UNSUPERVISED FEATURE LEARNING FOR SPEECH AND MUSIC DETECTION IN RADIO BROADCASTS

Jan Schlüter
Austrian Research Institute for
Artificial Intelligence, Vienna
jan.schlueter@ofai.at

Reinhard Sonnleitner
Dept. of Computational Perception,
Johannes Kepler University Linz
reinhard.sonnleitner@jku.at

ABSTRACT

Detecting speech and music is an elementary step in extracting information from radio broadcasts. Existing solutions either rely on general-purpose audio features, or build on features specifically engineered for the task. Interpreting spectrograms as images, we can apply unsupervised feature learning methods from computer vision instead. In this work, we show that features learned by a mean-covariance Restricted Boltzmann Machine partly resemble engineered features, but outperform three hand-crafted feature sets in speech and music detection on a large corpus of radio recordings. Our results demonstrate that unsupervised learning is a powerful alternative to knowledge engineering.

1. INTRODUCTION

Radio broadcasts are composed of two main types of audio content: Speech and music. Discriminating them is a basic first step in making their content accessible to further information retrieval. Often, music and speech do not just alternate, but overlap, and some applications require both classes of content to be detected independently. For example, in automatic broadcast transcription, only segments that contain speech should be passed to a speech recognition system. In this scenario, speech detection must be invariant to background music: Some radio stations constantly play music, even during the news, and a simple speech/music *discriminator* may give wrong predictions in this case. Another application is in collecting royalties from radio stations: In many countries, performance rights organizations charge royalties depending on the amount of music played, sometimes with a lower rate for music overlaid by speech. Automatic discrimination between pure music, music with speech, and non-music would facilitate a fair distribution of charges.

Existing approaches to speech and music detection either train standard classifiers on general-purpose audio features, or design new features based on observations on the structure of speech or music signals. A promising alternative is to learn features from data instead – in Computer Vision, learned features often outperform engineered features in object recognition tasks [1]. A particularly successful model for learning features from images, the mean-covariance Restricted Boltzmann Machine (mcRBM) [2], has already successfully been applied to spectrograms of speech [3] and music [4], and thus seems to be an ideal candidate for our task.

In this work, we build a speech and a music detector based on an mcRBM. Section 2 reviews existing work on speech and music detection, and Section 3 gives an introduction to mcRBMs and their application to audio data. In Section 4, we apply our system to a large corpus of radio broadcasts, analyze the features learned by the mcRBM and evaluate its classification performance in comparison to three approaches using hand-crafted features. Section 5 concludes with a discussion and outlook on future work.

2. RELATED WORK

Many methods for detection or discrimination of speech and music have been proposed in literature. We will discuss some prominent examples to illustrate their common strategies.

One class of approaches uses general-purpose audio features, hoping to capture distinct properties of speech or music. As speech is a sequence of vowels and consonants, it tends to exhibit quick changes in Zero Crossing Rate (ZCR), while music has a relatively constant ZCR. With a classifier trained on statistics capturing the ZCR variability, Saunders [5] achieves a speech/music discrimination accuracy of 90%, and adding an energy contour feature improves accuracy to 98%. Carey et al. [6] compare Mel-Frequency Cepstral Coefficients (MFCCs), amplitude, pitch and ZCR features classified with Gaussian Mixture Models (GMMs). They observe best performance in music/speech discrimination for MFCCs supplemented with their first-order derivative (i.e., delta). Pinquier et al. [7] train a speech detector on MFCCs augmented with delta-MFCCs, and a music detector on spectral frames. Both detectors use GMMs for classification, and achieve an accuracy of 99.5% for speech and 93% for music detection, respectively. Liu et al. [8] extract Linear Predictive Coefficients (LPCs), Line Spectral Pairs (LSPs), MFCCs, ZCR, and the spectral centroid, flux, rolloff, and kurtosis, augmented with the variance of each feature over a window of 1.28 s. Classification with a Multi-Layer Perceptron (MLP) yields an F-Score of 98% for music and speech detection.

Other authors design very smart features exploiting inherent characteristics of music or speech signals. A basic observation is that music contains long sustained tones of constant frequency. Hawley, [9, pp. 78–87], Minami et al. [10] and Seyerlehner et al. [11] all implemented algorithms for finding sustained frequency peaks in spectrograms, reporting up to 90% accuracy for music detection. Zhu et al. [12] add the observation that most music is tuned to equal temperament, and design a feature assessing whether spectral peaks are tuned to a common reference pitch which does not change over time. They report about 96% precision and recall in music detection, even when mixed with speech. Voiced speech, on the other hand, contains harmonics that quickly vary in frequency. Hawley uses a comb filter to detect such harmonics, and filters for varying pitches to distinguish speech from music [9, pp. 112–115]. Minami et al. [10] instead remove long sustained peaks from the spectrogram, and assume remaining harmonics to indicate speech, reporting 80% accuracy in speech detection. Scheirer and Slaney [13] observe that speech signals contain many short pauses, and compute the amount of low-energy frames per time unit as well as the 4 Hz modulation energy (matching the syllabic rate of speech). Adding a rhythm detector and several other hand-designed features, they report an accuracy of 94.5% in discriminating pure speech from pure music in radio broadcasts, virtually independent of the classifier used.

Few approaches employ unsupervised dimensionality reduction, interpretable as a form of feature learning: Mesgarani et al. [14] apply multilinear SVD to a high-dimensional biologically inspired audio representation. On a corpus of pure speech, music, environmental and animal sounds, they report a speech detection accuracy of 100% using an RBF-kernel SVM. Izumitani et al. [15] compress mel-spectral frames with PCA. They achieve an accuracy of 92% in discriminating pure speech from speech mixed with music using a GMM classifier.

