COVER SONG IDENTIFICATION USING A CONVOLUTIONAL RECURRENT NEURAL NETWORK

Xiaochen Liu
University of California, Davis, CA, USA
[1]xchliu@math.ucdavis.edu

ABSTRACT

In this report, we propose a cover song identification (CSI) system that utilizes a convolutional recurrent neural network (CRNN) to embed variations on harmonic pitch class profile (HPCP) features from music recordings into a low dimensional space. We train this system using a contrastive loss function, optimizing the CRNN so that Euclidean distances between embeddings of different songs are maximized and the distances between embeddings of covers of the same song are minimized. For training data, we use an internal dataset composed of thousands of songs with multiple covers. We test our system on both a small and a large dataset and compare it with recent state-of-the-art CSI systems. The results show that the recurrent neural network (RNN) block of the model significantly improves cover identification compared with systems using only convolutional layers. In addition, experiments show that the system scales to recognize cover songs with tens of thousands of original works in the reference set.

1. INTRODUCTION

Cover song identification (CSI) is a popular task in music information retrieval (MIR) that aims to identify if two music recordings are different renditions or *covers* of the same composition. An efficient CSI system can find use in a number of important applications, such as classification of musical works, music rights management, and general music similarity search. Covers typically vary in terms of key, tempo, singer or instrumentation, which makes the identification challenging. A scalable solution is preferably desired to accommodate ever-growing music collections.

CSI systems generally compute audio features which are robust between covers and compare them using some similarity measure in order to identify music works with the same underlying composition. Tonal-based features such as the chromagram [29], harmonic pitch class profiles (HPCP) [19], and other variants have been customarily used in CSI as they tend to characterize the melody and chord progression which are typically common between covers. Additional mechanisms can be used to improve the identification, especially by handling key transpositions and tempo deviations, either at the feature level or the comparison level. These mechanisms can include, for example, finding the optimal transposition index to align features in key, using beat tracking to synchronize features in time, or applying dynamic time warping to handle local time misalignments between features.

One of the earliest works in CSI used cross-correlation of beat-synchronous chromagrams to compare recordings and identify covers [14]. Other works also used the chromagram, directly or indirectly, along with various comparison methods, for example, by deriving from it a chord sequence [2], a tempo-insensitive feature [21], a covariance matrix [23], a dynamic chroma feature [24], or cross-correlation features [31]. HPCP, which is closely related to the chromagram, has also been used, directly [7, 34, 35] or indirectly, by deriving cross-recurrence plots [38] or time series models from it (and other features) [37]. For a summary and evaluation of early CSI works, the reader is referred to [36] and [12], respectively.

More recent works in CSI experimented with other types of audio features, such as a locally-binarized spectrogram based on the constant-Q transform (CQT) [30] and further processed by means of a sliding 2D Fourier transform (2DFT) [33], or self-similarity matrices computed from mel-frequency cepstrum coefficients (MFCC) [41] or a combination of MFCCs and HPCPs [40]. Other works focused on designing scalable systems for large databases, for example, via hashing techniques, by encoding pairs of landmarks from beat-synchronous chromagrams [3] or by using locality-sensitive hashing with chord profiles [22]; or via embedding techniques, by applying principal component analysis on the averaged 2DFTs of beat-synchronous chroma patches [4] and further augmented using a data-driven approach [20].

In recent years, deep learning methods have been successfully applied to a wide range of tasks, including CSI. Some of the earliest works used an autoencoder to learn a lower dimensional encoding from chroma features to identify covers [15–17]. Most works preferred to employ a convolutional neural network (CNN) to tackle the problem of CSI. For example, CNNs were trained on cross-similarity matrices generated from pairs of songs using chroma features [6, 26, 28]. CNNs were also used to learn key-invariant representations, with an additional scheme to handle tempo variations [43, 45, 46]. CNNs were also trained with triplet loss to learn embeddings from the dominant melody extracted from a pretrained U-Net (a CNN variant) [10, 11].

While CNNs are efficient at extracting local spatial fea-

tures, recurrent neural networks (RNN) are capable of extracting temporal features. Since music, and more generally audio signals, are essentially time series data, we reason that exploiting the time information should be beneficial when performing audio analysis. Indeed, RNNs have been used in a number of MIR tasks already, for example, music improvisation [13], chord identification [5], and rhythm prediction [25]. To the best of our knowledge, the only work that applied RNN for CSI did it by using long-short term memory (LSTM) layers, a type of RNN, in a Siamese architecture to learn a discriminative binary representation for each music recording [44].

