-aware recommender systems. The result of this training process is that each song will have two embedding vectors: the general embedding vector and the contextual embedding vector. Thus, the flexibility of neural networks in a single model can be explored to learn the two embedding vectors.

5 EMPIRICAL EVALUATION

This section describes the empirical evaluation conducted to evaluate our two proposals. Subsection 5.1 presents the datasets and their main statistics. Subsection 5.2

describes the context-aware recommender systems used to evaluate our methods. The evaluation setup and the metrics used in our evaluation are described in Subsection 5.3. Then, the results are discussed in Subsection 5.4.

5.1 Datasets

The empirical evaluation used two different datasets which contain the listening history for each user, as well as the timestamp for each listening event.

The first dataset, called Xiami[1], was proposed by [6]. It was built using a web crawler on the Xiami Music[2] application. This dataset has 361 899 songs and 4 284 users with 1 000 listened to songs in their listening history.

The second dataset, called Music4All[3], was proposed by [24]. It was created using the last.fm[4] official API and has 15 602 users and 109 269 songs, with an average of 361 songs per user. The dataset also has 853 unique genres and 19 541 unique tags.

It is worth to say that the Xiami dataset does not have metadata atributes. So, our metadata proposal was evaluated only in the Music4All dataset.

5.2 Context-Aware Recommender Systems

The evaluation of our embeddings (i.e. contextual information) was carried out by using the four context-aware recommender systems proposed by [6]. The recommenders make use of a general preference and a contextual preference for each user, which are built based on the learned embedding vectors. The general preference for a user u can be learned from its entire listening history $H^u = \left\{m_1^u, m_2^u, \ldots, m_{|H^u|}^u\right\}$ and is defined as:

$$\mathbf{p}_g^u = \frac{1}{|H^u|} \sum_{m_i^u \in H^u} \mathbf{v}_{m_i^u}^{g2v} \tag{8}$$

where $\mathbf{v}_{m_i^u}^{g2v}$ is defined as the general embedding vector. The contextual preference for the user u, given their current session $S_n^u = \left\{m_{n,1}^u, m_{n,2}^u, \ldots, m_{n,|S_n^u|}^u\right\}$ can be defined as:

$$\mathbf{p}_c^u = \frac{1}{|S_n^u|} \sum_{m_{n,i}^u \in S_n^u} \mathbf{v}_{m_{n,i}^u}^{c2v} \tag{9}$$

where $\mathbf{v}_{m_{n,i}^u}^{c2v}$ corresponds to the contextual embedding vector for the song. As can be seen in Equation (8), the general preference is defined as an average of all the general embedding vectors of the songs in the user's listening history. On the other hand, the contextual preference, defined in Equation (9), is the average of all the

[1] `https://1drv.ms/f/s!ApojZBGe9UzXgaI6x8pBf8JgN4PfZg`

[2] `https://www.xiami.com`

[3] `https://sites.google.com/view/contact4music4all`

[4] `https://www.last.fm`

contextual embedding vectors of the songs in the user's current session. Given the general and contextual embeddings as well as the preferences for each user, four context-aware recommender systems were defined in [6]:

- Music2vec-TopN (M-TN);
- Session-Music2vec-TopN (SM-TN);
- Context-Session-Music2vec-TopN (CSM-TN); and
- Context-Session-Music2vec-UserKNN (CSM-UK).

Among all context-aware recommender systems, the M-TN is the only one that uses only the general preference (i.e. the general embedding vector) to recommend songs to the users. Given a user u and their general preference $\mathbf{p}_g^u$ for songs, the recommender system measures the cosine similarity between $\mathbf{p}_g^u$ and the general embedding vector of all the songs in the set of songs M. The top-N songs with the highest value of cosine similarity are recommended to the user. Formally, the predicted preference of the user u to the song m can be defined as:

$$pp_{M-TN}(u,m) = \cos\left(\mathbf{p}_g^u, \mathbf{v}_m^{g2v}\right). \tag{10}$$

The SM-TN recommender system is similar to M-TN, but it uses contextual information instead of the general information. Given a user u and their contextual preference $\mathbf{p}_c^u$, the SM-TN measures the cosine similarity between the contextual embedding vector $\mathbf{v}_{m_{n,i}^u}^{c2v}$ of the songs and the contextual preference of the user. The top-N songs with the highest cosine similarity are then recommended to the user. Formally, the preference can be defined as:

$$pp_{SM-TN}(u,m) = \cos\left(\mathbf{p}_c^u, \mathbf{v}_m^{c2v}\right). \tag{11}$$

The CSM-TN recommender system is a combination of the previous recommender systems: M-TN and SM-TN. After the similarity of each recommender is calculated for each song, they are summed to obtain the most similar songs according to both the general and contextual preferences of the user. Formally, the preference is defined as:

$$PP_{CSM-TN}(u,m) = \cos\left(\mathbf{p}_g^u, \mathbf{v}_m^{g2v}\right) + \cos\left(\mathbf{p}_c^u, \mathbf{v}_m^{c2v}\right). \tag{12}$$

The last recommender system, CSM-UK, proposes a combination of the traditional recommender system, UserKNN [25], with the learned embedding vectors. The UserKNN recommender system needs a similarity function to build a neighborhood of similar users. In [6], the similarity function between two users, u and v is defined as follows:

$$\mathrm{sim}(u,v) = \sum_{m \in M^u \cap M^v} \frac{1}{\sqrt{|M^u| \times |M^v|} + \cos\left(\mathbf{p}_g^u, \mathbf{p}_g^v\right)} \tag{13}$$

where M^u and M^v are the set of songs listened by the users u and v, respectively. With the similarity function, the CSM-UK system recommends the top-N most similar songs for each user, given the user contextual preference and their most similar users. The predicted preference for the target user u to a song m can be defined as:

$$pp_{CSM-UK}(u,m) = \left(\sum_{v \in U^{u,K} \cap U^m} \frac{\text{sim}(u,v)}{|U^{u,k} \cap U^m|} \right) + \cos\left(\mathbf{p}_c^u, \mathbf{v}_m^{c2v}\right) \tag{14}$$

where $U^{u,K}$ is the set with the K users more similar to u, and U^m is the set of users who have listened to song m.