To the best of our knowledge, none of the methods published in the literature perform more sophisticated feature learning, let alone using (mc)RBMs. On other tasks in the audio domain, however, these models have already proven useful: Dahl et al. [3] and Mohamed et al. [16] learn features for speech recognition from spectrograms and waveforms, respectively, and Lee et al. [17], Hamel/Eck [18] and Schlüter/Osendorfer [4] learn features for musical genre classification or music similarity estimation.

In conclusion, prior research focused on using existing audio features, or put much effort into designing new features by hand. While all authors report promising results, they all use different datasets, making it hard to valuate their methods – as an example, the music detector of Minami et al. only yielded about 56% accuracy on a larger corpus of Seyerlehner et al. [11]. Besides, some approaches only discriminate pure speech from pure music [5, 6, 13, 14], and are likely to fail for the kind of mixed signals we are interested in. In this work, we investigate whether unsupervisedly learned features are competitive to hand-crafted features for music and speech detection.

3. FEATURE LEARNING WITH MC-RBMS

We will now introduce Restricted Boltzmann Machines (RBMs) and the variant used in this work, the mean-covariance Restricted Boltzmann Machine (mcRBM), as well as explain how they can be applied to audio data for both unsupervised feature learning and supervised classification. Note that we can only give a compact description of the models here, for a more slow-paced introduction please see [19, pp. 6–16] or [20, pp. 61–77], for example.

3.1. Restricted Boltzmann Machines and Deep Belief Nets

An RBM [21] is an undirected graphical model consisting of visible units $\boldsymbol{v}$ representing observable data, and hidden units $\boldsymbol{h}$ giving a latent representation of the data. Visible and hidden units form two layers fully connected to each other, without within-layer connections (Figure 1).

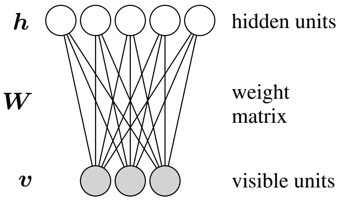


Figure 1: *A Restricted Boltzmann Machine (RBM)*

The RBM defines a joint probability distribution of visible and hidden states in terms of an energy function, such that configurations of low energy are more probable:

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

where $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{u},\boldsymbol{g}} e^{-E(\boldsymbol{u},\boldsymbol{g},\boldsymbol{\theta})}$ is the normalizing *partition function*.[1] The energy function defines the space of possible energy surfaces – hence, the type of RBM –, and the model parameters $\boldsymbol{\theta}$, which include the connection weights, shape the energy surface.

The most basic type of RBM restricts all unit states to be binary and uses the following energy function:

$$E_b(\boldsymbol{v},\boldsymbol{h},\boldsymbol{\theta}) = -\boldsymbol{v}^T\boldsymbol{W}\boldsymbol{h} - \boldsymbol{v}^T\boldsymbol{a} - \boldsymbol{h}^T\boldsymbol{b}, \tag{2}$$

where $\boldsymbol{\theta} = (\boldsymbol{W},\boldsymbol{a},\boldsymbol{b})$ are the connection weights, visible and hidden bias terms, respectively. Inserting (2) into (1), we can derive the probability of a unit taking its "on" state (the unit's *activation*) conditioned on the other units:[2]

$$P(h_k = 1|\boldsymbol{v},\boldsymbol{\theta}) = \sigma\Big(b_k + \sum_i v_i W_{ik}\Big) \tag{3}$$

$$P(v_k = 1|\boldsymbol{h},\boldsymbol{\theta}) = \sigma\Big(a_k + \sum_j W_{kj} h_j\Big), \tag{4}$$

where $\sigma(\cdot)$ denotes the logistic sigmoid function. As the activation of a hidden unit h_k only depends on the visible units, all hidden activations can be computed in parallel. Thus, determining the latent representation $\boldsymbol{h}$ of a data point $\boldsymbol{v}$ in this model is equivalent to passing it through a single-layer feedforward net with logistic units, then sampling binary states from the activations. Likewise, constructing a data point from a latent representation amounts to passing it back through the same network, again with a logistic activation function and binary sampling. The weight matrix $\boldsymbol{W}$ thus plays a double role: Its columns either act as feature detectors or as templates for generating data, each controlling or controlled by a single hidden unit.

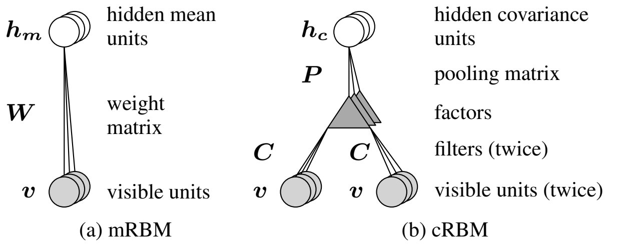
Training an RBM means adjusting $\boldsymbol{\theta}$ such that $p(\boldsymbol{v}|\boldsymbol{\theta})$, the marginal probability density of the visible units, approximates the observed distribution of a set of training data; this is equivalent to maximizing the likelihood of the model under the data. Gradient ascent on the log likelihood yields a simple learning rule for a single connection weight:

$$W_{ij} \leftarrow W_{ij} + \eta\Big(\big\langle v_i h_j\big\rangle_{\text{data}} - \big\langle v_i h_j\big\rangle_{\text{model}}\Big), \tag{5}$$

where η denotes the learning rate, $\langle v_i h_j\rangle_{\text{data}}$ is the correlation of a visible and hidden unit in the training data, and $\langle v_i h_j\rangle_{\text{model}}$ is the same correlation in data probable under the model. Considering Equation 2, each update lowers the energy for training data and raises the energy in low-energy regions. The first correlation can be easily computed by applying Equation 3 to each training data point. For the second correlation, we need to sample data points from the model. Directly sampling $p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{\theta})$ is intractable due to

[1] For reasonably sized models, the partition function is computationally intractable, as it requires enumerating all possible configurations. However, we will not need to compute it.