Some researchers proposed to combine the strengths of both CNN and RNN by designing convolutional recurrent neural network (CRNN) architectures to solve MIR problems, for example, music classification. In [8], the authors used a CRNN by concatenating two gated recurrent unit layers, a type of RNN, with 4 CNN layers and showed that their hybrid model was more efficient in terms of number of parameters and training time compared to models using only CNNs. In [18], the authors used a CRNN by concatenating the outputs from CNN and RNN layers into a fully connected layer and showed that their model improved music classification compared to models with only CNN layers. In [42], the authors proposed a CRNN where the RNN is used in time but also in the frequency dimension. CRNNs were also used for music emotion recognition, for example, in [1] and [27].

In this work, we propose to build an efficient CSI system using a CRNN, combining the strengths of both CNN and RNN models. To the best of our knowledge, CRNN models have never been applied to the problem of CSI. Our CRNN architecture is thus composed of a CNN module followed by an LSTM module. We train the CRNN to embed music works into vectors of fixed length in an Euclidean space, employing a Siamese framework with contrastive loss to maximize the distances between works from different songs, and minimize the distances between works covering the same song. We define a *song* as the musical composition, e.g., Hotel California by the Eagles, and a *work* as the recorded performance of a song, e.g., the original recording of Hotel California by the Eagles in 1976, the Hell Freezes Over version by also the Eagles, and the one covered by Marilyn Manson are three different works covering the same song.

Taking advantage of the large database of metadata along with hundreds of various audio features precomputed from millions of songs that we have at our disposal internally, we build a cover song dataset with thousands of different songs with multiple works each. We run a feature selection process to identify the features that lead to the best accuracy for the problem of CSI and propose to use a combination of derivatives of HPCP features that have been aligned with the measure of the music track and correlated with major and minor chord profiles. Our contributions are therefore the following: we propose a novel CSI system using a CRNN architecture; we compare multiple variants of audio features on the problem of CSI; we use a combination of HPCP features that are aligned with the musical measure and correlated with chord profiles to learn efficient embeddings for CSI.

The rest of the article is organized as follows. In section 2, we present our dataset, the feature selection process, and the selected features. In section 3, we describe the network architecture of our proposed system and the training process. In section 4, we evaluate our system, testing it on both a small and a large dataset. In section 5, we conclude this article.

2. DATASET AND FEATURES

2.1 Dataset

We make use of a large database of metadata associated with 90 million of up-to-date music tracks that we have at our disposal internally to build a cover song dataset from scratch. We use information such as artist names, song titles, album names, radio edit, live versions, etc., to curate a total of 29,730 different songs and 71,369 different works. Each song has between 2 and 4 distinct works, and there is no duplicate.

Every music track in our metadata database, and so every work in our cover song dataset, is associated with a set of 207 precomputed audio features representing various characteristics of the music signal, namely spatial, tonal, timbral, rhythmic, percussive, and spectral. Among these features, 110 are HPCPs and various derivatives of HPCPs, which are of particular interest for the problem of CSI.

By building our own cover song dataset, we are able to experiment with a large number of various precomputed audio features, including novel variants of popular features, as opposed to using existing datasets with a limited number of audio files or with only a handful of precomputed features. In the next subsection, we use this large set of audio features to run a feature selection process in order to select the features that lead to the best accuracy on the problem of CSI.

2.2 Feature Selection

Works which are covers of the same song are typically characterized by the same melody and thus have similar tonal characteristics, but not necessarily similar timbral or rhythmic characteristics. We therefore focus our attention on the 110 tonal features in our set of 207 precomputed features, which correspond to HPCPs and various derivatives of HPCPs. The HPCP derivatives essentially consist of various post-processing of the original HPCP features and include, for example, HPCPs correlated with chord profiles, and/or aligned with the measure of the music track, and/or various statistics computed from them. Although HPCPs have been commonly used, such derivatives have never been tested for CSI, to the best of our knowledge.

We use a simpler and more common CNN model to compare these precomputed audio features in order to identify those which would be the most effective for the problem of CSI. The architecture of this CNN model is described in more details in section 4.3. As a first step, we select out of the set of 207 features, 84 which are designed to be robust to changes in timbre and instrumentation and

which are believed to be orthogonal to each other, including 70 HPCP derivatives, and use them separately as input to this CNN system. We train the model on a small training subset containing 1,000 songs and 2,000 works for 50 epochs on each feature and validate it on a small validation subset containing 200 songs and 400 works. We compute the receiver operating characteristic (ROC) curve for each feature and select 8 that lead to the highest values for the area under the ROC curve (AUC). These top 8 features turned out to be HPCP derivatives. As a second step, each of these 8 HPCP-derived features is again used as input to the same CNN system for further selection. We train the model on a larger training subset containing 20,911 songs and 49,952 works and validate it on a larger validation subset of 8,919 songs and 21,417 works.

