LEARNING FEATURES FROM MUSIC AUDIO WITH DEEP BELIEF NETWORKS

Philippe Hamel and Douglas Eck
DIRO, Université de Montréal
CIRMMT
{hamelphi,eckdoug}@iro.umontreal.ca

ABSTRACT

Feature extraction is a crucial part of many MIR tasks. In this work, we present a system that can automatically extract relevant features from audio for a given task. The feature extraction system consists of a Deep Belief Network (DBN) on Discrete Fourier Transforms (DFTs) of the audio. We then use the activations of the trained network as inputs for a non-linear Support Vector Machine (SVM) classifier. In particular, we learned the features to solve the task of genre recognition. The learned features perform significantly better than MFCCs. Moreover, we obtain a classification accuracy of 84.3% on the Tzanetakis dataset, which compares favorably against state-of-the-art genre classifiers using frame-based features. We also applied these same features to the task of auto-tagging. The autotaggers trained with our features performed better than those that were trained with timbral and temporal features.

1. INTRODUCTION

Many music information retrieval (MIR) tasks depend on the extraction of low-level acoustic features. These features are usually constructed using task-dependent signal processing techniques. There exist many potentially-useful features for working with music: spectral, timbral, temporal, harmonic, etc (see [21] and [3] for good reviews), and it is not always obvious which features will be relevant for a given MIR task. It would be useful to have a system that can automatically extract relevant features from the audio, without having to depend on *ad-hoc* domain-dependent signal processing strategies.

Among the most widely used frame-level features for audio-related MIR tasks Mel-Frequency Cepstral Coefficients (MFCCs). MFCCs take advantage of source/filter deconvolution from the cepstral transform and perceptually-realistic compression of spectra from the Mel pitch scale. Because the first few MFCC values capture pitch-invariant timbral characteristics of the audio, they are commonly used in tasks where it is useful to generalize across pitch, such as multi-speaker speech recognition and musical timbre recognition.

Practically all audio-based music genre classification models use different types of acoustic features to drive supervised machine learning [4, 13, 14, 23]. These include sparse audio encodings in the time domain [17] and in the frequency (spectral) domain [8]. Other approaches use a Hidden Markov Model (HMM) to build a semantic representation of music [7, 22]. The best reported accuracy on the Tzanetakis dataset [23] for genre classification was achieved by a system that used auditory cortical representations of music recordings and sparse representation-based classifiers [20]. The challenges and motivations of genre classification are discussed in [18]. In these approaches it is difficult to know whether the acoustic features or the machine learning techniques are responsible for success. To address this we apply our model to the Tzanetakis dataset.

A closely related task to genre classification is that of "autotagging" (automatic tag-based annotation of music audio). As for genre classification, timbral and temporal features are often used to solve this task [5]. To test the robustness of our learned features, we applied them to the task of autotagging on the Majorminer dataset [16].

Some work in automatic feature extraction for genre classification have been done. In [19], automatic feature selection was done with genetic algorithms, and used for one-on-one genre classification. In our approach, we use a Deep Belief Network (DBN) [10] to learn a feature representation. DBNs have already been applied in some MIR tasks. In [9], a DBN is compared to other classifiers for the instrument recognition task. In [12], convolutional DBNs are used to learn features for speech recognition and for genre and artist classification.

Can we learn features for a given task directly from musical audio that would better represent the audio than engineered signal-processing features? In this work, we investigate this question.

We propose a method to automatically extract a relevant set of features from musical audio. We will show that these learned features compare favorably against MFCCs and other features extracted by signal-processing.

The paper is divided as follows. In Section 2, we describe the datasets that were used in our experiments. We then explain briefly the DBN model in Section 3. In Section 4 we describe the feature learning process. Then, in Section 5 we give the results of our features used in genre classification and autotagging tasks. Finally, we conclude and propose future work in Section 6.

2. DATASETS

We used two different datasets in our experiments. The first one is the Tzanetakis' dataset for genre recognition. We trained our feature extractor over this dataset. To test the robustness of our learned features, we then applied these same features to the task of autotagging on the Majorminer dataset.

2.1 Tzanetakis

This dataset consists of 1000 30-second audio clips assigned to one of 10 musical genres. The dataset is balanced to have 100 clips for each genre. The dataset was introduced in [24], and have since been used as a reference for the genre recognition task.

