Published version: IEEE Transactions on Audio, Speech, and Language Processing, 18(8):1955-1967, Nov. 2010. doi: 10.1109/TASL.2010.2040784

Non-negative Tensor Factorization Applied to Music Genre Classification

Emmanouil Benetos and Constantine Kotropoulos, *Senior Member, IEEE*

***Abstract*—Music genre classification techniques are typically applied to the data matrix whose columns are the feature vectors extracted from music recordings. In this paper, a feature vector is extracted using a texture window of one sec, which enables the representation of any 30 sec long music recording as a time sequence of feature vectors, thus yielding a feature matrix. Consequently, by stacking the feature matrices associated to any dataset recordings, a tensor is created, a fact which necessitates studying music genre classification using tensors. First, a novel algorithm for non-negative tensor factorization (NTF) is derived that extends the non-negative matrix factorization. Several variants of the NTF algorithm emerge by employing different cost functions from the class of Bregman divergences. Second, a novel supervised NTF classifier is proposed, which trains a basis for each class separately and employs basis orthogonalization. A variety of spectral, temporal, perceptual, energy, and pitch descriptors is extracted from 1000 recordings of the GTZAN dataset, which are distributed across 10 genre classes. The NTF classifier performance is compared against that of the multilayer perceptron and the support vector machines by applying a stratified 10-fold cross validation. A genre classification accuracy of 78.9% is reported for the NTF classifier demonstrating the superiority of the aforementioned multilinear classifier over several data matrix-based state-of-the-art classifiers.**

***Index Terms*—Non-negative tensor factorization, Bregman divergences, Music genre classification.**

I. INTRODUCTION

CURRENT advances in multimedia data management and retrieval have enabled the creation, distribution, and availability of vast amounts of music data including new content as well as digitized one from analog archives. Aided by the growth of the internet, these databases have become highly popular for personal as well as commercial use (e.g. online music retailers or digital libraries). Accordingly, the demand for tools to analyze and retrieve music content has emerged, leading to flourishing music information retrieval (MIR) research.

Music genres are the most popular music content descriptors, since they are employed by both the users and the music industry [1]. One may argue that the genres summarize music recordings based on some common perceptual characteristics [2]. However, music genres have not yet been precisely defined, because they are primarily determined by users' taste and may be culturally dependent. Not to mention that more than one music genres may be associated with a certain recording. Accordingly, the creation of a universal genre taxonomy remains still infeasible. Automatic genre classification techniques classify recordings into distinguishable genres by extracting relevant features and employing pattern recognition algorithms [3]. The accuracy of such genre classification techniques often exceeds that reported for humans with moderate music training [4]. However, the research on automatic genre classification appears to have reached a local maximum recently due to the lack of carefully annotated music corpora with ground truth [5].

Most genre classification approaches represent each music recording by a feature vector and consequently employ pattern recognition algorithms in order to perform classification. In this paper, each recording is represented by a time sequence comprising feature vectors extracted every one sec, thus forming a feature matrix. Starting with a comprehensive set of features measuring spectral, temporal, perceptual, energy, and pitch characteristics of the recordings, feature selection is applied next by using a branch-and-bound search strategy in order to determine the subset of the most discriminative features with respect to the ratio of the inter-class dispersion over the intra-class dispersion [9] and keep the number of the features to be processed into a manageable size. By stacking the feature matrices associated to recordings, a tensor is created, which provides a more detailed representation of music characteristics. Tensors are considered as extensions of matrices or vectors [6]–[8]. A novel non-negative tensor factorization (NTF) algorithm is proposed whose roots are traced back in the non-negative matrix factorization (NMF). The algorithm is able to decompose a tensor in Kruskal format [8]. That is, to decompose a tensor into a sum of elementary rank-1 tensors. The algorithm can employ several cost functions, which belong to the class of Bregman divergences [10]. The Bregman divergences have previously been used to solve the non-negative matrix approximation problem [11]. Details on the derivation of the tensor element update equations are provided and the computational cost of the algorithm is estimated. In addition, a novel supervised classifier based on the NTF is proposed, which trains a basis for each class separately and employs basis orthogonalization. The proposed classifier extends a similar classifier that was based on the NMF [34]. Preliminary results for music genre classification using the NTF classifier with the Frobenius norm as cost function were reported in [33]. Here, starting from a larger feature set than that used in [33], results are reported on extended experiments performed on the GTZAN database, which contains 1000 music recordings covering

E. Benetos is currently with the Dept. of Electronic Engineering, Queen Mary University of London, Mile End Road, London, E1 4NS, United Kingdom. He was with the Dept. of Informatics, Aristotle University of Thessaloniki, Box 451, Thessaloniki 54124, Greece. C. Kotropoulos is with the Dept. of Informatics, Aristotle University of Thessaloniki, Box 451, Thessaloniki 54124, Greece. E-mail: emmanouil.benetos@elec.qmul.ac.uk, costas@aiia.csd.auth.gr

10 music genre classes [12]. Several variants of the NTF classifier employing different feature subset sizes are tested and their music genre classification accuracy is measured using a stratified 10-fold cross-validation. For comparison purposes, support vector machines and multilayer perceptrons are also tested on the same database. In addition, experiments are performed using the features extracted by the MARSYAS platform [39]. An average genre classification accuracy of 78.9% with standard deviation equal to 2.6% is reported for the NTF classifier, when the Frobenius norm is utilized with a subset of 80 features. The aforementioned accuracy places the proposed NTF classifier within the most performing state-of-the-art genre classification methods. The superiority of the NTF classifier against the state-of-the-art data matrix-based classifiers is also demonstrated. Such results motivate further research using tensorial representations into audio processing applications.

The outline of the paper is as follows. Related work on automatic music genre classification is discussed in Section II. Section III details the proposed NTF method and establishes links with the NMF as well as other methods proposed for the NTF. In this section, a supervised classifier based on the proposed NTF is also described. Section IV briefly presents the dataset used, the feature set employed in the experiments, and thoroughly assesses the music genre classification accuracy of the proposed NTF classifier against that of state-of-the-art classifiers. Conclusions are drawn and future directions are indicated in Section V.

II. Related Work

Several benchmark datasets have been collected making the performance of the various music genre classification approaches comparable. Such benchmark datasets are listed in Table I along with the best accuracy reported for state-of-the-art classifiers in chronological order. The GTZAN dataset was introduced by Tzanetakis and Cook [12]. It contains 1000 audio recordings split into 10 genre classes. The parameters of a Gaussian mixture model (GMM) classifier were estimated by the iterative expectation-maximization (EM) algorithm in [12]. A 61% correct classification was reported for timbre, rhythmic, and pitch features. The same dataset was used by Li et al. who employed support vector machines (SVMs) and linear discriminant analysis (LDA) for classification [13]. The Daubechies wavelet coefficient histograms were used as features and the reported classification accuracy of the SVM classifier reached 78.5%. Lidy and Rauber employed a pairwise SVM classifier applied to the GTZAN dataset [3]. The extracted features include rhythm patterns, a statistical spectrum descriptor, and rhythm histogram features. The reported best classification accuracy was 74.9%. Bergstra et al. tested the mel-frequency cepstral coefficients (MFCCs), the fast Fourier Transform coefficients, the linear prediction coefficients (LPCs), and the zero-crossing rate (ZCR) on the GTZAN dataset [14] and reported a classification accuracy reaching 82.5% for the AdaBoost meta-classifier. It should be noted, however, that the classification accuracy in [14] was measured without cross-validation. Holzapfel and Stylianou also employed the same dataset. By utilizing a spectral basis derived by the NMF, that is fed to a GMM classifier, they obtained a 74.0% classification accuracy [15].

TABLE I
Classification accuracy (in %) of several music genre classifiers in chronological order. The highest accuracy is shown in boldface.

Reference	Dataset	Classifier	Best Accuracy
[12]	GTZAN	GMM	61.0
[13]	GTZAN	SVM - LDA	78.5
[3]	GTZAN	SVM	74.9
[14]	GTZAN	AdaBoost	**82.5**
[15]	GTZAN	GMM	74.0
[17]	MIREX 2004	NN - GMM	82.3
[3]	MIREX 2004	SVM	70.4
[15]	MIREX 2004	GMM	**83.5**

Another collection used extensively is the MIREX 2004 dataset, released for the MIREX genre and rhythm classification contests [16]. The MIREX 2004 genre dataset contains 1458 recordings belonging into 6 genre classes. The best accuracy (i.e. 83.5%) was reported by Holzapfel and Stylianou [15]. Pampalk et al. used the nearest neighbor (NN) classifier with GMMs and obtained an accuracy of 82.3% [17]. In our experiments, we are confined to the GTZAN dataset, because it contains more genre classes than the MIREX 2004 one, thus being a more comprehensive dataset for genre classification.

Other notable music genre classification approaches include that of Burred and Lerch, who proposed a 3-level music genre taxonomy covering 13 genres [18]. In addition, 3 speech classes, and one class for background noise were also considered. A dataset was created containing 50 recordings for each genre. Timbral and rhythmic features were extracted along with MPEG-7 audio descriptors. A GMM classifier reached classification accuracy of 59.76% for all classes. In 2005, Meng et al. created two datasets for genre classification [19]. The first dataset contains 100 recordings from 5 genres and the second dataset 354 music samples from the amazon.com database. Short, medium, and long-time features were extracted, and two classifiers were tested. The first classifier was a single-layer neural network and the second one was a Gaussian classifier that employs full covariance matrices. The reported best accuracy was 95% on the first dataset and about 68% on the second dataset. More recently, Barbedo and Lopes proposed a 4-level hierarchical genre taxonomy covering 29 music genres [20]. Several timbre features were selected and a classification procedure was developed that uses pairwise genre comparison. Overall, a genre classification accuracy of 61% at the lowest genre level was reported. Finally, Cataltepe et al. employed 225 MIDI music pieces covering 9 genre classes [21]. Timbral, rhythmic, and pitch content features were extracted, and the recordings were classified using a 10-nearest neighbor classifier (10-NN) or a normalized compression distance classifier. Using a combination of the aforementioned classifiers, a genre classification accuracy of 62% was reported.

