Playlist-Based Tag Propagation for Improving Music Auto-Tagging

Yi-Hsun Lin, Chia-Hao Chung, and Homer H. Chen
National Taiwan University
{r04942055, f03942038, homer}@ntu.edu.tw

***Abstract*—The performance of a music auto-tagging system highly relies on the quality of the training dataset. In particular, each training song should have sufficient relevant tags. Tag propagation is a technique that creates additional tags for a song by passing the tags from other similar songs. In this paper, we present a novel tag propagation approach that exploits the song coherence of a playlist to improve the training of an auto-tagging model. The main idea is to share the tags between neighboring songs in a playlist and to optimize the auto-tagging model through a multi-task objective function. We test the proposed playlist-based approach on a convolutional neural network for music auto-tagging and show that it can indeed provide a significant performance improvement.**

Keywords*—Music auto-tagging, tag propagation, multi-task learning, music playlist, convolutional neural network.

I. Introduction

Music auto-tagging has received considerable attention in the field of music information retrieval and recommendation, as is evident by the release of the Million Song Dataset (MSD) [1], [2]. Many auto-tagging systems have been developed using machine learning techniques with hand-crafted features [3] or using neural network techniques to extract latent features from audio signals [4]–[6]. These systems all have one thing in common: The system performance depends on the quality of training data.

However, it is difficult to have uniform quality for the data collected from collaborative social networks, such as Last.fm,[1] because each annotator may be familiar with only a limited set of songs and tags. For example, Last.fm has over 150 million songs, but each song is labeled with only 0.26 tags in average [2]. A long-tail shape of tag distribution may be resulted if too many tags are rarely used. In the case of MSD, more than 80% of tags are linked to less than 5% of the songs (see Fig. 1). For tags in the tail region, their association with songs can be sparse, which hinders the training of music auto-tagging model.

Tag propagation is a technique developed to create additional tags for a song by passing the tags from other similar songs [7]–[9]. With this technique, a song with insufficient tags can find potentially relevant tags from similar songs. The main issue is how to determine the similarity between songs. One obvious approach is to use the hand-crafted features of audio signals for the similarity measurement [7], [8]. However, the performance of such an approach is limited by the semantic gap between music content and tags [9].

[1]https://www.last.fm/

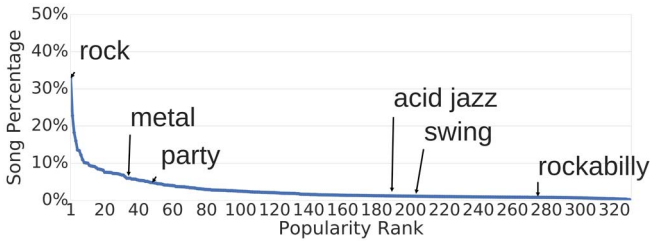

Fig. 1. Illustration of the long-tail of the Million Song Dataset. A popular tag, such as rock, metal, and party, is linked to much more songs than a rare tag, such as acid jazz and rockabilly. Over 80% of the tags are assigned to less than 5% of the songs.

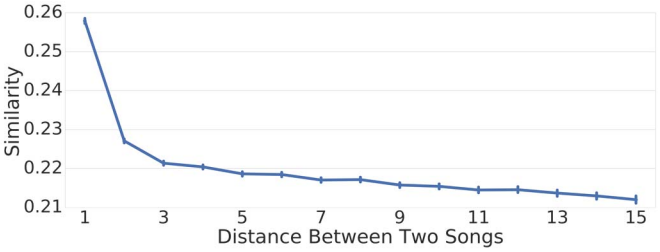

Fig. 2. Average tag similarity versus distance between two songs in a playlist. The tag similarity is calculated using the cosine similarity between the tag vectors of two songs. We can see that the tag similarity decreases as the song distance increases.

In light of the large quantity of playlists made available by music streaming services [10], [11], we propose a novel playlist-based tag propagation approach. Because songs in the same playlist are normally similar [13]–[15], their tags can be shared with each other to create additional song-tag links. Furthermore, neighboring songs in a playlist tend to have more similar tags than songs that are far apart (see Fig. 2). These characteristics of playlists are considered in our tag propagation scheme.

We further use the proposed playlist-based tag propagation to improve the training of an auto-tagging model based on a convolutional neural network. The auto-tagging model is optimized using both the original and the additional song-tag links. The optimization is carried out by a multi-task objective function to learn a generalized representation of the original and additional tags [15]–[18]. It leads to a significant performance improvement for music auto-tagging.

