Sound Event Detection in Multisource Environments Using Source Separation

Toni Heittola[1], *Annamaria Mesaros*[1], *Tuomas Virtanen*[1], *Antti Eronen*[2]

[1]Department of Signal Processing, Tampere University of Technology, Tampere, Finland
[2]Nokia Research Center, Tampere, Finland

{toni.heittola, annamaria.mesaros, tuomas.virtanen}@tut.fi, antti.eronen@nokia.com

Abstract

This paper proposes a sound event detection system for natural multisource environments, using a sound source separation front-end. The recognizer aims at detecting sound events from various everyday contexts. The audio is preprocessed using non-negative matrix factorization and separated into four individual signals. Each sound event class is represented by a Hidden Markov Model trained using mel frequency cepstral coefficients extracted from the audio. Each separated signal is used individually for feature extraction and then segmentation and classification of sound events using the Viterbi algorithm. The separation allows detection of a maximum of four overlapping events. The proposed system shows a significant increase in event detection accuracy compared to a system able to output a single sequence of events.

Index Terms: sound event detection, sound source separation, non-negative matrix factorization

1. Introduction

Humans live in a complex audio environment, and have very good skills of following a specific sound source while ignoring or simply acknowledging the others. For example we can follow a conversation in a busy background consisting of other people talking or music. The performance of automatic methods in computational auditory scene analysis (CASA) is much more limited in this task. Acoustic mixture signals contain multiple simultaneously occurring sound events, and machine listening systems are still far from the level of human performance in recognizing them.

Individual sound events can be used to describe an audio scene: they could represent in a symbolic way a scene on a busy street, with cars passing by, car horns and footsteps of people rushing. The different level descriptors represent context (street) and characteristic events (car, car horn, footsteps). Sound event detection and classification aims at processing the acoustic signal and converting it into such symbolic descriptions of the corresponding sound events present at the scene, for applications related to automatic tagging, automatic sound analysis or audio segmentation.

Previous studies related to sound event detection consider audio scenes with overlapping events that are explicitly annotated, but the detection results are presented as a sequence that is assumed to contain only the most prominent event at each time. In this respect, the systems are finding one event at each time, and the evaluation considers the output correct if the detected event is inlcuded in the annotations. The performance of such systems is very limited in case of rich multisource environments.

[1]This work was financially supported by the Academy of Finland.

Sound source separation methods have emerged in recent years for extracting a specific sound source from the mixture. Supervised sound source separation methods are used to separate the signal belonging to one sound source of interest. Unsupervised methods do not use any knowledge about the sound sources, and will usually not separate a specific sound source but a signal with roughly homogeneous spectral content that differs significantly from the background.

In this paper, we propose a sound event detection system that can recognize and temporally locate overlapping sound events in recordings belonging to various audio contexts. The signals are preprocessed using an unsupervised non-negative matrix factorization (NMF) based algorithm in order to separate sound sources into *tracks*. Each of these tracks represents a combination of the physical sources present in the original signal. Event detection is performed on each track. The separation offers the possibility of surpassing the performance levels of previous systems, by giving the possibility of detecting simultaneous events in the multisource environment. The system is evaluated with a database of audio recordings from ten everyday contexts.

The rest of this paper is organized as follows. Section 2 presents a review of related work in event detection and sound source separation. Section 3 presents the sound source separation and Section 4 presents the event detection system. Section 5 explains the database used in the evaluations and the experimental results. Section 6 provides conclusions and suggestions for further study.

2. Related work

Applications of sound event detection from audio include analysis of video sound tracks [1, 2], audio scene recognition [3, 4, 5], audio context recognition [6] or plain acoustic event detection [7]. The cited studies are done on small sets of sound events and small set of environments, and usually the sound events and audio examples are chosen so to minimize overlapping between different categories. In case of overlapping sound events, the annotation considers the most prominent one. There are few studies that consider the case of overlapping sound events. In [7] and [8], the annotation was done to include overlapping events, but the output of the systems is a sequence of non-overlapping events. The detection result ideally consists of a sequence of the most prominent sound events, and the evaluation metric in [7] is developed for that situation. To our knowledge, there is no work that considers modeling and detecting overlapping events for event detection.