III. Non-negative Tensor Factorization

In this section, a novel non-negative tensor factorization (NTF) technique is developed. First, the non-negative matrix factorization (NMF) is briefly discussed, because the NTF could be treated as a high-order generalization of the NMF for tensorial data. Some definitions from tensor algebra follow and the motivation for using tensors is described. Previous NTF algorithms are reviewed next and the proposed one is detailed. Contrary to previous approaches, the proposed NTF algorithm is not limited to 3rd order tensors, but can be applied to nth ($n > 3$) order tensors. In addition, the algorithm can be formulated using a variety of objective functions. Obviously, its use is not restricted to audio processing only. Finally, a novel supervised classifier based on NTF for 3rd order tensors is proposed, which performs separate training for each class and employs basis orthogonalization.

Throughout the paper, tensors are denoted by boldface Euler script calligraphic letters (e.g. $\boldsymbol{\mathcal{A}}$), matrices are denoted by uppercase boldface letters (e.g. $\mathbf{U}$), and vectors are denoted by lowercase boldface letters (e.g. $\mathbf{u}$). The elements of all the aforementioned mathematical structures are denoted by lowercase letters indexed by one or more indices. For example, the elements of matrix $\mathbf{U}$ are denoted as $u_{i_1 i_2}$. Let $\mathbb{R}$ and $\mathbb{Z}$ be the sets of real and integer numbers, respectively.

A. Non-negative Matrix Factorization

Subspace analysis seeks low dimensional structures of patterns within high dimensional spaces. NMF is a subspace method able to obtain a parts-based representation of objects by imposing non-negative constraints. It was first introduced as positive matrix factorization by Paatero et al. [22] and was re-termed as NMF by Lee and Seung [23]. The problem addressed by the NMF is as follows. Given a non-negative real-valued data matrix $\mathbf{V} \in \mathbb{R}_+^{n \times m}$, find the non-negative matrix factors $\mathbf{W} \in \mathbb{R}_+^{n \times k}$ and $\mathbf{H} \in \mathbb{R}_+^{k \times m}$ so that

$$\mathbf{V} \approx \mathbf{W}\mathbf{H} = \sum_{i=1}^{k} \mathbf{w}_i \, \mathbf{h}_i^T \triangleq \sum_{i=1}^{k} \mathbf{w}_i \circ \mathbf{h}_i \tag{1}$$

where $\circ$ stands for the outer product. Obviously, $\mathbf{w}_i$ are the columns of $\mathbf{W}$ and $\mathbf{h}_i^T$ are the rows of $\mathbf{H}$. $\mathbf{W}$ contains the basis vectors, while the column vectors of $\mathbf{H}$ contain the weights needed to properly approximate the corresponding column vector of $\mathbf{V}$ as a linear combination of the column vectors of $\mathbf{W}$. Usually, k is chosen so that $(n+m)k < nm$, thus resulting in a compressed version of the original data matrix. To find the approximate factorization (1), a suitable objective function has to be minimized. Let $\mathbf{V} = [v_{ij}]$ and $\mathbf{W}\mathbf{H} \triangleq \mathbf{Y} = [y_{ij}]$. In [23], the generalized Kullback-Leibler (KL) divergence between $\mathbf{V}$ and $\mathbf{W}\mathbf{H}$ was used:

$$D(\mathbf{V}||\mathbf{W}\mathbf{H}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\left[v_{ij}\log\frac{v_{ij}}{y_{ij}} - v_{ij} + y_{ij}\right] \tag{2}$$

The minimization of (2) can be solved by using iterative multiplicative rules [23]. Frequently, additional constraints are incorporated into (2). For example, the local NMF algorithm imposes spatial locality in the solution and consequently reveals local features in the data matrix $\mathbf{V}$ [24].

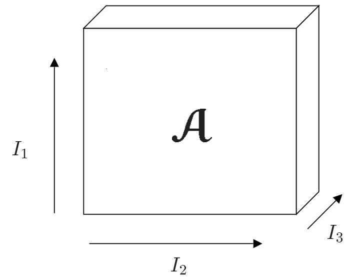

Fig. 1. 3rd order real-valued tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$.

A more general view of the NMF is set under the so-called non-negative matrix approximation (NNMA) in [11]. In NNMA, instead of minimizing a specific objective function, the minimization of a class of objective functions, called *Bregman divergences*, is proposed. The same approach will be adopted for the derivation of the NTF in Subsection III-D.

B. Tensors and Multilinear Algebra Basics

Quantities addressed by more than two indices are often employed in signal processing applications. To describe such quantities, tensors need to be employed. In multilinear algebra, tensors are considered as high-order generalizations of matrices and vectors [6]–[8]. A real-valued vector $\mathbf{a} \in \mathbb{R}^I$, $I \in \mathbb{Z}$ is treated as a first-order tensor. Similarly, a real-valued matrix $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$ with $I_1, I_2 \in \mathbb{Z}$ is defined as a second-order tensor. A real-valued tensor $\boldsymbol{\mathcal{A}}$ of order n is defined over the vector space $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, where $I_i \in \mathbb{Z}$, $i = 1, \ldots, n$. Each element of $\boldsymbol{\mathcal{A}}$ is addressed by n indices, i.e. $a_{i_1 i_2 \ldots i_n}$, where $i_i = 1, 2, \ldots, I_i$. A 3rd order tensor is sketched in Figure 1.

Mode-i unfolding of the tensor $\boldsymbol{\mathcal{A}}$ yields the matrix $\mathbf{A}_{(i)} \in \mathbb{R}^{I_i \times \overline{I}_i}$, where $\overline{I}_i \triangleq I_1 I_2 \cdots I_{i-1} I_{i+1} \cdots I_n$. In the following, the operations on tensors are expressed in matricized form [8]. The symbol $\times_i$ stands for the i-mode product between a tensor and a matrix [6]. It can be computed via the matrix multiplication $\mathbf{B}_{(i)} = \mathbf{U}\mathbf{A}_{(i)}$ followed by re-tensorization to undo the mode-i unfolding for $i = 1, 2, \ldots, n$. For example, the 2-mode product between the 3rd order tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and the matrix $\mathbf{U} \in \mathbb{R}^{I_2 \times J}$ yields the 3rd order tensor $\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}} \times_2 \mathbf{U} \in \mathbb{R}^{I_1 \times J \times I_3}$. The inner product of two nth order tensors $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ is denoted as $< \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} >$. The norm of tensor $\boldsymbol{\mathcal{A}}$ is defined as $||\boldsymbol{\mathcal{A}}|| = \sqrt{< \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} >}$.

An n-order tensor $\boldsymbol{\mathcal{A}}$ has rank 1, when it can be decomposed as the outer product of n vectors $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(n)}$, i.e.

$$\boldsymbol{\mathcal{A}} = \mathbf{u}^{(1)} \circ \mathbf{u}^{(2)} \circ \ldots \circ \mathbf{u}^{(n)}. \tag{3}$$

That is, each element of the tensor in (3) is given by $a_{i_1 i_2 \ldots i_n} = u_{i_1}^{(1)} \, u_{i_2}^{(2)} \, \ldots \, u_{i_n}^{(n)}$ for all $i_\ell = 1, 2, \ldots, I_\ell$ and $\ell = 1, 2, \ldots, n$. The rank of an arbitrary n-order tensor $\boldsymbol{\mathcal{A}}$ is the minimal number of rank-1 tensors that yield $\boldsymbol{\mathcal{A}}$ when linearly combined.

In the following, several products between matrices shall be needed, such as the Kronecker product denoted by $\otimes$ and the

Khatri-Rao product denoted by $\odot$, whose definitions can be found elsewhere, e.g. [8].

C. Motivation for Using Tensors and the Proposed Non-negative Tensor Factorization

The NMF has been used extensively in signal processing yielding promising results in the past years. A list of the numerous NMF applications can be found in [11]. However, the NMF as any other subspace method deals only with vectorized data. By vectorizing a typical 3rd order tensor stemming from 900 training recordings, which are represented by 30 feature vectors of 34 dimensions each, one obtains 900 vectors of 1020 dimensions. Many pattern classifiers cannot cope with the aforementioned dimensionality given the small number of training samples. In addition, handling such high-dimensional samples is computationally expensive. For example, eigen-analysis or singular value decomposition cannot be easily performed. Despite implementation issues, it is well understood that vectorization breaks the natural structure and correlation in the original data. Thus, in order to preserve the natural data structure and correlation, dimensionality reduction operating directly on tensors rather than vectors is desirable. The concept of low-rank decomposition of high-order signal representations is addressed in [6], [25], where several algorithms are reviewed. Thus, a high-order generalization of the NMF could be of great importance in the analysis of such high-order signal/pattern representations.

Some NMF generalizations have been proposed mostly for 3rd order tensors in face detection or recognition applications. In 2005, Shashua and Hazan proposed a generalization of the NMF for nth order tensors [26]. The problem was formulated as the decomposition of a tensor into a sum of k rank-1 tensors using the Frobenius norm as distance. Multiplicative update rules were employed and an application to sparse image coding was discussed. Hazan et al. extended the previous work by employing the KL divergence (also known as relative entropy) as distance [27].

