ACOUSTIC SCENE CLASSIFICATION WITH MATRIX FACTORIZATION FOR UNSUPERVISED FEATURE LEARNING

Victor Bisot, Romain Serizel, Slim Essid, Gaël Richard

LTCI, CNRS, Télćom ParisTech, Université Paris-Saclay, 75013, Paris, France

ABSTRACT

In this paper we study the use of unsupervised feature learning for acoustic scene classification (ASC). The acoustic environment recordings are represented by time-frequency images from which we learn features in an unsupervised manner. After a set of preprocessing and pooling steps, the images are decomposed using matrix factorization methods. By decomposing the data on a learned dictionary, we use the projection coefficients as features for classification. An experimental evaluation is done on a large ASC dataset to study popular matrix factorization methods such as Principal Component Analysis (PCA) and Non-negative Matrix Factorization (NMF) as well as some of their extensions including sparse, kernel based and convolutive variants. The results show the compared variants lead to significant improvement compared to the state-of-the-art results in ASC.

Index Terms— Acoustic scene classification, unsupervised feature learning, matrix factorization

1. INTRODUCTION

Acoustic Scene Classification (ASC) is the task of identifying in which acoustic environment a sound was recorded using only the audio signal. The interest for ASC has been increasing in the last few years and is becoming an important challenge in the machine listening community [1]. Acoustic scene classification has a variety of real life applications such as robotic navigation [2] or forensics [3]. Whilst many context aware devices only use visual information to adapt to their current location, complementary information can be given by analyzing the surrounding audio environment.

Many early works in ASC have tried to use various methods from speech recognition or event classification methods. For instance features like Mel Frequency Cepstral Coefficients (MFCC) [4], linear predictive coefficients [5] or auditory filter features such as Gammatones [6] have been widely explored. Another notable trend in ASC is to use hand-crafted features designed to characterize acoustic environments. The need for more specific features is motivated by the fact that in general environmental sounds, the time and frequency information is not as structured as for speech and music signals This leads to more complex features such as expansion coefficients based on a decomposition over a Gabor dictionary [7, 8] or even minimum statistics of a spectrogram to describe the acoustical background of a scene [9]. To our best knowledge the combination of Histogram of Oriented Gradients (HOG) and Subband Power Distribution (SPD) image features, has provided the best results on the LITIS Rouen data set [10].

This work was partly funded by the European Union under the FP7-LASIE project (grant 607480)

In this paper, we propose to learn features in an unsupervised manner directly from time-frequency images. Automatically learning the features frees us from focusing on a specific aspect of the signal, by having feature extraction capable of adapting to the data specificities. Although some studies used feature learning in their ASC systems [11, 12] the benefits in performance are not clear yet when compared to hand-crafted features. We choose to focus on matrix factorization techniques as they have proven effective to learn features in previous works, while being simple. Matrix factorization techniques do not require the data to be labeled, they are only used to learn features from the decomposition of the data onto a learned basis. The labels are only needed to train a classifier based on the learned features. Moreover, in contrast to most deep learning techniques, choosing the feature learning to be unsupervised while separating it from the classification part, makes the system less demanding in terms of tuning and training and allows for using unlabeled data. After a series of preprocessing and pooling steps on the scenes spectrograms, some well known matrix decomposition techniques such as principal component analysis (PCA) and non negative matrix factorization (NMF) are exploited in this work to automatically learn relevant features. The regular PCA and NMF are then compared to some of their popular extensions, including ones with sparsity constraints, kernels and convolution. Finally, our system is evaluated on the LITIS Rouen dataset, the largest annotated ASC data base available.

The rest of the paper is organized as follows. Section 2 describes the general feature learning system. Section 3 details the different matrix factorization techniques studied. Section 4 describes the data set and our experiments, before Section 5 concludes the work.

2. FEATURE LEARNING SYSTEM

The feature learning system we propose for ASC can be decomposed in four main steps: spectrogram extraction, pooling, feature learning and classification. The general idea is to learn a dictionary from the spectrograms in the training set and then use the projections of the data on this dictionary as features for classification. In this section we present the data preprocessing steps, the feature learning step in the context of matrix decomposition and finally the use of the learned features for classification.

2.1. Time-frequency representation

We choose to use the constant Q-transform (CQT) as our time-frequency representation as it has proven to give good results in ASC when used to extract image-based features [10]. The CQT is computed using $P = 134$ frequency bands from 0 to 22050 Hz covering slightly more than the audible frequency range. Each CQT is built from a $T = 30$ s recording of an acoustic environment using 60 ms windows without overlap resulting in a $P \times 500$ image.

