Feature Learning with Matrix Factorization Applied to Acoustic Scene Classification

Victor Bisot, Romain Serizel, Slim Essid, Gael Richard

Feature Learning with Matrix Factorization Applied to Acoustic Scene Classification

Victor Bisot, Romain Serizel, Slim Essid, and Gaël Richard
LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75013, Paris, France

***Abstract*—In this paper, we study the usefulness of various matrix factorization methods for learning features to be used for the specific Acoustic Scene Classification problem. A common way of addressing ASC has been to engineer features capable of capturing the specificities of acoustic environments. Instead, we show that better representations of the scenes can be automatically learned from time-frequency representations using matrix factorization techniques. We mainly focus on extensions including sparse, kernel-based, convolutive and a novel supervised dictionary learning variant of Principal Component Analysis and Nonnegative Matrix Factorization. An experimental evaluation is performed on two of the largest ASC datasets available in order to compare and discuss the usefulness of these methods for the task. We show that the unsupervised learning methods provide better representations of acoustic scenes than the best conventional hand-crafted features on both datasets. Furthermore, the introduction of a novel nonnegative supervised matrix factorization model and Deep Neural networks trained on spectrograms, allow us to reach further improvements.**

***Index Terms*—Acoustic Scene Classification, Feature learning, Matrix Factorization**

I. Introduction

THE task of identifying the environment in which a sound has been recorded is now commonly referred to as Acoustic Scene Classification (ASC) [1]. The main objective is to attribute a semantic label to recorded *soundscapes*, often corresponding to the type of location in which the scene takes place, such as *a train station*, *in a bus* or *a residential area*. Recognizing acoustic environments is one of the challenging tasks of the more general Computational Auditory Scene Analysis (CASA) [2] research field and is receiving an increasing interest in the machine listening community. Analyzing the surrounding audio environment may allow devices to increase their context awareness capabilities, especially when geolocalization and visual data is not available [3]. Up to now, ASC has proven to be important in many real life applications such as robotic navigation [4], personal archiving [5] or surveillance [3]. The growing interest for ASC has also motivated the community to organize challenges such as the AASP DCASE challenge in 2013 [6] and in 2016 [7].

The analysis of environmental sounds, can be separated in two more specific problems: ASC and acoustic event classification. At first, the acoustic event and scene classification tasks were not always clearly distinguished, in fact both scene and event labels coexisted in many datasets [4], [8]. More recently, research in ASC mainly aims at classifying longer segments of audio recordings and thus needs methods capable of characterizing acoustic environments as a whole. This aspect is particularly challenging as acoustic scenes are constituted of superpositions of an important variety of sound events, which are not all relevant to describe the environment.

One of the first steps in designing an ASC system is usually the choice of the features used to describe the acoustic environments. Motivated by their success in speech analysis, one of the original choices has been the Mel Frequency Cepstral Coefficients (MFCC) [4], [9]. These features have shown to give somewhat limited performance for ASC. In fact, environmental sounds are a lot less structured than speech or music signals. They can be of very different nature and have a high variability in their temporal and spectral structures. Perceptual studies of acoustic scenes explained how humans use particular event-cues in order to differentiate acoustic environments [10]. This suggests that having a way of better characterizing the acoustic events occurring in the scenes will help to more accurately discriminate the acoustic environments. Following this idea, the authors in [11] proposed a system which uses sequences of labeled events in the training phase in order to improve the classification of acoustic scenes. In most cases, the various events occurring in the scene are not labeled. This encourages to choose or design features capable of describing these events in an unsupervised manner. To address this problem, notable trends in ASC are for example to extract features inspired from computer vision [12], [13] or focus on modeling the statistical distribution of more traditional features over time [14], [15].

In this paper, we study the benefits of automatically learning features directly from time-frequency representations of acoustic scenes. Motivated by the success of spectrogram image features for ASC [12], [13], learning features from spectrograms can provide representations that are adapted to the data while addressing the general lack of flexibility of hand-crafted features. In fact, most ASC works relied on feature engineering techniques, often inspired from other tasks, to extract representations. The main drawback of such approaches, is that the resulting hand-crafted features generally lack the capability to generalize to problems other than the ones they have been initially designed for. Here, the goal is to learn features by decomposing time-frequency representations on a dictionary of basis elements representing the data. In general, dictionaries can either be predefined or automatically learned form the data using dictionary learning techniques [16]. The learned features are then obtained by projecting the data on to the dictionary, often relying on sparse coding methods [17]. Here, we perform feature learning using matrix factorization techniques, which have the advantage of jointly learning the dictionary and the projections. In our case, the

dictionary elements can be seen as frequency templates of characteristic events occurring during the scenes which we will refer to as *basis events*. Then, the projections of each data example on the dictionary contains the activations coefficients of the basis events during the recording. Especially in the case of nonnegative decompositions, this process is similar to the human strategy for discriminating acoustic scenes [10], the learned basis events being the event cues used to identify the environments. Hence, the projections are used as the learned features for the classification step. Learning features from time-frequency representations has shown promising results in other sound classification fields such as noisy acoustic event classification [18], speaker identification [19] or music information retrieval [20] and often offers improvements over hand-crafted features.

The first part of this study extends our previous work [21] by comparing popular unsupervised matrix factorization techniques, as well as some of their well known extensions, on a new ASC dataset. The first category of methods considered here are unsupervised, meaning the labels are not used to inform the decompositions. Unsupervised matrix factorizations methods have the advantage of being simpler to tune and less complex than their supervised counterparts while providing good representations adapted to the data at hand. They include sparse, kernel-based and convolutive variants of Principal Component Analysis (PCA) and Nonnegative Matrix Factorization (NMF) [22]. Then, we present a new contribution by adapting a supervised matrix factorization model known as *Task-driven Dictionary Learning* (TDL) [23] to suit the ASC task. We introduce a nonnegative extension of the TDL model, including a modification of the original algorithm, where a nonnegative dictionary is jointly learned with a multi-class classifier. Finally, we compare the performance of the matrix factorization variants and deep learning methods such as feed-forward Deep Neural Networks (DNN) that are state-of-the-art for many classification tasks. An experimental study is performed on two different ASC datasets, the DCASE 2016 scene classification dataset [7] and the LITIS Rouen dataset [12]. The different methods are studied under the same simple preprocessing and classification blocks to ensure fair comparison. The experimental results suggest that unsupervised matrix factorization is a good alternative to hand-crafted features for the task. Indeed, many of the variants presented allow us to get significant improvements over the best state-of-the-art features for ASC. Finally, we reach further improvement by jointly learning nonnegative dictionaries and the classifier with our new TDL algorithm as well as with DNN.

The rest of the paper is organized as follows. Information about the previous works in ASC is presented in Section II. Section III details the preprocessing steps before the feature learning. The different unsupervised variants of matrix factorization considered are then presented in Section IV. The supervised dictionary learning model as well as the proposed modifications are detailed in Section V. Section VI presents an experimental evaluation of the compared techniques. The proposed systems are compared to the state-of-the-art results in Section VII. Finally, some conclusions are suggested in Section VIII.

II. Related works

The combination of MFCC with Hidden Markov Models (HMM) or Gaussian Mixture Models (GMM), inspired from speech recognition, has been a common way to approach ASC in early works. This combination still serves as a baseline in the recent DCASE evaluation campaigns [6], [7]. For improved performance, MFCC have often been combined with a set of popular low-level features such as zero-crossing rate, spectral-roll off as well as linear predictive coefficients [14], [24]. Alternatively, other filter-banks such as Gammatones were considered instead of the Mel-spectrum [25]. Another important part of ASC works focused on finding features better suited for the task. This lead to the introduction of more complex features such as expansion coefficients obtained by a decomposition over a Gabor dictionary [26] or features representing the background of acoustic environments using minimum statistics on spectrograms [27]. Moreover, in the last few years, particular attention has been dedicated to the use of image features which are extracted from time-frequency images of acoustic scenes. For instance, Histograms of Oriented Gradients (HOG) were exploited for ASC [12]. They aim at characterizing the spectro-temporal variations of acoustic events occurring in a scene by computing the gradient of pixels in time-frequency images. In addition to HOG, other spectrogram image-based features have been proposed such as the Subband Power Distribution (SPD) [13] or Local Binary Patterns [28].

