SPEECH-MUSIC DISCRIMINATION: A DEEP LEARNING PERSPECTIVE

Aggelos Pikrakis

Dept. of Informatics
University of Piraeus, Greece
pikrakis@unipi.gr

Sergios Theodoridis

Dept. of Informatics and Telecommunications
University of Athens, Greece
stheodor@di.uoa.gr

ABSTRACT

This paper is a study of the problem of speech-music discrimination from a deep learning perspective. We experiment with two feature extraction schemes and investigate how network depth and RBM size affect the classification performance on publicly available datasets and on large amounts of audio data from video-sharing sites, without placing restrictions on the recording conditions. The main building block of our deep networks is the Restricted Boltzmann Machine (RBM) with binary, stochastic units. The stack of RBMs is pre-trained in a layer-wise mode and, subsequently, a fine-tuning stage trains the deep network as a whole with back-propagation. The proposed approach indicates that deep architectures can serve as strong classifiers for the broad binary problem of speech vs music, with satisfactory generalization performance.

Index Terms— Deep learning, Speech-Music Discrimination

1. INTRODUCTION

The ever increasing availability of audio data over various distribution channels, including video sharing sites and social networks, has highlighted the need for audio content classification algorithms that are capable to deal with diverse audio types. Although a multiclass solution for the general audio classification scenario is the ultimate goal, in practice several subtasks have been studied in the literature over the years.

Such is the case with the binary problem of speech-music discrimination, with early work dating back to the late 90s. In its first definition [1], the problem referred to the classification of pre-segmented homogeneous audio segments to the speech or music class. Later work also dealt with the task of automatically segmenting uninterrupted audio streams and tagging the resulting segments with a speech or music label [2], [3], [4]. It is worth noting that the initial problem definition is still useful in the case of continuous audio streams, because it is always possible to break the audio signal into non-overlapping fixed length mid-term segments and classify each one of the resulting segments separately.

Recent advances in the field of deep networks have demonstrated remarkable classification performance in various disciplines, including the modeling of acoustic signals [5]. Therefore, in this paper, we are making an attempt to study the problem of speech-music discrimination in its original formulation, from a deep learning perspective, *without making any assumptions with respect to the origin of the signals*. This is a significantly harder task compared with subtasks that have been studied in the past, e.g., the subtask of speech / music detection in audio streams from radio broadcasts [3], [4].

To this end, we experiment with two different feature extraction schemes which are based on the spectrogram of the signal and on a variant of the Mel-Frequency Cepstrum Coefficients (MFCCs). We focus on using the standard RBM with binary, stochastic units as the main building block of our deep networks instead of resorting to more complicated RBMs that are harder to train with the Contrastive Divergence algorithm. For example, the method in [4] uses a mean-covariance RBM in the context of a shallow network for the more constrained problem of spech / music detection in radio broadcasts, but it only outperforms slightly simpler classifiers when tested on unseen data. In the current paper, the stack of RBMs is first pre-trained in a layer-wise mode and in the end a fine-tuning operation is performed via a supervised back-propagation scheme. We investigate how the depth of the network and RBM size affect the classification performance on publicly available datasets and on large amounts of audio data from video-sharing sites. Classification performance measurements are mainly provided at the frame level, but we also draw conclusions at the audio track level. The latter deals with the problem of classifying a whole recording as either music or speech, even though there can be certain misclassifications at the short-term frame level. The novelty of our approach lies in the fact that we are revisiting an old problem from a new perspective, placing emphasis on the *generalization capabilities of the method*. Our study concludes that deep architectures can serve as strong classifiers, with promising generalization performance, for the broad binary problem of speech vs music.

The paper is organized as follows: the next section presents the feature extraction stage, Section 3 describes the adopted deep architecture, Section 4 presents the experimental setup, and conclusions are drawn in Section 5.

2. FEATURE EXTRACTION

A short-term processing technique is first applied on each audio recording. The signal is thus divided into non overlapping frames and from each frame a feature vector is extracted. We investigate two feature extraction schemes, namely the spectrogram of the signal and a variant of the Mel-Frequency Cepstrum Coefficients (MFCCs) [6].