[2] For the derivation, see e.g. [19, p. 9] or [20, p. 143].



(a) mRBM
(b) cRBM

Figure 2: *Diagrams of the two parts of a mean-covariance RBM.*

the partition function, but we can apply *Gibbs sampling*: Starting from a random configuration, sampling $\boldsymbol{h}$ and $\boldsymbol{v}$ in turns (Equations 3 and 4) runs a Markov chain which, when converged, produces samples from the model. To make learning more efficient, Hinton [21] proposed to start the chain at actual training samples and run it for a small number of k steps only (*Contrastive Divergence* learning, or CD-k).

After training, an RBM will produce samples resembling the training data when starting from a randomized state and performing Gibbs sampling for long enough. For this to work, the weights $\boldsymbol{W}$ must have learned useful templates to generate (parts of) data points – i.e., typical features found in the data. Thus, an RBM's hidden representation of a data point is an abstract description in terms of typical features, which makes them attractive for unsupervised feature extraction.

To obtain even more abstract representations, we can train an RBM on the latent representations of another RBM, learning features of features that capture higher-order correlations in the data. Recursively applying this principle, this creates a stack of RBMs termed a Deep Belief Net (DBN) [22].

3.2. The mean-covariance Restricted Boltzmann Machine

As the type of RBM discussed above has binary visible units, its generative model $p(\boldsymbol{v}|\boldsymbol{\theta})$ cannot approximate non-binary data, and thus will not learn useful features for non-binary inputs.

With a slight change of the energy function, though, we obtain an RBM capable of modelling real-valued data, termed *mRBM*:

$$E_m(\boldsymbol{v},\boldsymbol{h},\boldsymbol{\theta}) = -\boldsymbol{v}^T\boldsymbol{W}\boldsymbol{h} + \frac{1}{2}(\boldsymbol{v}-\boldsymbol{a})^T(\boldsymbol{v}-\boldsymbol{a}) - \boldsymbol{b}^T\boldsymbol{h} \quad (6)$$

Its latent representations still follow Equation 3, but the distribution of visible states conditioned on the hidden states becomes:[3]

$$p(\boldsymbol{v}|\boldsymbol{h_m},\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{v}|\boldsymbol{a}+\boldsymbol{W}\boldsymbol{h_m},\boldsymbol{I}), \quad (7)$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma})$ denotes the multivariate Gaussian probability density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Constructing a data point from a latent representation is thus equivalent to passing it through a one-layer network with *linear* output units, then adding isotropic Gaussian noise of unit standard deviation.

However, independent Gaussian noise does not yield a good generative model for most real-world data. To take into account pairwise dependencies of input variables, a third-order RBM can be defined, with weights $W_{i,j,k}$ connecting hidden units h_k to *pairs* of visible units v_i, v_j. By factorizing and tying these weights [23, 24], parameters can be reduced to a filter matrix $\boldsymbol{C}$ connecting the input twice to a set of *factors* and a pooling matrix $\boldsymbol{P}$ mapping factors to hidden units (Figure 2b). The energy function is

$$E_c(\boldsymbol{v},\boldsymbol{h},\boldsymbol{\theta}) = -(\boldsymbol{v}^T\boldsymbol{C})^2\boldsymbol{P}\boldsymbol{h} - \boldsymbol{c}^T\boldsymbol{h}, \quad (8)$$

yielding

$$p(h_k = 1|\boldsymbol{v},\boldsymbol{\theta}) = \sigma\Big(\boldsymbol{c} + \big((\boldsymbol{v}^T\boldsymbol{C})^2\boldsymbol{P}\big)^T\Big) \quad (9)$$

$$p(\boldsymbol{v}|\boldsymbol{h_c},\boldsymbol{\theta}) = \mathcal{N}(0,(\boldsymbol{C}\mathrm{diag}(\boldsymbol{P}\boldsymbol{h}^c)\boldsymbol{C}^T)^{-1}). \quad (10)$$

Computing the latent representation now corresponds to passing the data through a *two-layer* feedforward net with a squared activation function in the hidden layer, and usual sigmoid output units. Constructing a data point can no longer be interpreted as a feedforward net: Instead, the data point is sampled from a Gaussian with a covariance matrix depending on the hidden unit states. However, this Gaussian is restricted to have zero mean.

The mean-covariance Restricted Boltzmann Machine [2] combines the former two models by adding their energy functions:

$$E_{mc}(\boldsymbol{v},\boldsymbol{h_m},\boldsymbol{h_c},\boldsymbol{\theta}) = E_m(\boldsymbol{v},\boldsymbol{h_m},\boldsymbol{\theta_m}) + E_c(\boldsymbol{v},\boldsymbol{h_c},\boldsymbol{\theta_c}) \quad (11)$$

$p(\boldsymbol{v}|\boldsymbol{h_m},\boldsymbol{h_c},\boldsymbol{\theta})$ becomes the product of the two original Gaussian distributions, resulting in a powerful generative model of two types of hidden units (Figure 2). It explains each datapoint as a linear combination of templates in $\boldsymbol{W}$, selectively smoothened with filters in $\boldsymbol{C}$. mcRBMs can still be trained with Contrastive Divergence, using a different sampling method to avoid the matrix inversion of Equation 10. To make learning more robust, input data and filters in $\boldsymbol{C}$ are normalized to unit ℓ_2 norm when computing the hidden covariance unit states, and the pooling matrix $\boldsymbol{P}$ is constrained to a topographic mapping (associating a small group of neighboring factors to each hidden covariance unit), kept nonpositive and normalized to unit ℓ_1 norm. For more details on the model and its training procedure, we refer the reader to [2, 24].