2.2. Spectorgram pooling

In the context of ASC, it is possible to perform the feature learning on either the full spectrogram image, time frequency slices (a group of consecutive frames) or individual frames. Classifying directly individual frames has shown to be a limited solution [13] as they lack the crucial temporal context needed to describe acoustic scenes. Using the full time-frequency representation to learn the features can lead to unreasonable computation times. Therefore, to reduce the dimensionality of the data, we start by dividing each time frequency image into m non-overlapping q-seconds long slices, with $m = T/q$. The CQT image $\mathbf{S}$ of a T-seconds long recording is now considered as a set of consecutive shorter spectrograms $\mathbf{S} = [\mathbf{S}_0, ..., \mathbf{S}_{m-1}]$. We use $\mathbf{S}_i$ to denote the q-s long spectrogram slice starting $q \times i$ seconds after the beginning of the recording. Then, to further reduce the dimensionality while keeping some temporal information, we perform a pooling on each of the m spectrogram slices. Each recording is represented by a set of vectors $\mathbf{s} = [\mathbf{s}_0, ..., \mathbf{s}_{m-1}]$ where $\mathbf{s}_i$ is a vector of size P (number of bands) obtained by averaging the slice $\mathbf{S}_i$ over time. The vectors in $\mathbf{s}$ will be used as the inputs of the feature learning step. Our system uses $q = 2$-s long slices leading to $m = 15$ slices per example. This choice is a compromise between computation time and having enough information to learn relevant features. Many more temporal integration techniques could be use in addition to the mean [14]. In our system, we choose to perform the pooling only using the average over time to keep the focus on the matrix factorization techniques.

2.3. Unsupervised feature learning

The feature learning is unsupervised, meaning the basis vectors in the dictionary are learned from the data in an unlabeled training set. The data in the test set is then projected on the same dictionary. After extracting the set of vectors $\mathbf{s}$ for each of the N training examples, we stack them horizontally to build the training data matrix $\mathbf{V}$ of size $P \times mN$. The different matrix factorization methods we propose to use for feature learning aim at decomposing the data matrix $\mathbf{V}$ such that $\mathbf{V} \approx \mathbf{W}\mathbf{H}_{tr}$. The matrix $\mathbf{W}$ of size $P \times K$ contains the K dictionary elements and $\mathbf{H}_{tr}$ of size $K \times mN$ contains the projections of each data vector on the elements of $\mathbf{W}$. The test set data is decomposed on the fixed dictionary $\mathbf{W}$ learned on the training set, which gives us the test set activation matrix $\mathbf{H}_{te}$.

2.4. Feature pooling and classification

In order to have only one feature vector per example, we introduce a second pooling step on the learned features. We build the feature vector for each T-second long example by averaging its corresponding m projection vectors leading to one mean projection vector of size K. This last step allows us to have a single feature vector for each acoustic scene recording in the training and test set. To summarize, the final feature vector for each data example is the averaged projections of each of its averaged CQT slice on the learned dictionary $\mathbf{W}$. Therefore, the features describe the scene as a whole without discarding the temporal context. Finally, a regularized logistic regression model trained on the learned features is used to classify the test-set data.

3. MATRIX FACTORIZATION TECHNIQUES FOR FEATURE LEARNING

The main motivation behind using matrix factorization is to automatically decompose data, regardless of what it represents, into meaningful parts without the need of predefining a dictionary. There exists a wide variety of different matrix factorization methods which mainly extend the formulation given in Section 2.3. In this section, we briefly present different variations of matrix factorization methods we intend to compare such as nonnegativity constraints, sparsity, non-linearity and convolution. The presented variants are mainly extensions of PCA, a common factorization method to whiten or reduce the dimension of data, or extensions of NMF, known to provide part based decompositions of nonnegative data.