5.3 Evaluation Setup

As a baseline for our methods, it was used the approach proposed by [6] which is considered state-of-the-art when it comes for acquiring embedding vectors from songs, and that empirically outperformed several other recommender systems (i.e. Temporal Recommendation Based on Injected Preference Fusion (IPF), Bayesian Personalized Ranking (BPR), FISMauc (FISM), Factorizing Personalized Markov Chains (FPMC), Hierarchical Representation Model (HRM), and User-based Collaborative Filtering (UserKNN)).

To verify if our methods are able to achieve better results than the one proposed by [6], the context-aware recommender systems were executed using the k-fold cross-validation protocol.

In this k-fold cross-validation, the users of the datasets are split into k mutually exclusive partitions, in which 1 of these partitions is chosen as the testing partition and the remaining are chosen as the training partition. The testing partition is chosen k times, without repeating the same partition, as seen in [26]. Figure 3 shows the process of splitting the users into partitions, assuming $k = 5$ that is the value used in our work.

Users that are in the training partition use all of their songs sessions to build their preferences (general and contextual), as it can be seen in Figure 3. As for the users that are in the testing partitions, they use only the first part of their sessions to build their preferences, and the second part for each session is used as the testing songs.

To evaluate the recommendations made by the recommender systems, it was used five different metrics, which three (Precision, Recall, and F-measure) are commonly used to evaluate the accuracy of the recommendations, and two metrics are used to evaluate if the ranking of the recommendations meets the user's preference, which are Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) [27].

Precision, as described by [28], is a metric that measures the proportion of satisfying recommendations made by the recommender system, indicating the quality of

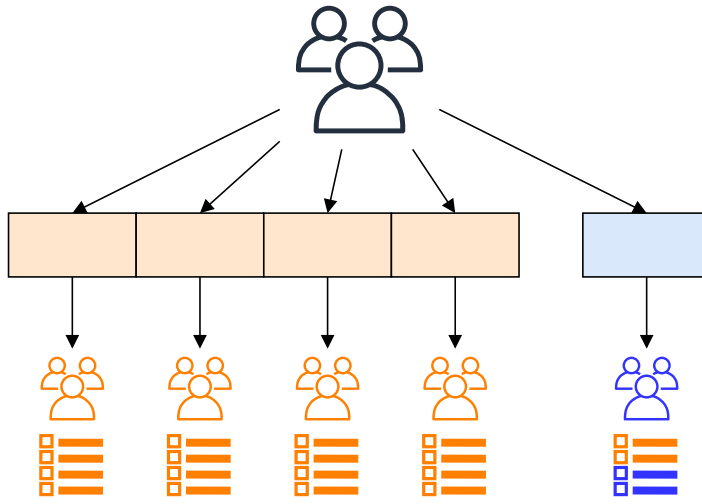

Figure 3. The k-fold cross-validation protocol with $k = 5$

recommendations made with an emphasis on the success of the recommendations. As seen in [29], Precision can be defined as:

$$Precision = \frac{|tp|}{|tp| + |fp|} \tag{15}$$

where, according to Table 1, tp means true positive and fp means false positive.

Recall, on the other hand, measures the proportion of the recommendations among the songs that the user is actually interested in. [29] defined Recall as:

$$Recall = \frac{|tp|}{|tp| + |fn|} \tag{16}$$

where, according to Table 1, tp means true positive and fn means false negative.

The F-measure metric is defined as the harmonic mean between the Precision and Recall metrics and as those metrics, its value varies between 0 and 1. As seen in [29], F-measure can be defined as:

$$F\text{-}measure = 2 * \frac{Precision * Recall}{Precision + Recall}. \tag{17}$$

The ranking metrics that were used in this work have different goals. MAP, for instance, as described in [29], has as its main focus to ensure that the first few items in the recommendation list are in the correct order. If they are not, the metric will penalize the recommendation list.

The MAP metric can be calculated as a mean of the Average Precision (AP) metric for each recommendation made for a single user. The metric AP can be

calculated for a recommendation list with N items (in our case, songs) as:

$$AP@N = \frac{1}{N}\sum_{k=1}^{N} P(k) \cdot \text{rel}(k) \tag{18}$$

where $P(k)$ refers to the Precision metric calculated to the first k elements of the recommendation list and rel(k) is the operation that verifies if the item that is in the recommendation list in the position k is the same that is in the position k in the testing list, returning 1 if it is true or 0 if it is false.

The NDCG metric, in contrast to the MAP metric, does not favor the items that appear first in the recommendation list. Its goal, as defined by [29], is to offer a metric that is appropriated to large recommendation lists in which the penalty is applied to the items that are further from the beginning of the list. Assuming that a user u has a gain $g_{u,i}$ for being recommended an item i to it, the mean of the metric Discounted Cumulative Gain (DCG) for a list of J items can be defined as:

$$\text{DCG} = \frac{1}{N}\sum_{u=1}^{N}\sum_{j=1}^{J} \frac{g_{u,i_j}}{\log_b(j+1)} \tag{19}$$

where i_j represents the item i in the position j of the list. The base b of the logarithm can be changed, but in general the value is 2 or 10. The normalized version of the metric DCG (NDCG), can be calculated as:

$$\text{NDCG} = \frac{\text{DCG}}{\text{DCG}^*} \tag{20}$$

where DCG* corresponds to the DCG computed using the set of songs that the users have listened to.