3.3. Discriminative Fine-Tuning

Using RBMs for feature extraction means computing their latent representation for given input data, usually omitting the binary sampling step. As explained in Sections 3.1 and 3.2, this can always be interpreted as passing the input data through a feedforward network, or Multi-Layer Perceptron (MLP), with either sigmoid or squared transfer functions. Thus, instead of merely training a classifier on the RBM's representations, we can add another layer on top and train the full network for classification with backpropagation, gently tuning the existing feature detectors to the task at hands. Unsupervised learning can then be seen as a pretraining step finding a good initialization for backpropagation [25] – especially for the lower layers, which are most strongly affected by the diminishing gradient effect (i.e., the effect that the error signal of the output layer gets weaker and weaker when backpropagated through the network).

3.4. Application to Audio Data

Originally, mcRBMs have been designed to model patches of natural images. Here, we follow [3] and [4] and apply it to excerpts of log-frequency spectrograms with standard preprocessing, allowing it to model local time-frequency structure in sounds.

Specifically, we convert our input data to 22.05 kHz monaural signals and perform a Short-Time Fourier Transform (STFT) with

[3] For the derivation, see e.g. [19, p. 13].

a 1024-sample Hanning window and a step size of 512 samples. The magnitude spectrum of each frame is passed through a bank of 70 triangular filters equally spaced on the Mel scale, covering the range from 50 Hz to 6854 Hz, and a log function is applied (resulting in log-frequency, log-magnitude spectral frames). Consecutive frames are joined to form overlapping blocks of 39 frames each, with a step size of 1 frame (i.e., 38 frames overlap). Each block covers about 0.93 s of audio. Blocks are decorrelated with PCA, compressed to 99% of their original variance by omitting the least significant principal components (in our case, from 2730 to 1606 dimensions), and whitened by dividing each component by its standard deviation. The resulting vectors form the input data for the mcRBM, mapping each component to a visible unit.

4. EXPERIMENTAL RESULTS

We performed a range of experiments to evaluate our approach. In this section, we will describe the dataset used in our experiments, the different variants of our method as well as approaches by other authors we evaluated, and then perform a qualitative analysis of the features learned as well as a quantitative analysis in terms of classification performance.

4.1. Dataset

Our dataset consists of 42 hours of radio broadcasts finely segmented (with a resolution of 200 ms) into speech/nonspeech and music/nonmusic sections by paid students. 30 hours have been recorded from web streams of 6 Swiss radio stations in segments of 30 minutes randomly distributed over the course of a week, to capture as many different shows as possible. The chosen radio stations (DRS Virus, RSI Rete 2, RSR Couleur 3, Radio Central, Radio Chablais, RTR Rumantsch) range from Indie rock to Classical music and cover the four official languages of Switzerland: Swiss German, French, Italian and Rumantsch. The remaining 12 hours have been captured from lower-bitrate web streams of 4 Austrian radio stations (Ö1, Ö3, FM 4, Life Radio) as continuous 3-hour recordings, again covering different music styles and two languages: Austrian German and English.

15 hours of the Swiss recordings were used for training, another 6 hours for validation (and tuning hyperparameters) and the remaining 9 hours for testing. The Austrian recordings served as an additional test set to evaluate robustness to different recording conditions and generalization to unseen radio stations.

4.2. Evaluated Methods

We will now detail the architecture and training procedure of our network, describe reduced variants for control experiments and introduce three approaches by other authors evaluated on our corpus.

4.2.1. This work

Our full system consists of an mcRBM of 256 mean units and 1296 factors mapped to 324 covariance units (the smaller of the two architectures in [4]), with two binary RBMs of 512 and 256 hidden units stacked on top. The mcRBM was trained unsupervisedly on spectral blocks extracted from the training set as detailed in Section 3.4. We trained it for 50 epochs on 453,120 training samples split into mini-batches of 128 data points with a learning rate of 0.02, ℓ_1 weight decay of 0.001 and pooling matrix P constrained to a 2D topographic mapping. Subsequently, the RBM of 512 hidden units was trained on the mcRBM's latent representations for 100 epochs with a learning rate of 0.01, ℓ_1 weight decay of 0.0001 and momentum 0.9 (reduced to 0.45 for the first 20 epochs, following [26]), linearly switching from CD-1 to CD-15 during training to counter the decreasing mixing rate in Gibbs sampling. The second RBM was trained on the first RBM's representations using the same settings.

Speech and music detection were treated as two separate classification problems handled by two separately fine-tuned instances of the network. For fine-tuning, we added a single sigmoid output unit and trained the resulting network on the full training set of 2,321,280 spectral blocks each paired with the binary label at its center. Training was performed by backpropagation with cross-entropy error, a learning rate of 0.01, and momentum 0.9 (reduced to 0.45 for the first 10 epochs, then raised in steps during the next 10 epochs). Each network was trained for 100 epochs, monitoring the classification error at threshold 0.5 on the validation set. The epoch of lowest validation classification error was selected for evaluation on the test sets.

As the networks' block-wise predictions tend to be noisy, we apply a sliding median filter[4] to the sequence of network outputs on a file before thresholding the values to obtain binary decisions. By optimization on the validation set, we set the filter length to 250 frames (5.8 s) for music and 100 frames (2.3 s) for speech.

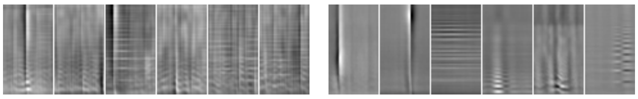
To understand by how much each component of our system influences the final result, we created four reduced variants in addition to the full system:

1. *MLP on mel:* To see how a classifier performs directly on the low-level audio representation, we train a Multi-Layer Perceptron (MLP) of 512 and 256 hidden units on the whitened mel-spectral blocks.
2. *P on mcRBM:* In a second step, we train a single Perceptron on the mcRBM output, to assess how useful the unsupervisedly learned features are to a linear classifier.
3. *MLP on mcRBM:* We repeat the same with an MLP of 512 and 256 hidden units.
4. *DBN on mcRBM:* We stack the 512-unit and 256-unit RBMs on top of the mcRBM and fine-tune them, still leaving the mcRBM unchanged as an unsupervised feature extractor (as in [3]). This tests whether pretrained RBMs outperform an MLP of randomly initialized weights (variant 3).
5. *DBN incl. mcRBM:* The final system includes the mcRBM in supervised fine-tuning with backpropagation.