The system we presented in [8] for event detection in real life recordings, is based on hidden Markov models (HMM). We trained HMMs for 61 sound event classes using mel-frequency cepstral coefficients (MFCC) extracted from the acoustic mixture signal. For event detection, the Viterbi algorithm was used to decode the best path through the HMM states, with the 61 model HMMs connected into a network. The system was evaluated against annotations that mark overlapping events, and the detection accuracy is therefore limited by the possibility of decoding only one event at each time, while the annotations contained simultaneous events. The sound separation will help overcome this limitation.

Sound source separation aims at separating a mixture signal consisting of multiple additive sources into source signals. Recently, NMF based source separation has produced good results in many applications [9]. The basic NMF separates a signal into sources in an unsupervised manner, i.e., without prior knowledge about the sources.

Supervised source separation utilizes some prior information about the sources. The prior information can include detailed models for the spectra of the source of interest [10, 11, 12], or pitch of the sound obtained by a pitch estimation algorithm in a preprocessing stage [13, 14]. These algorithms have been used to separate mixture signals and to classify the resulting sources in speech [10, 12], singing [13], instrument recognition [14], or music transcription applications [11].

3. Sound source separation

In sound source separation, a given input audio signal which consists of multiple overlapping sounds (mixture signal) is decomposed into its sound sources (ideally). For our sound event detection, we will use a sound source separation method that is based on non-negative matrix factorization of the magnitude spectrogram of the mixture signal [9].

When applied on a spectrogram representation of audio, NMF models the signal as a sum of components, each of which has a fixed magnitude spectrum and a time-varying gain. Since the algorithm is unsupervised, we cannot strictly control the outcome of the factorization, but the components correspond to redundant sound objects in the mixture signal. Each sound source in the mixture signal can become represented as the sum of one or more components. Each component can contain parts from one or more sound sources, but typically the factorization achieves good separation of sound sources.

The processing steps of our NMF based separation algorithm are as follows:

1. Window the input signal into frames using a 60 ms Hamming windows with a 25 % overlap and calculate the complex-valued spectrum $X_t(f)$ in each frame t using the fast Fourier transform. Here f denotes the discrete frequency index. Absolute values of the spectra are stored in to a magnitude spectrogram matrix $[\mathbf{X}]_{f,t} = |X_t(f)|$.
2. Calculate the non-negative matrix factorization $\mathbf{X} \approx \mathbf{SA}$ by minimizing the Kullback-Leibler divergence between the original spectrogram and a reconstructed spectrogram [15]. The number of components is fixed to four in our system. The initial reconstruction of the magnitude spectrogram matrix $\mathbf{Y}^n$ of component n is obtained as $[\mathbf{Y}^n]_{f,t} = [\mathbf{S}]_{f,n}[\mathbf{A}]_{n,t}$.
3. Reconstruct the complex spectrum $Y_t(f)^n$ of component n in frame t and frequency f as $Y_t(f)^n = X_t(f)[\mathbf{Y}^n]_{f,t}/(\sum_m [\mathbf{Y}^m]_{f,t})$. This corresponds to Wiener filtering where the source power spectrum estimate is given by the initial reconstruction of the magnitude spectrogram of the component, and the noise power spectrum estimate is given by the sum of the other components.
4. Convert the complex spectrum of each component in each frame to time domain by inverse fast Fourier transform, and combine the frames using overlap-add. The resulting time-domain signal of an individual component is dubbed a *track*.