Table 1 compares the top 8 features given their training loss, validation loss, and validation AUC, in decreasing order of AUC. These 8 HPCP derivatives correspond to various post-processing of the original HPCP features: aligned to the bar (or measure) or beat of the music track, correlated with major or minor chord profiles, and/or scaled in amplitude. We finally select the two HPCP-derived features with the highest AUC following this feature selection process, namely the derivatives which are aligned with the measure of the music track and also correlated with major and minor chord profiles, named here *bar_minor* and *bar_major*. In the next subsection, we explain in more details how these two features are derived.

Features	Train Loss	Val. Loss	AUC
bar_minor	114.8	133.1	.915
bar_major	116.0	140.7	.914
beat_minor	117.2	144.6	.908
beat_major	117.4	144.9	.906
scaled_beat_minor	118.0	144.5	.901
scaled_beat_major	118.3	147.2	.899
scaled_minor	118.1	146.9	.898
scaled_major	119.2	154.2	.890

Table 1. Training loss, validation loss, and AUC for the top 8 features, which are HPCP derivatives, following our feature selection process.

2.3 Selected Features

The selected HPCP-derived features, *bar_minor* and *bar_major*, are derived by first correlating the traditional HPCP features with known major and minor chord profiles. This is similar to the major and minor scale groupings performed in [19] but in this case, a dot product is performed between the HPCP vector and a table that contains either major or minor chord triads. The resulting chord features are then time aligned to the nearest bar line (measure) based on the estimated tempo and beat of the music track. A normalized estimate of the 12 possible major and minor chords is finally created for the two features, respectively. This process helps to derive HPCP features which emphasize the sequential structure of the song with the major or minor chords that are present in it. Since the melody line is present in the chord estimation, the addition of melody estimation as a feature did not result in a significant boost in accuracy.

For every work in our cover song dataset, each of these two HPCP-derived features has dimensions of 12 (rows) $\times$ T (columns), where T is the number of time frames in the audio signal. We resample the features to 256 time frames by linear interpolation on the time axis so that we are normalized across the durations for all the works. We then stack each resampled feature three times vertically on the frequency axis to avoid wrap around when a melodic pattern goes beyond the displayed 12 root notes. Finally, we combine the two post-processed features in two channels. The final feature input to our proposed neural network system has dimensions of 2 (channels) $\times$ 36 (frequency bands) $\times$ 256 (time frames).

3. PROPOSED SYSTEM

3.1 Network Architecture

We propose a novel CSI system using a CRNN architecture, combining the strengths of both CNN and RNN models. Our proposed system is thus composed of a CNN module with five convolutional blocks, followed by an RNN module with three bidirectional LSTM blocks, and finally three fully connected layers. The convolutional blocks, commonly used in computer vision, help to transform the audio features into key-invariant features, while the LSTM blocks, commonly used for time series data, aim at capturing the time information from the features.

The system is designed to learn a characteristic vector of fixed length, or embedding, from a musical work. We train it by using a Siamese framework with a contrastive loss function. Specifically, the system takes a pair of works as input, either a positive pair if the works are covering the same song or a negative pair if they are from different songs, and learns two embeddings such that the Euclidean distance between two positive embeddings is small while the distance between two negative embeddings is large. The training process is detailed in subsection 3.2.

Each convolutional block in the CNN module consists of a 2-dimensional convolutional layer, a rectified linear unit (ReLU) activation function, a max-pooling layer, and a batch normalization layer; the max-pooling layer is applied only to the first two convolutional blocks to maintain the temporal dimension. We use 3×3 kernels, and the number of such kernels in each of the 5 convolutional blocks is 8, 16, 32, 64, and 128, respectively. We use a stride of 1 and zero-pad the temporal and the spatial dimensions. The CNN module takes an input of dimensions $2 \times 36 \times 256$ (the size of our feature input) and returns an output of dimension $128 \times 9 \times 64$.