2.2 Majorminer

This dataset for autotagging was introduced in [16]. The tags were collected by using a web-based "game with a purpose". Over 300 tags have been assigned to more than 2500 10 second audio clips. For our experiment, we used only the 25 most popular tags and compared our results to those obtained in [16].

3. DEEP BELIEF NETWORKS

In the last few years, a large amount of research has been conducted around deep learning [1]. The goal of deep learning is to learn more abstract representations of the input data in a layer-wise fashion using unsupervised learning. These learned representations can be used as input for supervised learning in tasks such as classification and regression. Standard neural networks were intended to learn such deep representations. However, deep neural networks (i.e. networks having many hidden layers) are difficult or impossible to train using gradient descent [2]. The DBN circumvents this problem by performing a greedy layer-wise unsupervised pre-training phase. It has been shown [2, 10] that this unsupervised pre-training builds a representation from which it is possible to do successful supervised learning by "fine-tuning" the resulting weights using gradient descent learning. In other words, the unsupervised stage sets the weights of the network to be closer to a good solution than random initialization, thus avoiding local minima when using supervised gradient descent.

The Deep Belief Network (DBN) is a neural network constructed from many layers of Restricted Boltzmann Machines (RBMs) [2,10]. A schematic representation is shown in Figure 1. A RBM is structured as two layers of neurons: a visible layer and a hidden layer. Each neuron is fully connected to the neurons of the other layer, but there is no connection between neurons of the same layer. The role of a RBM is to model the distribution of its input. We can stack many RBMs on top of each other by linking the hidden layer of one RBM to the visible layer of the next

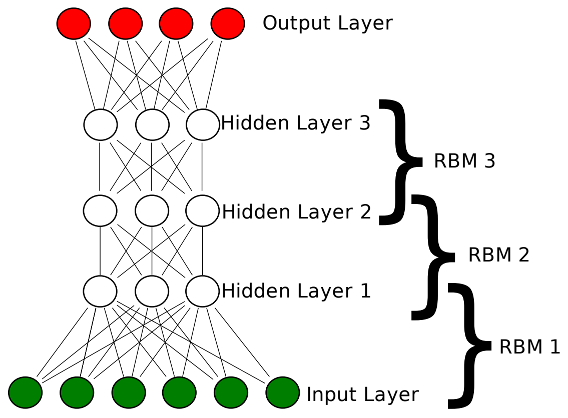

Figure 1. Schematic representation of a DBN. The number of layer and the number of units on each layer in the schema are only examples. We do not require to have the same number of units on each hidden layer.

RBM. In our experiments, we used an algorithm inspired by Gibbs sampling called Contrastive Divergence (CD) to optimize our RBMs. Our focus here is on analyzing the performance of the DBN, not in explaining the technical details of DBNs. The main idea for our purposes is that that DBNs offer an unsupervised way to learn multi-layer probabilistic representations of data that are progressively "deeper" (nonlinear) with each successive layer. For technical and mathematical details see [2, 10]. We used the Theano [1] python library to build and train our DBNs.

4. LEARNING THE FEATURES

Our goal is is to learn a representation of audio that will help us to solve the subsequent tasks of genre classification and autotagging.

4.1 Training the DBN

To learn our representation, we split the Tzanetakis' dataset in the following way: 50% for training, 20% for validation and 30% for testing. We divided the audio into short frames of 46.44ms (1024 samples at 22050 Hz sampling rate). For each of these frames, we calculated the discrete Fourier transform (DFT). We kept only the absolute values of the DFTs, and considering the symmetry in the DFT, we ended up with inputs of dimension 513.

The DBNs were first pre-trained with the training set in a unsupervised manner. We then proceeded to the supervised fine-tuning using the same training set, and using the validation set to do early-stopping. The supervised step used gradient descent to learn a weighted mixture of activations in the deepest layer to predict one of 10 genre. Both soft max and cross-entropy costs were minimized with comparable results.

We tried approximately 200 different hyper-parameters combinations and chose the model with the best validation error on the frame level. The chosen DBN model is described in Table 1.

[1] `http://deeplearning.net/software/theano/`

Number of hidden layers	3
Units per layer	50
Unsupervised learning rate	0.001
Supervised learning rate	0.1
Number of unsupervised epochs	5
Number of supervised epochs	474
Total training time (hours)	104
Classification accuracy	0.737

Table 1. Hyper-parameters and training statistics of the chosen DBN

The classifier trained from the last layer of the DBN yields a prediction of the genre for each frame. We average over all predictions for a song and choose the highest score as the wining prediction. This gave us a prediction accuracy of 73.7%.