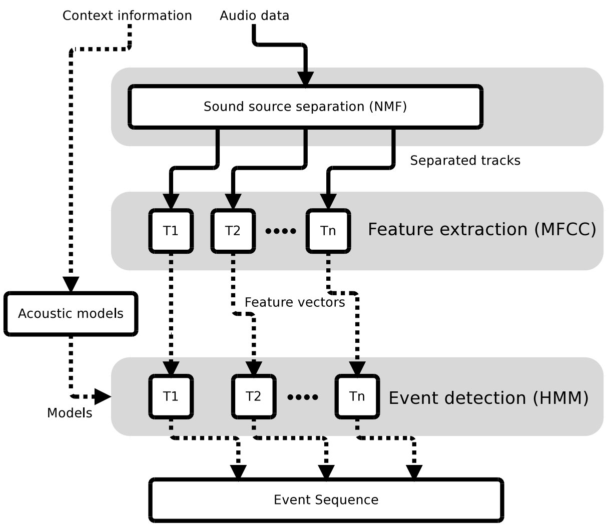

Figure 1: *System overview.*

It is clear that environmental sounds with diverse characteristics cannot be accurately modeled with the NMF model, i.e., with a fixed spectrum and time-varying gain. However, the reconstruction of tracks by Wiener filtering explains better the functioning of the algorithm: the time-varying Wiener filter of each component separates a track which contains roughly homogeneous spectral content that differs significantly from the other tracks.

The resulting separate tracks do not represent exact physical sound sources (like one track would be only sounds of footsteps), but a combination on the physical sources present in the signal. Sound event detection will be performed on each of the separated tracks. In this paper we are splitting the original multisource spectrum into four tracks. This limits the sound event detection to finding a maximum of four simultaneous sound events, in agreement with the average polyphony of our database.

4. Sound event detection

The overall system scheme is presented in Figure 1. Sound source separation is applied on the mixture signal to produce the separated tracks $(T1, T2, ..., Tn)$. Feature extraction and event detection is performed on each of these tracks separately. The results from different tracks are collected and combined into a multisource symbolic representation of the original signal, based on the total number of sound events that are recognized. This representation is then evaluated against ground truth annotations.

The event detection system consists of event class models trained from real-world audio recordings. Each event model is represented by a three-state left-to-right HMM with 16 Gaussians per state. The set of features used for constructing the models are the MFCCs (static, delta and acceleration): 16 MFCCs calculated on 20 ms windows with 50 % overlap.

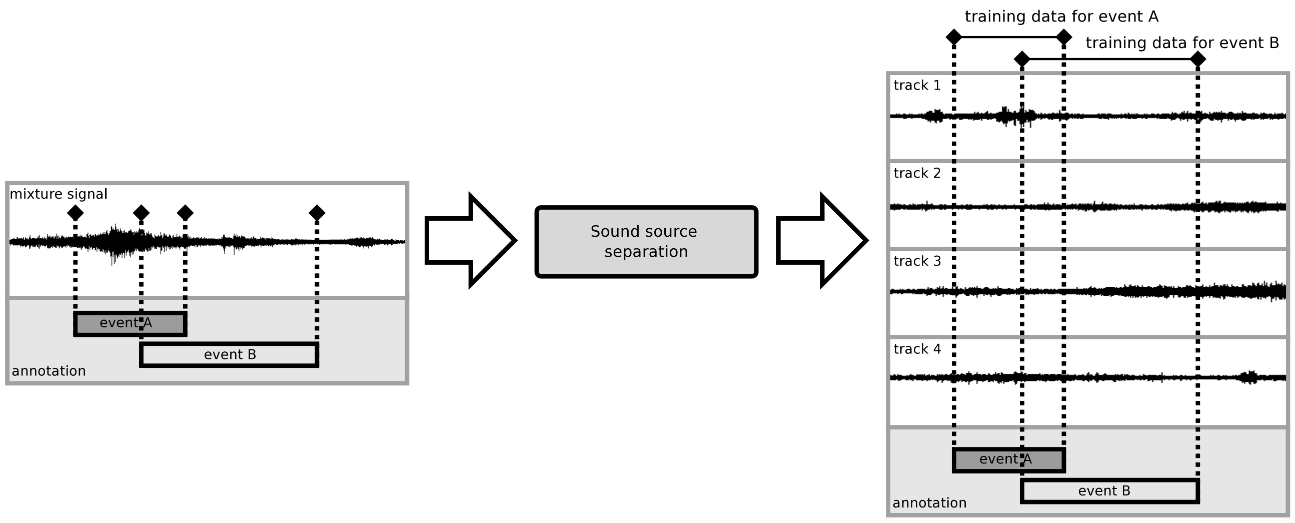

Figure 2: *Separation and segmentation procedure of the original audio for training sound event models.*

4.1. Model training

Each event instance annotated in the database represents one example for the event models. Regions of the sound that contain overlapping events were assigned to the relevant event classes and will be used as examples for all overlapping classes.