The output of the CNN module is then reshaped into 1152×64 and fed into the RNN module. The RNN module is composed of 3 blocks of bidirectional LSTM with a hidden size of 256. We concatenate the outputs of each LSTM blocks leading to an output of dimensions 6×256 for the RNN module. The output of the RNN module is then reshaped into 1536×1 and fed into three fully connected

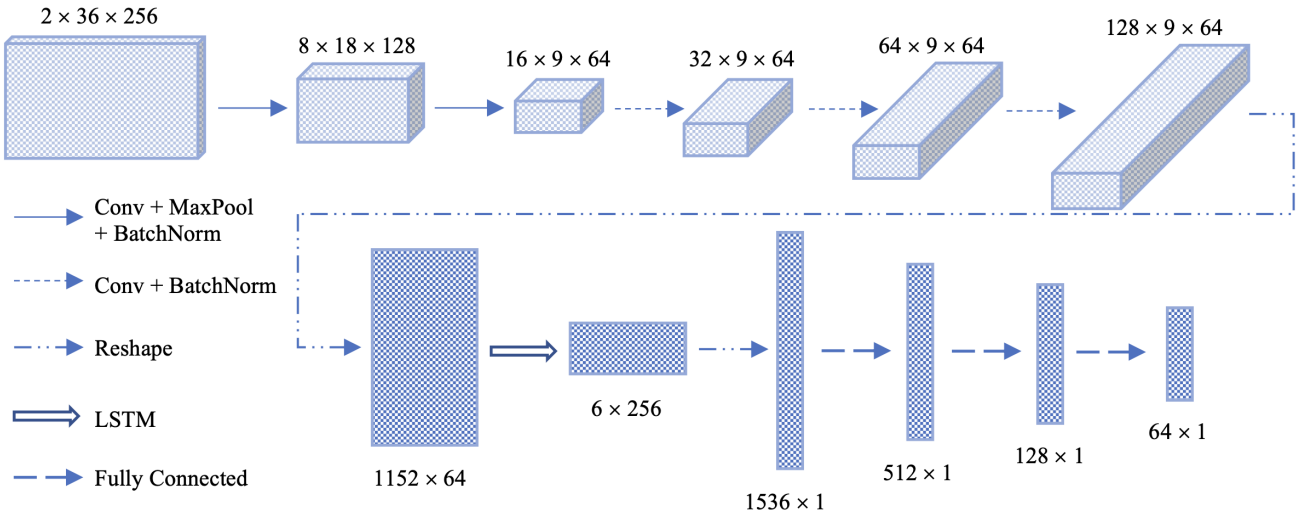

Figure 1. Overview of the proposed system.

layers with 512, 128, and 64 nodes, respectively. The final output of our proposed system is a vector of length 64 which corresponds to the embedding. Figure 1 shows an overview of our proposed system.

3.2 Training Process

We train our proposed system using a Siamese framework with a contrastive loss function [9]. The contrastive loss function is described as follows:

$$L(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, y_i) = \frac{1}{2} y_i \| f(\mathbf{x}_i^{(1)}) - f(\mathbf{x}_i^{(2)}) \|^2 + \frac{1}{2}(1 - y_j) \left[max(0, m - \| f(\mathbf{x}_i^{(1)}) - f(\mathbf{x}_i^{(2)}) \|) \right]^2 \quad (1)$$

where $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, y_i)$ defines the i^{th} triplet data, $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_i^{(2)}$ represent the feature inputs of two musical works, y_i indicates if the two works are covering the same song or not, f is the embedding function of our proposed system, $\|\cdot\|$ is the Euclidean distance, and m is a margin hyper-parameter.

As we can see, when the loss for the triplet i is minimized, the Euclidean distance between the embeddings of two works covering the same song is also minimized, while their distance is maximized by a margin of m if they are from different songs. A linear regularization with a parameter of 0.3 is added to the total loss to prevent overfitting.

During the training process, we use a batch size of 100 (i.e., 100 different pairs of works) and apply an online negative hard pair selector within each batch as described in [32]. To compute the total loss within a batch, we accumulate the loss obtained for *all* 100 positive pairs but account only for the loss of the 100 negative pairs that give the largest loss values.

We use an Adam optimizer and choose our margin m = 20. We use 0.001 as the initial learning rate and halve it every 50 epochs. We run the training process for 300 epochs. The hyper-parameters, including the margin, the regularization parameter, and the learning rate are all determined by a grid search. During the training process, we store the model that returns the largest AUC for the validation set.

4. EVALUATION

4.1 Metrics

To measure the performance of the system in finding if two music works are covering the same song, we use the AUC and the true positive rate when false positive rate is 5% (TPR@5%), which can both be derived from the ROC. We also use the mean average precision (MAP), precision at 10 (P@10) and mean rank of the 1^{st} match (MR1), metrics commonly used to measure performances in CSI.

