Greek folk music classification into two genres using lyrics and audio via canonical correlation analysis

Nikoletta Bassiou, Constantine Kotropoulos, and Anastasios Papazoglou-Chalikias
Department of Informatics, Aristotle University of Thessaloniki
Thessaloniki 54124, GREECE
Email: nbassiou@gmail.com, costas@aiia.csd.auth.gr, tasos@tpapazoglou.com

***Abstract*—We are interested in Greek folk music genre classification by resorting to canonical correlation analysis (CCA). Here, the genre is related to the place of origin of the song. The CCA learns a linear transformation of the song lyrics descriptors that is highly correlated with their genre labels as well as another linear transformation of the audio features extracted from music recordings, which is maximally correlated with their genre labels. In the latter task, thanks to the deep CCA (DCCA), deep nonlinear transformations of the audio features are learnt, which are maximally correlated with the genre labels. Experimental findings are disclosed for a two-class genre recognition problem, employing folk songs originated from Pontus and Asia Minor. It is demonstrated that the CCA achieves an average accuracy of 97.02% across the 5 folds, when the term frequency-inverse document frequency features model the song lyrics. By modeling the music signal of each song with 28 mel-frequency cepstral coefficients (MFCCs) extracted from each frame and averaged over all frames, the average accuracy of the CCA drops to 72.9% across the 5 folds. The DCCA yields an accuracy of 69% for audio-based genre recognition.**

***Keywords*—Canonical Correlation Analysis; Least-Squares Regression; Deep Canonical Correlation Analysis, Greek Folk Music Classification.**

I. Introduction

Music Information Retrieval (MIR) has been developed mainly for Western popular and classical music. However, the interest for non-Western music continuously grows as is evidenced by the increasing number of papers in recent MIR conferences. Computational methods for automatic classification and topological clustering of large folk music databases are described in [1]. A platform that extracts and explores pitch annotations in non-Western music, providing musicologically meaningful representations can be found in [2].

Greek folk music extends far back in time. It consists of compositions, usually characterized by the place of origin, where the songs are performed or created, such as Pontus, Asia Minor, Macedonia, Epirus, Thrace, Aegean islands, etc. Apart from regional criteria, Greek folk songs are classified into akritic, historical, klephtic, ballads, religious, love, wedding, satiric, immigrant, lament, work, proverbial, lullabies, and baby dandling ones [3]. They cover the whole spectrum of social life, including human life milestones, the nation's history, or community celebrations.

In this paper, we are interested in Greek folk music genre classification by resorting to canonical correlation analysis (CCA). Here, the genre is related to the place of origin of the song. The CCA finds the maximal correlations between two sets of random vectors [4]. Such random vectors may capture two "different views" of the same underlying pattern. One popular use of the CCA is in supervised learning. That is, when one view is derived from the data and the other view is derived from the class labels [5]–[7]. In particular, the CCA is exploited to learn a linear transformation of the song lyrics descriptors that is maximally correlated with the song genre labels as well another linear transformation of the audio features extracted from each song recording, which is also highly correlated with the genre labels. In the latter task, deep nonlinear transformations of the audio features maximally correlated with the genre labels are also learnt by means of the so-called deep CCA (DCCA) [8].

The major contribution of the paper is in the experimental findings reported for a two-class genre recognition problem, which employs folk songs originated from Pontus and Asia Minor. It is demonstrated that the CCA achieves an average accuracy of 97.02% across the 5 folds, when the term frequency-inverse document frequency (tf-idf) weights model the song lyrics. However, audio-based genre classification turned out to be a tough problem for the two-class problem under study. By modeling the music signal of each song with 28 mel-frequency cepstral coefficients (MFCCs) extracted from each frame and averaged next over all frames, the average accuracy of the CCA drops to 72.9% across the 5 folds. For audio-based genre classification using MFCCs, the DCCA yields an accuracy of 69%. In this task, the difference between the accuracy of the CCA and that of the DCCA is not statistically significant at 95% level of confidence. However, in certain folds of the cross-validation setting, the difference between the accuracies of the CCA and the DCCA is found to be statistically significant.