The training data is preprocessed using the described NMF-based method. Each recording is separated into four tracks and the annotations are used to provide the segmentation into event instances. Because of the unsupervised sound separation, we do not have any knowledge about which track contains what event from annotations. Therefore we assign the annotated event segment in all the separated tracks to training the annotated class. The assumption behind this is that in training, the tracks that do not contain relevant event classes will average out and the models will learn a reliable representation of the sound events. In general, the model should learn the acoustic representation of the event and average out the extra information. Figure 2 illustrates the procedure of assigning segments for training the sound event models.

We include a universal background model (UBM). This model represents the overall properties of the data and is trained by pooling all training material together. Its role in the event detection system is to capture regions when no events of interest are detected. This may happen at silent spots in the audio or in cases where the models do not score high enough to be considered plausible by the decoder.

4.2. Event Detection

For event detection, the event models are connected into a network HMM, having equal transition probabilities from one event model to another. The output of the detection step is an unrestricted sequence of the most likely model labels: any sound event can follow any other and there is no limit for the number of events. An optimal sequence of events is decoded using the Viterbi algorithm. The event detection is performed for each separated track. Then, the results from individual tracks are combined into a single description of the original audio. The final output contains timestamps and labels for all the recognized events; overlapping events from two or more tracks that carry the same label are combined into one compact representation.

5. Evaluation

The sound event detection system is trained and tested using an audio database collected from real-life contexts. The sound source separation based method is compared with a baseline system which is trained and tested on mixture signals. A detailed description of the approach used for constructing the baseline system can be found in [8]. The training and testing are done in a context-dependent manner, meaning that the number of sound event models trained and connected into a network for detection is limited to the events that are found in the annotation of the considered audio context.

5.1. Database

The material for the database was gathered by recording 10 to 30 minute long audio in ten different contexts. The selected audio contexts were basketball game, beach, inside a bus, inside a car, hallway, office, restaurant, grocery shop, street and stadium with track and field sports. The audio was recorded using binaural microphones placed inside the ears of the person recording. Each context is represented by 8 to 14 recordings, to a total of 103 recordings included in the database. In this study we are using monophonic versions of the recordings, i.e., the two channels are averaged to one channel.

The recordings were manually annotated indicating the start and end time of all clearly audible sound events in the auditory scene. Annotated sound events present in the recordings were grouped into 61 event classes. The event classes include e.g. speech, laughter, applause, car door, road, dishes, door, chair, music, and footsteps. Each context contains 9 to 16 event classes and many event classes appear in multiple contexts. There are also event classes which are context specific. The context-specific training and testing limits the number of models to 9-16 per context instead of training all 61 classes as we did in our previous work, and the material used for the training is also gathered only from the specific context.

5.2. Metric

Most of the previous studies found in the literature are concentrated on detecting non-overlapping events and the metrics presented in them are best suited for evaluating the monophonic output of the detection system. In the CLEAR evaluation [7], two metrics were defined for the sound event detection, one for detection accuracy and one for the temporal resolution of the detection. The detection accuracy was defined as the F-score between precision and recall. A detected event was regarded as correct if there was a certain degree of overlapping with an event with the same label in the annotation. The temporal resolution error was calculated by counting the entire amount of time that was wrongly attributed to events, divided by the total amount time covered by the events. The exact description of the two metrics can be found in [7]. We consider that these metrics are hard to interpret for evaluating an output with overlapping events, as it will be shown further in an example.

The recall of the system is limited by the number of events it can output, compared to the number of events that are annotated. As a consequence, even if the output contains only correct events, the accuracy is limited. The temporal resolution error represents all the erroneously attributed time, including events wrongly recognized and events missed altogether by the lack of sufficient polyphony in the detection. They are therefore complementary and tied to the polyphony of the annotation. This complicates optimization of the detection system into finding a balance between the two.

In order to tackle this problem and give a single understandable metric, we propose a block-wise accuracy for polyphonic case. This metric will evaluate how well the events detected in non-overlapping time blocks coincide with the annotations. The detected events are regarded only at the block level, for example within 30 seconds. This metric is designed for applications requiring fairly coarse time resolution, placing more importance into finding the right events within the block than finding their exact location.