4.2 Preliminary Tests

We divided the 29,730 songs of our cover song dataset into two sets according to a ratio of 7:3: a training set with 20,811 songs (and 49,952 works) and a validation set with 8,919 songs (and 21,417 works). Since we want our model to learn the characteristics of specific songs and what makes any two works covers of a same song, we made sure the two sets are fully separate, i.e., they do not share works from the same song.

We report preliminary results in terms of AUC and TPR@5% for the CRNN model on the training and validation sets in Table 2. We also plot the learning curves for the CRNN, but also for the CNN model used for the feature selection process, in Figure 2, where we can see that the CRNN model can achieve a smaller loss with fewer epochs compared with the CNN model.

	AUC	TPR@5%
Training Set	.991	.963
Validation Set	.978	.917

Table 2. Preliminary results for the CRNN model on the training and validation sets.

Inspired by [11], we plot the probability densities of the pairwise distances between cover works and non-cover works in Figure 3 (left). Additionally, the probability that

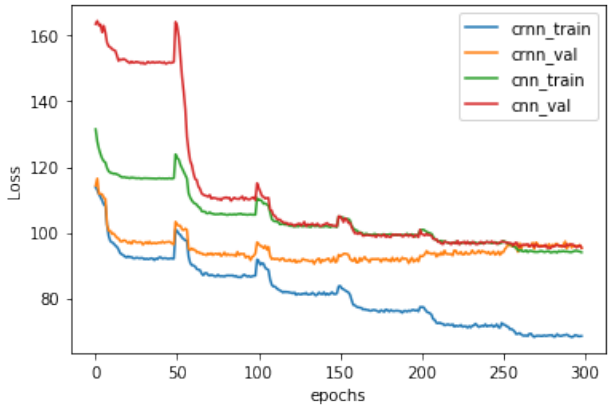

Figure 2. Learning curves for the CRNN model, and the CNN (used for feature selection), on the training and validation sets.

two works are covers given the distances between their embeddings is calculated using Bayes formula and depicted in Figure 3 (right).

4.3 Test with Small Dataset

We compare our CRNN model with the method presented in [11], which is, at the time of writing, state-of-the-art in CSI. In [11], the authors evaluate their system on the YouTube Covers dataset, which consists of 50 original songs, each with 7 different works, for a total of 350 works [39]. Out of the 7 works, 2 are used as references (for a total of 100 references) and 5 as queries (for a total of 250 queries) when evaluating on the dataset. As this dataset is not publicly available anymore, the authors in [11] mimic it using their own larger internal dataset. We proceed in a similar manner.

We first identify all the songs out of the 8,919 songs in our validation set which have at least 7 works, leading to 180 songs with a total of 1,422 works. We then randomly select 50 songs out of these 180 songs and 7 works for each song, among which, 2 are used as references and 5 as queries. With this process, we are able to randomly derive a dataset with a structure comparable to YouTube Covers.

We compute the embeddings for all the works using our CRNN model trained on the 20,811 songs and 49,952 works of our training set. We then compare each query with all the references by measuring the Euclidean distance between their embeddings. We repeat this comparison process 1,000 times for different randomizations of our mimicked dataset and report the results for the MAP, P@10, and MR1 (with means and standard deviations) in Table 3.

For the sake of comparison, we also build a simpler and more common CNN model using the same hyperparameters and training process that were used for our CRNN model. This CNN model essentially corresponds to the proposed CRNN architecture with the three LSTM blocks removed, where the output of the CNN module is reshaped from 1152×64 to 73728×1 and fed into the same three fully connected layers. This is the same CNN model used for the feature selection process in section 2.2. We evaluate this CNN model on the same mimicked dataset with the same randomizations and report the results for the same metrics in Table 3. We additionally perform a two-sample one-sided t-tests on all the results and also report the p-values.

For comparison, we also show the results reported by [11] on their own mimicked version. Note that the p-value here is obtained from a one-sample one-sided t-test.

	MAP	P@10	MR1
CRNN	.809 (.026)	.180 (.004)	2.482 (.481)
CNN	.789 (.027)	.176 (.005)	2.946 (.579)
p-value	< 0.001	< 0.001	< 0.001
[11]	.675 (.040)	.165 (.005)	3.439 (1.062)
p-value	< 0.001	< 0.001	$< .001$

Table 3. Results on the small dataset (means and standard deviations), for the proposed CRNN model, CNN model, and [11] (with p-values). Note that since there are only two works per song in the reference set, the maximum value of P@10 is 0.2.