In 2006, Boutsidis et al. introduced an algorithm for 3rd order tensor decomposition called projected alternating least squares with initialization and regularization (PALSIR) [28]. This algorithm also employed the Frobenius norm as distance and alternating least squares was used to derive the decomposition. Experiments were performed on eye image databases for biometric iris recognition applications. Heiler and Schnörr proposed a generalization of the sparse NMF algorithm for 3rd order tensors applied to face detection [29]. The Frobenius norm was used as distance in this case, too. The algorithm was termed as sparsity-constrained NTF, because a sparsity maximization algorithm was employed.

In 2007, Cichocki proposed algorithms for 3rd order NTF using alpha and beta divergences [30]. These algorithms employed alternating interior-point gradient and fixed point alternating least squares techniques incorporating sparsity constraints into the decomposition. The just described NTF method was incorporated into multilayer networks in order to improve the performance of multi-way blind source separation in EEG [31]. It should be noted that this model cannot be generalized to higher order tensors nor can degenerate to the NMF model for 2nd order tensors.

In this paper, we would like to derive a generic unified NTF algorithm, which can handle nth order tensors and can degenerate to the NMF, when $n = 2$.

D. Proposed NTF Algorithm

Having set our objectives, we decided to build upon the model proposed by Shashua and Hazan [26], which can be extended to nth order tensors and degenerates to the NMF, when $n = 2$. A preliminary version of the algorithm was first introduced in [33], which was also the basis for the Discriminant NTF algorithm in [32]. We resort to the Bregman divergences in order to offer a unified factorization framework, which includes as special cases the Frobenius norm, the KL-divergence, and the Itakura-Saito (IS) distance. The Bregman divergences, proposed by Bregman in 1967 [10], are defined as

$$D_\phi(x, y) = \phi(x) - \phi(y) - \phi'(y)(x - y), \tag{4}$$

where $\phi(x)$ is a strictly convex function defined on a convex set $S \subseteq \mathbb{R}$ and $\phi'(y)$ denotes the first derivative of $\phi()$ evaluated at y. By definition, the Bregman divergences are non-negative [11] and can be extended to tensors. Let us consider the nth order tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$. The following identity holds:

$$D_\phi(\mathcal{X}, \mathcal{Y}) = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_n=1}^{I_n} D_\phi(x_{i_1 i_2 \ldots i_n}, y_{i_1 i_2 \ldots i_n}). \tag{5}$$

For $\phi(x) = \frac{1}{2}x^2$, $D_\phi(x, y)$ corresponds to the Frobenius norm. For $\phi(x) = x \log(x)$, the Bregman divergence coincides with the KL divergence, whereas for $\phi(x) = -\log(x)$, the resulting $D_\phi(x, y)$ is recognized to be the Itakura-Saito (IS) distance.

Therefore, our goal is to decompose a tensor $\mathcal{V} \in \mathbb{R}_+^{I_1 \times I_2 \times \cdots \times I_n}$ into a sum of k rank-1 tensors:

$$\mathcal{V} = \sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \circ \mathbf{u}_j^{(n)} \tag{6}$$

where $\mathbf{u}_j^{(i)} \in \mathbb{R}_+^{I_i}$ and $j = 1, 2, \ldots, k$. Let $\mathbf{U}^{(i)} \triangleq [\mathbf{u}_1^{(i)} | \mathbf{u}_2^{(i)} | \ldots | \mathbf{u}_k^{(i)}]$, $i = 1, 2, \ldots, n$. Obviously $\mathbf{U}^{(i)} \in \mathbb{R}_+^{I_i \times k}$. Let us introduce the compact notation

$$\overline{\bigodot}_i \mathbf{U} \triangleq \mathbf{U}^{(n)} \odot \cdots \odot \mathbf{U}^{(i+1)} \odot \mathbf{U}^{(i-1)} \odot \cdots \odot \mathbf{U}^{(1)}. \tag{7}$$

In matricized form, the factorization (6) can be written as:

$$\begin{aligned} \mathbf{V}_{(i)} &\triangleq \underbrace{\mathbf{U}^{(i)}}_{\mathbf{W}^{(i)}} \Big(\underbrace{\mathbf{U}^{(n)} \odot \cdots \odot \mathbf{U}^{(i+1)} \odot \mathbf{U}^{(i-1)} \odot \cdots \mathbf{U}^{(1)}}_{\mathbf{H}^{(i)}}\Big)^T \\ &= \mathbf{U}^{(i)} \Big(\overline{\bigodot}_i \mathbf{U}\Big)^T. \end{aligned} \tag{8}$$

From the inspection of (8), one may readily see that the NMF results if $\mathbf{W} = \mathbf{W}^{(1)} = \mathbf{U}^{(1)}$ and $\mathbf{H} = [\mathbf{H}^{(1)}]^T = [\mathbf{U}^{(2)}]^T$. Let

$$\overline{\sum}_{\varrho_i} \triangleq \sum_{\varrho_1=1}^{I_1} \sum_{\varrho_2=1}^{I_2} \cdots \sum_{\varrho_{i-1}=1}^{I_{i-1}} \sum_{\varrho_{i+1}=1}^{I_{i+1}} \cdots \sum_{\varrho_n=1}^{I_n}. \tag{9}$$

The following minimization problem with Bregman divergences is solved

$$\min_{\mathbf{u}_j^{(i)} \geq \mathbf{0}} D_\phi\left(\sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \circ \mathbf{u}_j^{(n)}, \mathcal{V}\right) \tag{10}$$

by using auxiliary functions, as is analyzed in Appendix I. In particular, for the KL divergence (i.e. $\phi(x) = x\log(x)$), the following multiplicative update rule is obtained for the elements of $\mathbf{u}_j^{(i)}$ denoted as $u_{jl}^{(i)}$:

$$u_{jl}^{(i)} \leftarrow \tilde{u}_{jl}^{(i)} \cdot \exp\left(\frac{\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots\varrho_n} \cdot \log \frac{v_{\varrho_1\cdots\varrho_{i-1}l\varrho_{i+1}\cdots\varrho_n}}{\beta_{\varrho_1\cdots\varrho_{i-1}l\varrho_{i+1}\cdots\varrho_n}}}{\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots\varrho_n}}\right) \tag{11}$$

where $\tilde{u}_{jl}^{(i)}$ is the lth element of vector $\mathbf{u}_j^{(i)}$ before updating, $j = 1, 2, \ldots, k$, $i = 1, 2, \ldots, n$, $l = 1, 2, \ldots, I_i$, and

$$\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots\varrho_n} = u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)} \tag{12}$$

$$\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n} = \sum_{j'=1}^{k} u_{j'\varrho_1}^{(1)} \cdots u_{j'\varrho_{i-1}}^{(i-1)} u_{j'l}^{(i)} u_{j'\varrho_{i+1}}^{(i+1)} \cdots u_{j'\varrho_n}^{(n)}. \tag{13}$$

For $\phi(x) = \frac{1}{2}x^2$ (that is, when the Frobenius norm is used), the resulting update rule is:

$$u_{jl}^{(i)} \leftarrow \tilde{u}_{jl}^{(i)} \cdot \frac{\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots\varrho_n} \cdot v_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n}}{\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots\varrho_n} \cdot \beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n}}. \tag{14}$$

Finally, for $\phi(x) = -\log(x)$ (i.e. when the IS distance is employed), the update rule is:

$$u_{jl}^{(i)} \leftarrow \tilde{u}_{jl}^{(i)} \cdot \frac{\overline{\sum}_{\varrho_i} \frac{\alpha_{j\varrho_1\cdots\varrho_{i-1}\varrho_{i+1}\cdots\varrho_n}}{\beta_{\varrho_1\cdots\varrho_{i-1}l\varrho_{i+1}\cdots\varrho_n}}}{\overline{\sum}_{\varrho_i} \frac{\alpha_{j\varrho_1\cdots\varrho_{i-1}\varrho_{i+1}\cdots\varrho_n}}{v_{\varrho_1\cdots\varrho_{i-1}l\varrho_{i+1}\cdots\varrho_n}}}. \tag{15}$$

In order to apply the aforementioned NTF algorithms to an nth order tensor $\mathcal{V}$, the n matrices $\mathbf{U}^{(i)}$, $i = 1, 2, \ldots, n$, should be initialized by random numbers between 0 and 1. The update rules (11), (14), or (15) are applied to the column vectors $\mathbf{u}_j^{(i)}$ of matrix $\mathbf{U}^{(i)}$, $j = 1, 2, \ldots, k$. The proof of convergence for the Frobenius NTF algorithm can be found in Appendix I. The computational cost of the various NTF algorithms is derived in Appendix II.