	Song Recommended	**Song not Recommended**
Song Listened to	true positive (tp)	false negative (fn)
Song not Listened to	false positive (fp)	true negative (tn)

Table 1. Classification of the possible result of a recommended song to a user [29]

In this evaluation setup, the two sided paired t-test with a 95 % confidence level is applied to compare two context-aware recommender systems [30].

5.4 Results

This subsection presents the results for both methods proposed in Section 4. It is important to say that the first method, which uses metadata attributes to improve the embedding vectors, was evaluated only in the Music4All dataset, as it is the only dataset in this work that has these attributes. Both methods followed the same evaluation setup described in the previous subsection, and all values are statistically significant.

5.4.1 Results for the Metadata-Aware Method

Here, two different metadata attributes were used to evaluate the efficiency of our first method: tags and genres. Tags, as seen in [24], are provided by the users to a song based on their involvement with the song. Genre is an attribute that is filtered from the tags based on the application Every Noise at Once[5], meaning that they are a subset of tags and also assigned by users.

As tags and genres are lists of attributes, they were evaluated by using two different strategies. The first strategy consisted of using a random value from the list of genres/tags as the metadata attribute. The second strategy used all the values in the list of attributes, concatenating all the values in the list, separated by the symbol "_". As an example, if a song has the following genres *pop*, *rock*, *metal*; in the first strategy, one genre is randomly picked up as the metadata attribute. On the other hand, for the second strategy, the value for the metadata attribute will be "*pop_rock_metal*".

After obtaining the embedding vectors from both the songs and the metadata attributes, different operations are performed with those vectors, as seen in Figure 1. The operations that were performed in this work are ADD, MUL and AVG, as described in Subsection 4.2.

As can be seen in Figure 4, the use of genre as metadata attribute provides the best results for both strategies. The only exception was the random genre with the MUL operation on the SM-TN and CSM-UK recommender systems. In Figure 5, the same behavior can be observed with ten recommendations. The only attribute which the metrics did not surpass the baseline was the random genre with the MUL operation in the SM-TN and CSM-UK recommender systems. That indicates that both strategies, i.e. using a random genre or all the genres together, are viable options as metadata attribute to improve the embedding vectors from songs in context-aware recommender systems.

It is worth to say that for five recommendations, the M-TN algorithm presented an improvement for each metric, and we can highlight an improvement of 76.32 % in Precision and 75.89 % in F-measure using all genres with the MUL operation, which was the best attribute and operation combination for this recommender system.

On the other hand, the SM-TN recommender system did not show an improvement for all combinations. The combination that stood out the most was also using all genres with the MUL operation, which showed an improvement of 12 % in Precision, 11.28 % in F-measure and, 9.52 % in NDCG. The results for the CSM-UK recommender systems are pretty similar to the SM-TN recommender system.

For the CSM-TN recommender system, which combines both general and contextual preferences, all combinations showed good improvements. As a metadata attribute, the random genre was the best attribute for this recommender system, and the operations that performed better were ADD and AVG. With the ADD opera-

[5] `http://everynoise.com`

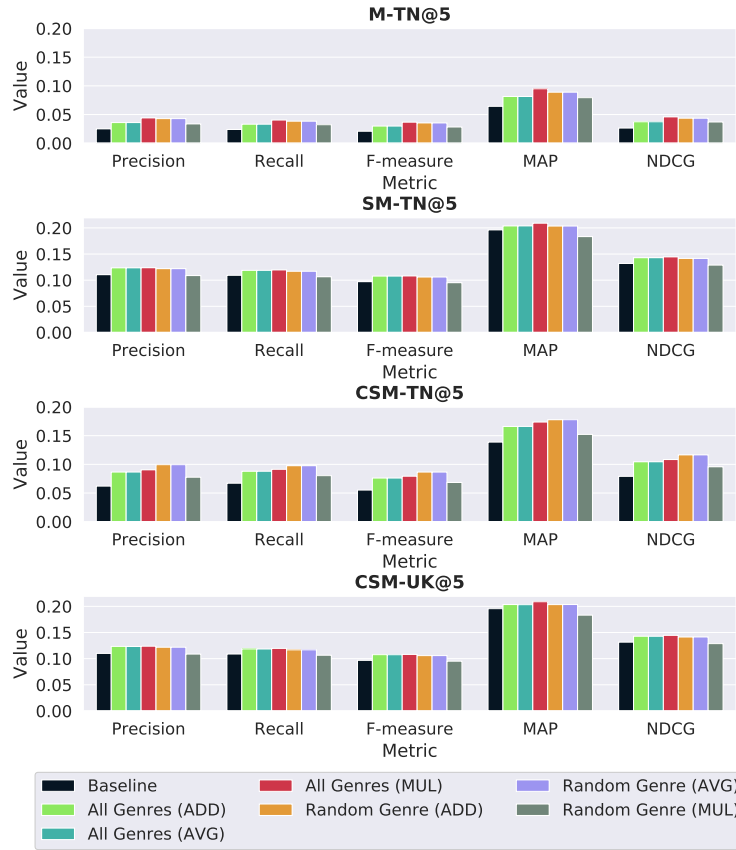

Figure 4. Results for the metadata-aware method with five recommendations using genres as the metadata attributes

tion, we obtained an improvement of 60.32 % in Precision and 56.92 % in F-measure, as well as in the AVG operation, that was an improvement of 45.57 % in Recall and 47.41 % in NDCG.

