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...
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].

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.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Dataset</th>
<th>Classifier</th>
<th>Best Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>[12]</td>
<td>GTZAN</td>
<td>GMM</td>
<td>61.0</td>
</tr>
<tr>
<td>[13]</td>
<td>GTZAN</td>
<td>SVM - LDA</td>
<td>78.5</td>
</tr>
<tr>
<td>[3]</td>
<td>GTZAN</td>
<td>SVM</td>
<td>74.9</td>
</tr>
<tr>
<td>[14]</td>
<td>GTZAN</td>
<td>AdaBoost</td>
<td>82.5</td>
</tr>
<tr>
<td>[15]</td>
<td>GTZAN</td>
<td>GMM</td>
<td>74.0</td>
</tr>
<tr>
<td>[17]</td>
<td>MIREX 2004</td>
<td>NN - GMM</td>
<td>82.3</td>
</tr>
<tr>
<td>[3]</td>
<td>MIREX 2004</td>
<td>SVM</td>
<td>70.4</td>
</tr>
</tbody>
</table>
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 method able to obtain a parts-based representation of objects could be treated as a high-order generalization of the NMF. The minimization of (2) can be solved by using iterative multiplicative rules [23]. Frequently, additional constraints are 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 \( \mathbf{A} \) 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 \( \mathbf{A} \) is addressed by \( n \) indices, i.e., \( a_{i_1 \ldots i_n} \), where \( i_1 = 1, 2, \ldots , I_1 \). A 3rd order tensor is sketched in Figure 1.

Mode-i unfolding of the tensor \( \mathbf{A} \) yields the matrix \( \mathbf{A}^{(i)} \in \mathbb{R}^{I_i \times \hat{T}_i} \), where \( \hat{T}_i \equiv I_1 I_2 \cdots I_{i-1} I_{i+1} \cdots I_n \). In the following, the operations on tensors are expressed in matricized form [8]. The symbol \( \times \) stands for the i-mode product between a tensor and a matrix [6]. It can be computed via the matrix multiplication \( \mathbf{B}^{(i)} = \mathbf{U}^{(i)} \mathbf{A} \) 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 \( \mathbf{A} \in \mathbb{R}^{3 \times 5 \times 2} \) and the matrix \( \mathbf{U} \in \mathbb{R}^{5 \times 2} \) yields the 3rd order tensor \( \mathbf{B} = \mathbf{A} \times_2 \mathbf{U} \in \mathbb{R}^{3 \times 2 \times 2} \). The inner product of two nth order tensors \( \mathbf{A} \) and \( \mathbf{B} \) is denoted as \( \langle \mathbf{A}, \mathbf{B} \rangle \). The norm of tensor \( \mathbf{A} \) is defined as \( ||\mathbf{A}|| = \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle} \).

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 \( \mathbf{A} \) 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 \( \mathbf{A} \) is addressed by \( n \) indices, i.e., \( a_{i_1 \ldots i_n} \), where \( i_1 = 1, 2, \ldots, I_1 \). A 3rd order tensor is sketched in Figure 1.

Mode-i unfolding of the tensor \( \mathbf{A} \) yields the matrix \( \mathbf{A}^{(i)} \in \mathbb{R}^{I_i \times \hat{T}_i} \), where \( \hat{T}_i \equiv I_1 I_2 \cdots I_{i-1} I_{i+1} \cdots I_n \). In the following, the operations on tensors are expressed in matricized form [8]. The symbol \( \times \) stands for the i-mode product between a tensor and a matrix [6]. It can be computed via the matrix multiplication \( \mathbf{B}^{(i)} = \mathbf{U}^{(i)} \mathbf{A}^{(i)} \) 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 \( \mathbf{A} \in \mathbb{R}^{3 \times 5 \times 2} \) and the matrix \( \mathbf{U} \in \mathbb{R}^{5 \times 2} \) yields the 3rd order tensor \( \mathbf{B} = \mathbf{A} \times_2 \mathbf{U} \in \mathbb{R}^{3 \times 5 \times 2} \). The inner product of two nth order tensors \( \mathbf{A} \) and \( \mathbf{B} \) is denoted as \( \langle \mathbf{A}, \mathbf{B} \rangle \). The norm of tensor \( \mathbf{A} \) is defined as \( ||\mathbf{A}|| = \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle} \).