3.1. Nonnegative matrix factorization

Nonnegative matrix factorization (NMF) is a well known data decomposition technique [15], to decompose nonnegative data into positive dictionary elements. In NMF, the goal is to find a decomposition that approximates the data matrix $\mathbf{V}$ such as $\mathbf{V} \approx \mathbf{W}\mathbf{H}$ with $\mathbf{W} \in \mathbb{R}_+^{P \times K}$ and $\mathbf{H} \in \mathbb{R}_+^{K \times N}$. Here, P represents the feature dimension, N the number of data points in $\mathbf{V}$ and K the number of basis vectors. NMF is obtained solving the following optimization problem:

$$\min D_\beta(\mathbf{V}|\mathbf{W}\mathbf{H}) \text{ s.t. } \mathbf{W}, \mathbf{H} \geq 0 \tag{1}$$

where D_β represents the β-divergence. We use the common multiplicative update rules [16] to optimize the problem for three different β-divergences: Euclidean ($\beta = 2$), Kullback-Leibler ($\beta = 1$) and Itakura-Saito ($\beta = 0$).

3.2. Sparse matrix factorization

Sparsity is often desired in matrix factorization in order to provide a more robust and interpretable decomposition. We look into using a sparse version of the PCA which can provide sparse dictionaries or sparse activations and an NMF with a sparsity constraint on the activation matrix.

3.2.1. Sparse PCA

There are many different formulations for the Sparse PCA model. In our work we use the one presented in [17] which considers sparse PCA as a dictionary learning problem. In the context of sparse dictionary learning, the matrices $\mathbf{W}$ and $\mathbf{H}$ are the solution of the following optimization problem:

$$\min_{\mathbf{W},\mathbf{H}} \|\mathbf{V} - \mathbf{W}\mathbf{H}\|_2^2 + \lambda \sum_{k=1}^{K} \|\mathbf{a}_k\|_1 \text{ s.t. } \|\mathbf{b}_{k:}\|_2 = 1 \tag{2}$$

where we set $\mathbf{A} = \mathbf{W}$ and $\mathbf{B} = \mathbf{H}$ to get sparse basis vectors and to get a sparse activation matrix, we set $\mathbf{A} = \mathbf{H}^T$ and $\mathbf{B} = \mathbf{W}^T$. The vector $\mathbf{b}_{k:}$ is the row in $\mathbf{B}$ and $\mathbf{a}_k$ the column in $\mathbf{A}$ indexed by k, $1 \leq k \leq K$.

3.2.2. Sparse activations with sparse NMF

As for sparse PCA there are many ways of enforcing sparsity in NMF. We use the sparse NMF formulation presented in [18], it is based on an optimization through multiplicative updates using the Euclidean distance. It was then extended in [19] for the other β-divergences. A l_1-norm constrains the activation matrix $\mathbf{H}$ while a unit l_2-norm constraint is forced on the dictionary elements. The matrices $\mathbf{W}$ and $\mathbf{H}$ are the solution of the following problem

$$\min_{\mathbf{W},\mathbf{H} \geq 0} \sum_i \|\mathbf{v}_i - \sum_k \mathrm{h}_{ki} \frac{\mathbf{w}_k}{\|\mathbf{w}_k\|}\|_2^2 + \lambda \sum_{i,k} \mathrm{h}_{ik}, \tag{3}$$

where $\mathbf{w}_k$ is the dictionary element indexed by k, $1 \leq k \leq K$.

3.3. Kernel-based non linear matrix factorization

Another possible variation of matrix factorization techniques is to decompose the data in a transformed feature space. Given a feature mapping function Φ from the original space to the transformed space, the desired decomposition approximates the data $\Phi(\mathbf{V})$ in the transformed space: $\Phi(\mathbf{V}) \approx \mathbf{W}_\Phi \mathbf{H}$. Our experiments include the use of Kernel PCA [20] (KPCA) and Kernel NMF [21] (KNMF), two kernel extensions of PCA and NMF for which we use a Gaussian kernel function.

3.4. Convolutive NMF

The convolutive NMF presented in [22] is an extension of the NMF, suited to decompose spectrograms. It extracts 2D basis vectors corresponding to groups of consecutive time frames. By doing so, convolutive NMF allows us to decompose the spectrogram of a scene in different slices, containing time-frequency images of acoustic events occurring during the scene. If one takes a spectrogram $\mathbf{S}$ of size $P \times T$, with P frequency bands and T time frames, the convolutive NMF searches for a decomposition in K different τ time frames long spectrogram slices. We search for the following approximation of $\mathbf{S}$

$$\mathbf{S} \approx \sum_{t=0}^{\tau-1} \mathbf{W}_t \overset{t\rightarrow}{\mathbf{H}}, \tag{4}$$