The second metadata attribute that was evaluated were the tags, and the results are presented in Figures 6 and 7. For five recommendations (Figure 6), M-TN and CSM-TN recommender systems obtained an improvement for all metadata attribute strategies and operations. Similar to what happened with the genres attribute, only the combination of the random tag with the MUL operation was not able to surpass the baseline in every recommender system and metric. It is worth to say that although we found a similar behavior between genres and tags, the absolute values obtained by the tags attributes are lower compared to the ones obtained by the genres attributes.

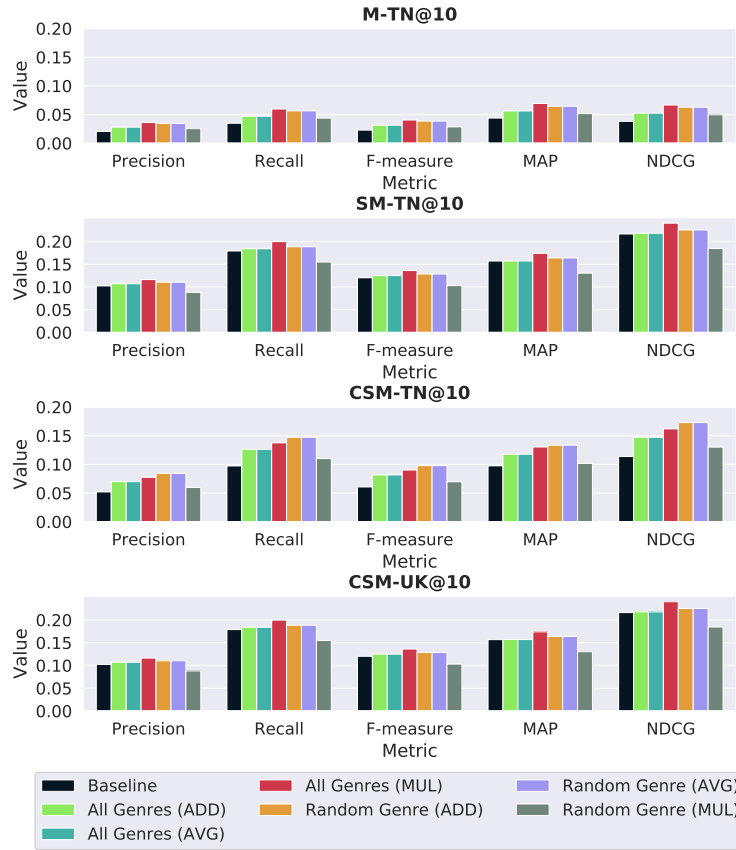

Figure 5. Results for the metadata-aware method with ten recommendations using genres as the metadata attributes

With respect to the ten recommendations (Figure 7), the random tag with the MUL operation was able to surpass the baseline only in the M-TN recommender systems, which was the only recommender system that got better results than the baseline for all possible combination (i.e. strategies and operations).

In general, by using tags, the combinations that performed better were the ones that used all the tags instead of a random tag. We can highlight the combination of using all the tags with the MUL operation, which was able to obtain an improvement of 77 % in F-measure and 75 % in NDCG for the M-TN recommender system, and an improvement of over 13 % in Precision and F-measure for the SM-TN recommender system.

Although user given tags are useful contextual information to be considered when recommending songs to users, as seen in [4], a subset of the tags, i.e. the gen-

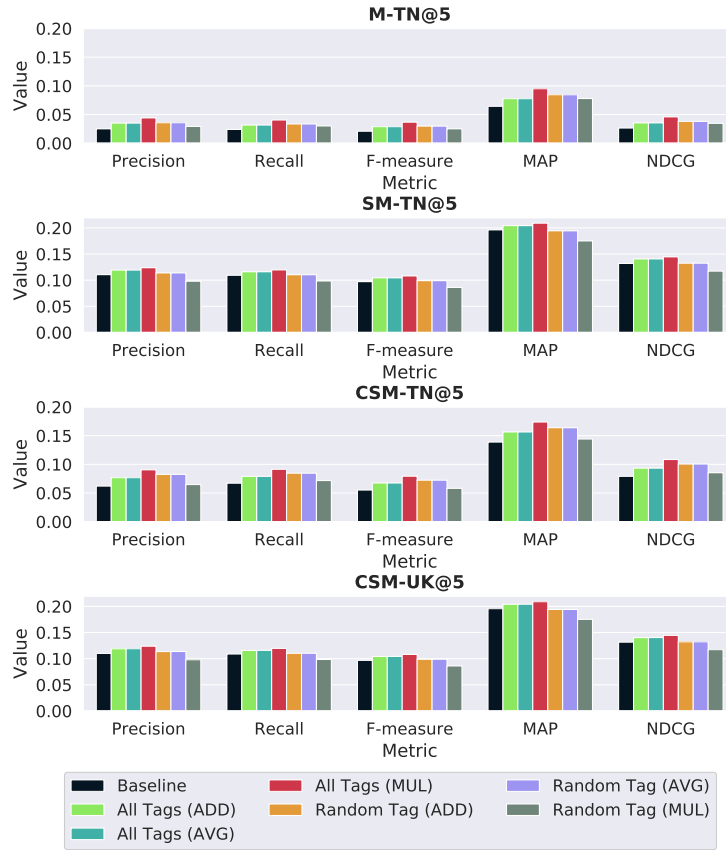

Figure 6. Results for the metadata-aware method with five recommendations using tags as the metadata attributes

res, may produce better results. Based on the results obtained with both metadata attributes, it can be observed that the genres in the Music4All dataset provided better embeddings to be used in context-aware music recommenders. The reason for this fact is that similar songs usually have similar genres, and songs of specific genres tend to be listened to by users that have similar general/contextual preferences. In other words, similar songs tend to appear in the same playing sequences. For example, a rock song is likely to appear in the playing sequences of rock users instead of classical song users. Furthermore, these co-occurrences that reflect the features of songs can be captured by our approach to learn the embeddings of songs.