An n-order tensor \( \mathbf{A} \) 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.,

\[
\mathbf{A} = \mathbf{u}^{(1)} \otimes \mathbf{u}^{(2)} \otimes \cdots \otimes \mathbf{u}^{(n)}. \tag{3}
\]

That is, each element of the tensor in (3) is given by \( a_{i_1 i_2 \ldots i_n}^{(\ell)} = u^{(\ell)}_{i_1} u^{(\ell)}_{i_2} \cdots u^{(\ell)}_{i_n} \) for all \( i_\ell = 1, 2, \ldots, I_\ell \) \( \ell = 1, 2, \ldots, n \). The rank of an arbitrary n-order tensor \( \mathbf{A} \) is the minimal number of rank-1 tensors that yield \( \mathbf{A} \) 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 $\circ$, 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 3rd 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 [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 $n$th 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 $n$th 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

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

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 $n$th order tensors $X, Y \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n}$. The following identity holds:

$$D_{\phi}(X, Y) = \sum_{i=1}^{I_1} \sum_{j=1}^{I_2} \cdots \sum_{i=1}^{I_n} D_{\phi}(x_{i_1 j_2 \ldots i_n}, y_{i_1 j_2 \ldots i_n}).$$

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 $V \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n}$ into a sum of $k$ rank-1 tensors:

$$V = \sum_{j=1}^{k} U^{(1)}_{j} \circ U^{(2)}_{j} \circ \cdots \circ U^{(n)}_{j},$$

where $U^{(j)}_{j} \in \mathbb{R}^{I_j}$ and $j = 1, 2, \ldots, k$. Let $U^{(i)} \triangleq \{u^{(i)}_{1}, u^{(i)}_{2}, \ldots, u^{(i)}_{k}\}$, $i = 1, 2, \ldots, n$. Obviously $U^{(i)} \in \mathbb{R}^{I_i \times k}$. Let us introduce the compact notation

$$\bigotimes_i U \triangleq U^{(n)} \odot \cdots \odot U^{(i+1)} \odot U^{(i-1)} \odot \cdots \odot U^{(1)}.$$