E. Proposed 3rd Order NTF Classifier

The novel NTF classifier for 3rd order tensors discussed next was inspired by the NMF classifier proposed in [34], where a basis for each class was trained separately and the test data were projected onto an orthogonalized basis. Preliminary results using the proposed classifier for 3rd order tensors in music genre classification were reported in [33]. Let C be the number of genre classes. The proposed 3rd order NTF classifier considers a tensor $\mathcal{V}_c \in \mathbb{R}^{I_{c1} \times I_2 \times I_3}$ with I_{c1} being the number of training recordings in class c, $c = 1, 2, \ldots, C$ (i.e. 90 for stratified 10-fold cross-validation in the GTZAN dataset), I_2 being the dimensionality of feature vectors, $I_3 = 30$ being the number of feature vectors extracted per recording (i.e. the number of 1 sec segments each recording is split to). The algorithm steps are as follows:

1) Decompose the training tensor for each genre $\mathcal{V}_c \in \mathbb{R}^{I_{c1} \times I_2 \times I_3}$, $c = 1, 2, \ldots, C$, i.e.

$$\mathcal{V}_c = \sum_{j=1}^{k} \mathbf{u}_{cj}^{(1)} \circ \mathbf{u}_{cj}^{(2)} \circ \mathbf{u}_{cj}^{(3)}. \tag{16}$$

2) Determine the 1st mode of the tensor $\mathcal{V}_c$ by unfolding [6], [8]:

$$\mathbf{V}_{c_{(1)}} = \mathbf{U}_c^{(1)} \left(\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)}\right)^T \tag{17}$$

where $\odot$ stands for the Khatri-Rao matrix product. Thus, $\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)}$ has dimensions $(I_3 I_2) \times k$, while $\mathbf{V}_{c_{(1)}}$ is a matrix with dimensions $I_{c1} \times (I_3 I_2)$. In the following, we deal with the transpose of matrix $\mathbf{V}_{c_{(1)}}$, i.e.

$$[\mathbf{V}_{c_{(1)}}]^T = \left(\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)}\right) [\mathbf{U}_c^{(1)}]^T. \tag{18}$$

3) Perform QR decomposition on the basis matrix $\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)}$:

$$\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)} = \mathbf{Q}_c \, \mathbf{R}_c \tag{19}$$

where $\mathbf{Q}_c$ is a $(I_3 I_2) \times k$ column-orthogonal matrix (i.e. $\mathbf{Q}_c^T \mathbf{Q}_c$ is the $k \times k$ identity matrix)[1] and $\mathbf{R}_c$ is a $k \times k$ upper triangular matrix. Store matrices $\mathbf{Q}_c$ and $\mathbf{H}_c = \mathbf{R}_c \, [\mathbf{U}_c^{(1)}]^T$. It is worth noting that the Gram-Schmidt orthogonalization does not affect the non-negativity of the basis matrix. It is used to calculate **correctly** the L_2 norms in a non-orthogonal basis.

4) For testing, the feature matrix $\mathbf{V}_t$ of dimensions $I_2 \times I_3$ is considered. The feature matrix is arranged to a column vector $\mathbf{v}_t$ of dimensions $I_2 I_3$ by concatenating its columns. The column vector $\mathbf{v}_t$ is projected onto the subspaces defined by the basis matrices of the classes:

$$\mathbf{h}_{ct} = \mathbf{Q}_c^T \, \mathbf{v}_t \tag{20}$$

and has length k.

5) Let $CSM_{tm}(c)$ be the cosine similarity measure (CSM) between $\mathbf{h}_{ct}$ and $\mathbf{h}_{cm}$, $m = 1, 2, \ldots, I_{c1}$ (i.e. the mth column of matrix $\mathbf{H}_c$):

$$CSM_{tm}(c) = \frac{\mathbf{h}_{ct}^T \, \mathbf{h}_{cm}}{||\mathbf{h}_{ct}|| \, ||\mathbf{h}_{cm}||}. \tag{21}$$

Let $CSM_{t[m]}(c)$ denote the mth largest element in the set $\{CSM_{tm}(c), \quad m = 1, 2, \ldots, I_{c1}\}$. The decision taken by the classifier is based on

$$\varpi_t(c) = \sum_{m=1}^{K} CSM_{t[m]}(c) \tag{22}$$

where $K \ll I_{c1}$ (e.g. $K = 3$). The class label of the test pattern $\mathbf{v}_t$ is determined by the maximum among $\varpi_t(c)$, i.e.:

$$\hat{c}_t = \underset{c=1,2,\ldots,C}{\operatorname{argmax}} \{\varpi_t(c)\}. \tag{23}$$

A block diagram of the testing procedure of the proposed supervised NTF classifier is sketched in Figure 2.

[1]Obviously, $\mathbf{Q}_c \mathbf{Q}_c^T$ is **not** equal to the identity matrix.

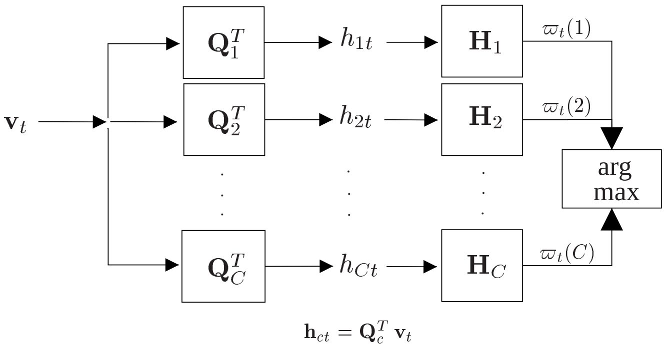

Fig. 2. The proposed supervised NTF classifier.

TABLE II
THE FEATURE SET.

No.	Feature	# Values per segment
1	Short-Time Energy (STE)	$1 \times 4 = 4$
2	Fundamental Frequency (FF)	$1 \times 4 = 4$
3	Total Loudness (TL)	$1 \times 4 = 4$
4	Specific Loudness Sensation (SONE)	$8 \times 4 = 32$
5	Spectrum Centroid (SC)	$1 \times 4 = 4$
6	Spectrum Rolloff Frequency (SRF)	$1 \times 4 = 4$
7	Spectrum Spread (SS)	$1 \times 4 = 4$
8	Spectrum Flatness (SF)	$4 \times 4 = 16$
9	Mel-frequency Cepstral Coefficients (MFCCs)	$24 \times 4 = 96$
10	Auto-Correlation Values (AC)	13
11	Log Attack Time (LAT)	1
12	Temporal Centroid (TC)	1
13	Zero-Crossing Rate (ZCR)	$1 \times 4 = 4$
14	Spectral Difference (SD)	$1 \times 4 = 4$
15	Bandwidth (BW)	$1 \times 4 = 4$
16	Phase Deviation (PD)	$1 \times 4 = 4$
17	Pitch Histogram (PH)	$1 \times 4 = 4$
18	Rhythmic Periodicity (RP)	$1 \times 4 = 4$
	Total number of features	**207**

IV. EXPERIMENTAL RESULTS

In this Section, music genre classification experiments are discussed. In subsection IV-A, the employed dataset is described. The feature extraction is detailed in subsection IV-B, while the feature selection method is discussed in subsection IV-C. Finally, the accuracy using the various classifiers is reported in subsection IV-D.

A. Dataset

The GTZAN database was employed for genre classification experiments. The database contains 1000 audio recordings distributed across 10 music genres [12], namely: Classical, Country, Disco, HipHop, Jazz, Rock, Blues, Reggae, Pop, and Metal. 100 recordings are collected for each genre. All recordings are mono channel, are sampled at 22050 Hz rate, and have a duration of approximately 30 sec. Each recording is separated into 30 segments (i.e. texture windows) of 1 sec duration. Such a texture window has commonly been used in genre classification experiments, because it increases the classification accuracy compared to direct analysis frames [2], [12], [19]. For each 1 sec long texture window, 207 features are extracted, which are described next.

B. Feature Extraction

In music genre classification experiments, the extracted features usually belong into 3 categories, namely timbre, rhythm, and pitch-based ones [1], [2]. In this paper, a combination of descriptors measuring energy, spectral, temporal, perceptual, and pitch characteristics of the music recordings is explored [35]. The complete list of the extracted features can be found in Table II.

The 1st feature measures the energy of the audio signal. Feature 2 is computed by maximum likelihood harmonic matching. Features 3 and 4 refer to the perceptual modeling of the human auditory system [17]. The spectral shape is captured by features 5-9 and 14-15. The temporal properties of the signals are correlated with features 10-13 and 16. Feature 17 describes the amplitude of the maximum peak of the folded histogram [36]. Feature 18 was proposed in [37]. Features 1, 2, 5, 7, 11, and 12 were computed using the definitions of the MPEG-7 audio framework [38]. It should be noted that 24 Mel-frequency cepstral coefficients and 8 specific loudness sensation (SONE) coefficients are extracted for each audio frame of 10 msec duration.

Except features 10-12, the remaining features are computed on frame basis and their 1st and 2nd moments are exploited by averaging over the frames within each 1 sec long texture window. Similarly, the 1st and 2nd moments of the first-order frame-based feature differences are computed. This explains the factor 4 appearing in Table II. In total, 207 features are extracted from each texture window. All features but the MFCCs are non-negative. Accordingly, they can be employed directly into the NTF. For the MFCCs, their magnitude is retained only. The computation of the aforementioned features every 1 sec yields the tensor $\mathcal{V}$ of dimensions $1000 \times 207 \times 30$.

For comparison purposes, a smaller feature set is also tested, which includes the features extracted by the Music Analysis, Retrieval and Synthesis for Audio Signals (MARSYAS) platform [39]. This feature set consists of the 1st order moments of the following timbral features: Spectral Centroid, Spectral Rolloff Frequency, Spectral Difference (also known as spectral flux), and 30 MFCCs per frame, which are averaged over 1 sec texture windows. Thus, the tensor of the MARSYAS features has dimensions $1000 \times 34 \times 30$.

C. Feature Selection

Careful feature selection is essential for classification. Here, the optimal feature subset maximizes the ratio of the inter-class dispersion over the intra-class dispersion: $J = \mathrm{tr}(\mathbf{S}_w^{-1}\mathbf{S}_b)$, where $\mathrm{tr}(\cdot)$ stands for the trace of a matrix, $\mathbf{S}_w$ is the within-class scatter matrix, and $\mathbf{S}_b$ is the between-class scatter matrix. Details on the computation of $\mathbf{S}_w$ and $\mathbf{S}_b$ can be found in any textbook on pattern recognition (e.g. [9]). Because, in our case, the number of distinct subsets having cardinality I_2 ($1 \leq I_2 \leq 207$) is $\frac{207!}{(207-I_2)!I_2!}$, the branch-and-bound search strategy is employed for complexity reduction. In this strategy, a tree structure of $(207 - I_2 + 1)$ levels is created, where every node corresponds to a subset. The tree root corresponds to the full set (e.g. 207 features), while each leaf node corresponds to a subset of cardinality I_2. The branch-and-bound search

TABLE IV
AVERAGE ACCURACY ACHIEVED BY SEVERAL CLASSIFIERS WHEN EITHER THE SUBSET OF 80 SELECTED FEATURES OR THE MARSYAS FEATURE SET WAS EMPLOYED.