Finally, the results showed that the M-TN recommender did not outperform so well as the other recommender systems. The reason for this is that the M-TN

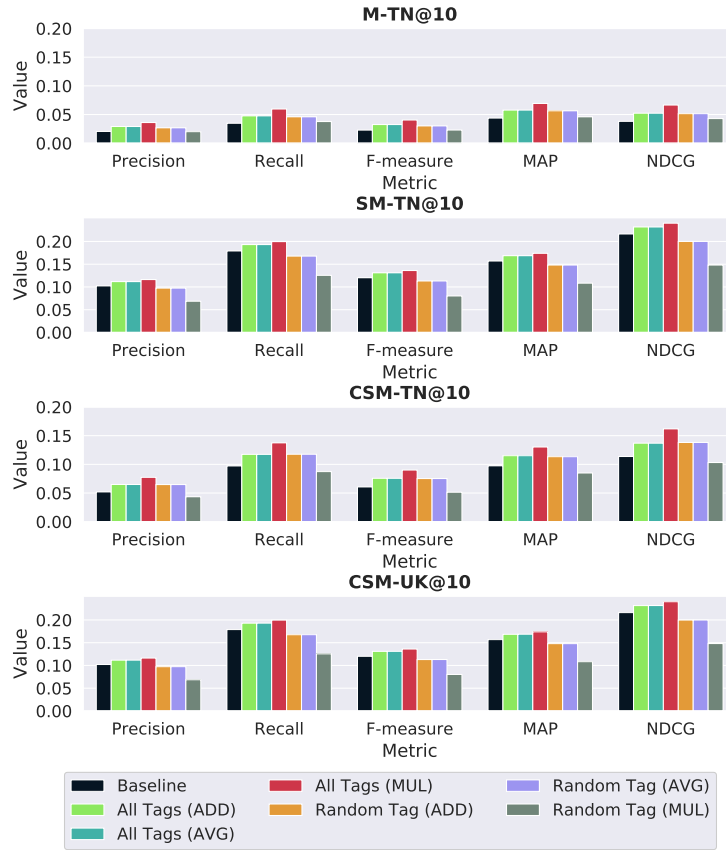

Figure 7. Results for the metadata-aware method with ten recommendations using tags as the metadata attributes

recommender uses only general preference embeddings obtained from long sequences of songs, and the model Music2Vec, used to obtain the embeddings, does not handle so well with such long sequences. The second method proposed in this work uses a recurrent neural network that handles better with long sequences of songs.

5.4.2 Results for the Dual Recurrent Neural Network Method

For the second method proposed in this work, several parameters were tested in order to obtain the best embedding vectors in both datasets, Xiami and Music4All. The parameters and their values, used by the second method, and that provided the best results can be seen in Table 2.

The best results were obtained with a contextual window of size 3, i.e. three songs before and after the target song. This value indicates that the proposed

method does not need too many songs around the target song to learn good general and contextual vector representations.

It is worth mentioning that although the Music4All is significantly lower then the Xiami dataset, when comparing the number of songs (30 % fewer songs), it needed more LSTM units and a larger embedding vector to obtain good results.

Dataset	Parameter	Value
Both	Size of the contextual window	3
Xiami	LSTM units	256
Music4All	LSTM units	512
Xiami	Embedding vectors size	256
Music4All	Embedding vectors size	1 024

Table 2. Parameters used to obtain the best results

Similar to the metadata-aware method, it was generated five and ten recommended songs for each user. The results with the Xiami dataset are presented in Tables 3 and 4, and the results with the Music4All dataset in Tables 5 and 6. The proposed method was able to outperform the baseline in both datasets. The metric that shows the best improvement over the baseline was F-measure, with an improvement of over 344 % for the M-TN recommender system in the Music4All dataset with ten recommendations.

Methods	RS	Precision	Recall	F-Measure	MAP	NDCG
Baseline	M-TN	0.0385	0.0285	0.0257	0.1324	0.0292
Dual RNN	M-TN	**0.0854**	**0.0868**	**0.0659**	**0.3346**	**0.0956**
Baseline	SM-TN	0.1189	0.1320	0.0948	0.4305	0.1532
Dual RNN	SM-TN	**0.1728**	**0.1674**	**0.1295**	**0.5269**	**0.1824**
Baseline	CSM-TN	0.0707	0.0768	0.0552	0.2780	0.0869
Dual RNN	CSM-TN	**0.1675**	**0.1670**	**0.1278**	**0.5286**	**0.1836**
Baseline	CSM-UK	0.1175	0.1310	0.0939	0.4257	0.1522
Dual RNN	CSM-UK	**0.1722**	**0.1671**	**0.1291**	**0.5261**	**0.1821**

Table 3. Results for the Xiami dataset with five recommendations. Best result is highlighted in bold (all differences are statistically significant at the 0.05 level using a two sided paired t-test). In the table, RS is a short for Recommender Systems.

For five recommendations, the proposed method was able to obtain embedding vectors that outperformed the baseline in both datasets. In the Music4All dataset, the method obtained better general embeddings (i.e. M-TN recommender) compared to the Xiami dataset, with improvements of 314 % in the F-measure and Precision, as an example. On the other hand, the method obtained better contextual embedding vectors (i.e. SM-TN, CSM-TN and CSM-UK recommenders) for the Xiami dataset in comparison to the Music4All dataset.