Because of the frame-based nature of many of these features, a particular focus on temporal integration is required in order to model the distribution of the features across the full duration of the scene. A common way of modeling the temporal information is either to extend the feature set with their first and second order derivatives or to compute their average over time, possibly combined with more complex statistical functions [14], [29]. The use of Recursive Quantitative Analysis (RQA) [15] on MFCC has also proven to be effective for modeling temporal information.

The combination of various frame based features often leads to high dimensional representations which can be reduced using Principal Component Analysis (PCA) or independent component analysis [24]. As for the decision stage, most ASC works use a maximum likelihood approach, modeling the data with GMM [4], [9] and/or HMM [24] or with max-margin classifiers such as Support Vector Machines [12], [14] with their linear or kernel-based formulations. Some also proposed the use of decision trees and random forests [30], [31] or a Deep Neural Network using MFCC as the input features [32].

Closer to our work, NMF [33] and Shift-Invariant Probabilistic Latent Component Analysis [34] have been used for ASC. In both cases, they were applied to decompose each data example, *i.e.* an audio excerpt, on a small dictionary. They aim at learning a few spectral templates per example, considering the dictionaries for each example as features. Instead, we will show, for various matrix factorization methods, how learning a common dictionary from the full training set, then using it to represent the audio data, by projection on to that dictionary, allows us to obtain competitive features for ASC. On the Litis

Rouen Dataset [12], the state-of-the-art hand crafted features is the combination of HOG and SPD [13]. More recently, [35] proposed a supervised linear combination of a bag of frame approach to learn features using Fisher vectors and HOG features, showing improvement compared to the hand-crafted features alone.

III. Learning features from time-frequency images

In this section, we describe the general framework under which we intend to compare matrix factorization techniques in a fair way. We start with the choice of a suited time-frequency representation for the acoustic scenes. We then introduce a set of simple pooling steps applied to the representations in order to build the data matrix from which we will learn the features. Furthermore, we explain how matrix factorization is applied on the data matrix to learn a common dictionary on the training set, the projections of the data on the dictionary being used as features for classification. Finally, we describe how a decision is taken on the full duration of the audio examples. The different construction steps of the data matrix are illustrated in Fig. 1.

A. Low-level time-frequency representation

The low-level time-frequency representations from the signals using a Constant Q-transform (CQT). Commonly used for music information retrieval tasks, the CQT has also often been the chosen representation to compute image-based features in ASC [12], [35]. It mainly helps by providing log-scaled frequency bands, closer to the behavior of the human auditory system, while being a lower dimensional representation of signals compared to short time Fourier transforms. We denote by $\mathbf{S} \in \mathbb{R}_+^{P\times T}$ the CQT of a given recording, where T is the number of time frames and P the number of frequency bands. Without loss of generality, in the remainder of this paper, the recordings are assumed to have equal length.

B. Matrix factorization for feature learning

The general idea of matrix factorization methods is to jointly learn a dictionary and the projections of the data on this dictionary. The dictionary is constituted of a set of basis vectors each representing certain aspects of the data. We recall that in our case, the basis vectors are referred to as "basis events" as they can be seen as representations of possible events occurring during the scenes. For both unsupervised and supervised techniques, the dictionary is only learned on a training set of the data. The features are obtained by projecting the data on the set of learned basis events in the dictionary.

The first feature learning approach studied here consists in directly learning a common dictionary from the full training set. In that case, we assume that the input data to decompose is stored in a common data matrix $\mathbf{V} \in \mathbb{R}^{P\times N}$, where P is the number of features and N the number of examples. Then, the matrix factorization techniques search for a factorization of the data matrix $\mathbf{V}$ as a product of two matrices such that $\mathbf{V} \approx \mathbf{WH}$. The matrix $\mathbf{W} \in \mathbb{R}^{P\times K}$ is the dictionary containing the K basis events. The projection matrix $\mathbf{H} \in \mathbb{R}^{K\times N}$ contains the projections (or event activations) of each data vector on the elements of $\mathbf{W}$. The test set data is decomposed on the fixed dictionary $\mathbf{W}$ learned from the training set.

C. Building the data matrix

Many of the confronted factorization techniques do not include any form of temporal modeling in their original formulations and would benefit from an adapted temporal preprocessing of the data. Moreover, trying to decompose the whole set of full time-frequency images would lead to unreasonable matrix sizes. Therefore, we apply two simple slicing and pooling steps aiming at reducing the dimensionality of the data while providing a suitable representation to the feature learning step. To do so, we start by dividing each time frequency image $\mathbf{S}$ into M non-overlapping slices of length $Q = T/M$. We use $\mathbf{S}_m$ to denote the Q-frames long spectrogram slice starting $Q \times m$ frames after the beginning of the recording. The CQT image $\mathbf{S}$ is now considered as a set of consecutive shorter spectrograms $\mathbf{S} = [\mathbf{S}_0, ..., \mathbf{S}_{M-1}]$. Each of the M spectrogram slices are then averaged over time resulting in M vectors. Assuming we have L training examples, every recording is now represented by a set of vectors $\mathbf{V}^{(l)} = [\mathbf{v}_0^{(l)}, ..., \mathbf{v}_{M-1}^{(l)}]$ where $\mathbf{v}_m^{(l)}$ is a vector of size P obtained by averaging the slice $\mathbf{S}_m^{(l)}$ over time. We extract the L sets of vectors $\mathbf{V}^{(l)}$ in the training set and stack them column-wise to build the data matrix $\mathbf{V} \in \mathbb{R}_+^{P\times N}$, where $\mathbf{V} = [\mathbf{V}^{(1)}, ..., \mathbf{V}^{(L)}]$ and $N = ML$.

Reshaping the time-frequency images in that manner helps representing the frequency signature of the scene at different times in the recording. In fact, each spectrogram slice will contain time-frequency information of various events occurring in the scene. The matrix factorization step will gather in the dictionary, a representation of the most frequent events, *i.e.* the averaged slices. On the other hand, the events that are occurring the less frequently will not have a great impact on the construction of the dictionary. Thus, only the most relevant sets of events to characterize the scenes will be modeled and, for a dictionary of sufficient size, the projections on these basis events will be able to discriminate between most acoustic environments.

Alternative temporal integration techniques could be used in addition to the average [36]. In this work, we choose to perform a simple pooling based on averaging over time to focus on the impact of the matrix factorization techniques.

D. Classification

At this stage we have M feature vectors per audio recording example and a decision needs to be taken for attributing a label to the full example. The final feature vector for each excerpt is built by averaging its M projection vectors, stored in the activation matrix $\mathbf{H}$. Finally, the classifier is a regularized linear logistic regression in its multinomial formulation [37], trained on the learned features. Logistic regression has the benefit of having a direct multinomial formulation not relying on one-versus strategies. It can also directly output the class probabilities for each data point. The classifier is kept linear

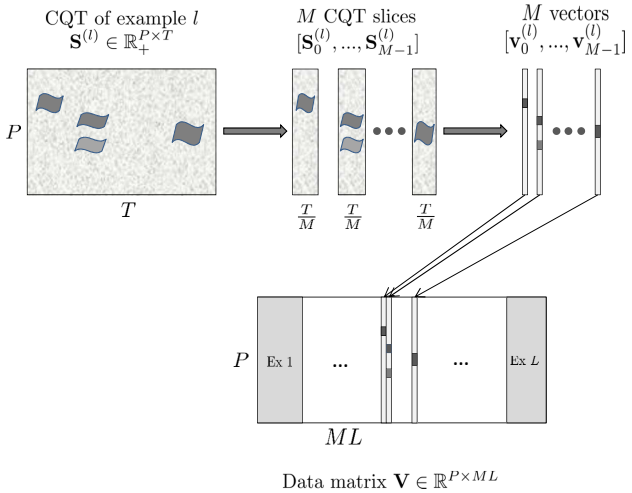

Fig. 1: Building steps of the data matrix $\mathbf{V}$, input representation for the matrix factorizations.

in the comparative study to better observe the ability of the feature learning methods to produce discriminative features.