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

$$V^{(i)} = \sum_{j=1}^{k} \sum_{i_1}^{I_1} \cdots \sum_{i_n}^{I_n} \sum_{g_{i_1}+1}^{I_1} \cdots \sum_{g_{i_n}+1}^{I_n} U^{(i)} = \sum_{j=1}^{k} \sum_{i_1}^{I_1} \cdots \sum_{i_n}^{I_n} \sum_{g_{i_1}+1}^{I_1} \cdots \sum_{g_{i_n}+1}^{I_n} H^{(i)}.$$
The following minimization problem with Bregman divergences is solved
\[
\min_{u^{(i)}_j \geq 0} D_\phi \left( \sum_{j=1}^{k} u^{(1)}_{j} \circ u^{(2)}_{j} \circ \cdots \circ u^{(n)}_{j}, \mathbf{V} \right)
\]
(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 $u^{(i)}_j$ denoted as $u^{(i)}_{j}$:
\[
u_{j}^{(i)} \leftarrow \nu_{j}^{(i)} \exp \left( \frac{\sum_{\theta=0}^{\infty} \alpha^{(i)}_{j \theta} \cdots \theta_{i-1} \cdots \theta_{n}}{\sum_{\theta=0}^{\infty} \alpha^{(i)}_{j \theta} \cdots \theta_{i-1} \cdots \theta_{n}} \right)
\]
(11)
where $\nu_{j}^{(i)}$ is the $i$th element of vector $u^{(i)}_j$ before updating, $j = 1, 2, \ldots, k$, $i = 1, 2, \ldots, n$, and $\alpha^{(i)}_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} = \nu_{j+1}^{(i)} \cdots (i-1) (i+1) \cdots (n)$.
\[
\beta_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} = \sum_{j=1}^{k} \nu_{j}^{(1)} \cdots (i-1) (i+1) \cdots (n) .
\]
(12)
For $\phi(x) = x^2$ (i.e., when the Frobenius norm is used), the resulting update rule is:
\[
u_{j}^{(i)} \leftarrow \nu_{j}^{(i)} \frac{\alpha^{(i)}_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} \nu^{(i)}_{j+1} \cdots (i-1) (i+1) \cdots (n)}{\sum_{j=1}^{k} \alpha^{(i)}_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} \nu^{(i)}_{j+1} \cdots (i-1) (i+1) \cdots (n)}
\]
(13)
Finally, for $\phi(x) = -\log(x)$ (i.e., when the IS distance is employed), the update rule is:
\[
u^{(i)}_{j} \leftarrow \nu^{(i)}_{j} \sum_{\theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} \frac{\alpha^{(i)}_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} \nu^{(i)}_{j+1} \cdots (i-1) (i+1) \cdots (n)}{\sum_{j=1}^{k} \alpha^{(i)}_{j \theta_1 \cdots \theta_{i-1} \theta_{i} \cdots \theta_{n}} \nu^{(i)}_{j+1} \cdots (i-1) (i+1) \cdots (n)}
\]
(14)
In order to apply the aforementioned NTF algorithms to an $n$th order tensor $\mathbf{V}$, 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 $u^{(i)}_j$ 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 $\mathbf{V}_c \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, with $I_3$ 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 $\mathbf{V}_c \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, $c = 1, 2, \ldots, C$, i.e.:
\[
\mathbf{V}_c = \sum_{j=1}^{k} u^{(1)}_{j} \circ u^{(2)}_{j} \circ u^{(3)}_{j}.
\]
(16)
2) Determine the 1st mode of the tensor $\mathbf{V}_c$ by unfolding [6], [8]:
\[
\mathbf{V}_{c_{[1]}} = \mathbf{U}_{c}^{(1)} \left( \mathbf{U}_{c}^{(3)} \odot \mathbf{U}_{c}^{(2)} \right)^T
\]
(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 $I_1 \times (I_3 I_2)$. In the following, we deal with the transpose of matrix $\mathbf{V}_{c_{[1]}}$ i.e.
\[
\mathbf{V}_{c_{[1]}}^T = \left( \mathbf{U}_{c}^{(3)} \odot \mathbf{U}_{c}^{(2)} \right) \left( \mathbf{U}_{c}^{(1)} \right)^T
\]
(18)
3) Perform QR decomposition on the basis matrix $\mathbf{U}_{c}^{(3)} \odot \mathbf{U}_{c}^{(2)}$:
\[
\mathbf{U}_{c}^{(3)} \odot \mathbf{U}_{c}^{(2)} = \mathbf{Q}_c \mathbf{R}_c
\]
(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 $\mathbf{H}_c = \mathbf{R}_c \left( \mathbf{U}_{c}^{(1)} \right)^T$. 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:
\[
\mathbf{h}_c = \mathbf{Q}_c^T \mathbf{v}_t
\]
(20)
and has length $k$.
5) Let $CSM_{tm}(c)$ be the cosine similarity measure (CSM) between $\mathbf{h}_c$ and $\mathbf{h}_m$, $m = 1, 2, \ldots, I_1$ (i.e. the $m$th column of matrix $\mathbf{H}_c$):
\[
CSM_{tm}(c) = \frac{\mathbf{h}_c^T \mathbf{h}_m}{||\mathbf{h}_c|| ||\mathbf{h}_m||}
\]
(21)
Let $CSM_{t_{[m]}}(c)$ denote the $m$th largest element in the set $\{CSM_{tm}(c)\}$, $m = 1, 2, \ldots, I_1$. The decision taken by the classifier is based on
\[
\widehat{c}_t = \arg\max_{c=1,2,\ldots,C} \{CSM_{tm}(c)\}
\]
(22)
where $K \ll I_1$ (e.g. $K = 3$). The class label of the test pattern $\mathbf{v}_t$ is determined by the maximum among $\widehat{c}_t$, i.e.:
\[
\hat{c}_t = \arg\max_{c=1,2,\ldots,C} \{CSM_{tm}(c)\}
\]
(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.
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 $\mathbf{V}$ 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, Spectrum Spread (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: $J = \text{tr}(\mathbf{S}_w^{-1} \mathbf{S}_b)$, where $\text{tr}(\cdot)$ 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 $2^{207} - 1$, 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

![Fig. 2. The proposed supervised NTF classifier.](image-url)
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 \( V(2) \in \mathbb{R}^{207 \times 30000} \) is computed from tensor \( V \in \mathbb{R}^{1000 \times 207 \times 300} \). 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 \( V_c \) should satisfy [8] (and references therein)

\[
k \leq \min \{I_1 I_2, I_1 I_3, I_2 I_3 \}. \tag{24}
\]

Since by the experimental protocol \( I_1 \) and \( I_3 \) are fixed to 90 and 30, respectively, the inequality (24) implies \( k \leq 30 I_2 \), 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 backpropagation 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 \( V(2) \in \mathbb{R}^{207 \times 30000} \) using 10-fold cross validation, where \( I_2 \in \{60, 70, 80, 90\} \).