4.2.2. MFCCs

To give our results some more context, we compare them to three hand-crafted feature sets proposed in literature. As a simple baseline, we extract 40 MFCCs, their first order derivative (delta) and second order derivative (acceleration) using yaafe [27] – these features have shown good results in [6, 7]. We normalize features by subtracting the mean and dividing each dimension by its standard deviation (both determined on the training set), then train two MLPs of 512 units in the first hidden layer and 256 units in the second hidden layer for speech and music detection, respectively. We use the same training parameters as in our own system, and post-process the predictions with the same sliding median filters.

[4] Compared to a sliding average, median filtering has the advantage of not blurring clearly localized decision boundaries.



(a) mean units (b) factors

Figure 3: *Exemplary features learned by the mcRBM. Each block represents 929 ms of a spectrogram: Time increases from left to right, mel-frequency from bottom to top, bright and dark indicate positive and negative values, respectively.*

4.2.3. Liu et al.

Liu et al. [8] extract a set of 8 standard audio features along with their variances over a short window, resulting in a 94-dimensional feature vector per audio frame. With a small MLP of 10 hidden units, they report near-perfect results of 98% F-Score for music and speech detection. Here, we extract the same set of features using yaafe [27]. To rule out any influence of the classifier, we then process the feature vectors just like the MFCCs above.[5]

4.2.4. Seyerlehner et al.

Seyerlehner et al. [11] engineered a feature for robust music detection in the presence of speech or noise, and demonstrated its performance on a corpus of TV recordings. It is based on the detection of horizontal structures in the spectrogram, i.e., sustained tones typical for music, and outputs a single value per timestep. We extract 5 such values per second, and apply a sliding median filter of 5.8 s as in our system. By design, this feature is only useful for music detection, but it is especially interesting for the qualitative analysis of our own features in the next subsection.

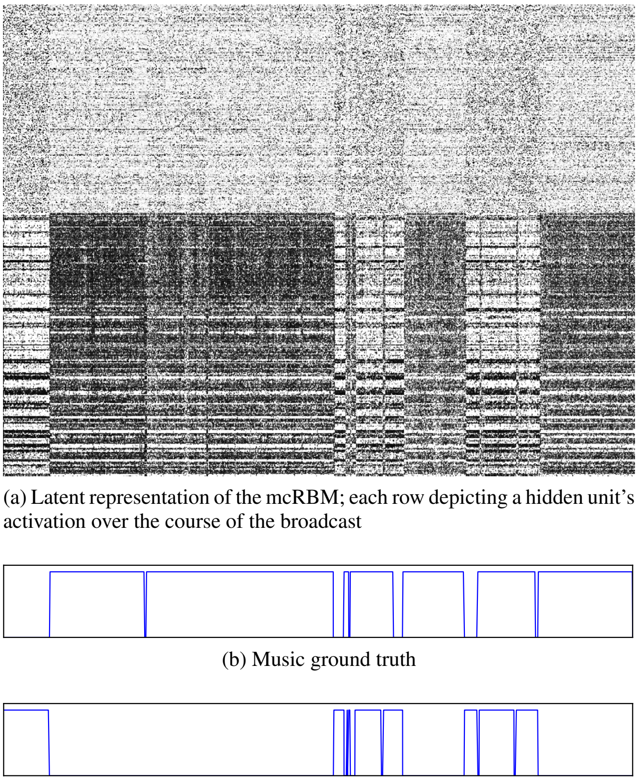
4.3. Learned Features

In Figure 3, we visualize a random selection of filters learned by the mcRBM *before discriminative fine-tuning*. Each block shows the unwhitened incoming weights of a mean unit (Figure 3a) or factor (Figure 3b), interpretable as a one-second spectrogram excerpt. Most filters exhibit distinct horizontal or vertical patterns, some even display structures faintly resembling formants in speech.

To better understand the filters, we show the activations of all hidden units over the course of a 30-minute recording (Figure 4a). Remarkably, the mcRBM's latent representations show quite clearly which sections are dominated by music and which are dominated by speech (cf. Figures 4b, 4c). Looking closely at the activations of the hidden covariance units (bottom part of Figure 4a), we can see units that are active during speech and inactive during music, and other units that behave the opposite.

In Figure 5, we zoom into a 10-seconds excerpt of the file. Figure 5a shows the mel-scaled spectrogram used as input for the mcRBM. It starts with a few seconds of pure speech and continues with music, clearly visible in the spectrogram. For the last few seconds, the moderator speaks again, with faint background music (barely visible). Figure 5b plots the corresponding activations of two covariance units that displayed roughly opposite behaviour on the whole file, and Figure 5c shows the respective factors connected to these units. Mind that the factors connect to the units

[5] Our MLP is considerably larger than in [8], but smaller networks performed similar or worse, consistent with findings in [28].