IV. Unsupervised matrix factorization variants

In this section, we briefly present different formulations of the matrix factorization problem we intend to compare experimentally. They mostly extend the basic formulation given in III-B, often by adding constraints on the matrices $\mathbf{W}$ and $\mathbf{H}$. All the methods can be regarded as variants of PCA or NMF. We will describe unsupervised extensions of PCA and NMF including sparsity, kernel-based and convolutive factorizations.

A. Nonnegative matrix factorization

NMF is a well known technique to decompose nonnegative data into nonnegative dictionary elements [22]. Many problems benefit from the nonnegativy of the decomposition to learn better representations of the data, especially in the audio processing field. In fact, most of the time-frequency representations for audio signals contain only nonnegative coefficients. For multi-source environments like acoustic scenes, the nonnegative constraints allows for interpreting the time-frequency representation as a sum of different nonnegative objects, corresponding to the different sources. In NMF, the goal is to find a decomposition that approximates the data matrix $\mathbf{V} \in \mathbb{R}_+^{P\times N}$ such as $\mathbf{V} \approx \mathbf{WH}$ with $\mathbf{W} \in \mathbb{R}_+^{P\times K}$ and $\mathbf{H} \in \mathbb{R}_+^{K\times N}$. NMF is obtained by solving the following optimization problem:

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

where D_β represents the β-divergence [38]. The particular cases of interest for the β-divergence are the Euclidean distance ($\beta = 2$), Kullback-Leibler ($\beta = 1$) and Itakura-Saito ($\beta = 0$). For more information about NMF generalities the interested reader is referred to the numerous publications on the topic [39], [40].

B. Sparse matrix factorization

Sparsity is often desired in matrix factorization in order to provide a more robust and interpretable decomposition. Here, we aim at building a common dictionary of basis events capable of representing all the labels in the dataset. In that case, for a more meaningful decomposition, each data point should be explained by a small subset of the dictionary elements, containing the most relevant basis events to describe the corresponding scene label. Therefore, we are particularly interested in adding sparsity constraints to the activation matrix in the PCA and NMF decompositions. In the case where we have an ℓ_1-norm penalty to promote sparsity of the activation matrix, the matrix factorization problem is generally defined as:

$$\min_{\mathbf{W},\mathbf{H}} D_\beta(\mathbf{V}|\mathbf{WH}) + \lambda \sum_{k=1}^{K} \|\mathbf{h}_{k:}\|_1 \text{ s.t. } \|\mathbf{w}_k\|_2 = 1; \qquad (2)$$

the vector $\mathbf{h}_{k:}$ is the row in $\mathbf{H}$ and $\mathbf{w}_k$ the column in $\mathbf{W}$ indexed by k, $1 \leq k \leq K$.

1) Sparse PCA: There are many different formulations for the sparse PCA model. In our work we use the one presented in [16] which presents sparse PCA as a more general dictionary learning problem. In the context of sparse dictionary learning, the matrices $\mathbf{W}$ and $\mathbf{H}$ are the solution of the problem (2), with D_β being the Euclidean distance ($\beta = 2$).

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 [41] which is simply obtained by adding nonnegative constraints on $\mathbf{W}$ and $\mathbf{H}$ to the problem equation (2). The advantages of this formulation and its optimization procedure with the general β-divergence are summarized in [42].

C. Kernel-based matrix factorizations

Some families of matrix factorization methods, such as Kernel PCA [43] (KPCA) and Kernel NMF [44] (KNMF), decompose the data in a transformed feature space. Kernel methods have the advantage of being able to deal with data that is non linearly separable, by projecting them into a higher dimensional feature space. Indeed, the kernel formulations of the matrix factorization problems can help exhibit more complex relations in the input data. 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: $\Phi(\mathbf{V}) \approx \mathbf{W}_\Phi\mathbf{H}$, where $\mathbf{W}_\Phi$ is the dictionary of basis vectors in the transformed space. Usually, the basis vectors in $\mathbf{W}_\Phi$ are defined as convex linear combinations of the data in the transformed space. Even though we do not necessarily have access to the data in the new feature space, a dot product in the transformed space, *i.e* the kernel function, is always defined. Thus, one can compute the projection matrix $\mathbf{H}$ without explicitly knowing $\Phi(\mathbf{V})$ and $\mathbf{W}_\Phi$, by having access to $\mathbf{W}_\Phi^{\mathrm{T}}\Phi(\mathbf{V})$. For more information about KNMF and KPCA, the interested reader is referred to [43], [44].

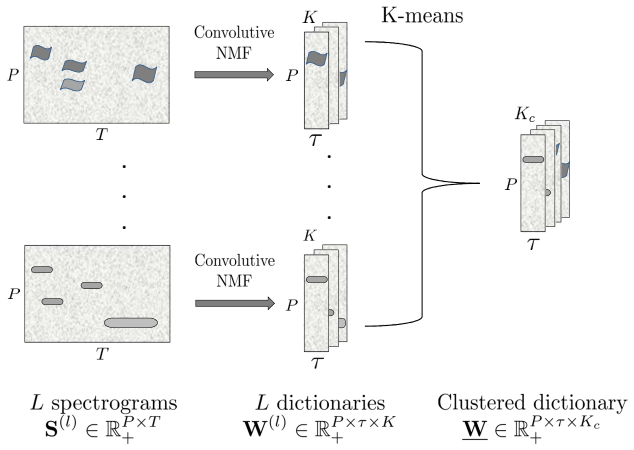

Fig. 2: Building steps of the data matrix $\mathbf{V}$, input representation for the matrix factorizations.

D. Convolutive NMF

The convolutive NMF presented in [45] is an extension of the NMF, suited to decompose time-frequency representations. 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 a dictionary of different slices, containing time-frequency images of acoustic events occurring during the scene. If one takes a spectrogram $\mathbf{S} \in \mathbb{R}_+^{P\times T}$, the convolutive NMF seeks the following approximation of $\mathbf{S}$:

$$\mathbf{S} \approx \sum_{t=1}^{\tau} \mathbf{W}_t \overset{t\rightarrow}{\mathbf{H}}, \tag{3}$$

where $\mathbf{W}_t \in \mathbb{R}_+^{P\times K}$ and the k^{th} column of $\mathbf{W}_t$ corresponds to the time frame $t \in [\![1, \tau]\!]$ of the 2D dictionary element indexed by k, $1 \leq k \leq K$. Applying the operation $t \rightarrow$ to $\mathbf{H}$ shifts its columns t indexes to the right while putting the first t columns to 0. For audio data, the convolutive NMF has proven to be effective to model the temporal evolution of acoustic events represented in time-frequency images [19]. However, applying it directly on the previously defined data matrix $\mathbf{V}$ makes less sense since the temporal structure of the events was in part discarded by the pooling steps. Therefore, the architecture of the feature learning system is changed when using the convolutive NMF and is illustrated in Fig. 2. Here, we extract a different 3D dictionary $\mathbf{W}^{(l)}$ for each audio example l in the training set. The $\mathbf{W}^{(l)}$ are concatenated to build a global dictionary $\hat{\mathbf{W}} = [\mathbf{W}^{(1)}, ..., \mathbf{W}^{(L)}]$. The resulting global dictionary is too large and possibly very redundant, therefore 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 l is obtained by projecting its spectrogram $\mathbf{S}^{(l)}$ on $\underline{\mathbf{W}}$, followed by computing the average of the resulting projection matrix $\underline{\mathbf{H}}^{(l)} \in \mathbb{R}_+^{K_c\times T}$ over time.