The outline of the paper is as follows. Section II describes the CCA and the DCCA. The dataset used in the experiments and the extracted features are discussed in Section III. The experiments conducted are detailed in Section IV and conclusions are drawn in Section V.

II. Classifiers based on Canonical Correlation

Throughout the paper, scalars appear as lowercase letters (e.g., λ_x), vectors are denoted by lowercase boldface letters (e.g., $\mathbf{x}$), and matrices are indicated by uppercase boldface letters (e.g., $\mathbf{X}$). $\mathbf{I}$ stands for the identity matrix of compatible dimensions, $\mathbf{1}$ is the vector of ones of compatible dimensions, $^\top$ denotes vector/matrix transposition, and $\|\mathbf{x}\|_2$ denotes the ℓ_2 norm of vector $\mathbf{x}$. Lowercase italic boldface letters are reserved for random vectors (e.g, $\boldsymbol{x}$). $\mathbb{R}$ and $\mathbb{Z}$ denote the fields of real, and integer numbers, respectively.

A. Canonical Correlation Analysis

CCA has been applied successfully in various applications [9], including natural language processing [10], [11], speech processing [12], [13], and multimodal signal processing [14]. It uses two views of a set of patterns and projects them onto a lower-dimensional space in which they are maximally correlated. CCA has also been used for supervised learning, where one view is derived from the data and the other view is derived from the class labels [5]–[7]. In this setting, the data are projected onto a lower-dimensional space dictated by the label information.

Formally, let $\boldsymbol{x} \in \mathbb{R}^{d\times 1}$ and $\boldsymbol{y} \in \mathbb{R}^{k\times 1}$ be two random vectors with covariance matrices $\boldsymbol{\Sigma_x} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{\Sigma_y} \in \mathbb{R}^{k\times k}$, respectively. Let $\boldsymbol{\Sigma_{xy}} \in \mathbb{R}^{d\times k}$ denote the cross-covariance matrix of the aforementioned random vectors. CCA computes two projection vectors $\mathbf{w}_x \in \mathbb{R}^{d\times 1}$ and $\mathbf{w}_y \in \mathbb{R}^{k\times 1}$, such that the correlation coefficient

$$\rho = \frac{\mathbf{w}_x^\top \, \boldsymbol{\Sigma_{xy}} \, \mathbf{w}_y}{\sqrt{\mathbf{w}_x^\top \, \boldsymbol{\Sigma_x} \, \mathbf{w}_x} \sqrt{\mathbf{w}_y^\top \, \boldsymbol{\Sigma_y} \, \mathbf{w}_y}} \tag{1}$$

is maximized. Let $\mathbf{X} = \left[\mathbf{x}_1|\mathbf{x}_2|\ldots|\mathbf{x}_n\right] \in \mathbb{R}^{d\times n}$ be the data matrix and $\mathbf{Y} = \left[\mathbf{y}_1|\mathbf{y}_2|\ldots|\mathbf{y}_n\right] \in \mathbb{R}^{k\times n}$ be the label matrix. Assume that both $\mathbf{x}_i$ and $\mathbf{y}_i$ are centered. If the covariance matrices in (1) are replaced by sample dispersion matrices, the following optimization problem should be solved:

$$(\mathbf{w}_x^*, \mathbf{w}_y^*) = \underset{\mathbf{w}_x,\, \mathbf{w}_y}{\operatorname{argmax}} \quad \frac{\mathbf{w}_x^\top \, \mathbf{X}\mathbf{Y}^\top \mathbf{w}_y}{\sqrt{\mathbf{w}_x^\top \, \mathbf{X}\mathbf{X}^\top \, \mathbf{w}_x} \sqrt{\mathbf{w}_y^\top \, \mathbf{Y}\mathbf{Y}^\top \, \mathbf{w}_y}} \tag{2}$$

The objective function in (2) is the sample correlation coefficient, which is invariant to the scaling of $\mathbf{w}_x$ and $\mathbf{w}_y$. Accordingly, the CCA optimization problem can be expressed as a constrained optimization problem, i.e.,

$$\begin{aligned} \underset{\mathbf{w}_x,\, \mathbf{w}_y}{\operatorname{argmax}} \quad & \mathbf{w}_x^\top \, \mathbf{X}\mathbf{Y}^\top \mathbf{w}_y \\ \text{subject to} \quad & \mathbf{w}_x^\top \, \mathbf{X}\mathbf{X}^\top \, \mathbf{w}_x = 1, \\ & \mathbf{w}_y^\top \, \mathbf{Y}\mathbf{Y}^\top \, \mathbf{w}_y = 1. \end{aligned} \tag{3}$$