In Table 5, it can be seen that there is only a small improvement in the SM-TN recommender system. For the NDCG metric, as an example, there was only

Methods	RS	Precision	Recall	F-Measure	MAP	NDCG
Baseline	M-TN	0.0297	0.0389	0.0274	0.0771	0.0395
Dual RNN	M-TN	**0.0622**	**0.1089**	**0.0630**	**0.1943**	**0.1176**
Baseline	SM-TN	0.0864	0.1586	0.0883	0.2512	0.1764
Dual RNN	SM-TN	**0.1269**	**0.2008**	**0.1224**	**0.3174**	**0.2145**
Baseline	CSM-TN	0.0514	0.0944	0.0520	0.1599	0.1027
Dual RNN	CSM-TN	**0.1269**	**0.2070**	**0.1249**	**0.3209**	**0.2224**
Baseline	CSM-UK	0.0860	0.1581	0.0879	0.2508	0.1759
Dual RNN	CSM-UK	**0.1266**	**0.2004**	**0.1220**	**0.3169**	**0.2141**

Table 4. Results for the Xiami dataset with ten recommendations. Best result is highlighted in bold (all differences are statistically significant at the 0.05 level using a two sided paired t-test). In the table, RS is a short for Recommender Systems.

Methods	RS	Precision	Recall	F-Measure	MAP	NDCG
Baseline	M-TN	0.0250	0.0239	0.0208	0.0642	0.0264
Dual RNN	M-TN	**0.1036**	**0.0953**	**0.0863**	**0.1798**	**0.1091**
Baseline	SM-TN	0.1105	0.1092	0.0971	0.1962	0.1320
Dual RNN	SM-TN	**0.1200**	**0.1165**	**0.1032**	**0.2156**	**0.1329**
Baseline	CSM-TN	0.0620	0.0670	0.0551	0.1388	0.0790
Dual RNN	CSM-TN	**0.1261**	**0.1213**	**0.1080**	**0.2230**	**0.1379**
Baseline	CSM-UK	0.1100	0.1088	0.0967	0.1956	0.1315
Dual RNN	CSM-UK	**0.1190**	**0.1155**	**0.1022**	**0.2138**	**0.1319**

Table 5. Results for the Music4All dataset with five recommendations. Best result is highlighted in bold (all differences are statistically significant at the 0.05 level using a two sided paired t-test). In the table, RS is a short for Recommender Systems.

an improvement of 0.74 %. The same metric in the Xiami dataset, however, had an improvement of 19 %, as it can be seen in Table 3. Results for the CSM-UK are similar to the SM-TN recommender system for both datasets, suggesting that the information about users was not relevant to the recommenders. The CSM-TN recommender system, which combines both general and contextual embedding vectors, showed better improvements in the Xiami dataset than in the Music4All dataset, with improvements of over 90 %.

For ten recommendations, the Xiami dataset had a similar behavior than with five recommendations, with improvements remaining similar for both cases. M-TN for five recommendations had slightly better improvements than M-TN for ten recommendations, however, the remaining recommender systems had similar improvements. The proposed method obtained the best improvement in the Music4All dataset with ten recommendations in the M-TN recommender system, with improvements of over 300 % in all metrics except for MAP, which had an improvement of 212 %.

In the SM-TN and CSM-UK, the baseline outperformed our method in some metrics, such as Recall, F-measure, and NDCG. For MAP, however, our method

Methods	RS	Precision	Recall	F-Measure	MAP	NDCG
Baseline	M-TN	0.0197	0.0336	0.0219	0.0427	0.0362
Dual RNN	M-TN	**0.0870**	**0.1435**	**0.0975**	**0.1331**	**0.1576**
Baseline	SM-TN	0.0994	**0.1747**	**0.1168**	0.1527	**0.1981**
Dual RNN	SM-TN	**0.1020**	0.1710	0.1156	**0.1601**	0.1913
Baseline	CSM-TN	0.0490	0.0930	0.0573	0.0936	0.1042
Dual RNN	CSM-TN	**0.1145**	**0.1917**	**0.1303**	**0.1733**	**0.2109**
Baseline	CSM-UK	**0.0992**	**0.1745**	**0.1166**	0.1526	**0.1979**
Dual RNN	CSM-UK	0.0978	0.1646	0.1107	**0.1546**	0.1835

Table 6. Results for the Music4All dataset with ten recommendations. Best result is highlighted in bold (all differences are statistically significant at the 0.05 level using a two sided paired t-test). In the table, RS is a short for Recommender Systems.

obtained an improvement of 5 % over the baseline. Although the SM-TN did not improve over the baseline, meaning that the contextual embedding vectors were not good; the CSM-TN, which combines both general and contextual embedding vectors, improved with respect to the baseline in all metrics of over 100 %.

Finally, by comparing the results in Tables 5 and 6 against the ones in Figures 4 to 7, it can be seen that for the Music4All dataset the second method (i.e. the Dual Recurrent Neural Network) provided better results than the first one (i.e. the Metadata Awareness) in most cases, showing that a recurrent neural network can obtain better embeddings. Additionally, the results also showed that the M-TN recommender outperforms as well as the other recommender systems. The reason for this fact is that the Recurrent Neural Network used by the second method handles better with long sequences of songs, being able to generate better general preference embeddings.

6 CONCLUSION AND FUTURE WORK

This work proposed two methods to obtain general and contextual preferences (i.e. embeddings) for context-aware music recommender systems: one that tries to improve general and contextual embeddings using metadata information, and one that uses LSTM units in a Dual Recurrent Neural Network to learn general and contextual embeddings for songs. Thus, both methods generate embeddings as contextual information to be used in context-aware music recommender systems.