V. Supervised dictionary learning

Supervised matrix factorization problems aim at finding decompositions that can, at the same time, provide good approximations of the data and are also adapted to address a target problem. For example, supervised decompositions have been applied to improve source separation [46], image denoising [23] or image classification tasks [47]. In our case, the goal is to make use of the labels in the decompositions to learn a dictionary that will help improving the classification performance. As mentioned previously, nonnegative factorizations are well suited for decomposing audio from multi-source environments. Therefore, we are particularly interested in having a supervised variant of NMF. Supervision has been introduced to NMF by using Fisher discriminant analysis [48], [49], adding a binary label matrix to the data [50] or by maximizing the margin between the projections [51]. In this work, we will focus on adapting the task driven dictionary learning (TDL) model introduced in [23] to our problem by applying it to nonnegative cases with a multinomial logistic regression classification scheme.

A. Task-driven dictionary learning model

The general idea of TDL is to group the dictionary learning and the training of the classifier in a joint optimization problem. Influenced by the classifier, the basis vectors are encouraged to explain the discriminative information in the data while keeping a low reconstruction cost. The TDL model first considers the optimal projections $\mathbf{h}^\star(\mathbf{v}, \mathbf{W})$ of the data point $\mathbf{v}$ on the dictionary $\mathbf{W}$. The projections are defined as solutions of the elastic-net problem [52] expressed as

$$\mathbf{h}^\star(\mathbf{v}, \mathbf{W}) = \min_{\mathbf{h}\in\mathbb{R}^K} \frac{1}{2}\|\mathbf{v} - \mathbf{W}\mathbf{h}\|_2^2 + \lambda_1\|\mathbf{h}\|_1 + \frac{\lambda_2}{2}\|\mathbf{h}\|_2^2; \tag{4}$$

where λ_1 and λ_2 are nonnegative regularization parameters. Given each data point $\mathbf{v}$ is associated with a label y in a fixed set of labels $\mathcal{Y}$, a classification loss $l_s(y, \mathbf{A}, \boldsymbol{h}^\star(\mathbf{v}, \mathbf{W}))$ is defined, where $\mathbf{A} \in \mathcal{A}$ are the parameters of the classifier. The TDL problem is then expressed as a joint minimization of the expected classification cost over $\mathbf{W}$ and $\mathbf{A}$:

$$\min_{\mathbf{W}\in\mathcal{W},\mathbf{A}\in\mathcal{A}} f(\mathbf{W}, \mathbf{A}) + \frac{\nu}{2}\|\mathbf{A}\|_2^2, \tag{5}$$

with

$$f(\mathbf{W}, \mathbf{A}) = \mathbb{E}_{y,\mathbf{v}}[l_s(y, \mathbf{A}, \mathbf{h}^\star(\mathbf{v}, \mathbf{W})]. \tag{6}$$

Here, $\mathcal{W}$ is defined as the set of dictionaries containing unit l_2-norm basis vectors and ν is a regularization parameter on the classifier's parameters to prevent over-fitting. The problem in equation (6) is optimized with stochastic gradient descent in [23]. After randomly drawing a data point $\mathbf{v}$, the optimal projection $\mathbf{h}^\star(\mathbf{v}, \mathbf{W})$ is first computed. Then, the classifier parameters $\mathbf{A}$ and the dictionary $\mathbf{W}$ are successively updated by projected gradient. The main steps of the original algorithm are presented in Algorithm 1. Here, I denotes the number of iterations, one iteration corresponds to an update of $\mathbf{A}$ and $\mathbf{W}$ with respect to one data point. The gradient of the classification cost with respect to the dictionary $\mathbf{W}$ is written as $\nabla_{\mathbf{W}} l_s(y, \mathbf{A}, \mathbf{h}^\star)$ and ρ is the projected gradient step size. The operation $\Pi_{\mathcal{W}}$ is the projection on $\mathcal{W}$, the set of dictionaries with unit ℓ_2 norm basis vectors. We refer the reader to [23] for a more complete description of the model.

Algorithm 1 Original stochastic gradient descent algorithm for the TDL model

Require: $\mathbf{V}, \mathbf{W} \in \mathcal{W}, \mathbf{A} \in \mathcal{A}, \lambda_1, \lambda_2, \nu, I, \rho$
  for $i = 1$ to I **do**
    Draw a random data point $\mathbf{v}$ and its label y
    Compute $\mathbf{h}^\star = \mathbf{h}^\star(\mathbf{v}, \mathbf{W})$ solution of Eq (4)
    $\mathbf{A} \leftarrow \Pi_{\mathcal{A}}[\mathbf{A} - \rho(\nabla_{\mathbf{A}} l_s(y, \mathbf{A}, \mathbf{h}^\star) + \nu\mathbf{A})]$
    Compute $\nabla_{\mathbf{W}} l_s(y, \mathbf{A}, \boldsymbol{h}^\star)$ as in [23]
    $\mathbf{W} \leftarrow \Pi_{\mathcal{W}}[\mathbf{W} - \rho\nabla_{\mathbf{W}} l_s(y, \mathbf{A}, \mathbf{h}^\star)]$
  end for
  return $\mathbf{W}, \mathbf{A}$

B. New formulation

In this section we present our modifications of the original TDL model in order to address the specificities of the task. In particular, it needs to classify averaged projections, to learn nonnegative dictionaries and to consider the multinomial logistic regression as the classifier in the model.

1) Classifying averaged projections: The original formulation supposes each projection $\mathbf{h}^\star(\mathbf{v}, \mathbf{W})$ is classified individually. Instead, we want to classify the mean of the projections of the data points $\mathbf{v}^{(l)}$ belonging to the sound example $l \in [\![1, L]\!]$ with $\mathbf{V}^{(l)} = [\mathbf{v}_0^{(l)}, ..., \mathbf{v}_{M-1}^{(l)}]$ (see section III-C). We define $\mathbf{h}^{(l)}$ as the averaged projection of $\mathbf{V}^{(l)}$ on the dictionary, where $\hat{\mathbf{h}}^{(l)} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{h}^\star(\mathbf{v}_m^{(l)}, \mathbf{W})$. Thus, the expected classification cost is now expressed as:

$$f(\mathbf{W}, \mathbf{A}) = \mathbb{E}_{y,\mathbf{V}}[l_s(y, \mathbf{A}, \hat{\mathbf{h}}^{(l)})]. \tag{7}$$

This alternate formulation only slightly modifies the gradients of $f(\mathbf{W}, \mathbf{A})$ with respect to $\mathbf{W}$ and $\mathbf{A}$.

2) Multinomial logistic regression: In order to stay consistent with the rest of the compared methods, we propose to use a multinomial logistic regression [37] as the classifier in the model. Compared to the two-class formulation chosen in [23], it has the advantage of learning a common dictionary for all the labels instead of relying on a one-versus-all strategy, without having to tune more parameters. The resulting supervised matrix factorization model can then be more clearly confronted to the *unsupervised matrix factorization plus multinomial logistic regression* feature learning systems.

3) A nonnegative version of the model: Motivated by the success of nonnegative decompositions for audio classification tasks, we believe the TDL model could benefit from having a nonnegative formulation. Although it was mentioned as possible by the authors in [23], it has not been applied and will lead to improved results in our case. For the nonnegative TDL, the projections $\mathbf{h}^\star$ are required to be nonnegative, resulting in equation (4) becoming a nonnegative version of the elastic-net problem. Moreover, the set $\mathcal{W}$ is now the set of dictionaries with unit ℓ_2-norm basis vectors having only nonnegative coefficients.