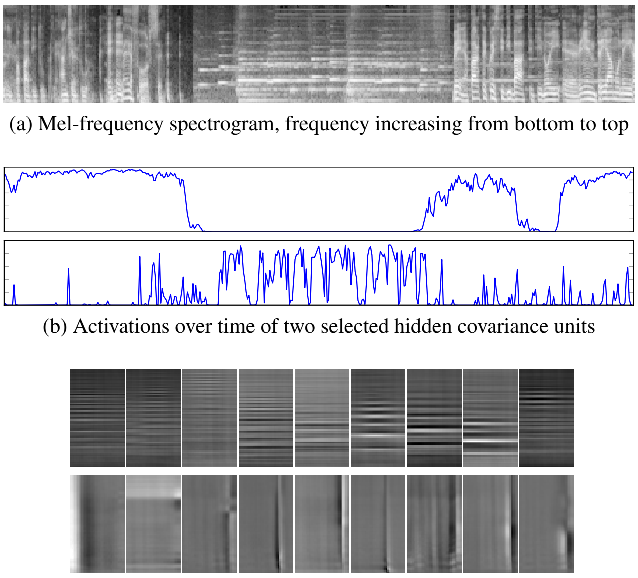


(a) Latent representation of the mcRBM; each row depicting a hidden unit's activation over the course of the broadcast

(b) Music ground truth

(c) Speech ground truth

Figure 4: *Unsupervisedly learned representation and ground truth for a 30-minute radio broadcast. Time increases from left to right.*

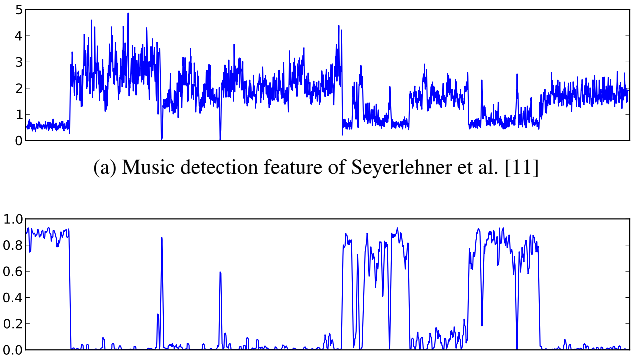


(a) Mel-frequency spectrogram, frequency increasing from bottom to top

(b) Activations over time of two selected hidden covariance units

(c) Filters negatively connected to the two covariance units

Figure 5: *Spectrogram and activations of two covariance units for a 10-second excerpt of pure speech, pure music, and speech with faint background music. The top unit is inactive at sustained notes, the bottom unit is inactive at sudden loudness changes.*



(a) Music detection feature of Seyerlehner et al. [11]

(b) Activations over time of a selected hidden covariance unit

Figure 6: *Comparison of an engineered music detector and a hidden unit's activation for the recording of Figure 4. The two curves act approximately inversely to each other.*

with *negative* weights, so a filter causes its covariance unit to become *inactive* if it detects anything.[6] Considering this, we can see that the first unit's filters detect sustained tones present in music (missing most background music, though), and the second unit's filters react on quick changes of loudness as occurring in speech.

As we learned in Section 2, similar characteristics have also been addressed by engineered features for music [9, 10, 11] and speech detection [13]. Exemplarily, we plot the feature of Seyerlehner et al. [11] in Figure 6a. By our reasoning, as it detects horizontal structure, it should respond contrary to the first covariance unit. Figure 6b shows that this is roughly the case, so the mcRBM seems to have re-invented their feature.

Note that these features were obtained with purely unsupervised learning – at this stage, the mcRBM has not seen any class labels, nor was it trained to find a binary segmentation. Merely the structure of the data drove it to develop two different sets of features for generating speech and music.

4.4. Classification Results

To assess how useful the features are for the task of speech and music detection, we evaluate their classification performance on our two test sets. Specifically, we compute the frame-wise real-valued predictions of each system, smoothen them as detailed in Section 4.2 and then apply a binary thresholding to obtain a label for each frame. Comparing the labels to ground truth provided by our annotators, we count the number of true positives tp, false positives fp, true negatives tn and false negatives fn. From these statistics, we compute four standard evaluation metrics: Accuracy $(tp + fn)/(tp + fp + tn + fn)$, precision $tp/(tp + fp)$, recall $tp/(tp+fn)$ and F-score: $2 \cdot \text{precision} \cdot \text{recall}/(\text{precision}+\text{recall})$.

For each method, we list results using a neutral threshold of 0.5, and a higher threshold of 0.7 which trades recall for higher precision. Predictions of Seyerlehner et al. are not limited between zero and one, so here we report results for the best thresholds (in terms of accuracy and F-score) on the validation set (0.85) and test sets (0.7) instead.[7]

Table 1 shows results for speech detection. Focusing on the Swiss test set, we can see that a linear classifier on features learned by an mcRBM (*P on mcRBM*) already performs quite well, but is inferior to what a discriminatively trained MLP learns from the same low-level spectrogram representation (*MLP on mel*). However, nonlinear classifiers on mcRBM features outperform both MFCCs and the feature set of Liu et al., and fine-tuning the mcRBM (*DBN incl. mcRBM*) brings an additional boost in performance. Interestingly, pre-training RBMs is not better than random initialization (*DBN on mcRBM* vs. *MLP on mcRBM*). On the Austrian test set, all classifiers perform worse, indicating that either they are slightly overfitted to Swiss radio stations, or they suffer from the reduced recording quality. This is especially true for the fully fine-tuned network: On the Austrian broadcasts, it is even inferior to an untuned mcRBM (*DBN incl. mcRBM* vs. *MLP/DBN on mcRBM*).

For music detection, results look a little different (Table 2). On the Swiss test set, the *DBN* and *MLP on mcRBM* are outperformed by hand-crafted features, but the fully fine-tuned network *DBN incl. mcRBM* performs best by a large margin (2.7% misclassifications compared to 3.4% for the second best model). Curiously, the simple *MLP on mel* beats more complex feature sets – possibly because it sees a longer context. Again, all classifiers perform worse on the Austrian test set, including Seyerlehner et al. which was never trained on Swiss broadcasts, suggesting that the Austrian set is generally more difficult to handle. The mcRBM-based methods now outperform all others, with the fully fine-tuned *DBN incl. mcRBM* still performing best – overfitting to Swiss stations seems to be less of a problem for music.