The results show that the CRNN model can achieve higher performances in terms of CSI than a simpler and more common CNN model, suggesting that the RNN module can help to improve CSI compared to using only convolutional blocks. The results also showed that our proposed CRNN model can compete with a state-of-the-art CSI method [10], suggesting the efficiency of the model and the features.

4.4 Test with Large Dataset

Finally, we test our CRNN model, and the CNN model, using all of our training set (20,811 songs and 49,952 works) and validation set (8,919 songs and 21,417 works). In each trial, we randomly select one work as the query and all the other works as the reference set. We sample 100 works to compute the MAP, P@10, and MR1 for each experiment and we repeat the experiment 1,000 times. We report the results (with means and standard deviations) in Table 4. We also compute the p-values.

The results show that a CRNN model gets higher performances in terms of CSI compared to a CNN model, for all the metrics, and for both the training set and validation set (except for the MAP for the validation set where there is no significant improvement), confirming the usefulness of an RNN module.

5. CONCLUSIONS

We have proposed a system which combines a CNN with an RNN for CSI. The system embeds music recordings such that covers from a same composition have embeddings close to each other in the embedding space and different compositions have embeddings far from each other. To our knowledge, this is the first CSI system which combines a CNN and an RNN in a Siamese setting. We showed that the proposed system is able to get state-of-the-art results in CSI compared to other recent systems based on a CNN only.

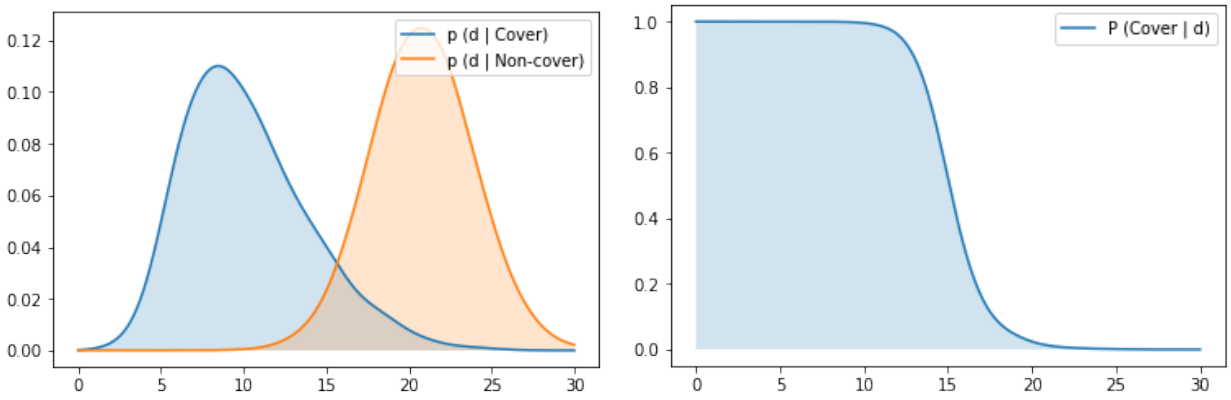

Figure 3. Left: probability densities of pairwise distances of cover works (blue) and non-cover works (yellow). Right: probability of two works being covers given the distances between their embeddings.

	MAP	P@10	MR1
	Training		
CNN	.435 (.041)	.087 (.007)	606 (297)
CRNN	.492 (.040)	.099 (.007)	293 (193)
p-value	<.001	<.001	<0.001
	Validation		
CNN	.446 (.040)	.088 (.007)	455 (188)
CRNN	.448 (.041)	.092 (.008)	363 (166)
p-value	.27	<.001	<.001

Table 4. Results on the large dataset (means and standard deviations), for the proposed CRNN model and CNN model (with p-values).

Such system could be commercially useful, for example, (1) for straightforward music identification, for detecting the name of a song being played; (2) for data curation, for accurately identifying all the versions of same song, and; (3) for copyright management, for finding cover songs uploaded online without the permission of the artist or the label.

6. ACKNOWLEDGMENT

The author would like to thank the applied research group at Gracenote, inc., especially Bob Coover, Josh Morris, Zafar Rafii and Joseph Renner, for their valuable support and useful suggestions during the author's internship.

7. REFERENCES

[1] Shahin Amiriparian, Maurice Gerczuk, Eduardo Coutinho, Alice Baird, Sandra Ottl, Manuel Milling, and Björn Schuller. Emotion and themes recognition in music utilising convolutional and recurrent neural networks. In *MediaEval Benchmarking Initiative for Multimedia Evaluation*, Sophia Antipolis, France, 2019.