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

<table>
<thead>
<tr>
<th>Classifier</th>
<th>80 Feature Subset</th>
<th>MARSYAS Feature Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTF Frobenius</td>
<td>78.9%</td>
<td>68.3%</td>
</tr>
<tr>
<td>SVM</td>
<td>77.2%</td>
<td>67.6%</td>
</tr>
<tr>
<td>MLP</td>
<td>77.0%</td>
<td>69.1%</td>
</tr>
<tr>
<td>NTF KL</td>
<td>70.4%</td>
<td>61.4%</td>
</tr>
<tr>
<td>NTF IS</td>
<td>63.6%</td>
<td>55.0%</td>
</tr>
</tbody>
</table>

**Fig. 3.** Average music genre classification accuracy for the various feature subsets.
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 is less performing than the NTF classifier with the Frobenius norm.

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 music genre and the rows to the actual one. For the NTF 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.

The 80 features selected by the branch-and-bound algorithm.
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.

<table>
<thead>
<tr>
<th>Genre</th>
<th>Blues</th>
<th>Classical</th>
<th>Country</th>
<th>Disco</th>
<th>Hiphop</th>
<th>Jazz</th>
<th>Metal</th>
<th>Pop</th>
<th>Reggae</th>
<th>Rock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blues</td>
<td>58</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Classical</td>
<td>1</td>
<td>56</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Country</td>
<td>2</td>
<td>0</td>
<td>87</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Disco</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Hiphop</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>11</td>
<td>67</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Jazz</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Metal</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Pop</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Reggae</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Rock</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Genre</th>
<th>Blues</th>
<th>Classical</th>
<th>Country</th>
<th>Disco</th>
<th>Hiphop</th>
<th>Jazz</th>
<th>Metal</th>
<th>Pop</th>
<th>Reggae</th>
<th>Rock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blues</td>
<td>58</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Classical</td>
<td>1</td>
<td>56</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Country</td>
<td>2</td>
<td>0</td>
<td>87</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Disco</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Hiphop</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>11</td>
<td>67</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Jazz</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Metal</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Pop</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Reggae</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Rock</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

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.

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.

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.

<table>
<thead>
<tr>
<th>Genre</th>
<th>Blues</th>
<th>Classical</th>
<th>Country</th>
<th>Disco</th>
<th>Hiphop</th>
<th>Jazz</th>
<th>Metal</th>
<th>Pop</th>
<th>Reggae</th>
<th>Rock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blues</td>
<td>58</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Classical</td>
<td>1</td>
<td>56</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Country</td>
<td>2</td>
<td>0</td>
<td>87</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Disco</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Hiphop</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>11</td>
<td>67</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Jazz</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Metal</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Pop</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Reggae</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Rock</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

APPENDIX I

A. Problem Formulation

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

$$\min \sum_{j=1}^{k} u_{j}^{(1)} \odot u_{j}^{(2)} \odot \cdots \odot u_{j}^{(n)} , \mathbf{V} .$$

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

$$D_{\phi} \left( \sum_{j=1}^{k} u_{j}^{(1)} \odot u_{j}^{(2)} \odot \cdots \odot u_{j}^{(n)} , \mathbf{V} \right) = \sum_{l=1}^{L} D_{\phi} \left( \sum_{j=1}^{k} u_{j}^{(1)} \odot u_{j}^{(2)} \odot \cdots \odot u_{j}^{(l-1)} u_{j}^{(l)} \odot u_{j}^{(l+1)} \odot \cdots \odot u_{j}^{(n)} , \mathbf{V}_{i=l-1} \right)$$

(25)

where $\mathbf{V}_{i=l-1} \in \mathbb{R}^{I_{l} \times I_{l+1} \times \cdots \times I_{n}}$ is a sub-tensor with the $i$th index fixed to $l$ whose elements are denoted by $v_{i_{1} \odot \cdots \odot i_{n}}, \forall l = 1, 2, \ldots, I_\ell$ 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 $F(u_{j}^{(i)})$ denote the divergence term in (25), i.e.