The results obtained by the methods in music datasets showed that they outperform the baseline (i.e. the method proposed by [6]) for four context-aware recommenders, using metrics that measured how good the recommendations are for the user and if the recommendations are ranked accordingly.

The results also showed that the first method, which combines metadata and song embeddings, can improve the recommendations generated by context-aware recommender systems. Additionally, results from the second method showed that a Recurrent Neural Network can be used effectively in capturing the intrinsic rela-

tionship between the sequence of songs that the user has listened to and generating better contextual information for context-aware recommender systems.

According to [5], listening to music is a kind of typical context-dependent behavior because users usually prefer different types of music under different contexts. As the methods proposed in this work are able to handle general and contextual user preferences, and outperformed other methods proposed in the literature, they can be seen as promising methods to implement real context-aware music recommender systems.

Besides, although both proposed methods can be used to implement real context-aware music recommender systems, the empirical evaluation showed that the second method, i.e. the Dual Recurrent Neural Network, can provide better results than the first method. However, the second method is not able to handle the metadata attributes, and a future work intends to address this issue. Finally, it is worth to mention that the first method does not handle it so well with long sequences of songs.

For future work, other embedding vector operations will be evaluated with the first method. Regarding the second method, the plan is to use different Recurrent Neural Networks such as Gated Recurrent Units, proposed by [31], which has fewer parameters as LSTM and can be used to decrease the training time and memory consumption; or even a Transformer [32], which is a simple network architecture that adopts the mechanism of attention, differentially weighing the significance of each part of the input data. Another possibility to improve the second method is also to use the metadata in the Dual Recurrent Neural Network. Finally, both methods will be evaluated also in other datasets.

Acknowledgements

To Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq/Brazil (Grant No. 403648/2016-5) for financial support and NVIDIA Corporation for donation of a GPU used in this work. This work was also supported by the Grant No. 2019/25010-5, Sao Paulo Research Foundation (FAPESP).

REFERENCES

[1] Kim, J. Y.—Belkin, N. J.—Belkin, J.: Categories of Music Description and Search Terms and Phrases Used by Non-Music Experts. Proceedings of the Third International Conference on Music Information Retrieval (ISMIR 2002), 2002, pp. 209–214.

[2] Chen, H. C.—Chen, A. L.: A Music Recommendation System Based on Music and User Grouping. Journal of Intelligent Information Systems, Vol. 24, 2005, No. 2-3, pp. 113–132, doi: 10.1007/s10844-005-0319-3.

[3] Bu, J.—Tan, S.—Chen, C.—Wang, C.—Wu, H.—Zhang, L.—He, X.: Music Recommendation by Unified Hypergraph: Combining Social Media Information and

Music Content. Proceedings of the 18th ACM International Conference on Multimedia (MM 2010), ACM Press, 2010, pp. 391–400, doi: 10.1145/1873951.1874005.

[4] Celma, Ò.: Music Recommendation. Chapter 3. Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space, Springer, Berlin, Heidelberg, 2010, pp. 43–85, doi: 10.1007/978-3-642-13287-2_3.

[5] Kaminskas, M.—Ricci, F.: Contextual Music Information Retrieval and Recommendation: State of the Art and Challenges. Computer Science Review, Vol. 6, 2012, No. 2-3, pp. 89–119, doi: 10.1016/j.cosrev.2012.04.002.

[6] Wang, D.—Deng, S.—Xu, G.: Sequence-Based Context-Aware Music Recommendation. Information Retrieval Journal, Vol. 21, 2018, No. 2-3, pp. 230–252, doi: 10.1007/s10791-017-9317-7.

[7] Mikolov, T.—Sutskever, I.—Chen, K.—Corrado, G.—Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality. Advances in Neural Information Processing Systems, 2013, doi: 10.48550/arXiv.1310.4546.

[8] Wang, X.—Rosenblum, D.—Wang, Y.: Context-Aware Mobile Music Recommendation for Daily Activities. Proceedings of the 20th ACM International Conference on Multimedia (MM '12), 2012, pp. 99–108, doi: 10.1145/2393347.2393368.

[9] Hong, J.—Hwang, W. S.—Kim, J. H.—Kim, S. W.: Context-Aware Music Recommendation in Mobile Smart Devices. Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC '14), 2014, pp. 1463–1468, doi: 10.1145/2554850.2554991.

[10] Cheng, Z.—Shen, J.: On Effective Location-Aware Music Recommendation. ACM Transactions on Information Systems, Vol. 34, 2016, No. 2, pp. 13:1–13:32, doi: 10.1145/2846092.

[11] Chen, C. M.—Tsai, M. F.—Liu, J. Y.—Yang, Y. H.: Using Emotional Context from Article for Contextual Music Recommendation. Proceedings of the 21st ACM International Conference on Multimedia (MM '13), 2013, pp. 649–652, doi: 10.1145/2502081.2502170.

[12] Hinton, G. E.—McClelland, J. L.—Rumelhart, D. E.: Distributed Representations. In: Rumelhart, D. E., McClelland, J. L., PDP Research Group (Eds.): Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations. MIT Press, Cambridge, MA, USA, 1986, pp. 77–109.

[13] Wang, D.—Deng, S.—Zhang, X.—Xu, G.: Learning Music Embedding with Metadata for Context Aware Recommendation. Proceedings of the 2016 ACM International Conference on Multimedia Retrieval (ICMR '16), 2016, pp. 249–253, doi: 10.1145/2911996.2912045.

[14] Vötter, M.—Zangerle, E.—Mayerl, M.—Specht, G.: Autoencoders for Next-Track-Recommendation. In: Schenkel, R. (Ed.): Proceedings of the 31st GI-Workshop Grundlagen von Datenbanken, Saarburg, Germany, June 11-14, 2019. CEUR-Ws.org, CEUR Workshop Proceedings, Vol. 2367, 2019, pp. 20–25, http://ceur-ws.org/Vol-2367/paper_2.pdf.