[2] Juan Pablo Bello. Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats. In *8th International Society for Music Information Retrieval Conference*, Vienna, Austria, September 23-27 2007.

[3] Thierry Bertin-Mahieux and Daniel P. W. Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, USA, October 16-19 2011.

[4] Thierry Bertin-Mahieux and Daniel P. W. Ellis. Large-scale cover song recognition using the 2D fourier transform magnitude. In *13th International Society for Music Information Retrieval Conference*, Porto, Portugal, October 8-12 2012.

[5] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio chord recognition with recurrent neural networks. In *14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, November 4-8 2013.

[6] Sungkyun Chang, Juheon Lee, Sang Keun Choe, and Kyogu Lee. Audio cover song identification using convolutional neural network. In *Workshop Machine Learning for Audio Signal Processing at NIPS*, Long Beach, CA, USA, December 8 2017.

[7] Ning Chen, Wei Li, and Haidong Xiao. Fusing similarity functions for cover song identification. *Multimedia Tools and Applications*, 77:2629—2652, 2018.

[8] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. In *42nd IEEE International Conference on Acoustics, Speech and Signal Processing*, New Orleans, LA, USA, March 5-9 2017.

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, USA, June 20-26 2005.

[10] Guillaume Doras, Philippe Esling, and Geoffroy Peeters. On the use of U-Net for dominant melody estimation in polyphonic music. In *2019 International Workshop on Multilayer Music Representation and Processing*, Milano, Italy, January 23-24 2019.

[11] Guillaume Doras and Geoffroy Peeters. Cover detection using dominant melody embeddings. In *20th International Society for Music Information Retrieval Conference*, Delft, The Netherlands, November 4-8 2019.

[12] J. Stephen Downie, Mert Bay, Andreas F. Ehmann, and M. Cameron Jones. Audio cover song identification: MIREX 2006-2007 results and analyses. In *8th International Society for Music Information Retrieval Conference*, Philadelphia, PA, United States, September 14-18 2008.

[13] Douglas Eck and Juergen Schmidhuber. Finding temporal structure in music: Blues improvisation with lstm recurrent networks. In *12th IEEE workshop on neural networks for signal processing*, Martigny, Switzerland, September 6 2002.

[14] Daniel PW Ellis and Graham E Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *32nd IEEE International Conference on Acoustics, Speech and Signal Processing*, Honolulu, HI, USA, April 15-20 2007.

[15] Jiunn-Tsair Fang, Yu-Ruey Chang, and Pao-Chi Chang. Fast cover song retrieval in advanced audio coding domain based on deep learning technique. In *Data Compression Conference*, Snowbird, UT, USA, March 30-April 1 2016.

[16] Jiunn-Tsair Fang, Yu-Ruey Chang, and Pao-Chi Chang. Deep learning of chroma representation for cover song identification in compression domain. *Multidimensional Systems and Signal Processing*, 29(13):887–902, February 2017.

[17] Jiunn-Tsair Fang, Chi-Ting Day, and Pao-Chi Chang. Deep feature learning for cover song identification. *Multimedia Tools and Applications*, 76(22):23225–23238, November 2017.

[18] Lin Feng, Shenlan Liu, and Jianing Yao. Music genre classification with paralleling recurrent convolutional neural network. *CoRR*, 2017.

[19] Emilia Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):283–406, August 2006.

[20] Eric J. Humphrey, Oriol Nieto, and Juan Bello. Data driven and discriminative projections for large-scale cover song identification. In *14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, November 4-8 2013.

[21] Jesper Hojvang Jensen, Mads G. Christensen, Daniel P. W. Ellis, and Soren Holdt Jensen. A tempo-insensitive distance measure for cover song identification based on chroma features. In *33rd IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, USA, March 31-April 4 2008.

[22] Maksim Khadkevich and Maurizio Omologo. Large-scale cover song identification using chord profiles. In *14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, November 4-8 2013.

[23] Samuel Kim, Erdem Unal, and Shrikanth Narayanan. Music fingerprint extraction for classical music cover song identification. In *IEEE International Conference on Multimedia and Expo*, Hannover, Germany, June 23 -26 2008.

[24] Sy Kim and Shrikanth Narayanan. Dynamic chroma feature vectors with applications to cover song identification. In *IEEE 10th Workshop on Multimedia Signal Processing*, Cairns, Qld, Australia, October 8-10 2008.