4) Modified algorithm: We propose a novel modification of the original algorithm presented in [23]. The modified algorithm is presented in Algorithm 2. It alternates between an update of the classifier using the full set of projections and an update of the dictionary by stochastic projected gradient on a full epoch. [1] The multinomial logistic regression parameters $\mathbf{A}$ are no longer updated with stochastic gradient descent but with one iteration of the L-BFGS algorithm [53] using the full set of averaged projections in $\hat{\mathbf{H}}^\star(\mathbf{V}, \mathbf{W}) = [\hat{\mathbf{h}}^{(1)}, ..., \hat{\mathbf{h}}^{(L)}]$. Therefore, the gradient step ρ only impacts the dictionary update step, making the results less sensible to its tuning. Finally, the dictionary update follows the same steps as in Algorithm 1. but is done separately on a full epoch at each iteration, after the classifier has been updated. The projection $\Pi_{\mathcal{W}}$ on $\mathcal{W}$ guaranties the dictionary only has nonnegative coefficients. In practice, we always observed a decrease in the objective function, *i.e.* the regularized multinomial logistic loss, when using the Algorithm 2. An example of the evolution of the values of the objective function on two examples is given in Fig. 3.

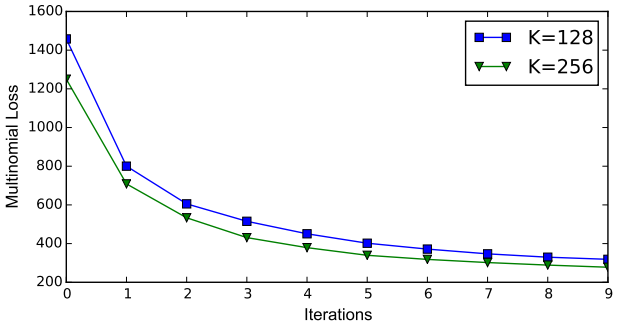

Fig. 3: Evolution of the objective function for the nonnegative TDL model through iterations with Algorithm 2 on two dictionary sizes K. The two examples are from the first training set of the LITIS dataset, following the same experimental setting as described in Section VI-H.

Algorithm 2 Modified algorithm for the nonnegative TDL model

Require: $\mathbf{V}, \mathbf{W} \in \mathcal{W}, \mathbf{A} \in \mathcal{A}, \lambda_1, \lambda_2, \nu, I, \rho$
  for $i = 1$ to I **do**
    $\forall l \in [\![1, L]\!]$ compute $\hat{\mathbf{h}}^{(l)} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{h}^\star(\mathbf{v}_m^{(l)}, \mathbf{W})$
    Set $\hat{\mathbf{H}}^\star(\mathbf{V}, \mathbf{W}) = [\hat{\mathbf{h}}^{(1)}, ..., \hat{\mathbf{h}}^{(L)}]$
    Update $\mathbf{A}$ with one iteration of L-BFGS
    for $n = 1$ to N **do**
      Draw a random data point $\mathbf{v}$ and its label y
      Compute $\mathbf{h}^\star = \mathbf{h}^\star(\mathbf{v}, \mathbf{W})$
      Compute $\nabla_{\mathbf{W}} l_s(y, \mathbf{A}, \mathbf{h}^\star)$ as in [23]
      $\mathbf{W} \leftarrow \Pi_{\mathcal{W}}[\mathbf{W} - \rho\nabla_{\mathbf{W}} l_s(y, \mathbf{A}, \mathbf{h}^\star)]$
    end for
  end for
  return $\mathbf{W}, \mathbf{A}$

[1] An epoch is defined as a full pass through a random permutation of the training set resulting in the number of iterations I being the number of passes through the data.

VI. Experimental evaluation

A. The acoustic scene classification datasets

We evaluate the different feature learning methods on two of the largest publicly available ASC datasets.

a) LITIS Rouen Dataset : The LITIS dataset [12] is to our knowledge the largest ASC datasets publicly available. It contains 25 hours 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*. The authors provided 20 training-test splits where 80% of the examples are kept for training and the other 20 % for testing. We keep the same splits to guaranty comparable results to previous publications on the dataset.

b) DCASE 2016 Scene dataset : The DCASE dataset corresponds to the development dataset provided for the scene classification task of the 2016 edition of the IEEE AASP DCASE challenge [7]. It contains 10 hours of urban audio scenes recorded with an electret binaural microphone, split into 1170 examples of 30 s without overlap forming 15 different classes. It has some labels in common with the Litis Dataset. We use the same 4 training-test splits provided by the authors, where 25% of the examples are kept for testing. Special care has been put into grouping recordings from similar locations in the same fold. Moreover, despite being smaller than the LITIS dataset, the set of recordings for a given label seems to contain more variability. The only available results on this dataset are from the baseline system provided for the challenge. The baseline is a GMM model trained on MFCC features extracted from each scene.

B. Time-frequency representation extraction

The CQT were extracted with the *YAAFE* toolbox [54] after rescaling the signals in $[-1, 1]$. For the LITIS dataset, the CQT has 12 bands per octave from 5 to 11025 Hz resulting in $P = 134$ frequency bands. For the DCASE dataset, since the recordings are of better quality, the frequency range for the CQT extraction goes from 5 to 22050 Hz resulting in $P = 146$ frequency bands. For both datasets, the examples are 30-second long, the CQT are extracted using 60-ms windows without overlap resulting in $T = 500$ time frames. Some ASC systems benefited from using shorter windows or a higher number of frequency bands, whereas in our case, increasing those values has not provided any notable increase in performance. In order to build the data matrix (see also Section III-C and Fig. 1), we use 2-s long slices leading to $M = 15$ slices per example. In most cases, keeping shorter slices did not help to significantly improve the performance. We found $M = 15$ slices to be a good compromise between complexity and performance.

C. Evaluation protocol

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 with a square root compression. The data is scaled to have unit variance and centered for the PCA variants only. The obtained projections are also scaled and standardized before classification. The classifier is a linear multi-class logistic regression trained with scikit-learn [55]. In order to compute the results for each training-test split we use the average F1 score over all classes. The final F1 score is its average value over all splits. Finally, statistical significance is asserted via a cross-validated student t-test ($p < 0.05$). A summary of the matrix factorization variants compared is presented in Table I. The results for the state-of-the art hand-crafted features on both datasets are given in Table II in order to discuss the interest of the feature learning methods presented for the task. Some of the results are presented only in terms of precision as they were in the original publication [12].

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

TABLE I: Summary of the variants tested for PCA and NMF. Max K specifies the highest dictionary size tested for each technique. The different variants are marked with an "o" if they are presented in the experimental evaluation and with an "×" if they are not.

DCASE 2016	F1 score	
MFCC + GMM baseline [7]	70.9	
LITIS Rouen	Precision	F1 score
MFCC + RQA [12]	86.0	-
HOG [13]	91.2	90.5
HOG+SPD [13]	**93.3**	**92.8**

TABLE II: Results with the best hand-crafted features on both datasets.

D. Basic matrix factorizations

We first focus on basic matrix factorizations techniques. F1 scores are presented in Table III. For PCA, the dictionary size is limited to the dimension of the feature space (the number of frequency bands P). Therefore the results are presented for $K = 128$. The performance for NMF on the LITIS dataset slightly differs from [21] where the projection matrix $\mathbf{H}$ was jointly learned with $\mathbf{W}$. The approach used here is to, first learn the dictionary during the training stage, then separately fully reproject the training data on the fixed dictionary. This leads to better results for higher values of K. The NMF problem does not have a unique solution and the estimated model is known to be sensible to initialization. Therefore NMF was tested on 5 different random initializations, where we kept the one providing the lowest reconstruction cost on the training data.