Classifier	80 Feature Subset	MARSYAS Feature Set
NTF Frobenius	**78.9%**	68.3%
SVM	77.2%	67.6%
MLP	77.0%	**69.1%**
NTF KL	70.4%	61.4%
NTF IS	63.6%	55.0%

strategy traverses the structure using a depth-first search with backtracking [9].

In order to apply the feature selection algorithm, the data tensor should be transformed into a matrix by unfolding [6]. Thus, the unfolding $\mathbf{V}_{(2)} \in \mathbb{R}_+^{207\times 30000}$ is computed from tensor $\mathcal{V} \in \mathbb{R}_+^{1000\times 207\times 30}$. Several feature subsets were derived with respect to maximizing J comprising $I_2 \in \{60, 70, 80, 90\}$ features out of the 207 initial features. The subset comprising 80 features listed in Table III is found to yield the highest music genre classification accuracy, when it is employed in the proposed NTF Frobenius classifier (cf. subsection IV-D). It is seen that 31 out of the 80 selected features are moments of the MFCCs or their first-order differences.

D. Performance Assessment

Experiments were performed by employing various subsets of selected features as well as the MARSYAS feature set using a stratified 10-fold cross validation, which is widely used in genre classification experiments on the GTZAN dataset.

The rank of the 3rd order genre class-dependent tensor $\mathcal{V}_c$ should satisfy [8] (and references therein)

$$k \leq \min\{I_{c1}I_2, I_{c1}I_3, I_2I_3\}. \tag{24}$$

Since by the experimental protocol I_{c1} and I_3 are fixed to 90 and 30, respectively, the inequality (24) implies $k \leq 30I_2$, if $I_2 \leq 90$. Various values of k were tested for the NTF algorithms within the proposed NTF classifier. The highest genre classification accuracy was obtained for the following values of k: $k = 55$, when the feature subset comprises $I_2 = 60$ features; $k = 61$, when the number of selected features I_2 is 70 or 80; and $k = 64$, when $I_2 = 90$ features are selected. For the MARSYAS set, k was set to 22. The number of terms K taken into account in (22) was set to 3 for all NTF classifiers.

The performance of the NTF classifier was compared against that of multilayer perceptron (MLP) and SVMs. In particular, a 3-layered perceptron with the logistic activation function was used. Its training was performed by the back-propagation algorithm with learning rate equal to 0.3 and momentum equal to 0.2 for 500 training epochs. A multi-class SVM classifier with a 2nd order polynomial kernel with unit bias/offset was also tested [40]. The experiments with the aforementioned classifiers were conducted on the matrix unfolding $\mathbf{V}_{(2)} \in \mathbb{R}_+^{I_2\times 30000}$ using 10-fold cross validation, where $I_2 = \{60, 70, 80, 90\}$.

The average music genre classification accuracy achieved by the classifiers over the 10 folds is listed in Table IV, when

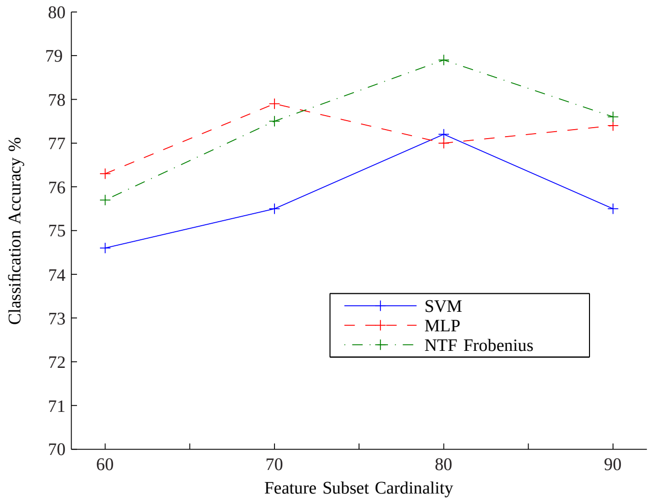

Fig. 3. Average music genre classification accuracy for the various feature subsets.

either the subset of 80 selected features or the MARSYAS feature set was employed. In Figure 3, the average accuracy of the SVM, MLP, and NTF Frobenius classifier is plotted versus several subset cardinalities. From Figure 3, it can be seen that the highest average accuracy of 78.9% was obtained by the proposed NTF Frobenius classifier, when it is applied to the subset of 80 selected features listed in Table III. The standard deviation of the accuracy achieved by the NTF Frobenius classifier is found to be 2.60%. The aforementioned classification accuracy outperforms that reported in [12] (i.e. 61.0%), [3] (i.e. 74.9%), [15] (i.e. 74.0%), and slightly exceeds that reported in [13] (i.e. 78.5%). In [14], a greater classification accuracy than ours is reported (i.e. 82.5%) by employing boosting. However, since cross-validation was not used, the latter accuracy is not directly comparable with ours (i.e. 78.9%). The NTF classifier, when the Frobenius norm was used, attained a higher accuracy than that achieved by the SVM or the MLP for 80 selected features. This was not the case when 60 or 70 features were selected, as can be seen in Figure 3.

Concerning the classification accuracy when the full set of 207 features is used with , it was measured 54.7%, 49.2%, and 40.8% for the NTF Frobenius, NTF KL, and NTF IS classifiers, respectively, with k set to 86. The corresponding accuracies for the SVM or the MLP classifiers were 51.8% and 53.5%, respectively.

The NTF Frobenius classifier outperforms the SVM for all subset feature cardinalities tested. The NTF classifier achieved a lower accuracy, when either the KL divergence or the IS distance was employed, than that when the Frobenius norm was used.

If the comparison is made across the sets of features employed, the inspection of Table IV reveals that the set of 80 features listed in Table III clearly performs better than the MARSYAS feature set within all classifiers. Using the MARSYAS features, the best classification accuracy of 69.1% was achieved by the MLP classifier. When the NTF Frobenius

TABLE III
THE 80 FEATURES SELECTED BY THE BRANCH-AND-BOUND ALGORITHM.

No.	Selected Feature	No.	Selected Feature
1	Mean of 2nd SF coefficient	41	Mean of 1st order difference of 2nd SF coefficient
2	Variance of 1st SONE	42	Variance of SS
3	Mean of SF	43	Mean of 8th SONE
4	Mean of 2nd SONE	44	Variance of 2nd SF coefficient
5	Mean of BW	45	Variance of 1st order difference of 20th MFCC
6	Variance of 1st order difference of 13rd MFCC	46	Mean of 9th MFCC
7	Mean of 12th MFCC	47	Variance of 1st order difference of 2nd MFCC
8	Mean of SC	48	Mean of 1st order difference of 10th MFCC
9	Mean of 1st SF coefficient	49	Mean of 5th MFCC
10	Mean of SS	50	Mean of 3rd MFCC
11	Variance of TL	51	Variance of 1st order difference of 1st SF coefficient
12	Variance of 1st order difference of 5th MFCC	52	7th AC coefficient
13	Mean of 3rd SF coefficient	53	Mean of SRF
14	Variance of 8th MFCC	54	Variance of 1st order difference of 24th MFCC
15	Variance of SD	55	3rd AC coefficient
16	Mean of RP	56	Variance of 1st order difference of 12th MFCC
17	Variance of PD	57	Mean of 6th SONE
18	Mean of 4th SF coefficient	58	4th AC coefficient
19	Variance of 1st order difference of SS	59	Variance of 9th MFCC
20	Variance of 1st order difference of 8th MFCC	60	Variance of 2nd SONE coefficient
21	Variance of 1st order difference of 11th MFCC	61	Variance of 1st MFCC
22	Variance of 1st order difference of 10th MFCC	62	Mean of 14th MFCC
23	Mean of 11th MFCC	63	Mean of 1st MFCC
24	Variance of 4th SF	64	Variance of 1st order difference of 4th SF coefficient
25	Mean of TL	65	Mean of 2nd MFCC
26	Variance of 1st order difference of 1st SONE	66	Variance of 1st order difference of 3rd SF coefficient
27	Mean of 1st order difference of RP	67	Variance of 1st order difference of 8th SONE coefficient
28	Variance of 3rd SONE	68	Variance of 1st order difference of 5th SONE coefficient
29	Mean of FF	69	Variance of 1st order difference of 3rd SONE coefficient
30	Variance of 1st order difference of 9th MFCC	70	Mean of 1st order difference of 1st SF coefficient
31	Mean of 1st order difference of 11th MFCC	71	Variance of STE
32	Variance of 1st order difference of RP	72	Variance of BW
33	Mean of 13th MFCC	73	Mean of 1st order difference of 12th MFCC
34	Variance of 1st order difference of SC	74	Mean of 16th MFCC
35	Mean of 1st order difference of 14th MFCC	75	Variance of 1st order difference of 7th MFCC
36	Variance of SC	76	Variance of SRF
37	Variance of 2nd MFCC	77	Variance of 1st order difference of 2nd SF coefficient
38	Variance of 1st SF coefficient	78	Variance of 7th MFCC
39	Variance of 3rd SF coefficient	79	Mean of 4th MFCC
40	Variance of 5th SONE	80	Mean of PD

classifier was used with MARSYAS features, the second best accuracy 68.3% was obtained. The superiority of the extracted features over the MARSYAS ones is partially attributed to the fact that the latter features roughly consist a subset of the former ones. For the MARSYAS features, the NTF classifier with either the KL divergence or the IS distance is less performing than the NTF classifier with the Frobenius norm.