where $\mathbf{W}_t \in \mathbb{R}_+^{P \times K}$ and the k^{th} column of $\mathbf{W}_t$ corresponds to the time frame t of the 2D dictionary element indexed by $k, 1 \leq k \leq K$. Applying the operation $t \rightarrow$ to $\mathbf{H}$ shift its columns t indexes to the right while putting the first t columns to 0. Since the convolutive NMF is primarily suited for decomposing time-frequency images, we will not directly apply it on the previous data matrix $\mathbf{V}$. Instead, we extract a different 3D dictionary $\mathbf{W}_i$ for each audio example i in the training set. The $\mathbf{W}_i$ are concatenated to build a global dictionary $\hat{\mathbf{W}} = [\mathbf{W}_1, ..., \mathbf{W}_N]$. Since the size of the global dictionary is too high, we perform a K-means clustering on $\hat{\mathbf{W}}$ in order to build a reduced dictionary $\underline{\mathbf{W}}$, containing the K_c cluster centers. The feature vector learned for a given data example is obtained by decomposing its spectrogram on $\underline{\mathbf{W}}$ and computing the average of the projection vectors over time (the average of each row in $\mathbf{H}$).

4. EXPERIMENTAL EVALUATION

4.1. The dataset

The unsupervised feature learning variants considered are evaluated on the LITIS Rouen data set [13]. To our knowledge it is the largest publicly available dataset for ASC. It contains 25 h of urban audio scenes recorded with a smart-phone, split into 3026 examples of 30 s without overlap forming 19 different classes. Each class corresponds to a specific location such as *in a train station* or *at the market*.

4.2. Evaluation protocol

All experiments use the training-test splits suggested by the authors of the dataset to guaranty comparable results. The classifier is a multi-class linear logistic regression. In order to compute the results for each training-test split we use the mean average F1 score. The final F1 score is the average value of F1 over the 20 splits. In each split 80% of the examples are kept for training and the other 20 % for testing. A log compression is applied to spectrograms before the pooling step when we use the PCA and its variants. For the different NMF extensions, we tried using a $\log(1 + x)$ type compression since negative data points are not permitted but better results were obtained a square root compression. Because the focus is on feature learning, the classifier is kept linear to permit easier comparisons between the tested techniques. Better results could probably be obtained by using a non linear kernel but this is left for future works. Finally, when we claim a method significantly outperforms another one, statistical significance is asserted via a cross-validated student t-test ($p < 0.05$).

	PCA		NMF	
Variants	Tested	Max K	Tested	Max K
Non modified	o	128	o	1024
Sparse basis	o	128	×	×
Sparse activations	o	128	o	1024
Kernel-based	o	1024	o	256
Convolution	×	×	o	1024

Table 1: Summary of the variants tested for PCA and NMF. Max K specifies the highest dictionary size tested for each technique.

4.3. Results for the basic matrix factorizations

The first results presented in Table 2 are obtained using the basic PCA and NMF to learn the features. We also indicate the results when using Independent Component Analysis (ICA) [23] and Factor Analysis (FA) which are two other popular data decomposition techniques. Because the input data points are of dimension 134, we can not search for a higher number of components for PCA or ICA but NMF has no such limitations.

	$K = 128$	$K = 256$	$K = 512$	$K = 1024$
PCA	89.8	-	-	-
ICA	**90.0**	-	-	-
FA	**90.0**	-	-	-
NMF $\beta = 2$	87.9	89.7	89.4	89.9
NMF $\beta = 1$	88.4	90.1	89.6	90.3
NMF $\beta = 0$	88.7	90.5	90.2	**90.7**

Table 2: F1 scores for PCA, ICA, NMF and Factor Analysis on different dictionary sizes K

As shown in Table 1, none of the average F1 scores obtained for PCA, ICA, factor analysis and NMF significantly stands out. The classification results reaching up to a 90.7% F1 score but are still far from improving the 92.8% state-of-the-art F1 score results on the dataset [10]. The scores obtained are promising but indicate the need for more complex decompositions capable of better capturing the specificities of the data.

4.4. Influence of sparsity

The results when adding sparsity constraints for PCA and NMF are presented Table 3. The λ parameter is the regularization parameter controlling the influence of the l_1 norm constraints in equations (2) and (3). For Sparse NMF the results are given for the Euclidean distance ($\beta = 2$) as it gave the best results.