If $\mathbf{Y}\mathbf{Y}^\top$ is non-singular, $\mathbf{w}_x^*$ can be found by solving

$$\begin{aligned} \max_{\mathbf{w}_x} \quad & \mathbf{w}_x^\top \, \mathbf{X}\mathbf{Y}^\top \left(\mathbf{Y}\mathbf{Y}^\top\right)^{-1} \mathbf{Y}\mathbf{X}^\top \, \mathbf{w}_x \\ \text{subject to} \quad & \mathbf{w}_x^\top \, \mathbf{X}\mathbf{X}^\top \, \mathbf{w}_x = 1. \end{aligned} \tag{4}$$

The just mentioned assumption for $\mathbf{Y}\mathbf{Y}^\top$ can be easily maintained in practice. Simply, start with a class membership indicator matrix (i.e., append for each pattern $\mathbf{x}_i$, a vector having 1 in the entry associated to the class it belongs to and 0 to all other entries) and apply centering [5]. The solution of (4) is the eigenvector corresponding to the top eigenvalue η of the following generalized eigenvalue problem:

$$\mathbf{X}\mathbf{Y}^\top \left(\mathbf{Y}\mathbf{Y}^\top\right)^{-1} \mathbf{Y}\mathbf{X}^\top \, \mathbf{w}_x = \eta \, \mathbf{X}\mathbf{X}^\top \, \mathbf{w}_x. \tag{5}$$

Under certain orthonormality constraints, it is possible to obtain multiple projection vectors by retaining the top k eigenvectors of the generalized eigenvalue problem (5) [7], [9]. To prevent overfitting and to avoid the singularity of $\mathbf{X}\mathbf{X}^\top$ and $\mathbf{Y}\mathbf{Y}^\top$ two regularization terms, $\lambda_x \, \mathbf{I}$ and $\lambda_y \, \mathbf{I}$ with $\lambda_x > 0$ and $\lambda_y > 0$ can be inserted in (5), arriving at the so called regularized CCA [9], [15], i.e.,

$$\mathbf{X}\mathbf{Y}^\top \left(\mathbf{Y}\mathbf{Y}^\top + \lambda_y \mathbf{I}\right)^{-1} \mathbf{Y}\mathbf{X}^\top \, \mathbf{w}_x = \eta \left(\mathbf{X}\mathbf{X}^\top + \lambda_x \mathbf{I}\right) \mathbf{w}_x. \tag{6}$$

Pattern classification can be addressed in a least-squares formulation. More specifically, starting from a data matrix $\tilde{\mathbf{X}} \in \mathbb{R}^{d\times n}$ and scalar class labels $y_i \in \{1, 2, \ldots, k\}$, $i = 1, 2, \ldots, n$, where k is the number of classes, create a centered data matrix $\mathbf{X}$ and centered targets $t_i = y_i - \overline{y}$, where $\overline{y}$ denotes the average class label. Collect the centered targets in the row vector $\mathbf{t} \in \mathbb{R}^{1\times n}$ and seek for the projection vector $\mathbf{w} \in \mathbb{R}^{d\times 1}$ minimizing the sum-of-squares cost function [5], [6]:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} |\mathbf{w}^\top \, \mathbf{x}_i - t_i|^2 = \|\mathbf{w}^\top \mathbf{X} - \mathbf{t}\|_2^2. \tag{7}$$