	DCASE 2016			LITIS Rouen		
	K=128	K=256	K=512	K=128	K=256	K=512
PCA	73.7 ± 3	-	-	89.8 ± 2	-	-
NMF $\beta = 2$	78.5 ± 5	79.6 ± 4	**80.1** ± 4	87.7 ± 2	89.8 ± 1	90.8 ± 1
NMF $\beta = 1$	79.3 ± 5	79.7 ± 3	79.5 ± 4	88.8 ± 1	90.1 ± 1	**91.5** ± 1
NMF $\beta = 0$	79.5 ± 4	79.5 ± 4	**80.1** ± 4	88.9 ± 1	90.4 ± 1	**91.1** ± 1

TABLE III: F1 scores and standard deviations for PCA and NMF on different dictionary sizes K

As shown in Table III, for both datasets better performance is obtained with the NMF decompositions, which confirms the usefulness of the nonnegativity constraint for acoustic scenes. For NMF, the choice of the β-divergence has almost no impact on the results. Indeed, applying the square root compression on the data compensates for some of the drawbacks of the Euclidean distance (β=2) by diminishing the differences in scales between data points. The performance of NMF on the DCASE dataset shows that even with a simple matrix factorization approach, we can obtain better performance than the typical MFCC + GMM method. We can also observe that the performance of NMF is improved when taking higher values of K on the LITIS dataset, whereas it is not clear for the DCASE dataset. It can be due to the LITIS dataset containing 3 times more training data, thus needing more basis events to describe the whole data matrix. In that case, the NMF system is still sufficient to outperform the results obtained with MFCC + RQA (Table II), but not yet those obtained with more complex image features.

E. Influence of sparsity

In this section we propose to study the effects of adding sparsity constraints to the PCA and NMF-based feature learning system. The F1 scores are presented in Table IV. The λ parameter is the regularization parameter controlling the influence of the ℓ_1-norm penalty in equation (2). For Sparse NMF the results are only presented when obtained with the Euclidean distance ($\beta = 2$). In most cases, the Euclidean distance showed a slight increase in performance compared to other divergences without any significant improvements.

As expected, the results in Table IV show that sparsity helps in most cases for both dataset, except for $K = 128$ on the LITIS one, where adding sparsity to the activation matrix in the PCA and NMF decompositions always decreases the results. This can be attributed to differences in size between the two datasets. While the sparsity constraint generally improves performance for the DCASE dataset, the impact is particularly clear on the LITIS dataset where a significant improvement is obtained for higher values of K. Moreover, we reach a 94.1% F1 score obtained with $\lambda = 0.25$ and $K = 1024$, which is a first significant improvement over the 92.8% F1 score obtained with the best image features on the dataset [13].

F. Influence of non-linearity

In this section we study the influence of using kernel-based extensions of the PCA. A Gaussian kernel was used for the KPCA. The σ parameter for the Gaussian kernel function is tuned using cross-validation on a sub-set of the data. The results are reported in Table V. Unfortunately, for KNMF, the computation time gets prohibitive for high values of K, preventing us from providing results. The performance of KNMF for lower values of K were reported in [21], the F1 scores obtained were significantly below the regular NMF for similar dictionary sizes. 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 than NMF when $N >> P$.

	DCASE 2016		LITIS Rouen	
	K=512	K=1024	K=512	K=1024
Kernel PCA	79.7 ± 3	79.5 ± 3	94.3 ± 1	95.6 ± 1

TABLE V: F1 scores for Kernel PCA on different dictionary sizes K

Unlike regular PCA, for Kernel PCA, the dictionary size is not limited to the dimension of the input feature space since we decompose the data in the transformed space. By using KPCA with 1024 components, we obtain a 95.6% F1 score on the LITIS dataset which significantly outperforms the previous results for PCA as well as the best spectral image features. KPCA also improves the performance compared to PCA for the DCASE dataset but does not manage to outperform the regular NMF. In fact, the gap of performance between regular PCA and NMF is larger for the DCASE data, suggesting that even the kernel formulation of PCA does not compensate for the benefits of the nonnegative decomposition.

G. Convolutive NMF

Since the convolutive NMF is applied on full spectrograms, the feature learning architecture is different from the previous experiments as described in Section IV-D. The spectrograms are decomposed using 2D dictionary elements of $\tau = 4$ time frames (0.25 seconds) for the LITIS dataset and $\tau = 8$ time frames (0.5 seconds) for the DCASE dataset. Decomposing on longer slices did not provide better results. Each full duration spectrogram in the training set is approximated by a dictionary of 40 basis slices (for the DCASE) or 80 basis slices (for the LITIS). The results shown in Table VI are given for different number K_c of cluster centers obtained after applying the K-means to $\underline{\mathbf{W}}$ in order to reduce the size of the dictionary. The convolutive NMF is compared to the regular NMF using the same alternate feature learning architecture illustrated in Fig. 2. The regular NMF applied on this architecture is referred to as *NMF + clustering*. This method uses the regular NMF to

	DCASE 2016					LITIS Rouen				
Sparse PCA	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$
K=128	73.7 ± 3	77.5 ± 5	**79.7** ± 3	78.3 ± 3	76.8 ± 4	**90.0** ± 2	**90.0** ± 2	89.1 ± 2	82.6 ± 2	65.2 ± 4
Sparse NMF	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1$
K=128	**78.5** ± 5	78.1 ± 5	77.7 ± 4	**78.3** ± 3	77.0 ± 3	**88.5** ± 2	88.2 ± 2	88.0 ± 2	86.7 ± 2	87.1 ± 2
K=256	78.4 ± 5	**79.4** ± 5	**79.3** ± 4	**79.4** ± 4	78.6 ± 4	88.9 ± 2	**90.8** ± 1	**90.6** ± 1	90.1 ± 2	90.1 ± 2
K=512	80.4 ± 5	80.2 ± 4	**82.3** ± 5	80.8 ± 4	**81.7** ± 2	91.2 ± 1	92.0 ± 1	**93.3** ± 1	91.9 ± 1	91.1 ± 1
K=1024	81.4 ± 5	81.4 ± 5	**82.0** ± 4	81.4 ± 2	**82.1** ± 3	92.0 ± 1	93.1 ± 1	**94.1** ± 1	92.1 ± 1	91.8 ± 2

TABLE IV: F1 scores and standard deviation for Sparse NMF and Sparse PCA for different dictionary sizes K and sparsity constraints λ. The bold values denote the best score for each value of K and the underlined values highlight the best scores for all values of K.

learn a separate basis of 5 vectors on each 2-s spectrogram slice. Similarly to convolutive NMF, the concatenation of all basis vectors is processed by clustering 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.

First, the convolutive NMF appears to be well suited to address 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, the basis events being 2D slices, their temporal structure is also modeled by the dictionary. On both datasets, the results obtained with convolutive NMF are slightly better than with the *NMF + clustering* method. Similar observations have been made for speaker identification in noisy conditions [19]. In line with the sparse NMF and the kernel PCA, convolutive NMF also significantly improves the baseline reaching a 82.5% F1 score for the DCASE dataset and significantly outperforms the reference best spectral image features for the LITIS dataset with a 94.5% F1 score. However, even the best results obtained with convolutive NMF do not present any significant improvements compared to the best sparse NMF scores given in Table IV. This observation suggests that the way we build the data matrix in Section III-C offers a sufficient representation of acoustic scenes while being a lot more compact. Meaning that keeping only the averaged frequency information in slices of the spectrograms may be enough to learn good features for discriminating acoustic scenes.