For the first feature, we compute the short-time Fourier transform every 30 ms and keep the frequencies up to 3 kHz, approximately. This frequency range carries sufficient discriminatory information for the problem at hand. For the second feature, we compute a vector of 13 MFCCs, using a filter-bank of triangular non overlapping filters, whose center frequencies have been tuned to coincide with the semitones of the chromatic scale [7]. Let $\mathbf{X} = \{x_1, x_2, \dots, x_N\}$ be the sequence of extracted feature vectors, irrespective of the adopted feature, where N is the length of the feature sequence. Each x_i is then augmented with its delta and delta-delta coefficients, a common context modeling technique in speech processing. To preserve the simplicity of presentation, we keep the notation $x_i, i = 1, \dots, N$, for the sequence of augmented feature vectors.

At a next step, we form "slices" of feature vectors, i.e., we use x_{i-1}, x_i and x_{i+1} to compile the column vector $y_i = [x_{i-1}^T \; x_i^T \; x_{i+1}^T]^T, i = 2, \dots, N-1$, where ^{T} denotes matrix transposition. Recent deep learning studies for acoustic modeling [5] have shown that feature slicing yields good feature candidates in the context of phoneme modeling. In the sequel, we will refer to the y_is with the term *patterns*. In other words, *a pattern is a column version of a slice of adjacent feature vectors*. As an example, if a vector of 13 MFCCs is augmented with its delta and delta-delta coefficients, it becomes $3 \times 13 = 39$-dimensional and each slice yields in turn a $3 \times 39 = 117$-dimensional pattern.

Finally, each feature dimension is soft-max normalized in the interval $[0, 1]$ as follows: $\hat{x}_i(k) = \frac{1}{1+exp(-z(k))}$, where k is the index of dimensions, $z(k) = \frac{x_i(k)-\mu(k)}{\sigma(k)}$, and $\mu(k)$ and $\sigma(k)$ are the mean value and standard deviation over the k-th feature dimension. In our study, the softmax normalization step is important because *it assigns a probabilistic interpretation to each feature dimension*, thus making it possible to feed the normalized pattern to the first RBM of the architecture which consists of binary, stochastic nodes.

After the normalization step has been completed, all patterns from the same audio file are given the same class label. The resulting dataset is therefore a set of pattern-label pairs, $(\hat{x}_i, c_i), i = 1, \dots, L$ where L is the total number of patterns and the c_i s are binary labels (music vs speech). We have used a binary vector representation for the class labels, i.e., $[1 \;\; 0]$ and $[0 \;\; 1]$ are the music and speech labels, respectively. This representation is dictated by the nature of the associative memory, which lies on top of the stack of RBMs and maps the output of the last RBM to the desired class labels. In the rest of the paper, we drop the hat notation from the patterns, so the i-th pattern is simply written as x_i.

3. DEEP ARCHITECTURE

The proposed architecture is a stack of Restricted Boltzmann Machines (RBMs). Each RBM is a two-layer network with binary, stochastic units. These two layers are commonly referred to as the "visible" and "hidden" layers. Each unit, $v_i, i = 1, \dots, M$, of the visible layer is connected with all hidden units with undirected weights and, similarly, each hidden unit, $h_j, j = 1, \dots, N$, is connected with all visible units. The units of the same layer are not connected with each other and this is a major difference compared with the more general Boltzmann machine.

Let $\mathbf{v}$ and $\mathbf{h}$ denote the visible and hidden nodes, respectively, and W the connection weights, i.e., w_{ij} is the weight connecting v_i with h_j. The energy, $(\mathbf{v}, \mathbf{h})$, of the joint configuration $(\mathbf{v}, \mathbf{h})$ is then defined as

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{M} \alpha_i v_i - \sum_{j=1}^{N} \beta_j h_j - \sum_{i=1}^{M} \sum_{j=1}^{N} v_i h_j w_{ij}$$

and the respective probability, $p(\mathbf{v}, \mathbf{h})$, of the configuration is

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h})} \tag{1}$$

where Z is known as the partition function: $Z = \sum_{\mathbf{v},\mathbf{h}}^{-E(\mathbf{v},\mathbf{h})}$.