II. Related Work

In this section, we describe previous work related to tag propagation, music playlist, and multi-task learning.

ISBN 978-90-827970-1-5 © EURASIP 2018

A. Tag Propagation

Various music similarity estimations have been adopted for tag propagation [7]–[9]. Sordo *et al.* [7] applied a content-based similarity estimation using acoustic features and showed that 65% testing songs can be correctly labeled. On the other hand, music similarity can be estimated using the contextual information of music, for example, user listening logs and web-pages. With this observation, Kim *et al.* [9] developed a context-based tag propagation method and showed that it outperforms content-based methods. Yang *et al.* [8] further showed that tag propagation can improve the performance of a music auto-tagging model. To the best of our knowledge, little has been studied to exploit music playlists for tag propagation and auto-tagging.

B. Music Playlist

Music playlist has been used for music information retrieval and recommendation [10], [19]–[21]. Moore *et al.* [19] and Chen *et al.* [20] assumed that nearby songs in the same playlist are relevant to each other and incorporated the song-tag relationship for music retrieval. Motivated by this innovation, Chung *et al.* [21] exploited the co-occurrence of music and text (more precisely, playlist names) for music retrieval. These studies suggest that songs in the same playlist are similar to each other and relevant to the tags for certain music styles. We believe that the relevance between the tags and songs in the playlists is useful for tag propagation.

C. Multi-Task Learning

The goal of multi-task learning (MTL) is to improve the performance of a machine learning model for a target task by exploiting the knowledge obtained from related tasks [15]–[18]. It is done by learning the target task and the related tasks simultaneously. With the flexibility of deep learning framework [22], MTL has also been applied to avoid overfitting for noisy labels [15]–[18]. For example, Yang *et al*. [16] adopted a neural network-based MTL for chord recognition (the target task) and root note recognition (a related task) to regularize the representation of chords. In light of its effectiveness, we apply MTL to joint learning of the prediction of the tags of a song (the target task) and the prediction of the tags of a similar song (a related task) for auto-tagging.

III. Proposed Approach

Our goal is to improve the performance of an auto-tagging model by associating each song with more potentially relevant tags through the use of playlist information. In this section, we first describe the objective function of an auto-tagging model and the proposed playlist-based tag propagation algorithm for creating additional song-tag links. Then, we describe the incorporation of the proposed tag propagation algorithm into the objective function.

A. Objective Function for Auto-Tagging

A music auto-tagging model based on a neural network usually uses binary cross entropy:

$$\sum_{s\in S,t\in T} y_{s,t}\, log\big(f(x_s,t,\theta)\big) + (1-y_{s,t})\, log\big(1-f(x_s,t,\theta)\big) \quad (1)$$

as the objective function, where s denotes a song in a song set S, t denotes a tag in a tag set T, and $y_{s,t} \in \{1,0\}$ denotes whether or not a link exists between s and t. Here, an auto-tagging model is expressed by a function f that maps the audio signal x_s of song s to the probability of tag t being relevant to song s. All parameters in the auto-tagging model are expressed by θ for simplicity. By minimizing (1), we reduce the discrepancy between the probability of t being relevant to s and the likelihood that s is linked with t. In other words, the model is optimized to reproduce $y_{s,t}$.

However, the training dataset may not have sufficient song-tag links. The consequence is that, even if $y_{s,t}$ has value 0, there still can be relevance between s and t in reality. The purpose of tag propagation is to create such potential links.

B. Playlist-Based Tag Propagation Algorithm

To increase song-tag links, we proposed a playlist-based tag propagation algorithm to pass tags to an input song s from a relevant song s'. Denote a playlist set by P and a playlist in the set by $p = \{s_1^p, s_2^p, \dots, s_{|p|}^p\}$, where s_i^p denotes the ith song and $|p|$ the number of songs in the playlist p. As shown in the pseudocode, our algorithm first randomly selects, among P, a playlist $\tilde{p}$ containing the input song s and identifies the position (referred to by the index $\tilde{i}$) of s in $\tilde{p}$. Then, a song s' nearby s is selected by increasing or decreasing the index $\tilde{i}$, and all the tags linked with s' are passed to the input song s.