Inside the blocks, we calculate precision and recall. Precision is defined as the number of correctly detected sound event classes divided by the total number of event classes detected within the block. Recall is defined as the number of correctly detected sound event classes divided by the number of all reference event classes within the block. We calculate the accuracy in each block by the F-score, based on precision and recall by the formula:

$$fscore = \frac{2 * precision * recall}{precision + recall} \quad (1)$$

An illustration of how this metric works can be seen in Figure 3. In block 1, the reference events are A, B, C and D; the system output contains A, C and D. The A and C are correctly detected. This means that for this block, precision is 2 out of 3 (2/3) and recall 2 is out of 4 (2/4). The calculated accuracy for this block is 57.1 %. For the entire example, the average block accuracy is 57.3 %.

For comparison, the CLEAR metrics calculated on the same illustration are the following: precision is 6/10 and recall is 6/9, resulting detection accuracy of 63.2 %. For calculating the time resolution error we count the units that are wrongly labeled/missed: there are 56 of them. The total number of units covered by the annotated events is 51, with a resulting time resolution error of 109.8 %.

5.3. Results

The database is divided into sets in a five-fold manner. One set is used as development set and the remaining sets are used for evaluating the system. Inside a set, 70 % of the material is used for training and 30 % for the testing.

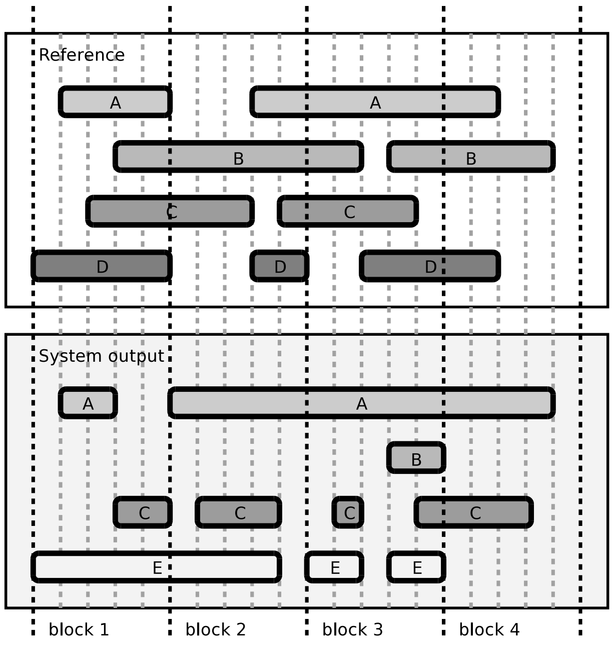

Figure 3: *Block-wise accuracy for sound event detection.*

The non-restricted Viterbi search for the optimum path results in an output containing a very large number of short events, up to ten times more events than the total annotated events. To control the length of the events, we introduce an extra cost at the inter-model transitions. This will result in the Viterbi path staying longer in each model if the cost of transitioning to a new model is higher. The development set is used to search the optimum value for this parameter. The cost value was chosen to be the one that produced a reasonable number of output events - order-wise close to the number of reference (ground truth) events.

The baseline system is trained and tested using the original mixture signal; the audio segment examples for training the classes are extracted based on the annotations, and the same region of audio was included in all annotated overlapping classes too. The baseline system uses the same development set as the proposed system, but with independent cost parameter search.

The event detection results for the baseline system and the proposed system are presented in Table 1. The overall performance of the baseline system is 25.8 % for evaluating precision and recall within 30 second blocks. This value is lower than the accuracy presented in [8], where the system was using more general models and outputting only a sequence of events. As presented in Section 5.2, the proposed block-wise accuracy is lower than the CLEAR evaluation accuracy. The 30 % performance calculated according to the CLEAR evaluation metrics is therefore meaningless without mentioning at the same time the time resolution error, which was 84 %. The block-wise accuracy could be seen as the system performance in detecting the correct events with a coarse time resolution, representing in a way a combination of the CLEAR accuracy and time resolution performance (opposite of the time resolution error).

Overall performance of the detection increases significantly by using sound source separation as preprocessing in training of the models and also in testing. Context-wise, the proposed system performs much better than the baseline system, almost doubling the overall accuracy. Individual contexts show 17 to 38 percent units improvement.