The scores for sparse PCA with sparsity constraints on the basis vectors show that enforcing more sparsity on the dictionary leads to

Sparse PCA with sparse basis vectors					
K	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$
128	**90.0**	89.0	88.7	88.5	86.8
Sparse PCA with sparse activations					
K	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$
128	**90.0**	90.0	89.1	82.6	65.2
Sparse NMF					
K	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$
128	**88.5**	88.2	88.0	86.7	87.1
256	89.9	**90.8**	90.6	90.1	90.1
512	91.2	92.0	**93.3**	91.9	91.1
1024	92.0	93.1	**94.1**	92.1	91.8

Table 3: F1 scores for Sparse NMF and Sparse PCA for different dictionary sizes K and sparsity constraints λ

decreasing performance. A possible explanation is that the basis vectors, regardless of the dictionary size, can not exceed P dimensions and forcing them to have more coefficients set to 0 may lead to discarding important information. In Sparse NMF, adding the sparsity constraint on the activation matrix allows us to reach better results when using a higher number of dictionary elements (K=1024). For lower values of K, adding sparsity to the activation matrix in the PCA and NMF decompositions decreases the results. The sparsity on the activation matrix enforces each data point to be explained by only a few basis vectors and thus often leads to a more realistic decomposition, where each scene is only explained by a few basis vectors. The best result for sparse NMF is a 94.1% F1 score obtained with $\lambda = 0.25$. It is a significant improvement over the 92.0% F1 score obtained with $\lambda = 0$ and over the 92.8% F1 score obtained by the state-of-the-art method on the same dataset [10].

4.5. Influence of non-linearity

We now look at the influence of using kernel extensions of the PCA and NMF. A Gaussian kernel was used for both the Kernel PCA and Kernel NMF. The σ parameter for the Gaussian kernel function is tuned using cross-validation on a sub-set of the data. The results of our experiments are presented in Table 4.

	$K = 128$	$K = 256$	$K = 512$	$K = 1024$
KPCA	91.6	93.3	94.3	**95.6**
KNMF	78.1	84.1	-	-

Table 4: F1 scores for Kernel PCA and Kernel NMF compared to NMF and PCA on different dictionary sizes K

Since the data is decomposed in a transformed feature space, this allows us to build larger over-complete dictionaries with Kernel PCA than with the regular PCA. For KNMF, the computation time gets too prohibitive for high values of K, preventing us from providing results for K above 256. Indeed, the presence of the Gram matrix $\Phi(\mathbf{V})^T\Phi(\mathbf{V}) \in \mathbb{R}^{N\times N}$ in the multiplicative update rules makes KNMF much more complex then NMF when $N >> P$. By using the KPCA with 1024 components, we obtain a 95.6% F1 score which significantly outperforms the previous results for PCA and the state-of-the-art method.

4.6. Results with the convolutive NMF

As we have mentioned, since the convolutive NMF is applied on full spectrograms, the feature learning architecture is different from the previous experiments and was described in Section 3.4. The spectrograms are decomposed using 2D dictionary elements of 4 consecutive time frames ($\tau = 0.25$ s). We considered decomposing on longer slices (8 or 16 consecutive frames) but it did not provide better results. Each example in the training set is approximated by a dictionary of 80 slices leading to a total training dictionary of size $80N$ before clustering. The results shown in Table 5 are given for different number K_c of cluster centers obtained after applying the K-means to $\underline{\mathbf{W}}$. The convolutive NMF and regular NMF are compared using the same type of feature learning architecture. The *NMF + clustering* method uses the regular NMF to learn a separate basis of 5 vectors on each 2-s spectrogram slice. Similarly to convolutive NMF, the concatenation of all basis is clustered to keep a dictionary of size K_c used to extract the projection features. The best results were obtained with the Itakura-Saito divergence ($\beta = 0$) for both methods.

Cluster centers	$K_c = 256$	$K_c = 512$	$K_c = 1024$
Convolutive NMF	90.5	92.6	**94.5**
NMF + clustering	90.1	92.2	**93.7**

Table 5: F1 scores for convolutive NMF and NMF with clustering in function of the dictionary sizes K_c

The convolutive NMF appears to be a well suited model to answer the specific difficulties of ASC. In fact, it decomposes an acoustic scene as a superposition of different short acoustic events. Contrarily to the regular NMF, because we consider slices of the spectrogram, the time-frequency structure of acoustic events is less altered. The results with the *NMF + clustering* technique shows that an important part of the improvement obtained with the convolutive NMF has to be attributed to the change of architecture compared to the other methods presented. The results obtained with *NMF + clustering*, up to a 93.7 % F1 score, are slightly improved when using the convolutive NMF, reaching a 94.5% F1 score. In line with the Sparse NMF and the Kernel PCA it also significantly improves the previous state-of-the-art result on the data set.