As shown in Fig. 2, closer songs in a playlist are more similar to each other. To appropriately propagate the tags of songs in a playlist, we design a searching mechanism in the song

---

Proposed Playlist-based Tag Propagation Algorithm
Inputs: s, P
Output: $\{y_{s,t'}\}_{t'\in T}$
Randomly select $\tilde{p} \in P \cap \{p | s \in p\}_{p\in P}$
for i from 1 to $|\tilde{p}|$ **do**
  if $s = s_i^{\tilde{p}}$
    $\tilde{i} \leftarrow i$
    break
  end if
end for
Randomly select Direction ∈{Forward, Backward}
while $\tilde{i} > 0$ and $\tilde{i} < |p|$ **do**
  $q \leftarrow$ Uniformly select a number ∈ [0,1]
  if q< r do
    if Direction = Forward **do**
      $\tilde{i} \leftarrow \tilde{i} + 1$
    else if Direction = Backward **do**
      $\tilde{i} \leftarrow \tilde{i} - 1$
    end if
  else do
    $s' \leftarrow s_{\tilde{i}}^{\tilde{p}}$
    for each $t' \in T$ **do**
      $y_{s,t'} \leftarrow y_{s',t'}$
    end for
  end if
end while

ISBN 978-90-827970-1-5 © EURASIP 2018

selection phase of our algorithm. During this phase, our algorithm either continues visiting the next song (by incrementing the index $\tilde{i}$ by 1) or terminates the search. We denote the probability of continuation by r. In this way, the probability of s' being selected given the input song s and the selected playlist $\tilde{p}$ can be formulated as

$$P(s'|s,\tilde{p}) = \frac{r^{d(s',s)}}{\sum_{s'\in\tilde{p}} r^{d(s',s)}}, \quad (2)$$

where $d(s',s)$ is the distance between s' and s. Since r controls the probability of the next neighboring song to be visited in the playlist $\tilde{p}$, we refer to r as probabilistic search range or, in short, search range. We can see that the probability of the tags of a song to be propagated decays with the distance between s' and s. By setting r close to 0, we allow the tags of more nearby songs to be propagated, and vice versa.

C. Incorporating Tag Propagation into Objective Function

To incorporate a propagated tag t' into the optimization of an auto-tagging model, we modify the objective function (1) as follows:

$$\sum_{s\in S,t\in T}\left(y_{s,t}\log\left(f(x_s,t,\theta)\right)+(1-y_{s,t})\log\left(1-f(x_s,t,\theta)\right)\right)+$$
$$\sum_{s\in S,t\in T}P(s'|s)\left(y_{s,t'}\log\left(f(x_s,t',\theta)\right)+(1-y_{s,t'})\log\left(1-f(x_s,t',\theta)\right)\right), \quad (3)$$

where the first summation term is the original binary cross entropy and the second summation term is the supportive binary cross entropy, which allows the minimization of the discrepancy between the probability that t' is relevant to s and the likelihood that t' is propagated to s.

Note that, the supportive objective function is weighted by

$$P(s'|s) = \frac{\sum_{\tilde{p}\in\tilde{P}} P\left(s'|s,\tilde{p}\right)}{|\tilde{P}|}, \quad (4)$$

which represents the probability that s' is sampled by our algorithm. Here, $\tilde{P}$ is the set of playlists that contain s and $P(s'|s,\tilde{p})$ is the probability of the song s' being selected given s and a playlist $\tilde{p}\in\tilde{P}$. By minimizing (3), the auto-tagging model is optimized to reproduce not only $y_{s,t}$ but also the additional song-tag link created by our algorithm (i.e., $y_{s,t'}$). This way, even if $y_{s,t}$ has value 0, the model can learn to predict the relevance between s and t'. Meanwhile, by using (4), the model can focus on t' that is more relevant to s.

IV. Experiment Setup

The proposed tag propagation algorithm is incorporated into a state-of-the-art auto-tagging model, and the performance of the resulting auto-tagging model is measured to test the effectiveness of our approach. In this section, we describe the details of the model, the dataset, and the evaluation metrics used in the experiment.

A. Model

The state-of-the-art convolutional neural network (CNN) [6], which consists of 11 convolution layers with 1.9×10^6 parameters, was adopted as the auto-tagging model. Specifically, the software[2] of this network released by Lee *et al.* [6] was adopted with (3) as the objective function. The model was optimized (or trained) by using stochastic gradient descent with learning rate = 0.01 and momentum = 0.9. During the training, the learning rate was decayed by five once an increase of the validation loss was found. To avoid overfitting, the training was terminated when the validation loss kept increasing for 12 epochs.