Next, the statistical significance of the accuracy differences between the classifiers was addressed by employing the method described in [41], where the number of correctly classified patterns is assumed to be distributed according to the binomial distribution. It can easily be shown that the performance gains obtained by the NTF Frobenius classifier against the SVM and MLP classifiers are not statistically significant at 95% confidence level. On the contrary, the accuracy difference between the NTF classifier with the Frobenius norm and the same classifier, when either the KL divergence or the IS distance is used, is found to be statistically significant at 95% confidence level. It should be noted that the difference of 0.4% between the one-vs-the-rest SVMs [13] and the NTF Frobenius classifier is statistically insignificant as well. However, the performance gain obtained by the NTF Frobenius against the classifiers employed in [3], [12], [15] is statistically significant.

Insight to the performance of the NTF Frobenius, SVM, and MLP classifiers is offered by the confusion matrices averaged over the 10 splits determined by 10-fold stratified cross-validation, in Tables V, VI, and VII, respectively. The columns of the confusion matrix correspond to the predicted music genre and the rows to the actual one. For the NTF Frobenius classifier, most misclassifications occur among the Hiphop, Pop, and Rock genres. Concerning the SVM classifier, most misclassifications occur for Rock recordings, which are

TABLE V
AVERAGE CONFUSION MATRIX FOR THE NTF FROBENIUS CLASSIFIER USING THE 80 SELECTED FEATURES OVER THE 10 SPLITS DETERMINED BY 10-FOLD STRATIFIED CROSS-VALIDATION.

Genre	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	Rock
Blues	**86**	1	6	1	0	1	0	0	1	4
Classical	1	**88**	9	0	0	1	0	0	0	1
Country	2	0	**87**	1	0	3	0	0	1	6
Disco	2	0	4	**76**	4	0	0	3	3	8
Hiphop	3	0	2	11	**67**	0	3	5	8	1
Jazz	4	1	10	2	0	**77**	4	0	0	2
Metal	1	0	0	1	0	1	**92**	0	0	5
Pop	3	1	3	15	3	0	0	**72**	1	2
Reggae	2	0	2	5	5	1	0	5	**73**	7
Rock	4	0	5	4	0	2	7	2	5	**71**

TABLE VI
AVERAGE CONFUSION MATRIX FOR THE SVM CLASSIFIER USING THE 80 SELECTED FEATURES OVER THE 10 SPLITS DETERMINED BY 10-FOLD STRATIFIED CROSS-VALIDATION.

Genre	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	Rock
Blues	**84**	0	4	2	1	2	0	1	2	4
Classical	0	**97**	1	0	0	1	0	0	0	1
Country	6	2	**73**	4	0	2	0	1	0	12
Disco	2	0	3	**77**	2	0	1	4	3	8
Hiphop	2	0	0	8	**73**	0	2	3	11	1
Jazz	5	5	7	0	1	**80**	0	0	0	2
Metal	0	0	0	1	3	0	**90**	1	0	5
Pop	1	1	5	3	3	0	2	**77**	3	5
Reggae	4	0	1	9	11	1	0	3	**67**	4
Rock	7	0	17	11	1	0	5	2	3	**54**

misclassified as either Country or Disco ones. The same occurs for the MLP classifier. It is worth noting that the boundaries between genres such as Pop and Rock as well as Rock and Metal still remain fuzzy [2], a fact that is reflected in the annotations accompanying the dataset.

V. CONCLUSIONS - FUTURE WORK

In this paper, music genre recognition experiments have been performed using a variety of sound description features and multilinear classification techniques. Novel algorithms for the NTF have been derived from first principles and their computational cost has been estimated. An NTF classifier that trains a basis for each class separately and employs basis orthogonalization has also been proposed. The NTF classifier has been tested against state-of-the-art classifiers. It has been found to be slightly superior than them. This superiority is attributed to the higher expressive power of the multilinear representations than that of the pattern matrix the standard pattern recognition algorithms they depend on.

NTF classifiers compared to standard machine learning approaches, such as MLPs and SVMs, are not limited to vectorized data, but can be used for higher order representations.

TABLE VII
AVERAGE CONFUSION MATRIX FOR THE MLP CLASSIFIER USING THE 80 SELECTED FEATURES OVER THE 10 SPLITS DETERMINED BY 10-FOLD STRATIFIED CROSS-VALIDATION.

Genre	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	Rock
Blues	**80**	3	4	0	0	3	2	1	1	6
Classical	0	**93**	0	0	0	3	0	1	0	3
Country	3	1	**74**	3	0	5	0	2	1	11
Disco	4	1	8	**72**	3	0	2	3	4	3
Hiphop	2	0	1	2	**80**	0	2	5	7	1
Jazz	9	4	2	0	0	**82**	1	0	1	1
Metal	1	0	0	2	2	0	**87**	3	0	5
Pop	1	0	6	3	2	0	1	**79**	1	7
Reggae	4	0	3	3	9	1	0	3	**72**	5
Rock	6	0	19	8	1	2	6	2	5	**51**

For example, tensors can be used for modeling any recording as a time series of potentially different genre labels. NTF offers an attractive framework for modeling data as a multilinear combination of features and can extract basis features enabling a greater interpretability of the basis contributions to the classification than SVMs and MLPs. Such factorizations can also be used for dimensionality reduction prior to the application of standard machine learning algorithms (e.g. SVMs). NTF is not limited to music genre classification, but it can be utilized in various cases associated with feature vectors computed over time, such as for multiple frequency estimation in audio recordings in order to provide a global spectral basis for a whole set instead of a basis for each recording [42].

In the future, the performance of NTF will be assessed on hierarchical music genre databases, which offer additional flexibility over the flat classification approaches. In addition, the NTF algorithms could be enhanced by incorporating penalty functions into the factorization problem, which can control the outcome of the factorization. Finally, various initialization techniques similar to those proposed for the NMF algorithm [44], could be developed for the NTF algorithms aiming to reduce the number of iterations.

APPENDIX I

A. Problem Formulation

As stated in Section III-D, the following minimization problem is treated:

$$\min_{\mathbf{u}_j^{(i)} \geq \mathbf{0}} D_\phi \left(\sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \circ \mathbf{u}_j^{(n)}, \mathcal{V} \right).$$

Let $l = 1, 2, \ldots, I_i$ and $i = 1, 2, \ldots, n$. The goal is to find a multiplicative updating rule for the elements of vectors $\mathbf{u}_l^{(i)}$ denoted as $u_{jl}^{(i)}$, $j = 1, 2, \ldots, k$. From (5), it can be seen that:

$$D_\phi \left(\sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \circ \mathbf{u}_j^{(n)}, \mathcal{V} \right) = \sum_{l=1}^{I_i} D_\phi \left(\sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \mathbf{u}_j^{(i-1)} \, u_{jl}^{(i)} \circ \mathbf{u}_j^{(i+1)} \circ \cdots \circ \mathbf{u}_j^{(n)}, \mathcal{V}_{\varrho_i = l} \right) \tag{25}$$

where $\mathcal{V}_{\varrho_i = l} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{i-1} \times I_{i+1} \times \cdots \times I_N}$ is a sub-tensor with the ith index fixed to l whose elements are denoted by $v_{\varrho_1 \ldots \varrho_{i-1} l \varrho_{i+1} \ldots \varrho_n}$, $\varrho_\ell = 1, 2, \ldots, I_\ell$ and $\ell = 1, 2, \ldots, i-1, i+1, \ldots, n$.

B. Auxiliary function

The minimization problem can be solved using auxiliary functions [11], [23]. Let $F(\mathbf{u}_l^{(i)})$ denote the divergence term in (25), i.e.

$$F(\mathbf{u}_l^{(i)}) = D_\phi \left(\sum_{j=1}^{k} \mathbf{u}_j^{(1)} \circ \mathbf{u}_j^{(2)} \circ \cdots \mathbf{u}_j^{(i-1)} u_{jl}^{(i)} \circ \mathbf{u}_j^{(i+1)} \circ \cdots \circ \mathbf{u}_j^{(n)}, \mathcal{V}_{\varrho_i = l} \right). \tag{26}$$

The application of (4) and (5) to (26) yields:

$$F(\mathbf{u}_l^{(i)}) = \overline{\sum}_{\varrho_i} \phi\Big(\sum_{j=1}^{k} u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}\Big) - \phi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}) - \psi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n})\cdot \cdot\Big(\sum_{j=1}^{k} u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)} - v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}\Big), \tag{27}$$

where $\psi(x) = \phi'(x)$. The following auxiliary function for $F(\mathbf{u}_l^{(i)})$ is proposed:

$$G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)}) = \overline{\sum}_{\varrho_i} \Bigg(\sum_{j=1}^{k} \lambda_{\varrho_1\dots j\dots\varrho_n} \cdot \phi\Bigg(\frac{u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}}{\lambda_{\varrho_1\dots\varrho_{i-1}j\varrho_{i+1}\dots\varrho_n}}\Bigg)\Bigg) - \phi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}) - \psi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n})\cdot \cdot\Big(\sum_{j=1}^{k} u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)} - v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}\Big) \tag{28}$$

where

$$\lambda_{\varrho_1\dots\varrho_{i-1}j\varrho_{i+1}\dots\varrho_n} = \frac{u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} \tilde{u}_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}}{\sum_{j'=1}^{k} u_{j'\varrho_1}^{(1)} \cdots u_{j'\varrho_{i-1}}^{(i-1)} \tilde{u}_{j'l}^{(i)} u_{j'\varrho_{i+1}}^{(i+1)} \cdots u_{j'\varrho_n}^{(n)}}. \tag{29}$$

It can easily be shown that $G(\mathbf{u}_l^{(i)}, \mathbf{u}_l^{(i)}) = F(\mathbf{u}_l^{(i)})$. In addition, using Jensen's inequality for convex functions, it can be verified that $G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)}) \geq F(\mathbf{u}_l^{(i)})$. Accordingly, indeed $G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})$ is an auxiliary function for $F(\mathbf{u}_l^{(i)})$.