5. CONCLUSION

In this paper we have studied and compared different popular matrix factorization methods to perform unsupervised feature learning for acoustic scene classification. Our experiments on the largest available ASC dataset compare the use of extensions of the regular PCA and NMF such as sparsity, kernels and convolution. The classification scores show that these different variants of matrix factorization all allow us to get improved results. We manage to outperform the previous state-of-the-art results on the LITIS Rouen dataset with Sparse NMF (94.1% F1-score), Kernel PCA (95.6% F1-score) and convolutive NMF (94.5% F1-score). In the future we intend to combine some of the good performing matrix factorization variants presented to take advantage of more then one of the sparsity, kernel or convolution extensions at time. For instance, methods have already been developed to introduce sparsity in kernel PCA or in convolutive NMF.

6. REFERENCES

[1] D. Barchiesi, D. Giannoulis, D. Stowel, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[2] S. Chu, S. Narayanan, C.-C. J. Kuo, and M. J. Mataric, "Where am i? scene recognition for mobile robots using audio features," in *IEEE International Conference on Multimedia and Expo*, 2006, pp. 885–888.

[3] G. Muhammad, Y. A. Alotaibi, M. Alsulaiman, and M. N. Huda, "Environment recognition using selected mpeg-7 audio features and mel-frequency cepstral coefficients," in *Fifth International Conference on Digital Telecommunications (ICDT)*, 2010.

[4] J.-J. Aucouturier, B. Defreville, and F. Pachet, "The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music," *The Journal of the Acoustical Society of America*, vol. 122, no. 2, pp. 881–891, 2007.

[5] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 321–329, 2006.

[6] X. Valero and F. Alías, "Gammatone cepstral coefficients: biologically inspired features for non-speech audio classification," *IEEE Transactions on Multimedia*, vol. 14, no. 6, pp. 1684–1689, 2012.

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

[8] R. Mogi and H. Kasaii, "Noise-robust environmental sound classification method based on combination of ica and mp features," *Artificial Intelligence Research*, vol. 2, no. 1, pp. p107, 2012.

[9] S. Deng, J. Han, C. Zhang, T. Zheng, and G. Zheng, "Robust minimum statistics project coefficients feature for acoustic environment recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 8232–8236.

[10] V. Bisot, S. Essid, and G. Richard, "Hog and subband power distribution image features for acoustic scene classification," in *European Signal Processing Conference (EUSIPCO)*, 2015.

[11] E. Benetos, M. Lagrange, and S. Dixon, "Characterisation of acoustic scenes using a temporally constrained shift-invariant model," in *Digital Audio Effects*, 2012.

[12] J. Nam, Z. Hyung, and K. Lee, "Acoustic scene classification using sparse feature learning and selective max-pooling by event detection," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events*, 2013.

[13] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene classification," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 1, pp. 142–153, 2015.

[14] C. Joder, S. Essid, and G. Richard, "Temporal integration for audio classification with application to musical instrument classification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 1, pp. 174–186, 2009.

[15] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[16] C. Févotte and J. Idier, "Algorithms for nonnegative matrix factorization with the β-divergence," *Neural Computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[17] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the ACMl International Conference on Machine Learning*, 2009, pp. 689–696.

[18] J. Eggert and E. Körner, "Sparse coding and nmf," in *2004 IEEE International Joint Conference on Neural Networks*, 2004, vol. 4, pp. 2529–2533.

[19] J. Le Roux, F. J. Weninger, and J. R. Hershey, "Sparse nmf–half-baked or well done?," Tech. Rep., Mitsubishi Electric Research Labs (MERL), 2015.

[20] B. Schölkopf, A. Smolar, and K.-R Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[21] D. Zhang, Z. Zhou, and S. Chen, "Non-negative matrix factorization on kernels," in *PRICAI 2006: Trends in Artificial Intelligence*, pp. 404–412. Springer, 2006.

[22] P. D. O'Grady and B. A. Pearlmutter, "Convolutive non-negative matrix factorisation with a sparseness constraint," in *Workshop on Machine Learning for Signal Processing*, 2006, pp. 427–432.

[23] P. Comon, "Independent component analysis, a new concept?," *Signal processing*, vol. 36, no. 3, pp. 287–314, 1994.