B. Dataset and Preprocessing

The Million Song Dataset (MSD) was used in our experiment. The audio signal and tags of each song used in the experiment were collected from 7digital.com and Last.fm dataset,[3] respectively. All audio signals were resampled to 22,050 Hz and randomly clipped into 50,000 samples (2.27 seconds) for training. Unlike other approaches that considered only the most popular 50 tags [5], [6], we also considered the most popular 1000 tags to study the behavior of the tags in the tail region. That is, we considered two training setups: 50-tags and 1000-tags.

On the other hand, the playlists used in the experiment came from the Art-of-the-Mix 2011 dataset[4] [10]. Each song in the playlist is refer to by a song ID number defined by the MSD. However, songs without tags were excluded from the experiment. At the end, a total of 80,578 playlists with average length 6 were used in our experiment. Among the songs of these playlists, there were 70,427 unique (i.e., distinct) songs. Furthermore, among these unique songs, only 26,844 songs have both audio and tags. These were the final song used for training. The songs for validation and testing were chosen from the MSD as well but excluding songs already used for training. In total, 10,054 songs were used for validation and 6,354 songs for testing.

C. Evaluation

For comparison purpose, the original CNN model [6] with the objective function (1) is referred to as the baseline. Both models were trained the same way. To evaluate the performance of both models, a retrieval task and an annotation task [3]–[7] were conducted. The performance was measured by the mean average precision (MAP), which measures the percentage that a predicted tag (in the annotation task) or a retrieved song (in the retrieval task) is correct. Besides, the receiver operating characteristic curve (AUC-ROC) was used to measure the performance of the retrieval task [4]–[6]. This metric measures the likelihood that a model ranks a relevant song higher than an irrelevant song.

V. Result and Discussion

In this section, we examine the impact of the search range on the effectiveness of our approach. We also compare the performance of the modified CNN model with the baseline CNN model. Then, we examine the performance improvement of our model for tags located at various positions of the long tail

[2]https://github.com/tae-jun/sample-cnn
[3]http://labrosa.ee.columbia.edu/millionsong/lastfm
[4]https://bmcfee.github.io/data/aotm2011.html

ISBN 978-90-827970-1-5 © EURASIP 2018

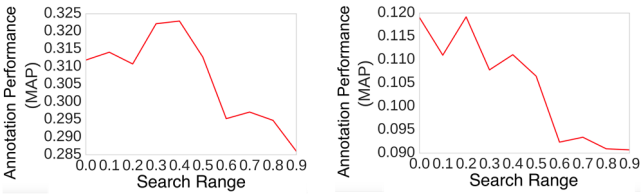

Fig. 3. Annotation performance of the modified CNN model versus search range. Left: trained with the most popular 50 tags. Right: trained with 1000 tags.

Fig. 4. Retrieval performance of the modified CNN model versus search range. Left: trained with the most popular 50 tags. Right: trained with 1000 tags.

(Fig. 1). Finally, we divide the tags into six categories and investigate the effectiveness of our model for each category.

A. *Impact of the Search Range*

Figs. 3 and 4 depict the annotation and retrieval performance of the modified CNN model versus search range. When only the most popular 50 tags are used for training, the best performance of our model is achieved when the probability is 0.4, which is a medium value. This may be explained by considering 1) The smaller the search range, the more likely our approach selects the closer neighboring song for tag propagation, and, as a result, less additional song-tag links will be created, and 2) As the search range increases, the chance that our approach selects distant songs for tag propagation increases. As a result, less relevant tags are created since the tag similarity decreases with the distance between songs (Fig. 2). It takes some trials to determine an appropriate value for the search range.

It is interesting to note that the best search range for the model trained with 1000 tags is 0.2, which is lower than the model trained with the most popular 50 tags. This implies that the tags in the long tail region are more diverse, so are the music styles labeled by these tags. In other words, it is less likely for a song in this region to have a neighboring song with the same style. Therefore, a lower search range is resulted.