[25] Andrew J. Lambert, Tillman Weyde, and Newton Armstrong. Perceiving and predicting expressive rhythm with recurrent neural networks. In *12th International Conference in Sound and Music Computing*, Maynooth,Ireland, July 26-August 1 2015.

[26] Juheon Lee, Sungkyun Chang, Sang Keun Choe, and Kyogu Lee. Cover song identification using song-to-song cross-similarity matrix with convolutional neural network. In *43rd IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, AB, Canada, April 15-20 2018.

[27] Maximilian Mayerl, Michael Vötter, Hsiao-Tzu Hung, Bo-Yu Chen, Yi-Hsuan Yang, and Eva Zangerle. Recognizing song mood and theme using convolutional recurrent neural networks. In *Working Notes Proceedings of the MediaEval 2019 Workshop*, 2019.

[28] Manan Mehta, Anmol Sajnani, and Radhika Chapaneri. Cover song identification with pairwise cross-similarity matrix using deep learning. In *IEEE Bombay Section Signature Conference*, Mumbai, India, July 26-28 2019.

[29] Meinard Müller. *Fundamentals of Music Processing*. Springer, 2015.

[30] Zafar Rafii, Bob Coover, and Jinyu Han. An audio fingerprinting system for live version identification using image processing techniques. In *39th IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, Italy, May 4-9 2014.

[31] Suman Ravuri and Daniel P.W. Ellis. Cover song detection: From high scores to general classification. In *35th IEEE International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX, USA, March 14-19 2010.

[32] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[33] Prem Seetharaman and Zafar Rafii. Cover song identification with 2D fourier transform sequences. In *42nd IEEE International Conference on Acoustics, Speech and Signal Processing*, New Orleans, USA, March 5-9 2017.

[34] Joan Serrà and Emilia Gómez. Audio cover song identification based on tonal sequence alignment. In *33rd IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, USA, March 31-April 4 2008.

[35] Joan Serrà, Emilia Gómez, Perfecto Herrera, and Xavier Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1138–1151, August 2008.

[36] Joan Serrà, Emilia Gómez, Xavier Serra, and Perfecto Herrera. Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In Zbigniew W. Raś and Alicja A. Wieczorkowska, editors, *Advances in Music Information Retrieval*, volume 274, chapter 14, pages 307–332. Springer-Verlag Berlin / Heidelberg, 2010.

[37] Joan Serrà, Holger Kantz, Xavier Serra, and Ralph G. Andrzejak. Predictability of music descriptor time series and its application to cover song detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):514–525, February 2012.

[38] Joan Serrà, Xavier Serra, and Ralph G Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11:1138–1151, September 2009.

[39] Diego F. Silva, Vinícius M. A. Souza, and Gustavo E. A. P. A. Batista. Music shapelets for fast cover song recognition. In *16th International Society for Music Information Retrieval Conference*, Málaga, Spain, October 26-30 2015.

[40] Christopher J. Tralie. Early MFCC and HPCP fusion for robust cover song identification. In *18th International Society for Music Information Retrieval Conference*, Suzhou, China, October 23-27 2017.

[41] Christopher J. Tralie and Paul Bendich. Cover song identification with timbral shape sequences. In *16th International Society for Music Information Retrieval Conference*, Málaga, Spain, October 26-30 2015.

[42] Zhen Wang, Suresh Muknahallipatna, Maohong Fan, Austin Okray, and Chao Lan. Music classification using an improved CRNN with multi-directional spatial dependencies in both time and frequency dimensions. In *International Joint Conference on Neural Networks*, Budapest, Hungary, July 14-19 2019.

[43] Xiaoshuo Xu, Xiaoou Chen, and Deshun Yang. Key-invariant convolutional neural network toward efficient cover song identification. In *IEEE International Conference on Multimedia and Expo*, San Diego, CA, USA, July 23-27 2018.

[44] Zhaoqin Ye, Jaeyoung Choi, and Gerald Friedland. Supervised deep hashing for highly efficient cover song detection. In *IEEE Conference on Multimedia Information Processing and Retrieval*, San Jose, CA, USA, March 28-30 2019.

[45] Zhesong Yu, Xiaoshuo Xu, Xiaoou Chen, and Deshun Yang. Temporal pyramid pooling convolutional neural network for cover song identification. In *28th International Joint Conference on Artificial Intelligence*, Macao, China, August 10-16 2019.

[46] Zhesong Yu, Xiaoshuo Xu, Xiaoou Chen, and Deshun Yang. Learning a representation for cover song identification using convolutional neural network. In *45th IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, May 4-8 2020.