H. Supervised dictionary learning

The results obtained when applying the task-driven dictionary learning model to perform supervised matrix factorization are presented in Table VII. We especially highlight the results obtained with our novel variant of the model in the nonnegative case (nonnegative TDL). The nonnegative TDL is also compared to results obtained with sparse NMF which can be seen as its unsupervised counterpart. The dictionaries for the TDL model are initialized using unsupervised sparse NMF for the nonnegative case and the dictionary learning function from the *spams* toolbox [56] in the general case. The weights of the classifier are initialized by applying multinomial logistic regression to the projections on the initialized dictionary. In the algorithm, the projections on the dictionary (corresponding to equation (4)) are computed using the *lasso* function from the *spams* toolbox [56]. Then, the classifier is updated using one iteration of the scikit-learn [55] implementation of the multinomial logistic regression with the L-BFGS solver. The model is trained over $I = 10$ iterations with a 0.001 initial gradient step for dictionary update. The decaying of the gradient steps over iterations follows the same heuristic as suggested in [23]. The ℓ_1 regularization parameter and the logistic regression's regularization ν were tuned on a development set of the first training fold, where λ_2 was set to 0, λ_1 was varied in the set $\{0.1, 0.25, 0.5\}$ and ν in the set $\{0.1, 1, 10\}$.

First, we can see from the results in Table VII that for all cases, using the nonnegative formulation of TDL helps improving the results compared to the more general model. It demonstrates that adding the nonnegative constraints to the model can be beneficial when applied to sound classification tasks, especially to decompose audio from multi-source environments like urban acoustic scenes. The performance without the nonnegative constraint is particularly low for the DCASE dataset, this could have been hinted at by the performance of PCA compared to regular NMF (see Table III). When comparing to sparse NMF, the results obtained with smaller dictionary sizes ($K = 128$ or $K = 256$) particularly stand out. Indeed, by learning basis vectors adapted to the classification task, the model is capable of factorizing the relevant information in much smaller dictionaries. The whole dataset can be well explained by only a small number of basis events, giving a more compact common representation. This may be particularly interesting when fast projections of new incoming data on the learned dictionary is needed. For example, with $K = 128$, the results increase from a 78.5 % F1 score to a 81.0 % F1 score for the DCASE dataset and from a 88.5 % F1 score to a 94.7 % F1 score on the LITIS dataset. On the two datasets, the results stop increasing at $K = 512$ and in both cases, they improve the best sparse NMF results. The proposed nonnegative TDL model succeeds in introducing supervision to the regular sparse NMF-based feature learning systems for ASC. It is beneficial both for improving performance over unsupervised decompositions and to learn smaller representations with good discriminative power.

I. Deep neural networks

DNN are the state-of-the-art approach in many classification tasks, and attempts have been made to apply them to ASC [32]. Therefore, we decided to compare the matrix factorization

	DCASE 2016			LITIS Rouen		
	K_c=256	K_c=512	K_c=1024	K_c=256	K_c=512	K_c=1024
NMF + *clustering*	76.1 ± 5	**79.6** ± 3	**79.9** ± 3	90.1 ± 2	92.2 ± 1	**93.7** ± 1
Convolutive NMF	77.7 ± 2	80.8 ± 2	**82.5** ± 2	90.5 ± 2	92.6 ± 1	**94.5** ± 1

TABLE VI: F1 scores and standard deviation for convolutive NMF and NMF with clustering for different dictionary sizes K_c.

	DCASE 2016				LITIS Rouen			
	K=128	K=256	K=512	K=1024	K=128	K=256	K=512	K=1024
Sparse NMF	78.5 ± 5	79.4 ± 4	82.3 ± 5	82.0 ± 4	88.5 ± 2	90.8 ± 1	93.3 ± 1	94.1 ± 1
TDL	77.0 ± 3	75.9 ± 3	73.9 ± 3	73.5 ± 3	94.0 ± 1	94.2 ± 1	94.3 ± 1	94.1 ± 1
Nonnegative TDL	81.0 ± 3	**83.1** ± 3	**83.3** ± 3	82.1 ± 3	94.7 ± 1	**95.7** ± 1	**96.0** ± 1	**95.8** ± 1
DNN	78.5 ± 5				**96.9** ± 1			

TABLE VII: F1 scores and standard deviations for the TDL model and nonnegative TDL model compared to the Sparse NMF results on different dictionary sizes K. The last row reports the F1 scores and standard deviations obtained with the DNN.

approaches described here to a feed-forward fully connected DNN used for feature extraction. The DNN is composed of three hidden layers, the first two layers contain 1500 elements when applied to the LITIS dataset and 500 for the DCASE. For both datasets, the last hidden layer is composed of 100 elements. This latter layer is used to extract features at runtime. During training, the targets of the DNN are the classes of the scene classification problem such that the network architecture can be summarized as follows: 134 x 1500 x 1500 x 100 x 19 for the LITIS dataset and 146 x 500 x 500 x 100 x 15 for the DCASE.

The DNN is trained with the Lasagne toolkit.[2] The DNN weights are initialized with Glorot weights sampled from a uniform distribution [57]. Rectified linear unit (ReLU) activations [58] are used in hidden layers and softmax activations are applied to the output layer. The objective function is a categorial cross-entropy that is suited to multinomial classification problems. The DNN is trained with the Adam algorithm [59] over 100 epochs with a constant 0.001 learning rate. In order to prevent over-fitting, dropout with probability 0.5 is applied to the hidden layers [60].

The results obtained with the DNN architectures described above are reported in Table VII. They are compared to those obtained with the nonnegative TDL which is the best performing approach of all the matrix factorization variants. The DNN improves the results compared to best matrix factorization variants for the LITIS Rouen dataset reaching a 96.9 % F1 score, although the difference is not significant with the 96.0% F1 score obtained with nonnegative TDL. On the other hand, for the DCASE dataset, DNN is outperformed by most of the matrix factorization methods. This can be attributed to the lack of training data for the DCASE dataset, where in the case of smaller training sets, decompositions such as NMF can be a good alternative to learn meaningful representations. Testing more complex architectures for the network lead to more over-fitting and consequently worst results. The higher intra-class variability due to the design of the dataset tends to make the appropriate discriminative information harder to learn leading to the supervised methods being more sensible to over-fitting.

[2]http://lasagne.readthedocs.io/

VII. Comparison to the state of the art

Table VIII summarizes some of the results obtained with the matrix factorization variants presented here and compares them to the state-of-the-art on the LITIS and DCASE datasets. In particular we include results from some of the best hand-crafted features [12], [13], other results obtained with DNN and a method which uses a combination of hand-crafted features with feature learning [35]. For the LITIS dataset, we highlight two other published methods of interest. The first is a deep learning approach from [32], where a feed-forward deep neural network is fit with MFCC combined with other low-level features which we refer to as DNN+MFCC. The second one is the work from [35] which uses a supervised combination of two probabilistic SVMs, one is fit with HOG features and the other with a bag-of-frame approach on the same features. The results will be given in mean accuracy over all folds, this being the measure used for most previous works.

	DCASE	LITIS
Feature-based methods		
MFCC + GMM baseline	72.5	-
HOG+SPD [13]	-	93.4
Feature learning-based methods		
Kernel PCA $K = 1024$	80.2	**96.0**
Sparse NMF $K = 1024$	**82.7**	94.6
Nonnegative TDL $K = 512$	**83.8**	**96.4**
Ye et al.'s [35]'s	-	96.1
Deep learning methods		
DNN +MFCC [32]	-	92.2
Our DNN	79.0	**97.1**

TABLE VIII: Accuracy scores of the best feature learning variants presented here compared to the state-of-the-art results on both datasets.