During the training stage, the training patterns are "clamped" on the visible nodes and the goal of the training algorithm is to seek the weights that maximize, from a maximum likelihood perspective, the sum of the logarithms of the probabilities of the training patterns, i.e., the function $\sum_{i=1}^{L} \log p(\mathbf{v})$, where $p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$ and L is the number of patterns. In this study, we are using an approximation of the log-likelihood gradient, known as Contrastive Divergence (CD) [8], which has become very popular in practice. A detailed treatment of CD can be found in [9]. Basically, the CD procedure uses the following equation to update the RBM weights:

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon})$$

where angular brackets denote expectations over the training data and their reconstruction, respectively, and ϵ is the learning rate. To estimate the quantities inside the angular brackets, alternate Gibbs sampling is used [8], [9]. Due to the absence of direct connections among nodes of the same layer, the conditional probabilities, $p(h_j = 1 \mid \mathbf{v})$ and $p(v_i = 1 \mid \mathbf{h})$ are logistic sigmoid functions that can be easily sampled during the Gibbs sampling procedure.

As a performance measure during the training stage of each RBM, we adopt the Signal-to-Reconstruction Error-Ratio (SRER), measured in dBs. Given a training set of L

patterns (M-dimensional), we define the SRER as:

$$SRER = 10 \log_{10} \frac{\sum_{n=1}^{L} \sum_{k=1}^{M} x_n^2(k)}{\sum_{n=1}^{L} \sum_{k=1}^{M} [x_n(k) - \hat{x}_n(k)]^2} \qquad (2)$$

where x_n and $\hat{x}_n$ are the i-th pattern and its reconstruction, respectively. If the change of the SRER value is negligible between successive iterations, training is terminated. We have adopted a layer-wise pre-training scheme. It means that the RBMs are trained in succession. Specifically, after the first RBM has been trained, we feed each training pattern to its visible nodes and obtain the conditional probabilities of the hidden nodes, $p(h_j = 1 \mid \mathbf{v}), j = 1, \ldots, N$. Therefore, each M-dimensional visible pattern generates a N-dimensional probabilistic pattern that becomes the visible pattern of the next RBM in the stack. Instead of probabilities, we could have used binary codewords, i.e., force unit binarization after the conditional probability computation. However, we have observed that the propagation of binary codewords degrades the classification performance which is in accordance with similar observations in other application fields of deep learning [8]. Practical issues related to the implementation of the CD algorithm are presented in Section 4.

Further to the stack of RBMs, a top layer of weights is used to connect the output of the last RBM with a layer consisting of two softmax units. We refer to this weighting scheme by the term "associative memory", because it associates the output of the stack of RBMs with the class labels. The output of a softmax unit is a weighted sum of its input (i.e., of the output of the last RBM), divided by the sum of outputs of all softmax units (two units for the binary problem at hand). The result can be interpreted as an estimate of the posterior probability of the respective class given the input pattern. The associative memory is pre-trained for a few iterations, during which the binary representations of the class labels are clamped on the softmax units.

After the associative memory has been trained, a back propagation algorithm that minimizes the cross-entropy error function is used to fine-tune the network as a whole [8].

During the classification stage, each pattern is clamped on the visible nodes of the first RBM and the activation of hidden nodes propagates through the network until it reaches the associative memory, the output of which determines the softmax node which has produced the highest posterior probability.

4. EXPERIMENTS

The proposed method was applied on two publicly available datasets and on several hours of YouTube© audio data, without restricting the origin of the recordings.