Table 1: *Sound event detection results, accuracy calculated using the block-wise accuracy metric inside 30 second blocks.*

	baseline system	proposed system
Overall	28.2	52.6
Context		
Basketball	30.3	68.2
Beach	23.0	38.7
Bus	24.4	57.6
Car	18.8	46.7
Hallway	37.0	51.1
Office	30.1	49.7
Restaurant	25.4	54.2
Shop	27.7	56.2
Street	26.4	50.1
Track & Field	41.7	57.4

The sound source separation algorithm brings important improvement not just in the numbers, but conceptually. The proposed system is able to detect overlapping events, whereas the baseline system is only producing monophonic output.

6. Conclusions

This paper presented a sound event detection system capable of detecting overlapping events in natural multisource environments. The audio is preprocessed in the sound source separation stage, and separated into four individual tracks representing combinations of the physical sources present in the signal. Sound event detection is applied to each track separately. We use recordings from ten everyday environments. In the evaluations, sound source separation was found to substantially increase the sound detection accuracy. In addition to this, the proposed system produces a conceptually accurate symbolic representation of the environment by detecting overlapping events. Thus, we conclude that the proposed method improves the sound event detection performance by producing more accurate and more realistic results.

7. References

[1] R. Cai, L. Lu, A. Hanjalic, H. Zhang, and L.-H. Cai, "A flexible framework for key audio effects detection and auditory context inference," *IEEE Trans. on Audio, Speech and Language Process.*, vol. 14, no. 3, pp. 1026–1039, 2006.

[2] M. Xu, C. Xu, L. Duan, J. S. Jin, and S. Luo, "Audio keywords generation for sports video analysis," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 4, no. 2, pp. 1–23, 2008.

[3] C. Clavel, T. Ehrette, and G. Richard, "Events detection for an audio-based surveillance system," in *Proceedings of the 2005 IEEE International Conference on Multimedia and Expo, ICME 2005, July 6-9, 2005, Amsterdam, The Netherlands*, pp. 1306–1309, 2005.

[4] A. Härmä, M. F. McKinney, and J. Skowronek, "Automatic surveillance of the acoustic activity in our living environment," *IEEE International Conference on Multimedia and Expo*, pp. 634–637, 2005.

[5] A. Eronen, V. Peltonen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 14, pp. 321–329, Jan. 2006.

[6] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Trans. on Audio, Speech and Language Process.*, vol. 17, no. 6, pp. 1142–1158, 2009.

[7] R. Stiefelhagen, R. Bowers, and J. Fiscus, eds., *Multimodal Technologies for Perception of Humans: International Evaluation Workshops CLEAR 2007 and RT 2007.* Berlin, Heidelberg: Springer-Verlag, 2008.

[8] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real-life recordings," in *18th European Signal Processing Conference*, (Aalborg, Denmark), pp. 1267–1271, 2010.

[9] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 15, pp. 1066–1074, March 2007.

[10] B. Raj, R. Singh, P. Smaragdis, and B. R. R. Singh, "Recognizing speech from simultaneous speakers," in *Proc. Interspeech*, pp. 3317–3320, 2005.

[11] J. Paulus and T. Virtanen, "Drum transcription with nonnegative spectrogram factorisation," in *EUSIPCO*, pp. 4–8, 2005.

[12] P. Smaragdis, "Convolutive speech bases and their application to supervised speech separation," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, pp. 1 –12, jan. 2007.

[13] A. Mesaros, T. Virtanen, and A. Klapuri, "Singer identification in polyphonic music using vocal separation and pattern recognition methods," in *Proceedings 7th Internationl Society for Music Information Retrieval Conference (ISMIR)*, (Vienna, Austria), pp. 375–378, 2007.

[14] T. Heittola, A. Klapuri, and T. Virtanen, "Musical instrument recognition in polyphonic audio using source-filter model for sound separation," in *Proceedings 10th Internationl Society for Music Information Retrieval Conference (ISMIR)*, (Kobe, Japan), pp. 327–332, 2009.

[15] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *NIPS*, pp. 556–562, MIT Press, 2000.