In Figure 7, we additionally plot the precision/recall curves for Liu et al., Seyerlehner et al. (if applicable), our *MLP on mcRBM* and *DBN incl. mcRBM* on the Austrian test set. The curves show that our conclusions from the table are valid for a range of reasonable thresholds.

5. DISCUSSION

Similar to how another feature learning approach unsupervisedly found that images of digits come in 10 different shapes [29], the mcRBM discovered that radio stations produce two different kinds of content that are to be modeled separately (the same lateralization of speech and music processing has also been observed in the human brain [30]). Exploiting this, we were able to build a highly accurate speech and music detector outperforming hand-crafted features on a large corpus of recorded radio broadcasts.

Fine-tuning the mcRBM on labeled data noticeably improved results on a test set of similar inputs, but worsened speech detection accuracy on broadcasts of lower quality recorded in a different country. It may be worthwhile to investigate ways to regularize fine-tuning of mcRBMs, to reduce overfitting.

Of course, our classifier is still not perfect – for example, it may miss background music if it is too faint. Different low-level representations, different preprocessing or a generative model directly targeted to audio data could improve results.

For real-world applications, it would be interesting to evaluate if smaller models yield similar results at lower computational costs. Although our classifier works at about 50 x real-time on a consumer graphics card, the architecture used in our experiments might be oversized for a binary classification problem.

[6] Conversely, when reconstructing a data point from its latent representation, an inactive covariance unit causes its factors to *not* smoothen the data, leaving intact the structure the filters detect. See also [24, Sec. 5].

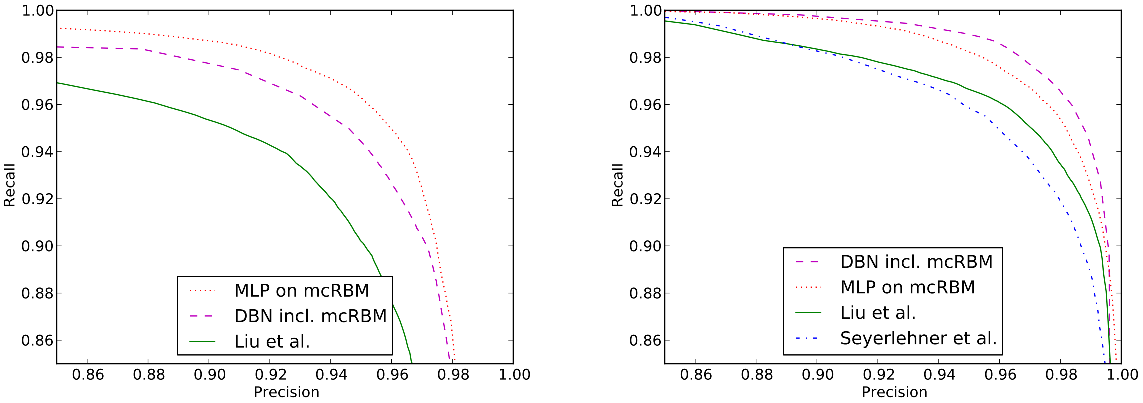
[7] Optimizing the threshold on the test sets is unfair, but shows that, on our corpus, the feature of Seyerlehner et al. is inferior to multi-feature approaches even with this radical measure.

Table 1: *Speech detection performance of all methods on both test sets. For each method, we report the accuracy, precision, recall and F-score in percent at binarization thresholds of 0.5 and 0.7. The best accuracy and F-score per column are marked in bold.*

		Swiss test set				Austrian test set			
Method	threshold	acc.	prec.	rec.	f-sc.	acc.	prec.	rec.	f-sc.
P on mcRBM	0.5	96.3	89.0	95.8	92.3	95.8	93.8	93.7	93.8
	0.7	97.0	94.0	93.0	93.5	95.1	96.0	89.4	92.6
MLP on mel	0.5	97.3	92.8	95.7	94.2	94.5	95.5	88.0	91.6
	0.7	97.6	95.0	94.4	94.7	93.8	96.3	85.1	90.4
MFCCs	0.5	97.3	94.2	94.1	94.1	94.6	95.3	88.5	91.8
	0.7	97.6	97.6	91.6	94.5	93.4	96.9	83.2	89.5
Liu et al.	0.5	97.4	93.5	95.4	94.4	95.2	94.7	90.9	92.7
	0.7	97.5	95.5	93.7	94.6	94.6	95.9	87.8	91.7
DBN on mcRBM	0.5	97.4	92.5	96.7	94.6	96.6	95.4	94.4	94.9
	0.7	97.8	95.4	95.1	95.3	96.0	96.7	91.3	93.9
MLP on mcRBM	0.5	97.9	93.9	97.1	95.4	**97.0**	95.9	95.1	**95.5**
	0.7	98.1	95.9	95.7	95.8	96.6	96.8	93.1	94.9
DBN incl. mcRBM	0.5	98.3	96.0	96.8	**96.4**	95.9	96.7	91.2	93.9
	0.7	**98.4**	96.4	96.5	**96.4**	95.8	97.0	90.3	93.6