- The first dataset (D_1) originally appeared in [1] and was subsequently refined in [10]. This corpus is a relatively small collection of 240 randomly chosen extracts from radio recordings. Each resulting file is 15 s long and stored in WAVE format (sampling frequency 22050Hz, single channel, 16 bits per sample). The dataset is partitioned by its creators into a training subset and a test subset. However, as we are using a repeated cross-fold validation scheme in our study, we have ignored the initial data partitioning scheme. D_1 consists of four classes, namely, pure music (101 files), pure speech (80 files with male, female and conversational speech), mixed tracks (60 files with speech over music), and a very small class with other types of sounds (4 files). Some of the music extracts are instrumental. The third and fourth classes were not taken into account in this study.

- The second dataset (D_2) is available via the Marsyas website [11]. It consists of a total of 120 tracks, evenly distributed among the classes of music and speech. Each track is 30 seconds long and stored in WAVE format (sampling frequency 22050Hz, single channel, 16 bits per sample). Some of the music extracts are instrumental. The music class covers a wide variety of music genres. The speech class contains both male and female speakers and in some cases dialogue. There are not any mixed tracks in this corpus (cases of speech over music).

- The third dataset (D_{INT}) was only used for testing purposes and consists of several hours of pure music and pure speech data stemming from various YouTube© audio streams. This corpus was compiled to enable testing on a large scale, on totally unseen data and on a variety of recording conditions. In this way, we can measure the generalization capabilities of our algorithm. The music class encompasses a wide variety of music genres, namely a 5-hour atmospheric video game music compilation, music collections from the 50's, 80's and 90's (2 hours) and a Celtic Music Collection (3 hours). The speech class consists of audiobooks (3 hours, narration by a single male or female speaker), discussions evolving around matters of law (2.5 hours) and briefings of officials to journalists (2 hours). For data engineering purposes and better content understanding, each uninterrupted recording was broken into a sequence of segments, where each segment was 30 s long.

4.1. Feature Extraction Details

For the computation of the spectrogram, we have used a short-frame length equal to 30 ms with zero overlap between successive frames. Given that the sampling frequency, F_s, is 22050 Hz in all our experiments, this frame length amounts to 662 samples, which are zero padded with 362 zeros to yield 1024 DFT coefficients, of which 128 are kept to cover the frequency range up to 2750Hz. After the magnitude of each DFT coefficient is computed, the vector of DFT coefficients is augmented with the delta and delta-delta coefficients, yielding a 384-dimensional feature vector. As a result, the resulting pattern (slice)is $384 \times 3 = 1152$-dimensional.

For the computation of the MFCCs, we use non overlapping frames, 50 ms long. In this case, the longer window permits a finer frequency resolution, which is necessary because the lower semitones of the chromatic scale are closely spaced, starting from the center frequency at 110Hz. Each frame yields 13 MFCCs, which eventually produce 117-dimensional patterns.

After the softmax normalization step has been completed, the dataset under study is used in a cross-fold validation scheme. At each fold, 70% of the audio files are randomly chosen for training and the rest for testing. Table 1 presents a description of D_1, D_2 and D_{INT} for each feature extraction scheme with respect to the number of patterns.

Feature	D_1-*Tr*	D_2-*Tr*	D_1-*Te*	D_2-*Te*	D_{INT}-*Te*
DFT	59400	83580	25460	35820	2.05 M
MFCC	35400	49980	15170	21420	1.23 M

Table 1. Description of datasets. *Tr* and *Te* stand for training and testing respectively. Numbers refer to patterns.

4.2. Network Training

The patterns of the training set are randomly shuffled and are grouped to form minibatches. Each minibatch consists of 100 patterns. During the RBM training stage, the weights of the RBM are updated after each minibatch has been processed and a training epoch is completed after all the minibatches have been processed. We experimented with several different network depths, with at least 3 RBMs, and with different RBM sizes (number of hidden nodes per RBM). During the layer-wise training scheme [8], [12], the value of SRER at the end of the i-th epoch ($SRER_i$) is examined and if $| SRER_i - SRER_{i-1} | < e$, RBM training terminates, where $e = 0.01$ dB in our study. If, on the other hand, the computed SRER is not negligible, training continuous for at most 100 epochs. Figure 1 presents the learning curves of the RBMs of a 500x500x2000 architecture, for the case of DFT-based features. In this figure the SRER is plotted versus the epoch index. Similarly, the associative memory