C. Minimization of the auxiliary function

In order to derive a multiplicative update rule for $u_{jl}^{(i)}$, the auxiliary function $G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})$ should be minimized with respect to $u_{jl}^{(i)}$. The partial derivative of $G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})$ with respect to $u_{jl}^{(i)}$ is set to zero:

$$\frac{\partial G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})}{\partial u_{jl}^{(i)}} = \overline{\sum}_{\varrho_i} \lambda_{\varrho_1\dots\varrho_{i-1}j\varrho_{i+1}\dots\varrho_n} \cdot \psi\Bigg(\frac{u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{jl}^{(i)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}}{\lambda_{\varrho_1\dots\varrho_{i-1}j\varrho_{i+1}\dots\varrho_n}}\Bigg) \cdot \frac{u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}}{\lambda_{\varrho_1\dots\varrho_{i-1}j\varrho_{i+1}\dots\varrho_n}} - \overline{\sum}_{\varrho_i} \psi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}) \cdot \Big(u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}\Big) \tag{30}$$

By replacing (29) into (30) and after performing some algebraic manipulations, we obtain:

$$\frac{\partial G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})}{\partial u_{jl}^{(i)}} = \overline{\sum}_{\varrho_i} \Big(u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}\Big) \cdot \psi\Bigg(\sum_{j'=1}^{k} u_{j'\varrho_1}^{(1)} \cdots u_{j'\varrho_{i-1}}^{(i-1)} \tilde{u}_{j'l}^{(i)} u_{j'\varrho_{i+1}}^{(i+1)} \cdots u_{j'\varrho_n}^{(n)} \cdot \frac{u_{jl}^{(i)}}{\tilde{u}_{jl}^{(i)}}\Bigg) - \overline{\sum}_{\varrho_i} \psi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}) \cdot \Big(u_{j\varrho_1}^{(1)} \cdots u_{j\varrho_{i-1}}^{(i-1)} u_{j\varrho_{i+1}}^{(i+1)} \cdots u_{j\varrho_n}^{(n)}\Big) \tag{31}$$

The equation

$$\frac{\partial G(\mathbf{u}_l^{(i)}, \tilde{\mathbf{u}}_l^{(i)})}{\partial u_{jl}^{(i)}} = 0 \tag{32}$$

cannot be analytically solved for any $\psi()$. If $\psi()$ is assumed to be multiplicative as in [11] (i.e. $\psi(xy) = \psi(x)\psi(y)$), the substitution of (31) in (32) yields the following update rule:

$$u_{jl}^{(i)} \leftarrow \tilde{u}_{jl}^{(i)} \cdot \psi^{-1}\Bigg(\frac{\overline{\sum}_{\varrho_i} \psi(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}) \cdot \alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots\varrho_n}}{\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots\varrho_n} \cdot \psi(\beta_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n})}\Bigg) \tag{33}$$

where $\alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots\varrho_n}$ and $\beta_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}$ are defined in (12) and (13), respectively. The update rule (33) should be applied to all elements $u_{jl}^{(i)}$ for $j = 1, 2, \dots, k$, $i = 1, 2, \dots, n$, and $l = 1, 2, \dots, I_i$. It should be noted that $\psi()$ is multiplicative for the Frobenius norm, since $\phi(x) = \frac{1}{2}x^2 \Rightarrow \psi(x) = x$. The update rule for the Frobenius norm given in (14) can be easily derived from (33).

However, (33) cannot be applied to the KL divergence or the IS distance, since the associated functions $\psi()$ are not multiplicative. Indeed, we have $\phi(x) = x\log(x) \Rightarrow \psi(x) = \log(x) + 1$ for the KL divergence. By replacing the explicit form of $\psi(x)$ into (31), we obtain

$$\overline{\sum}_{\varrho_i} \alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots\varrho_n} \cdot \Bigg(\log\beta_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n} + \log\frac{u_{jl}^{(i)}}{\tilde{u}_{jl}^{(i)}}\Bigg) - \overline{\sum}_{\varrho_i} \log\psi\Big(v_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots\varrho_n}\Big) \cdot \alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots\varrho_n} = 0, \tag{34}$$

which leads to the update rule (11). Similarly, for the IS distance, $\phi(x) = -\log(x) \Rightarrow \psi(x) = -\frac{1}{x}$. By replacing the explicit form of $\psi(x)$ into (31), we obtain the update rule (15).

D. Proof of Convergence

To prove the convergence of the multiplicative update rule for the Frobenius NTF algorithm (14), one needs to show that $F(\mathbf{u}_l^{(i)}) \leq F(\tilde{\mathbf{u}}_l^{(i)})$. Using the definition of $F(\mathbf{u}_l^{(i)})$, $\alpha_{j\varrho_1\dots\varrho_{i-1}\varrho_{i+1}\dots,\varrho_n}$ and $\beta_{\varrho_1\dots\varrho_{i-1}l\varrho_{i+1}\dots,\varrho_n}$ given in (27), (12), and (13), respectively, it can be shown that:

$$F(\tilde{\mathbf{u}}_l^{(i)}) - F(\mathbf{u}_l^{(i)}) =$$
$$\overline{\sum}_{\varrho_i}\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)} - v_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n}\right)\cdot$$
$$\cdot\left(\psi(\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n}) - \psi(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)})\right)$$
$$+\overline{\sum}_{\varrho_i}D_\phi\left(\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n},\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)}\right). \tag{35}$$

Since the 2nd term is by definition non-negative, it suffices to prove that the first term in (35) is non-negative. For the Frobenius norm, $\psi(x) = x$, a fact that facilitates further the derivations. Moving the denominator in (14) to the left hand side part and summing over j, we get

$$\overline{\sum}_{\varrho_i}\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n}\cdot v_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n} =$$
$$\overline{\sum}_{\varrho_i}\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)}\right)\cdot\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n}. \tag{36}$$

Using (36) into the algebraic manipulations of the first term in (35), we conclude that it is sufficient to prove

$$\overline{\sum}_{\varrho_i}v_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n}\cdot\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)}\right)\geq$$
$$\overline{\sum}_{\varrho_i}\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)}\right)\cdot$$
$$\left(\sum_{j'=1}^{k}\alpha_{j'\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{j'l}^{(i)}\right) \tag{37}$$

(14) implies that

$$\overline{\sum}_{\varrho_i}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot v_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots\varrho_n} =$$
$$\overline{\sum}_{\varrho_i}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n}\cdot\frac{u_{jl}^{(i)}}{\tilde{u}_{jl}^{(i)}} \tag{38}$$

Using (38), the inequality (37) is rewritten as

$$\overline{\sum}_{\varrho_i}\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot\frac{[u_{jl}^{(i)}]^2}{\tilde{u}_{jl}^{(i)}}\right)\cdot\beta_{\varrho_1\ldots\varrho_{i-1}l\varrho_{i+1}\ldots,\varrho_n}\geq$$
$$\overline{\sum}_{\varrho_i}\left(\sum_{j=1}^{k}\alpha_{j\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{jl}^{(i)}\right)\cdot$$
$$\cdot\left(\sum_{j'=1}^{k}\alpha_{j'\varrho_1\ldots\varrho_{i-1}\varrho_{i+1}\ldots,\varrho_n}\cdot u_{j'l}^{(i)}\right) \tag{39}$$

Inequality (39) holds thanks to Lemma 4 [11], which concludes the proof.

APPENDIX II

The computational cost of the NTF is derived for $n = 3$ (3rd order tensors), when the Frobenius norm is used. We assume that one flop corresponds to a single floating point operation, i.e. a floating point addition or a floating point multiplication [43]. The computational cost is detailed in Table VIII. We explicitly calculate the cost for the computation of the matrix $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1\times k}$ having columns $\mathbf{u}_j^{(i)}$. The first entry refers to terms $\alpha_{j\varrho_2\varrho_3}$ defined by (12), which are kI_2I_3 in total and each of them requires 1 multiplication. The second entry refers to terms $\beta_{l\varrho_2\varrho_3}$ defined by (13), which are $I_1I_2I_3$ in total and each of them requires $2k$ multiplications and $k-1$ additions. The cost for one update of $u_{jl}^{(1)}$ given by (14) is $2(I_2I_3+1)$ multiplications and $2(I_2I_3-1)$ additions, therefore in total $4I_2I_3$. There are I_1k elements that should be computed. In total, one needs $(7k-1)I_1I_2I_3 + kI_2I_3$ flops per iteration in order to compute the full matrix $\mathbf{U}^{(1)}$. By repeating the same computation for $\mathbf{U}^{(2)}$ and $\mathbf{U}^{(3)}$ and multiplying by the number of iterations r needed for convergence, we obtain:

$$3r(7k-1)I_1I_2I_3 + rk(I_2I_3 + I_1I_3 + I_1I_2). \tag{40}$$

It can be said that the Itakura-Saito NTF and the Kullback-Leibler NTF algorithms have a computational cost of the same order to that given by (40), if a constant cost is assumed for the computation of logarithms and exponentials.