Having learnt $\mathbf{w}^*$, which minimizes (7) in a training set created by sampling $\mathbf{X}$, the class label of an unseen test data sample $\mathbf{z}$ can be predicted by rounding

$$\hat{y}(\mathbf{z}) = \overline{y} + (\mathbf{w}^*)^\top \left(\mathbf{z} - \overline{\mathbf{x}}\right) \tag{8}$$

where $\overline{\mathbf{x}}$ is the average data sample in the training set. The just described regression framework was extended for class labels coded as multivariate centered targets, i.e., $\mathbf{t}_i \in \mathbb{R}^{k\times 1}$ [5]. Let $\mathbf{T} = \left[\mathbf{t}_1|\mathbf{t}_2|\ldots|\mathbf{t}_n\right] \in \mathbb{R}^{k\times n}$. Then, (7) is generalized to

$$\min_{\mathbf{W}} \sum_{i=1}^{n} \|\mathbf{W}^\top \, \mathbf{x}_i - \mathbf{t}_i\|_2^2 = \|\mathbf{W}^\top \mathbf{X} - \mathbf{T}\|_F^2 \tag{9}$$

where $\mathbf{W} \in \mathbb{R}^{d\times k}$ is a projection matrix and $\|\mathbf{A}\|_F$ denotes the Frobenius norm of matrix $\mathbf{A}$. The solution of (9) is given by [5], [6]:

$$\mathbf{W}_{LS} = \left(\mathbf{X}\mathbf{X}^\top\right)^\dagger \, \mathbf{X} \, \mathbf{T}^\top \tag{10}$$

where $\mathbf{A}^\dagger$ denotes the Moore-Penrose pseudo-inverse of matrix $\mathbf{A}$. Having learnt $\mathbf{W}_{LS}$ in a training set, an unseen test data sample $\mathbf{z}$ is classified to the class

$$\underset{j=1,2,\ldots,k}{\operatorname{argmax}} \quad \overline{y}_j + \mathbf{w}_j^\top \left(\mathbf{z} - \overline{\mathbf{x}}\right) \tag{11}$$

where $\overline{y}_j$ is the jth element of the average class indicator vector $\overline{\mathbf{y}}$ and $\mathbf{w}_j$ is the jth column of the projection matrix $\mathbf{W}_{LS}$.

Under mild conditions, for the particular choice $\mathbf{T} = \left(\mathbf{Y}\mathbf{Y}^\top\right)^{-\frac{1}{2}} \mathbf{Y}$, an equivalence exists between the solution of the least squares problem (10) and the matrix $\mathbf{W}_{CCA}$ formed by the top k eigenvectors of the generalized eigenvalue problem (5) for classifiers, such as the k-Nearest Neighbor and the linear support vector machines (SVMs) employing the Euclidean distance [7]. Moreover, if the class indicator vectors are centered (i.e., $\mathbf{Y}\mathbf{1} = \mathbf{0}$), then the target vectors in $\mathbf{T}$ are also centered. Otherwise, centering is needed for $\mathbf{T}$. In addition to the straightforward choice $\mathbf{Y}$ with elements $Y_{ij} = 1$, if $\mathbf{x}_i$ belongs to class j and 0 otherwise, other choices are the matrix $\mathbf{Y}'$ with elements

$$Y'_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ belongs to class } j \\ -\frac{1}{k-1} & \text{otherwise,} \end{cases} \tag{12}$$

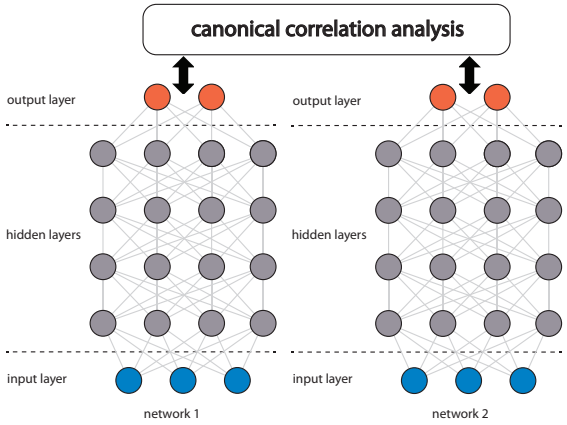


Fig. 1. A schematic representation of DCCA that consists of two deep neural networks. The networks are jointly trained so that the correlation between the output layers of the two networks is maximized. In this example, both networks have $L = 4$ hidden layers with $c_1 = c_2 = 4$ nodes (in grey), $n = 3$ input nodes (in blue), and $o = 2$ output nodes (in orange).

or the matrix $\mathbf{Y}''$ with elements

$$Y''_{ij} = \begin{cases} \sqrt{\frac{n}{n_j}} - \sqrt{\frac{n_j}{n}} & \text{if } \mathbf{x}_i \text{ belongs to class } j \\ -\sqrt{\frac{n_j}{n}} & \text{otherwise,} \end{cases} \quad (13)$$

where n_j is the sample size of the j-th class [16]. For the centered matrix $\mathbf{Y}''$, an equivalence between the multivariate linear regression and the linear discriminant analysis was established in [16].