Once trained, we can use the activations of the DBN hidden units as a learned representation of the input audio. We analyzed the performance of each layer of the network independently, and also all the layers together. To illustrate what is learned by the DBN, in Figure 2 we have plotted a 2-dimensional projection of some of the representations used. The projection was done by using the t-SNE algorithm described in [25]. Notice how the clustering of the activations of the hidden layers is more definite than for the input or the MFCCs. As we will see in Section 5, this will improve the accuracy of the classifiers.

5. CLASSIFICATION USING OUR LEARNED FEATURES

In this section, we use our learned features as inputs for genre classification and autotagging. In the first task we explore different ways of using our features to get the best classification accuracy. In the second task, we use the method that gave us the best result in the genre recognition in order to do autotagging.

For both experiments, we use a non-linear Support Vector Machine (SVM) with a radial basis function kernel [6] as the classifier. It would also be possible to train our DBN directly to do classification. However our goal is to compare the DBN learned representation with other representations. By using a single classifier we are able to carry out direct comparisons.

5.1 Genre classification

5.1.1 Frame-level features

In our first experiment, we used our frame-level features as direct input to the SVM. Since the SVM doesn't scale well with large datasets, we subsampled the training set by randomly picking 10,000 frames. We compared these accuracies to the accuracy of the SVM trained with MFCCs over these same frames of audio. As in Section 4.1, we used the frame predictions of a whole song and voted for the best genre in order to compute the test accuracy. The results for this experiments are shown in Table 2. We see

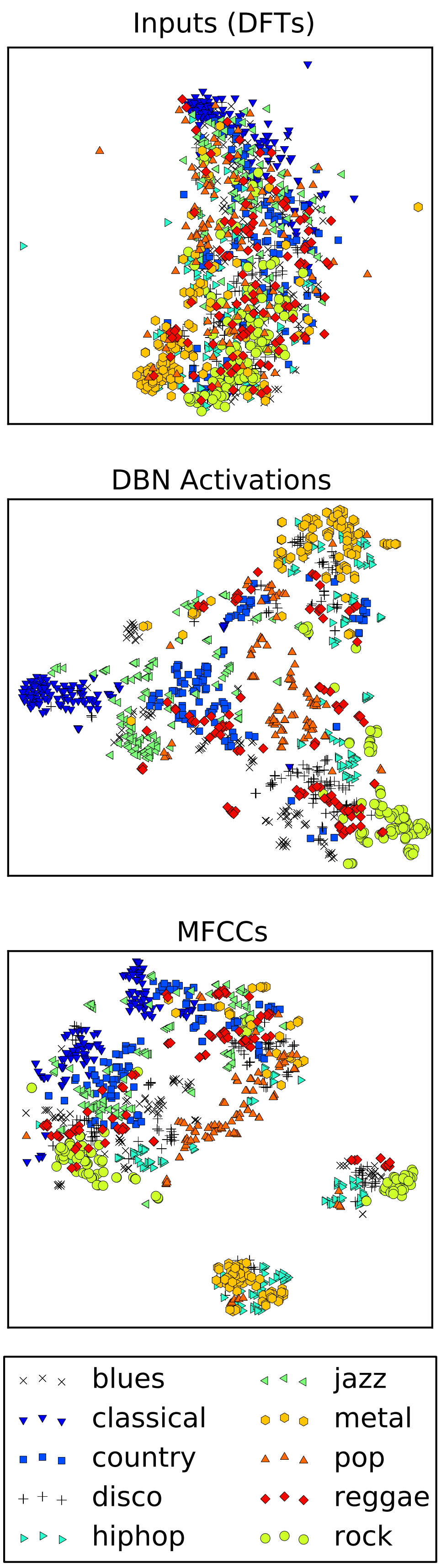

Figure 2. 2-Dimensional projections of different representations of the audio with respect to their genre.

	Accuracy
MFCCs	0.630
Layer 1	0.735
Layer 2	0.770
Layer 3	0.735
All Layers	0.770

Table 2. Classification accuracy for frame-level features

	Accuracy
MFCCs	0.790
Layer 1	0.800
Layer 2	0.837
Layer 3	0.830
All Layers	**0.843**

Table 3. Classification accuracy for features aggregated over 5 seconds

that, at the frame level, our learned features performed significantly better than the MFCCs alone. We also see that the second layer seems to have the best representation out of the three layers. By using all the layers as the input, we don't see any improvement compared to the second layer alone. Since we used the same dataset here that we used for learning the features, we took care to reuse that same training, validation and testing splits as in Section 4, so as not to contaminate our testing set. Because our learned DBN representation was learned on a single test/train split, we were unable to do cross-validation on this dataset with the SVM classifier, since this would have given us a biased result.