B. *Comparison with the Baseline*

Table I shows the annotation and retrieval performance of the modified CNN (our model) and the baseline CNN model trained with the most popular 50 tags and 1000 tags, for which the search range was set to 0.4 and 0.2, respectively. We can see

TABLE I. PERFORMANCE COMPARISON

		Annotation	Retrieval	
Model	***# Training Tags***	**MAP**	**MAP**	**AUC-ROC**
Baseline	50	0.296	0.172	0.701
Proposed		0.323 (9.0%)	0.193 (12.2%)	0.721 (1.2%)
Baseline	1000	0.0948	0.0115	0.559
Proposed		0.119 (25.8%)	0.0179 (55.7%)	0.648 (15.9%)

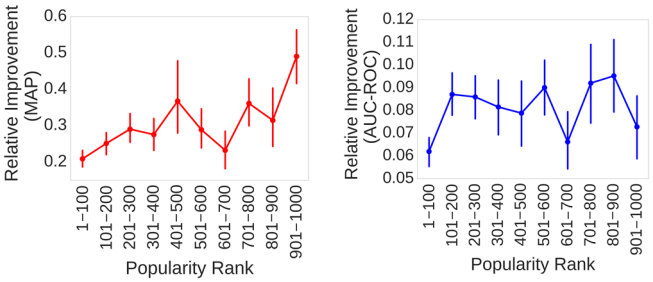

Fig. 5. The relative improvement of retrieval performance vs. tag popularity rank. The improvement is averaged per 100 tags. The popularity rank indicates the position of a tag on the long tail.

that our model outperforms the baseline model in all cases. The number in each pair of parentheses indicates the improvement percentage of our model over the baseline. Comparing the percentage improvements, we can see that the improvement of our method for the 1000-tag case is higher than that for the 50-tag case. The results clearly show that our method improves the performance of the baseline model, even more so when the training data include long-tail tags.

C. *Performance Improvement for Long-tail Tags*

Fig. 5 depicts the relative improvement of retrieval performance versus tag popularity rank, which indicates the position of a tag in the long tail. In terms of MAP, the improvement generally increases with the popularity rank. In terms of AUC-ROC, the improvement steeply increases for the most popular 200 tags and remains high for the rest 800 tags. A positive correlation can be observed between the improvement and the popularity rank. This shows the effectiveness of the song-tag links created by our method. The lower the popularity of a tag, the less the relevant songs linked to the tag, and the more beneficial the creation of song-tag links.

D. *Effectiveness for Different Tag Categories*

We divided the tags into six categories as shown in Table II. The relative improvement and the average popularity for the six tag categories is shown in Fig. 6. We can see that our approach is more sensitive to the category than the popularity. For example, the improvement for the "era" tags is higher than that for the "mood" tags, even though the two categories have similar popularity. This interesting observation indicates that it can be advantageous to consider tag category in our approach. A possible way is to assign an appropriate search range to each category. Another way is to weight the supportive objective

ISBN 978-90-827970-1-5 © EURASIP 2018

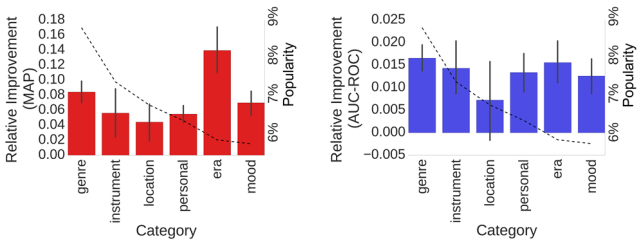

Fig. 6. Relative improvement and average popularity plotted against different tag categories. The popularity is the percentage of songs linked to a tag.

function according to the category of the propagated tags. These ideas will be further explored in future work.

VI. Conclusion and Future Work

We have described a novel playlist-based tag propagation approach for improving a convolutional neural network-based music auto-tagging model. Experimental results show that our approach improve the performance of the auto-tagging model in most cases, especially for tags in the long-tail region. For future work, we are interested in 1) applying our approach to other neural network-based models, 2) extending the objective function with tag-dependent weights to control the influence of tags according to the diversity of tags as discussed in Sec. V. A, and 3) exploiting further the similarity of music contextual information, such as artist and user listening log, for tag propagation.

TABLE II. Tag Categories

Category	Tags
genre	Hip-Hop, acoustic, alternative, alternative rock, blues, classic rock, country, dance, electronic, electronica, experimental, folk, hard rock, indie, indie rock, jazz, metal, pop, punk, rock, soul
instrument	instrumental, guitar, female vocalist, female vocalists, male vocalists
personal	Awesome, Favorite, Favourites, favorites, Love, cool, seen live, singer-songwriter
era	00s, 70s, 80s, 90s
mood	oldies, ambient, chillout, chill, Mellow, party, beautiful, easy listening, sexy, catchy
location	american, british

References

[1] B. McFee, T. Bertin-Mahieux, D. P. W. Ellis, and G. R. G. Lanckriet, "The million song dataset challenge," in *Proc. 21st Annu. Conf. World Wide Web Companion*, pp. 909, 2012.