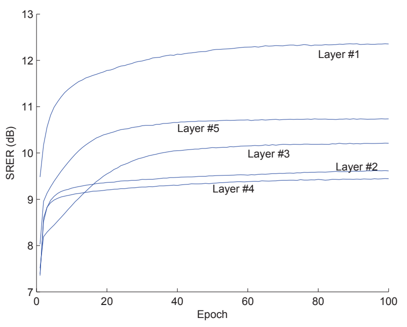


Fig. 1. Learning curves for the RBMs of a 5x200 network configuration which used DFT-based features.

is trained for a small number of epochs ($5 - 10$ epochs are enough). Finally, the whole network is trained as a whole for 50 epochs with a back-propagation algorithm which minimizes the cross-entropy error function. The weights from the pre-training stage serve to initialize the back propagation algorithm.

In all our training experiments, the cross-fold training error was measured at the pattern level and was less than 1.5% on D_1 and approximately 2% on D_2. Figure 2 presents the cross-fold classification error on D_1 and D_2 for selected network architectures. We use abbreviated notation to denote network depth and RBM size. For example, $100 \times 100 \times 400$ means that we are dealing with a 3-layer network, where the first and second RBMs have 100 hidden nodes and the last RBM has 400 nodes. It follows that the size of the associative memory in this case is 400×2 nodes, where 2 is the number of classes.

We present performance measurements using a confidence threshold, T_h. The goal of the threshold is to reject any classification decision if the estimated posterior probability of the winning class fails to exceed T_h. As complementary information, we also provide the percentage of patterns that have been left unclassified.

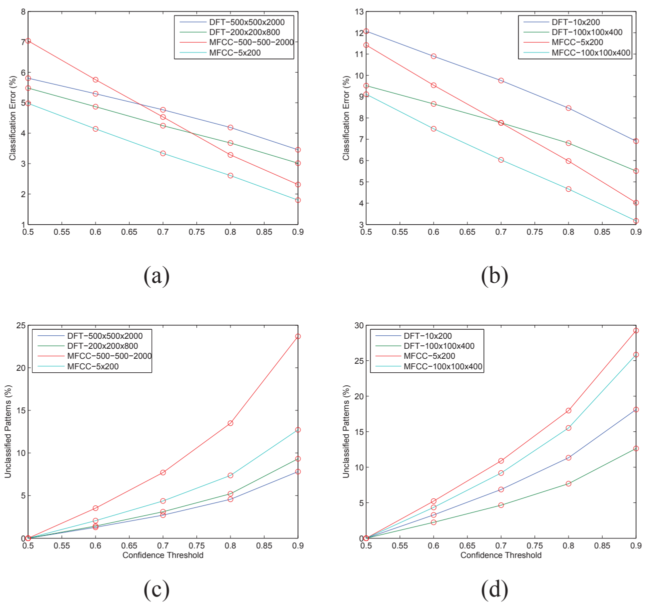


Fig. 2. (a) Cross-fold validation error on D_1 and (b) on D_2. (c) Unclassified patterns on D_1 and (d) on D_2.

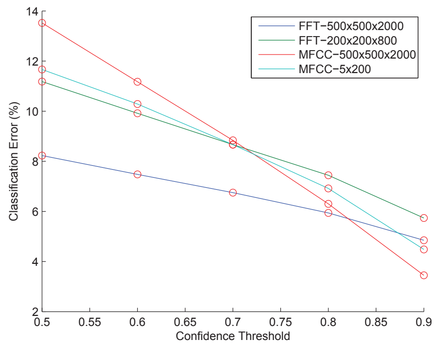
Figure 3 presents classification results on dataset D_{INT}, for selected network configurations trained on D_1. Due to space limitations, we have omitted the respective plot for D_2. However, it has to be noted that the performance based on D_2 follows closely Figure 3. Based on figures 2 and 3, we can draw the following conclusions:

(i) At the pattern level, the best performance on D_{INT} (8% error) was achieved by a 500x500x2000 network with

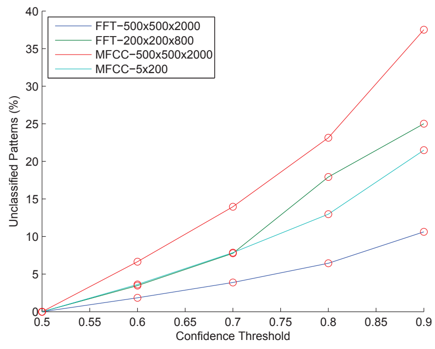
DFT-based features, when the confidence threshold was 0.5. If a strict threshold is applied, the lowest error is 3.5 and stems from a 500x500x2000 network operating on MFCCs. However, the percentage of unclassified patterns in this case is almost 40%. So, a more realistic network choice would be again the 500x500x2000 network with DFT-based features (10% of unclassified patterns).

(ii) The use of a confidence threshold decreases significantly the pattern classification error, especially in D_{INT}. However, as the threshold increases, the percentage of unclassified patterns increases more rapidly for certain architectures, as it is for example the case with the 500x500x2000 MFCC-based network. The DFT-based approaches appear to be less sensitive to the increasing threshold.

To compliment the results in the figures, we report that, at the audio file level (30 s level), the cross-fold error is zero on D_1 and almost zero on D_2, for a 500x500x2000 DFT-based network. More specifically, if the label which corresponds to the majority of decisions at each 30 s segment is selected as the label of the segment, we always get correct results in D_1 and at most 1 misclassified segment in D_2. These numbers are not affected by the value of the confidence threshold. In this line of thinking, the error rate at the 30 s level on D_{INT} is equal to 0.63%, i.e, only 13 (music) files were misclassified (out of 2066 in total).



(a)



(b)

Fig. 3. (a) Error on D_{INT} (b) Unclassified patterns.

5. CONCLUSIONS

In this paper, the problem of speech-music discrimination was studied from a deep learning perspective. We experimented with different feature extraction schemes and investigated how the network depth and RBM size affect the classification performance on publicly available datasets and on large amounts of audio data from video-sharing sites. It can be concluded that deep architectures provide the basis for building strong classifiers for the broad binary problem of speech vs music, with satisfactory generalization performance.

REFERENCES

[1] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *IEEE ICASSP*, 1997, vol. 2, pp. 1331–1334.

[2] C. Panagiotakis and G. Tziritas, "A speech/music discriminator based on rms and zero-crossings," *Multimedia, IEEE Trans. on*, vol. 7, no. 1, pp. 155–166, 2005.

[3] A. Pikrakis, T. Giannakopoulos, and S. Theodoridis, "A speech/music discriminator of radio recordings based on dynamic programming and bayesian networks," *Multimedia, IEEE Trans. on*, vol. 10(5), 2008.

[4] Jan Schlüter and Reinhard Sonnleitner, "Unsupervised feature learning for speech and music detection in radio broadcasts," in *Proceedings of DAFx*, 2012.

[5] Abdel-rahman Mohamed, George E Dahl, and Geoffrey Hinton, "Acoustic modeling using deep belief networks," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 14–22, 2012.

[6] L. R. Rabiner and R. W. Schafer, *Theory and Applications of Digital Speech Processing*, Prentice Hall, 2010.

[7] I. Antonopoulos, A. Pikrakis, and S. Theodoridis, "Self-similarity analysis applied on tempo induction from music recordings," *Journal of New Music Research*, vol. 36, no. 1, pp. 27–38, 2007.

[8] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18(7), pp. 1527–1554, 2006.

[9] Yoshua Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[10] G. Williams and D. PW Ellis, "Speech/music discrimination based on posterior probability features," in *Eurospeech 99, Budapest*. ESCA, 1999.

[11] G. Tzanetakis and P. Cook, "Marsyas: A framework for audio analysis," *Organised sound*, vol. 4(3), 2000.

[12] Y Bengio et al., "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, pp. 153, 2007.