The inspection of (40) reveals that the cost of the non-negative training tensor $\mathcal{V}_c$ factorization depends linearly on the number of the training recordings I_{c1} for each genre class. Besides the NTF, the computational cost of the proposed NTF classifier training in Section III-E involves tensor unfolding (of no cost), the Khatri-Rao matrix product $\mathbf{U}_c^{(3)} \odot \mathbf{U}_c^{(2)}$, its QR decomposition, and the matrix product $\mathbf{H}_c = \mathbf{R}_c\,[\mathbf{U}_c^{(1)}]^T$ for each genre. The just mentioned Khatri-Rao matrix product is computed at a cost of I_3I_2k flops. The QR decomposition can be performed at a cost of $2I_3I_2k^2$ flops, if the modified Gram-Schmidt method is used [43]. $\mathbf{H}_c$ can be computed at a cost of $I_{c1}(2k-1)k$ flops. The test phase involves (20)-(22), which implies a computational cost of $\mathcal{O}(I_{c1}k^2)$ for each genre.

TABLE VIII
3RD ORDER FROBENIUS NTF COMPUTATIONAL COST.

Term	Flops
$\alpha_{j\varrho_2\varrho_3}$	I_2I_3k
$\beta_{l\varrho_2\varrho_3}$	$(3k-1)I_1I_2I_3$
$u_{jl}^{(1)}$ given $\alpha_{j\varrho_2\varrho_3}$ and $\beta_{l\varrho_2\varrho_3}$	$4I_1I_2I_3k$
$\mathbf{U}^{(1)}$ per iteration	$(7k-1)I_1I_2I_3 + kI_2I_3$
$\mathbf{U}^{(i)}$, $i = 1,2,3$, per iteration	$3(7k-1)I_1I_2I_3 + k(I_2I_3 + I_1I_3 + I_1I_2)$
Frobenius NTF for r iterations	$3r(7k-1)I_1I_2I_3 + rk(I_2I_3 + I_1I_3 + I_1I_2)$

REFERENCES

[1] J.-J. Aucouturier and F. Pachet, “Representing musical genre: a state of the art,” *J. New Music Research*, Vol. 32, No. 1, pp. 83-93, 2003.

[2] N. Scaringella, G. Zoia, and D. Mlynek, “Automatic genre classification of music content,” *IEEE Signal Processing Mag.*, Vol. 23, No. 2, pp. 133-141, March 2006.

[3] T. Lidy and A. Rauber, “Evaluation of feature extractors and psycho-acoustic transformations for music genre classification,” in Proc. *6th Int. Conf. Music Information Retrieval*, pp. 34-41, September 2005.

[4] D. Perrott and R. O. Gjerdingen, "Scanning the dial: An exploration of factors in the identification of musical style," *Dept. Music, Northwestern University, Illinois, Res. Notes*, 1999.

[5] C. McKay and I. Gujinaga, "Musical genre classification: is it worth pursuing and how can it be improved?," in Proc. *7th Int. Conf. Music Information Retrieval*, pp. 34-41, September 2006.

[6] L. De Lathauwer, "*Signal Processing Based on Multilinear Algebra*", Ph.D. thesis, K.U. Leuven, E.E. Dept.-ESAT, Belgium, 1997.

[7] D. C. Kay, *Theory and Problems of Tensor Calculus*, New York: McGraw-Hill, 1988.

[8] T. G. Kolda and B. W. Bader: "Tensor Decompositions and Applications," *SIAM Review*, Vol. 51, No. 3, to appear.

[9] F. van der Heijden, R. P. W. Duin, D. de Ridder, and D. M. J. Tax, *Classification, Parameter Estimation and State Estimation*, London UK: Wiley, 2004.

[10] L. M. Bregman, "The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming," *USSR Computational Mathematics and Mathematical Physics*, Vol. 7, pp. 200-217, 1967.

[11] S. Sra and I. S. Dhillon, "Nonnegative matrix approximation: algorithms and applications," *Technical Report TR-06-27*, Computer Sciences, University of Texas at Austin, 2006.

[12] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech and Audio Processing*, Vol. 10, No. 5, pp. 293-302, July 2002.

[13] T. Li, M. Ogihara, and Q. Li, "A comparative study on content-based music genre classification," in Proc. *26th Annual ACM Conf. Research and Development in Information Retrieval*, pp. 282-289, July-August 2003.

[14] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kegl: "Aggregate Features and AdaBoost for Music Classification," *Machine Learning*, Vol. 65, No. 2-3, pp. 473–484, 2006.

[15] A. Holzapfel and Y. Stylianou, "Musical genre classification using nonnegative matrix factorization-based features," *IEEE Trans. Audio, Speech, and Language Processing*, Vol. 16, No. 2, pp. 424-434, February 2008.

[16] P. Cano, E. Gómez, F. Gouyon, P. Herrera, M. Koppenberger, B. Ong, X. Serra, S. Streich, and N. Wack, "ISMIR 2004 audio description contest," *MTG Technical Report: MTG-TR-2006-02*, 2006.

[17] E. Pampalk, A. Flexer, and G. Widmer, "Improvements of audio based music similarity and genre classification," in Proc. *6th Int. Symp. Music Information Retrieval*, pp. 628-633, 2005.

[18] J. J. Burred and A. Lerch, "A hierarchical approach to automatic musical genre classification," in Proc. *6th Int. Conf. Digital Audio Effects (DAFx)*, September 2003.

[19] A. Meng, P. Ahrendt, and J. Larsen, "Improving music genre classification by short-time feature integration," in Proc. *6th Int. Symp. Music Information Retrieval*, pp. 604-609, September 2005.

[20] J. G. A. Barbedo and A. Lopes, "Automatic genre classification of musical signals," *EURASIP Journal on Advances in Signal Processing*, Vol. 2007, Article ID 64960, 2007.

[21] Z. Cataltepe, Y. Yaslan, and A. Sonmez, "Music genre classification using MIDI and audio features," *EURASIP Journal on Advances in Signal Processing*, Vol. 2007, Article ID 36409, 2007.

[22] P. Paatero, U. Tapper, R. Aalto, and M. Kulmala. "Matrix factorization methods for analysing diffusion battery data," *Journal of Aerosol Science*, Vol. 22 (Supplement 1), pp. 273-276, 1991.

[23] D. D. Lee and H. S. Seung, "Algoritnms for non-negative matrix factorization," *Advances in Neural Information Processing Systems*, Vol. 13, pp. 556-562, 2001.

[24] S. Z. Li, X. Hou, H. Zhang, and Q. Cheng, "Learning spatially localized, parts-based representation," in Proc. *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1-6, 2001.

[25] N. D. Sidiropoulos, " Low-rank decomposition of multi-way arrays: A signal processing perspective," in Proc. *Sensor Array and Multichannel Signal Processing Workshop*, pp. 52-58, July 2004.

[26] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in Proc. *22nd Int. Conf. Machine Learning*, pp. 792-799, August 2005.

[27] T. Hazan, S. Polak, and A. Shashua, "Sparse image coding using a 3D non-negative tensor factorization," in Proc. *10th IEEE Int. Conf. Computer Vision*, Vol. 1, pp. 50-57, October 2005.

[28] C. Boutsidis, E. Gallopoulos, P. Zhang, and R. Plemmons, "PALSIR: A new approach to nonnegative tensor factorization", Poster presented at *Workshop Algorithms for Modern Massive Data Sets*, June 2006.

[29] M. Heiler and C. Schnörr, "Controlling sparseness in non-negative tensor factorization," in Proc. *9th European Conf. Computer Vision*, Vol. 1, pp. 56-67, May 2006.

[30] A. Cichocki, R. Zdunek, S. Choi, R. Plemmons, and S. Amari, "Non-negative tensor factorization using alpha and beta divergences," in Proc. *2007 IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Vol. III, pp. 1393-1396, April 2007.

[31] A. Cichocki, R. Zdunek, S. Choi, R. Plemmons, and S. Amari, "Novel multi-layer nonnegative tensor factorization with sparsity constraints," in Proc. *8th Int. Conf. Adaptive and Natural Computing Algorithms*, pp. 271-280, April 2007.

[32] S. Zafeiriou, "Discriminant non-negative matrix factorization algorithms," *IEEE Trans. Neural Networks*, Vol. 20, No. 2, pp. 217-235, February 2009.

[33] E. Benetos and C. Kotropoulos, "A tensor-based approach for automatic music genre classification," in Proc. *16th European Signal Processing Conf.*, August 2008.

[34] E. Benetos, M. Kotti, and C. Kotropoulos, "Large scale musical instrument identification," in Proc. *4th Sound and Music Computing Conference*, pp. 283-286, July 2007.

[35] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," *CUIDADO IST Project Report*, 2004.

[36] G. Tzanetakis, A. Ermolinskyi, and P. Cook, "Pitch Histograms in Audio and Symbolic Music Information Retrieval," *J. New Music Research*, Vol. 32, No. 2, pp. 143-152, 2003.

[37] E. Pampalk, S. Dixon, and G. Widmer, "Exploring Music Collections by Browsing Different Views," *Computer Music J.*, Vol. 28, No. 2, pp. 49-62, June 2004.

[38] MPEG-7, "Information Technology-Multimedia Content Description Interface-Part 4: Audio," *ISO/IEC JTC1/SC29/WG11 N5525*, March 2003.

[39] G. Tzanetakis, *Music Analysis, Retrieval and Synthesis for Audio Signals*, http://marsyas.sness.net/.

[40] B. Schölkopf, C. J. C. Burges, and A. J. Smola, *Advances in Kernel Methods: Support Vector Learning*, Cambridge MA, USA: MIT Press, 1999.

[41] I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik, "What size test set gives good error rate estimates?," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 52-64, January 1998.

[42] P. Smaragdis and J. Brown, "Non-negative matrix factorization for polyphonic music transcription," in Proc. *IEEE Workshop Applications of Signal Processing to Audio and Acoustics*, pp. 177-180, 2003.

[43] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Baltimore, MD: Johns Hopkins Univ. Press, 1996.

[44] R. Albright, J. Cox, D. Duling, A. Langville, and C. D Meyer, "Algorithms, initializations, and convergence for the nonnegative matrix factorization," preprint, 2006.