[15] Mayerl, M.—Vötter, M.—Zangerle, E.—Specht, G.: Language Models for Next-Track Music Recommendation. In: Schenkel, R. (Ed.): Proceedings of the 31st GI-Workshop Grundlagen von Datenbanken, Saarburg, Germany, June 11-14, 2019.

CEUR-Ws.org, CEUR Workshop Proceedings, Vol. 2367, 2019, pp. 15–19, http://ceur-ws.org/Vol-2367/paper_1.pdf.

[16] Hansen, C.—Hansen, C.—Maystre, L.—Mehrotra, R.—Brost, B.—Tomasi, F.—Lalmas, M.: Contextual and Sequential User Embeddings for Large-Scale Music Recommendation. Fourteenth ACM Conference on Recommender Systems (RecSys '20), 2020, pp. 53–62, doi: 10.1145/3383313.3412248.

[17] Adomavicius, G.—Mobasher, B.—Ricci, F.—Tuzhilin, A.: Context-Aware Recommender Systems. AI Magazine, Vol. 32, 2011, No. 3, pp. 67–80, doi: 10.1609/aimag.v32i3.2364.

[18] Tso-Sutter, K. H. L.—Marinho, L. B.—Schmidt-Thieme, L.: Tag-Aware Recommender Systems by Fusion of Collaborative Filtering Algorithms. Proceedings of the 2008 ACM Symposium on Applied Computing (SAC '08), 2008, pp. 1995–1999, doi: 10.1145/1363686.1364171.

[19] Goodfellow, I. J.—Bengio, Y.—Courville, A. C.: Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org/.

[20] Bengio, Y.—Courville, A.—Vincent, P.: Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 35, 2013, No. 8, pp. 1798–1828, doi: 10.1109/TPAMI.2013.50.

[21] Bengio, Y.—Ducharme, R.—Vincent, P.—Jauvin, C.: A Neural Probabilistic Language Model. Journal of Machine Learning Research, Vol. 3, 2003, pp. 1137–1155, http://jmlr.org/papers/v3/bengio03a.html.

[22] Mikolov, T.—Chen, K.—Corrado, G.—Dean, J.: Efficient Estimation of Word Representations in Vector Space. International Conference on Learning Representations (ICLR 2013) – Workshop Track Proceedings, 2013, doi: 10.48550/arXiv.1301.3781.

[23] Hochreiter, S.—Schmidhuber, J.: Long Short-Term Memory. Neural Computation, Vol. 9, 1997, No. 8, pp. 1735–1780, doi: 10.1162/neco.1997.9.8.1735.

[24] Santana, I. A. P.—Pinhelli, F.—Donini, J.—Catharin, L.—Mangolin, R. B.—da Costa, Y. M. G.—Feltrim, V. D.—Domingues, M. A.: Music4All: A New Music Database and Its Applications. Proceedings of the 27th International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE, 2020, pp. 399–404, doi: 10.1109/IWSSIP48289.2020.9145170.

[25] Resnick, P.—Iacovou, N.—Suchak, M.—Bergstrom, P.—Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW 1994), 1994, pp. 175–186, doi: 10.1145/192844.192905.

[26] Alpaydin, E.: Design and Analysis of Machine Learning Experiments. Introduction to Machine Learning, 2004, pp. 474–515.

[27] Järvelin, K.—Kekäläinen, J.: Cumulated Gain-Based Evaluation of IR Techniques. ACM Transactions on Information Systems, Vol. 20, 2002, No. 4, pp. 422–446, doi: 10.1145/582415.582418.

[28] Chung, Y.—Kim, N. R.—Park, C. Y.—Lee, J. H.: Improved Neighborhood Search for Collaborative Filtering. International Journal of Fuzzy Logic and Intelligent Systems, Vol. 18, 2018, No. 1, pp. 29–40, doi: 10.5391/IJFIS.2018.18.1.29.

[29] Shani, G.—Gunawardana, A.: Evaluating Recommendation Systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P. B. (Eds.): Recommender Systems Handbook. Springer US, Boston, MA, 2011, pp. 257–297, doi: 10.1007/978-0-387-85820-3_8.

[30] Mitchell, T. M.: Machine Learning. 1st Edition. McGraw-Hill, Inc., 1997, https://www.worldcat.org/oclc/61321007.

[31] Cho, K.—van Merriënboer, B.—Bahdanau, D.—Bengio, Y.: On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, ACL, 2014, pp. 103–111, doi: 10.3115/v1/W14-4012.

[32] Vaswani, A.—Shazeer, N.—Parmar, N.—Uszkoreit, J.—Jones, L.—Gomez, A. N.—Kaiser, L.—Polosukhin, I.: Attention Is All You Need. In: Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017). Curran Associates Inc., 2017, pp. 6000–6010, doi: 10.48550/arXiv.1706.03762.

Igor André Pegoraro Santana is currently Frontend Developer at Accountfy, Brazil. He obtained his Master degree in computer science (2020) at the State University of Maringá, Brazil; and he graduated with a degree in computer science (2017) at the State University of Maringá, Brazil. His main research interests are recommender systems and deep learning.

Marcos Aurélio Domingues is currently Professor at the State University of Maringá, Brazil. He received his Ph.D. (2010) in computer science at the University of Porto, Portugal; his Master degree in computer science and computational mathematics (2004) at the University of São Paulo, Brazil; and his degree in computer science (2002) at the Federal University of Lavras, Brazil. His main research interests are recommender systems, web personalization and web data mining.