5.1.2 Aggregated features

Bergstra et al [4] investigated the impact of feature aggregation on classification performance for genre recognition. It is demonstrated that aggregating frame-level features over a period of time increases classification accuracy. The optimal aggregation time depend depends on the nature of the features and the classifier, with many popular features having optimal aggregation times of between 3 and 5 seconds. With this in mind, we aggregated our features over 5 seconds periods. Thus, for each 5 seconds segment of audio (with 2.5 seconds overlap), we computed the mean and the variance of the feature vectors over time. This method not only raised our classification accuracy, but also reduced the number of training examples, thus accelerating the training of the SVMs. With the aggregation, our classification accuracy by jumped to 84.3%, which is better than the 83% accuracy reported in [4]. However, since this result was reported on a 5-fold cross-validation on the dataset, we cannot directly compare our results. More importantly we observe that our results are in general competitive with the state-of-the-art signal-processing feature extraction for the genre classification task. Also, given a fixed classifier (the nonlinear SVM) our learned representation outperforms MFCCs. As in Section 5.1.1, we see that the second layer gives the best representation of all the layers, but we gain a bit of accuracy by using all of the layers.

5.2 autotagging

To test the robustness of our learned features, we tested their performance on an autotagging task. Following the results in Section 5.1, we used the activations of all the layers of the DBN aggregated on 5 second windows as inputs for the SVMs. We will refer to this set of feature as the DBN feature set. We compare it to a set of timbral and temporal features presented in [15]. We will refer to this set of feature as the MIM feature set. We used the same method as in [16] to train the SVMs over the dataset. The results for the 25 most popular tags in the dataset are shown in Figure 3 and summarized in Table 4.

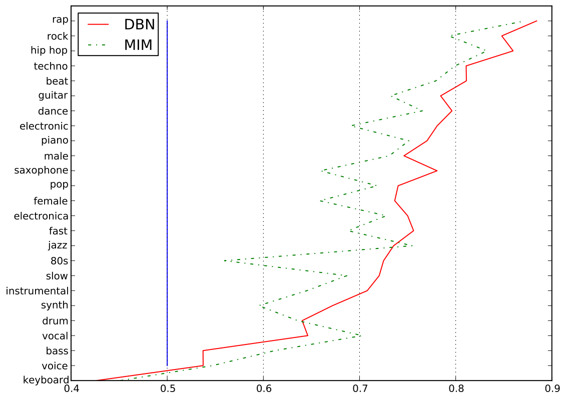

Figure 3. Accuracy of the DBN and the MIM feature sets for the 25 most popular tags. As each tag training set was balanced for positive and negative examples, the vertical line at 0.5 indicates chance accuracy.

	Mean Accuracy	Standard Error
DBN	0.73	0.02
MIM	0.70	0.02

Table 4. Mean and standard error of the autotagging results.

The results show that our features give a better classification performance for almost all the tags. In particular, our features performed significantly better better for tags such as 'rock', 'guitar', 'pop' and '80s'. Except for 'guitar', these particular tags represent genres, which is what our features were optimized to classify.

5.3 Discussion

From the results presented in Section 5.1 and Section 5.2, we see that it is indeed possible to learn features from audio relevant to a particular task. In the case of genre classification, our DBN features performed as well if not better than most signal-processing feature extraction approaches. The features were optimized to discriminate between the

10 genres shown in Figure 2, but we showed that these features were also relevant to describe many other tags, such as 'guitar', that were not related to genre. We believe this is evidence that a DBN can in fact learn to extract important and robust characteristics from audio. Another positive point is that, once the DBN is trained, the feature extraction from audio is very fast and can be done easily in real-time, which could be useful for many applications.

However, there are several areas for improvement. The main one is the long computation time necessary to train the DBN. The model that we used required a few days to train. This is mainly due to the size of the dataset. Since we used uncompressed audio frames overlapping over half a frame, the combination of the training and validation set required around 2 gigabytes of memory. There are many ways to reduce the size of the training set and to speed up the training. We could compress the DFTs with Principal Component Analysis (PCA). We could also aggregate the DFTs over small windows before sending them to the DBN. Randomly choosing a subset of the frames in the dataset could also help. Another solution would be to augment the mini-batch size to optimize the time of training process. However, it is not clear how each of these solutions will affect the quality of the representation. This requires further investigation.