Table 2: *Music detection performance of all method on both test sets.*

		Swiss test set				Austrian test set			
Method	threshold	acc.	prec.	rec.	f-sc.	acc.	prec.	rec.	f-sc.
P on mcRBM	0.5	94.4	98.2	95.2	96.7	94.1	98.6	94.2	96.4
	0.7	94.1	99.1	93.9	96.4	91.6	99.2	90.5	94.7
MFCCs	0.5	95.6	98.5	96.2	97.4	92.1	96.8	93.4	95.1
	0.7	94.4	99.0	94.3	96.6	92.0	98.0	92.1	95.0
DBN on mcRBM	0.5	95.7	98.6	96.2	97.4	94.7	97.6	95.8	96.7
	0.7	94.8	99.0	94.8	96.8	93.9	98.5	93.9	96.2
MLP on mcRBM	0.5	95.9	98.7	96.4	97.5	94.7	97.8	95.7	96.7
	0.7	95.1	99.1	95.1	97.0	93.9	98.6	93.9	96.2
Seyerlehner et al.	0.7	96.1	97.9	97.4	97.7	92.4	94.9	95.9	95.4
	0.85	93.8	99.0	93.6	96.2	89.0	99.2	87.2	92.9
Liu et al.	0.5	96.1	98.1	97.3	97.7	93.5	96.8	95.2	96.0
	0.7	95.6	98.7	96.1	97.4	93.1	97.9	93.6	95.7
MLP on mel	0.5	96.6	98.7	97.3	98.0	94.2	95.9	97.0	96.5
	0.7	96.0	98.9	96.3	97.6	93.9	96.9	95.7	96.3
DBN incl. mcRBM	0.5	**97.3**	98.7	98.1	**98.4**	**95.6**	97.0	97.7	**97.3**
	0.7	**97.3**	98.8	98.0	**98.4**	**95.6**	97.3	97.4	**97.3**



(a) Speech detection

(b) Music detection

Figure 7: *Precision/recall curves on the Austrian test set*

6. ACKNOWLEDGEMENTS

This work was supported by the Austrian FWF (proj. Z159, L511).

7. REFERENCES

[1] H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. of the 26th Int. Conf. on Machine Learning (ICML 2009)*, 2009.

[2] M. Ranzato and G.E. Hinton, "Modeling pixel means and covariances using factorized third-order boltzmann machines," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'10)*, 2010.

[3] G.E. Dahl, M. Ranzato, A. Mohamed, and G.E. Hinton, "Phone recognition with the mean-covariance restricted Boltzmann machine," in *Advances in Neural Information Processing Systems 23*, pp. 469–477. 2010.

[4] J. Schlüter and C. Osendorfer, "Music similarity estimation with the mean-covariance restricted boltzmann machine," in *Proc. of the 10th Int. Conf. on Machine Learning and Applications (ICMLA 2011)*, Honolulu, USA, 2011.

[5] J. Saunders, "Real-time discrimination of broadcast speech/music," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1996)*, 1996.

[6] M.J. Carey, E.S. Parris, and H. Lloyd-Thomas, "A comparison of features for speech, music discrimination," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1999)*, Phoenix, AZ, USA, 1999.

[7] J. Pinquier, C. Sénac, and R. André-Obrecht, "Speech and music classification in audio documents," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2002)*, 2002.

[8] C. Liu, L. Xie, and H. Meng, "Classification of music and speech in mandarin news broadcasts," in *Proc. of the 9th Nat. Conf. on Man-Machine Speech Communication (NCMMSC)*, Huangshan, Anhui, China, 2007.

[9] M.J. Hawley, *Structure out of sound*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, 1993.

[10] K. Minami, A. Akutsu, H. Hamada, and Y. Tonomura, "Video handling with music and speech detection," *IEEE Multimedia*, vol. 5, no. 3, pp. 17–25, 1998.

[11] K. Seyerlehner, T. Pohle, M. Schedl, and G. Widmer, "Automatic music detection in television productions," in *Proc. of the 10th Int. Conf. on Digital Audio Effects (DAFx-07)*, Bordeaux, France, 2007.

[12] Y. Zhu, Q. Sun, and S. Rahardja, "Detecting musical sounds in broadcast audio based on pitch tuning analysis," in *Proc. of the IEEE Int. Conf. on Multimedia and Expo (ICME 2006)*, 2006.

[13] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1997)*, Munich, Germany, 1997.

[14] N. Mesgarani, M. Slaney, and S. Shamma, "Discrimination of speech from nonspeech based on multiscale spectrotemporal modulations," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 3, pp. 920–930, 2006.

[15] T. Izumitani, R. Mukai, and K. Kashino, "A background music detection method based on robust feature extraction," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2008)*, 2008.

[16] A. Mohamed, G.E. Dahl, and G.E. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[17] H. Lee, Y. Largman, P. Pham, and A.Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems 22*, pp. 1096–1104. 2009.

[18] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *Proc. of the 11th Int. Soc. for Music Information Retrieval Conf. (ISMIR 2010)*, 2010.

[19] Alex Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. of Comp. Science, University of Toronto, 2009.

[20] Jan Schlüter, "Unsupervised audio feature extraction for music similarity estimation," M.S. thesis, Technische Universität München, Munich, Germany, 2011.

[21] Geoffrey E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[22] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[23] R. Memisevic and G.E. Hinton, "Learning to represent spatial transformations with factored higher-order boltzmann machines," *Neural Computation*, vol. 22, no. 6, pp. 1473–1492, 2010.

[24] M. Ranzato, A. Krizhevsky, and G.E. Hinton, "Factored 3-way restricted boltzmann machines for modeling natural images," in *Proc. of the 13th Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[25] D. Erhan, Y. Bengio, A.C. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?," *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[26] Geoffrey Hinton, "A practical guide to training restricted boltzmann machines," Tech. Rep. UTML TR 2010-003, Dept. of Comp. Science, University of Toronto, 2010.

[27] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "Yaafe, an easy to use and efficient audio feature extraction software," in *Proc. of the 11th Int. Soc. of Music Information Retrieval Conf. (ISMIR 2010)*, 2010.

[28] L. Giles R. Caruana, S. Lawrence, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Advances in Neural Information Processing Systems 13*. 2000.

[29] M. Ranzato, Y. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *Advances in Neural Information Processing Systems 20*, pp. 1185–1192. 2007.

[30] M. Tervaniemi and K. Hugdahl, "Lateralization of auditory-cortex functions," *Brain Research Reviews*, vol. 43, no. 3, pp. 231–246, 2003.