The performance of the spectral image features highlighted that important information could be extracted from the time-frequency images and that, averaging that information over time, can be a good way of characterizing acoustic scenes. To follow this trend, Ye et al. combined hand-crafted features with a feature learning approach also on spectrogram images which leads to a 96.1 % accuracy score [35]. The Kernel PCA

and Nonnegative TDL variants manage to reach similar results without using hand-crafted features with a 96% and 96.4% accuracy scores respectively. In [35], the feature learning part alone gives a 94.5% accuracy, suggesting that slightly better results could be obtained if the best variants presented were combined with some of the mentioned spectral image features. The two deep learning approaches compared mainly differ by the input representation given to the DNN. The results when fitting a DNN on MFCC and more traditional features are significantly lower than those obtained with the feature learning approaches. Meanwhile, when a DNN is trained using directly the preprocessed versions of the time-frequency images (data matrix $\mathbf{V}$ assembled as described in Section III-C), that is the same input representation as the matrix factorization methods, we reach a 97.1 % accuracy, outperforming the reference methods. This observation, combined with the success of image features for ASC, show that the task benefits from learning directly with the spectrograms unlike the more popular audio features such as MFCC, that struggle to give good performance even when given to learn more complex classifiers like DNNs.

VIII. Conclusion

In this paper we studied and compared different matrix factorization methods to perform feature learning for ASC. In order to focus on the impact of the learning approach, the methods presented were confronted on the same input representation of the data, and exploited with a simple linear classifier. This allowed us to emphasize the benefit of nonnegative constraints in the decompositions with the NMF variants, which helped improving the results in both the unsupervised and supervised settings. We also proposed a novel supervised dictionary learning model for nonnegative decompositions. By jointly optimizing the factorization and classification problems, the resulting nonnegative TDL decomposition is capable of providing better dictionaries that are more compact than their unsupervised counterparts. Finally, we confronted a DNN to supervised dictionary learning model. The nonnegative TDL model obtained similar or significantly better performance than the DNN

We evaluated the feature learning methods on two of the largest available ASC datasets. On both of them, the feature learning approaches presented here significantly improved results to the previous best compared hand crafted features. Engineering hand crafted features is still one of the more popular approaches to ASC. While their study is interesting to highlight the specificities of the task, they lack flexibility and often only describe certain aspects of the scenes. Instead, in this study, we have shown that a common representation of the data could be learned with matrix factorization, with the advantage of automatically adapting to the data at hand. In our future research, motivated by the good performance of the nonnegative TDL model for ASC, we plan to adapt the model to other sound classification tasks such as acoustic event detection. Indeed, the model has the advantage of being able to adapt to a specific target problem and could be linked to more complex or dynamical classifiers.

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] D. Wang and G. J. Brown, *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. IEEE Press, 2006.

[3] R. Serizel, V. Bisot, S. Essid, and G. Richard, "Machine listening techniques as a complement to video image analysis in forensics," in *Proc. International Conference on Image Processing*, 2016, pp. 5470–5474.

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

[5] D. P. W. Ellis and K. Lee, "Minimal-impact audio-based personal archives," in *Proc. ACM workshop on Continuous archival and retrieval of personal experiences*, 2004.

[6] D. Giannoulis, D. Stowell, E. Benetos, M. Rossignol, M. Lagrange, and M. D. Plumbley, "A database and challenge for acoustic scene classification and event detection," in *Proc. European Signal Processing Conference*, 2013.

[7] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *Proc. European Signal Processing Conference*, 2016.

[8] V. Peltonen, J. Tuomi, A. Klapuri, J. Huopaniemi, and T. Sorsa, "Computational auditory scene recognition," in *Proc. International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 2002, pp. II–1941.

[9] 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.

[10] V. T. Peltonen, A. J. Eronen, M. P. Parviainen, and A. P. Klapuri, "Recognition of everyday auditory scenes: Potentials, latencies and cues," in *Proc. 110th Audio Eng. Soc. Convention*, 2001.

[11] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Audio context recognition using audio event histograms," in *Proc. European Signal Processing Conference*, 2010, pp. 1272–1276.

[12] 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*, vol. 23, no. 1, pp. 142–153, 2015.

[13] V. Bisot, S. Essid, and G. Richard, "Hog and subband power distribution image features for acoustic scene classification," in *Proc. European Signal Processing Conference*, 2015, pp. 719–723.

[14] J. T. Geiger, B. Schuller, and G. Rigoll, "Large-scale audio feature extraction and svm for acoustic scene classification," in *Proc. Workshop on Applications of Signal Processing to Audio and Acoustics*, 2013.

[15] G. Roma, W. Nogueira, and P. Herrera, "Recurrence quantification analysis features for environmental sound recognition," in *Proc. Workshop on Applications of Signal Processing to Audio and Acoustics*, 2013.

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

[17] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[18] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 171–175.

[19] A. Hurmalainen, R. Saeidi, and T. Virtanen, "Noise robust speaker recognition with convolutive sparse coding," in *in Proc. Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[20] E. Benetos, M. Kotti, and C. Kotropoulos, "Musical instrument classification using non-negative matrix factorization algorithms," in *Proc. International Symposium on Circuits and Systems*, 2006.

[21] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Acoustic scene classification with matrix factorization for unsupervised feature learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 6445–6449.

[22] 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.

[23] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 791–804, 2012.

[24] 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.

[25] 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.

[26] 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.

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

[28] D. Battaglino, L. Lepauloux, L. Pilati, and N. Evansi, "Acoustic context recognition using local binary pattern codebooks," *Proc. Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 1–5, 2015.

[29] J. Krijnders and G. A. T. Holt, "A tone-fit feature representation for scene classification," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events*, 2013.

[30] D. Li, J. Tam, and D. Toub, "Auditory scene classification using machine learning techniques," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events*, 2013.

[31] E. Olivetti, "The wonders of the normalized compression dissimilarity representation," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events*, 2013.

[32] Y. Petetin, C. Laroche, and A. Mayoue, "Deep neural networks for audio scene recognition," in *Proc. European Signal Processing Conference*, 2015, pp. 125–129.

[33] B. Cauchi, "Non-negative matrix factorization applied to auditory scene classification," Master's thesis, ATIAM (UPMC / IRCAM / TELECOM ParisTech), 2011.

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

[35] J. Ye, T. Kobayashi, M. Murakawa, and T. Higuchi, "Acoustic scene classification based on sound textures and events," in *Proc. Annual Conference on Multimedia*, 2015, pp. 1291–1294.

[36] 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.

[37] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2001, vol. 1.

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

[39] N. Gillis, "The why and how of nonnegative matrix factorization," *Regularization, Optimization, Kernels, and Support Vector Machines*, vol. 12, no. 257, 2014.

[40] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

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

[42] J. L. Roux, F. J. Weninger, and J. R. Hershey, "Sparse nmf–half-baked or well done?" Mitsubishi Electric Research Labs (MERL), Tech. Rep. TR2015-023, 2015.

[43] 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.

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

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

[46] S. Wisdom, J. R. Hershey, J. L. Roux, and S. Watanabe, "Deep unfolding for multichannel source separation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 121–125.

[47] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. Bach, "Supervised dictionary learning," in *Proc. Advances in neural information processing systems*, 2009, pp. 1033–1040.

[48] Y. Wang, Y. Jia, C. Hu, and M. Turk, "Fisher non-negative matrix factorization for learning local features," in *Proc. Asian Conference on Computer Vision*, 2004, pp. 27–30.

[49] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas, "Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 683–695, 2006.

[50] A. Mesaros, T. Heittola, O. Dikmen, and T. Virtanen, "Sound event detection in real life recordings using coupled matrix factorization of spectral representations and class activity annotations," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 151–155.

[51] B. V. Kumar, I. Kotsia, and I. Patras, "Max-margin non-negative matrix factorization," *Image and Vision Computing*, vol. 30, no. 4, pp. 279–291, 2012.

[52] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[53] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[54] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "Yaafe, an easy to use and efficient audio feature extraction software." in *Proc. International Society for Music Information Retrieval*, 2010, pp. 441–446.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[56] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[57] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. International conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[58] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE International Conference on Computer Vision*, 2009, pp. 2146–2153.

[59] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[60] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.