Reducing the training time of a single model would also help to solve the second issue, which is the hyper-parameter search. As mentioned in Section 4.1, there are many hyper-parameters to optimize. It is not clear how the optimal hyper-parameters vary depending on the input and the task. Current research on deep learning is investigating the matter, and some techniques to automatically adjust the hyper-parameters are being developed.

Another flaw of our model is that the features are extracted at the frame level only, so that our model cannot model long-term time dependencies. To better represent musical audio, we would need features that are able to capture the long-term time structure. Convolutional DBNs might provide a suitable model for time hierarchical representations [11].

6. CONCLUSION AND FUTURE WORK

In this paper, we have investigated the ability for DBNs to learn higher level features from audio spectra. We showed that these learned features can outperform MFCCs and carefully-tailored feature sets for autotagging. These results motivate further research with deep learning applied to MIR tasks.

In future work, we will continue investigating ways to reduce the training time of our models. Furthermore, we will learn features over a wider range of datasets and MIR tasks. We are interested, for example, in using the unsupervised DBN training approach to observe a large amount of unlabeled audio data. Finally, we will continue to investigate how we can take advantage of structure found at multiple timescales in music. To this end, a hierarchical convolutional DBN may be appropriate.

7. ACKNOWLEDGEMENTS

Special thanks to Guillaume Desjardins and Michael Mandel for their contribution to the code used in the experiments. Thanks also to Simon Lemieux, Pierre-Antoine Manzagol and James Bergstra for helpful discussions, and to the Theano development team for building such useful tools. This research is supported by grants from the Quebec Fund for Research in Nature and Technology (FQRNT) as and the Natural Sciences and Engineering Research Council of Canada (NSERC).

8. REFERENCES

[1] Y. Bengio. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2:1–127, 2009.

[2] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, 2007.

[3] J. Bergstra. Algorithms for classifying recorded music by genre. Master's thesis, Université de Montréal, 2006.

[4] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65(2-3):473–484, 2006.

[5] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010.

[6] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[7] K. Chen, S. Gao, Y. Zhu, and Q. Sun. Music genres classification using text categorization method. *IEEE*, 2006.

[8] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng. Shift-invariant sparse coding for audio classification. In *Proceedings of the Conference on Uncertainty in AI*, 2007.

[9] P. Hamel, S. Wood, and D. Eck. Automatic identification of instrument classes in polyphonic and poly-instrument audio. In *in Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR'09) , Kobe, Japan*, 2009.

[10] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[11] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

[12] H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in Neural Information Processing Systems (NIPS) 22.*, 2009.

[13] T. Li and G. Tzanetakis. Factors in automatic musical genre classification. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003.

[14] M. I. Mandel and D. P. W. Ellis. Song-level features and support vector machines for music classification. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 594–599, 2005.

[15] M. I. Mandel and D. P. W. Ellis. Multiple-instance learning for music information retrieval. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 577–582, 2008.

[16] M. I. Mandel and D. P. W. Ellis. A web-based game for collecting music metadata. *Journal of New Music Research*, 37(2):151–165, 2008.

[17] P.-A. Manzagol, T. Bertin-Mahieux, and D. Eck. On the use of sparse time relative auditory codes for music. In *9th International Conference on Music Information Retrieval (ISMIR 2008)*, 2008.

[18] C. McKay and I. Fujinaga. Musical genre classification: is it worth pursuing and how can it be. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR 2006)*, 2006.

[19] I. Mierswa and K. Morik. Automatic feature extraction for classifying audio data. *Machine Learning Journal*, 58:127–149, 2005.

[20] Y. Panagakis, C. Kotropoulos, and G.R. Arce. Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, pages 249–254, 2009.

[21] G. Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Technical report, IRCAM, 2004.

[22] J. Reed and C.-H. Lee. A study on attribute-based taxonomy for music information retrieval. In *Proc. of Int. Symposium on Music Information Retrieval*, 2007.

[23] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, 10(5):293–302, 2002.

[24] G. Tzanetakis, G. Essl, and P. Cook. Automatic musical genre classification of audio signals. In *Proceedings of the 2nd International Conference on Music Information Retrieval (ISMIR 2001), Bloomington, Indiana*, 2001.

[25] L.J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.