[2] P. Lamere, "Social tagging and music information retrieval," *J. New Music Res.*, vol. 37, no. 2, pp. 101–114, 2008.

[3] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere, "Autotagger: A model for predicting social tags from acoustic features on large music databases," *J. New Music Res.*, vol. 37, no. 2, pp. 115–135, 2008.

[4] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Proc. IEEE Int. Conf. Acoustics Speech Signal Process.*, pp. 6964–6968, 2014.

[5] K. Choi, G. Fazekas, and M. Sandler, "Automatic tagging using deep convolutional neural networks," in *Proc. 17th Int. Soc. Music Inform. Retrieval Conf.*, pp. 805–811, 2016.

[6] J. Lee, J. Park, K. L. Kim, and J. Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," in *Proc. Sound Music Computing Conf.*, 2017.

[7] M. Sordo, C. Laurier, and Ò. Celma, "Annotating music collections : How content-based similarity helps to propagate labels," in *Proc. 8th Int. Soc. Music Inform. Retrieval Conf.*, pp. 531–534, 2007.

[8] Y.-H. Yang, D. Bogdanov, P. Herrera, and M. Sordo, "Music retagging using label propagation and robust principal component analysis," in *Proc. 21st Annu. Conf. World Wide Web Companion*, pp. 869–876, 2012.

[9] J. H. Kim, T. Brian, and T. Douglas, "Using artist similarity to propagate semantic information," in *Proc. 10th Int. Soc. Music Inform. Retrieval Conf.*, pp. 375–380, 2009.

[10] B. McFee and G. Lanckriet, "Hypergraph models of playlist dialects," in *Proc. 13th Int. Soc. Music Inform. Retrieval Conf.*, pp. 343–348, 2012.

[11] M. Schedl, H. Zamani, C. W. Chen, Y. Deldjoo, and M. Elahi, "Ressys challenge 2018: Automatic playlist continuation," in *Late-Breaking/Demos Proc. 18th Int. Soc. Music Inform. Retrieval Conf.*, 2017.

[12] J. H. Lee, "How similar is too similar?: Exploring users' perceptions of similarity in playlist evaluation," in *Proc. 12th Int. Soc. Music Inform. Retrieval Conf.*, pp. 109–114, 2011.

[13] D. Jannach, I. Kamehkhosh, and G. Bonnin, "Analyzing the characteristics of shared playlists for music recommendation," in *Proc. 6th Workshop Recommender Systems Social Web*, 2014.

[14] G. Bonnin and D. Jannach, "Automated generation of music playlists: Survey and experiments," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 1–35, 2014.

[15] B. Ahmed, T. Thesen, K. Blackmon, R. Kuzniecky, O. Devinsky, J. Dy, and C. Brodley, "Multi-task learning with weak class labels: Leveraging iEEG to detect cortical lesions in cryptogenic epilepsy," in *Proc. 1st Machine Learning Healthcare Conf.*, vol. 56, pp. 115–133, 2016.

[16] M. H. Yang, L. Su, and Y. H. Yang, "Highlighting root notes in chord recognition using cepstral features and multi-task learning," in *Proc. Asia-Pacific Signal Inf. Process. Association Annu. Summit Conf.*, pp. 1–8, 2016.

[17] S. Ruder, "An overview of multi-task learning in deep neural networks," *Int. J. Data Warehous. Min.*, 2007.

[18] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang, "Representation learning using multi-task deep neural networks for semantic classification and information retrieval," in *Proc. Conf. North American Chapter Association Computational Linguistics – Human Language Technologies*, pp. 912–921, 2015.

[19] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims, "Playlist prediction via metric embedding," in *Proc. 18th ACM Int. Conf. Knowledge Discovery Data Mining*, pp. 714, 2012.

[20] J. Moore, S. Chen, T. Joachims, and D. Turnbull, "Learning to embed songs and tags for playlist prediction," in *Proc. 13th Int. Soc. Music Inf. Retrieval*, pp. 349–354, 2012.

[21] C.-H. Chung, Y. Chen, and H. Chen, "Exploiting playlists for representation of songs and words for text-based music retrieval," in *Proc. 18th Int. Soc. Music Inf. Retrieval* , pp. 478–485, 2017

[22] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symposium Operating Systems Design Implementation*, pp. 265–284, 2016.

ISBN 978-90-827970-1-5