$$F(u_{j}^{(i)}) = D_{\phi} \left( \sum_{j=1}^{k} u_{j}^{(1)} \odot u_{j}^{(2)} \odot \cdots \odot u_{j}^{(l-1)} u_{j}^{(l)} \odot u_{j}^{(l+1)} \odot \cdots \odot u_{j}^{(n)} , \mathbf{V}_{i=l-1} \right).$$

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

\[ F(u_i^{(i)}) = \sum_{\phi} \phi \left( \sum_{j=1}^{k} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

\[ - \phi(v_{k+1} \cdots v_{l+1}) = \psi(v_{k+1} \cdots v_{l+1}) \]  

\[ \psi \left( \sum_{j=1}^{k} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

(27)

where \( \psi(x) = \phi(x) \). The following auxiliary function for \( F(u_i^{(i)}) \) is proposed:

\[ G(u_i^{(i)}, u_i^{(i)}) = \sum_{\phi} \phi \left( \sum_{j=1}^{k} \lambda_{\theta \cdots \theta - 1} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

\[ - \phi(v_{k+1} \cdots v_{l+1} \cdots v_{n}) = \psi(v_{k+1} \cdots v_{l+1} \cdots v_{n}) \]  

\[ \psi \left( \sum_{j=1}^{k} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

(28)

where

\[ \lambda_{\theta \cdots \theta - 1} = \frac{\lambda_{\theta \cdots \theta - 1}^{(1)} \cdots \lambda_{\theta \cdots \theta - 1}^{(i)} \cdots \lambda_{\theta \cdots \theta - 1}^{(i+1)} \cdots \lambda_{\theta \cdots \theta - 1}^{(n)}}{\lambda_{\theta \cdots \theta - 1}^{(1)} \cdots \lambda_{\theta \cdots \theta - 1}^{(i)} \cdots \lambda_{\theta \cdots \theta - 1}^{(i+1)} \cdots \lambda_{\theta \cdots \theta - 1}^{(n)}} \]

(29)

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

C. Minimization of the auxiliary function

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

\[ \frac{\partial G(u_i^{(i)}, u_i^{(i)})}{\partial u_i^{(i)}} = 0 \]

(30)

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

\[ \frac{\partial G(u_i^{(i)}, u_i^{(i)})}{\partial u_i^{(i)}} = \sum_{\phi} \phi \left( u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

\[ \psi \left( \sum_{j=1}^{k} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

\[ - \sum_{\phi} \phi(v_{k+1} \cdots v_{l+1} \cdots v_{n}) \psi \left( \sum_{j=1}^{k} u_j^{(1)} \cdots u_{j-1}^{(i-1)} u_j^{(i)} u_{j+1}^{(i+1)} \cdots u_{j+n}^{(n)} \right) \]

(31)

The equation

\[ \frac{\partial G(u_i^{(i)}, u_i^{(i)})}{\partial u_i^{(i)}} = 0 \]

(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:

\[ u_j^{(i)} \leftarrow u_j^{(i)} \cdot \psi^{-1} \left( \sum_{\phi} \phi(v_{k+1} \cdots v_{l+1} \cdots v_{n}) \cdot \left( \sum_{\phi} \phi(v_{k+1} \cdots v_{l+1} \cdots v_{n}) \right) \right) \]

(33)

where \( \alpha_{j \theta \cdots \theta - 1} \) and \( \beta_{j \theta \cdots \theta - 1} \) are defined in (12) and (13), respectively. The update rule (33) should be applied to all elements \( u_j^{(i)} \) for \( j = 1, 2, \ldots, k \), \( i = 1, 2, \ldots, n \), and \( l = 1, 2, \ldots, l_i \). It should be noted that \( \psi \) is multiplicative for the Frobenius norm, since \( \psi(x) = \frac{x^2}{\psi(x)} = \psi(x) \).

D. Proof of Convergence

To prove the convergence of the multiplicative update rule for the Frobenius NTF algorithm (14), one needs to show that \( F(u_i^{(i)}) \leq F(u_i^{(i)}) \). Using the definition of \( F(u_i^{(i)}) \), \( \alpha_{j \theta \cdots \theta - 1} \) and \( \beta_{j \theta \cdots \theta - 1} \) given in (27), (12),
F_{\mathbf{u}_j^{(i)}} - F_{\mathbf{u}_j^{(i)}} = \sum_{\psi_0} \left( \sum_{j=1}^{k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right)
\cdot \psi(\mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 - \mathbf{v}_j \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0)
\cdot \sum_{\psi_0} D_0 \left( \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right)
\cdot \psi(\mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 - \mathbf{v}_j \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0)
(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
\sum_{\psi_0} \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 = \sum_{\psi_0} \left( \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right) \cdot \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0
(36)

Using (36) into the algebraic manipulations of the first term in (35), we conclude that it is sufficient to prove
\sum_{\psi_0} \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \left( \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right) \geq \sum_{\psi_0} \left( \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right)
(37)

(14) implies that
\sum_{\psi_0} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 = \sum_{\psi_0} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 = \sum_{\psi_0} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0
(38)

Using (38), the inequality (37) is rewritten as
\sum_{\psi_0} \left( \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right) \cdot \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \geq \sum_{\psi_0} \left( \sum_{j=1} {k} \alpha_j \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0 \cdot \mathbf{u}_j^{(i)} \right)
\cdot \beta_0 \mathbf{v}_0 \cdots \mathbf{v}_{-1} \cdots \mathbf{v}_0
(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 \( \mathbf{U}^{(1)} \in \mathbb{R}^{k \times k} \) having columns \( \mathbf{u}_j^{(i)} \). The first entry refers to terms \( \alpha_j \mathbf{v}_0 \mathbf{w}_0 \mathbf{u}_j^{(i)} \) defined by (12), which are \( kI_2I_3 \) in total and each of them requires 1 multiplication. The second entry refers to terms \( \beta_j \mathbf{v}_0 \mathbf{w}_0 \mathbf{u}_j^{(i)} \) 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 \( \mathbf{u}_j^{(i)} \) given by (14) is \( 2(I_2I_3 + 1) \) multiplications and \( 2I_2I_3 - 1 \) 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:

\[ 3r(7k - 1)I_1I_2I_3 + r(kI_2I_3 + I_1I_3 + I_1I_2) \]
(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 \( \mathbf{\nabla}_c \) factorization depends linearly on the number of the training recordings \( I_1 \) 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 \( \mathbf{U}^{(3)} \) \( \odot \mathbf{U}^{(2)} \), its QR decomposition, and the matrix product \( \mathbf{H}_c = \mathbf{R}_c \mathbf{[U]^{(1)}} \) 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 \( 2I_2I_3k^2 \) flops, if the modified Gram-Schmidt method is used [43]. \( \mathbf{H}_c \) can be computed at a cost of \( I_1I_2(2k - 1)k \) flops. The test phase involves (20)-(22), which implies a computational cost of \( O(\sum I_1)k^2 \) for each genre.

### TABLE VIII
<table>
<thead>
<tr>
<th>Term</th>
<th>Flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha_j \mathbf{v}_0 \mathbf{w}_0 \mathbf{u}_j^{(i)} ) given ( \alpha_j \mathbf{v}_0 \mathbf{w}_0 \mathbf{u}_j^{(i)} ) and ( \beta_j \mathbf{v}_0 \mathbf{w}_0 \mathbf{u}_j^{(i)} )</td>
<td>( 4kI_2I_3 )</td>
</tr>
<tr>
<td>( \psi(\mathbf{v}<em>0 \cdots \mathbf{v}</em>{-1} \cdots \mathbf{v}_0 - \mathbf{v}<em>j \cdots \mathbf{v}</em>{-1} \cdots \mathbf{v}_0) ) per iteration</td>
<td>( 7k - 1)I_1I_2I_3 + kI_2I_3 )</td>
</tr>
<tr>
<td>( \mathbf{U}^{(j)}, j = 1,2,3, ) per iteration</td>
<td>( 3(7k - 1)I_1I_2I_3 + kI_2I_3 + I_1I_3 + I_1I_2) )</td>
</tr>
<tr>
<td>Frobenius NTF for ( r ) iterations</td>
<td>( 3r(7k - 1)I_1I_2I_3 + r(kI_2I_3 + I_1I_3 + I_1I_2) )</td>
</tr>
</tbody>
</table>

### REFERENCES