B. Deep Canonical Correlation Analysis

DCCA uses multiple stacked network layers of nonlinear transformations to simultaneously learn the representations of two views of data that are maximally correlated [8]. Two deep neural networks (i.e., one for each data view) are simultaneously trained, so that the output layers between the two networks are maximally correlated. A schematic representation of the two networks is illustrated in Fig. 1. In both networks, the input layer has as many nodes as the dimensionality of each data view (i.e., d for the data view and k for the class label view). The output layer has o nodes in both networks. There are L hidden layers in the first network all having the same number of nodes c_1. Similarly, there are M hidden layers with c_2 nodes in the second network.

Given an input data sample $\mathbf{x}_i$ in the first network, the output $\mathbf{h}_1 \in \mathbb{R}^{c_1 \times 1}$ of the first hidden layer is given by $\mathbf{h}_1 = s(\mathbf{W}_1^1\, \mathbf{x}_i + \mathbf{b}_1^1)$, where $\mathbf{W}_1^1 \in \mathbb{R}^{c_1 \times n}$ is the weight matrix, $\mathbf{b}_i^1 \in \mathbb{R}^{c_1 \times 1}$ is the vector of biases, and $s : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function. The output $\mathbf{h}_1$ of the first hidden layer serves as input to the second hidden layer, which in turn has $\mathbf{h}_2$ as output, and so on. The output $\mathbf{h}_l$ of each hidden layer, which has as input the output of the previous hidden layer, $\mathbf{h}_{l-1}$, is described by:

$$\mathbf{h}_l = s(\mathbf{W}_l^1\, \mathbf{h}_{l-1} + \mathbf{b}_l^1) \quad (14)$$

where $l = 1, 2, \ldots, L$. When $l = L$, (14) computes the final output representation $f_1(\mathbf{x}_i) \in \mathbb{R}^{o \times 1}$ for the given instance $\mathbf{x}_i$. Similarly, the output $\mathbf{h}_m$ of each hidden layer in the second network is obtained by:

$$\mathbf{h}_m = s(\mathbf{W}_m^2\, \mathbf{h}_{m-1} + \mathbf{b}_m^2) \quad (15)$$

where $m = 1, 2, \ldots, M$. When $m = M$, (15) gives the final output representation $f_2(\mathbf{y}_i) \in \mathbb{R}^{o \times 1}$ for the given multivariate label $\mathbf{y}_i$.

Denoting by $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ the vectors of all parameters $\mathbf{W}_l^1$, $\mathbf{b}_l^1$ and $\mathbf{W}_m^2$, $\mathbf{b}_m^2$ of the first and second network, respectively, the goal is to jointly learn $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ so that the correlation between $f_1(\mathbf{X})$ and $f_2(\mathbf{Y})$ is maximized [8]. Let $\mathbf{H}_X \in \mathbb{R}^{o \times n}$ and $\mathbf{H}_Y \in \mathbb{R}^{o \times n}$ be the matrices having as columns the output representations produced by the two deep networks and $\bar{\mathbf{H}}_X = \mathbf{H}_X - \frac{1}{n}\mathbf{H}_X\mathbf{1}$ and $\bar{\mathbf{H}}_Y = \mathbf{H}_Y - \frac{1}{n}\mathbf{H}_Y\mathbf{1}$ be the corresponding centered matrices. The sample dispersion matrices of the output representations, $\hat{\boldsymbol{\Sigma}}_X$ and $\hat{\boldsymbol{\Sigma}}_Y$, are described as follows [8]:

$$\hat{\boldsymbol{\Sigma}}_X = \frac{1}{n-1}\bar{\mathbf{H}}_X\bar{\mathbf{H}}_X^\top + r_X\mathbf{I}, \quad (16)$$

$$\hat{\boldsymbol{\Sigma}}_Y = \frac{1}{n-1}\bar{\mathbf{H}}_Y\bar{\mathbf{H}}_Y^\top + r_Y\mathbf{I}, \quad (17)$$

where $r_X > 0$ and $r_Y > 0$ are regularization constants guaranteeing that $\hat{\boldsymbol{\Sigma}}_X$ and $\hat{\boldsymbol{\Sigma}}_Y$ are positive definite. The sample cross-covariance matrix $\hat{\boldsymbol{\Sigma}}_{XY}$ is defined as:

$$\hat{\boldsymbol{\Sigma}}_{XY} = \frac{1}{n-1}\bar{\mathbf{H}}_X\bar{\mathbf{H}}_Y^\top. \quad (18)$$

When $k = o$, the correlation between $\bar{\mathbf{H}}_X$ and $\bar{\mathbf{H}}_Y$ is given by the matrix trace norm of $\mathbf{T} = \hat{\boldsymbol{\Sigma}}_X^{-1/2}\, \hat{\boldsymbol{\Sigma}}_{XY}\, \hat{\boldsymbol{\Sigma}}_Y^{-1/2}$, i.e.,

$$\mathrm{corr}(\bar{\mathbf{H}}_X, \bar{\mathbf{H}}_Y) = \mathrm{tr}\big(\mathbf{T}^\top\, \mathbf{T}\big)^{1/2}. \quad (19)$$

The parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ of DCCA are then estimated on the training data in a way to optimize the total correlation expressed by (19). To this end, back-propagation has been exploited to estimate the gradient of the total correlation (19) with respect to the parameters involved [8]. A quadratic penalty with weight $\lambda_b > 0$ is also added in (19) for regularization.

Stochastic optimization based on mini-batches has been found to perform poorly with respect to the correlation objective, since the correlation is a function defined on the entire training set. Accordingly, a full-batch optimization is performed based on a memory efficient quasi-Newton optimization algorithm that approximates the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, known as L-BFGS [17]. L-BFGS has been successfully applied to deep learning [18].

A further optimization improvement can be achieved by means of pre-training. The latter is a common practice used in deep learning for the initialization of the optimization parameters. In particular, a denoising autoencoder is used to initialize the parameters of each network layer [19]. A distorted matrix $\tilde{\mathbf{X}}$ is constructed by adding to the data matrix $\mathbf{X}$ independent identically distributed zero-mean Gaussian noise with variance σ_a^2. The reconstructed data $\hat{\mathbf{X}}$ are then formed as $\hat{\mathbf{X}} = \mathbf{W}^\top s(\mathbf{W}\tilde{\mathbf{X}} + \mathbf{b}\,\mathbf{1}^\top)$. Next, the L-BFGS algorithm is used to find a local minimum of the total reconstruction error plus a quadratic penalty, i.e.,

$$\varphi(\mathbf{W}, \mathbf{b}) = \|\hat{\mathbf{X}} - \mathbf{X}\|_F^2 + \lambda_a\, (\|\mathbf{W}\|_F^2 + \|\mathbf{b}\|_2^2), \quad (20)$$

where σ_a^2 and λ_a are hyperparameters optimized on a devel-

opment set [8]. The values $\mathbf{W}^*$ and $\mathbf{b}^*$ that minimize (20) are used to initialize the DCCA objective and to yield the representation for pre-training the next layer.

III. Dataset and feature extraction

A corpus of Greek folk songs has been collected and tagged from publicly available resources in the web, including the lyrics and musical audio recordings [20]. The raw data have been manually checked in order to maintain some minimum quality and consistency. Here, we are dealing with a subset of the corpus, which contains songs originated from the two corpus largest classes, namely Pontus and Asia Minor.

Song lyrics are a rich information carrier. Correlations between lyrical and audio features were used for mood detection in [21], while song lyrics along with rhythm were used for music emotion classification in [22]. Various text processing tasks were applied prior to lyrics feature modeling. First, punctuation marks, special characters, numbers, and redundant white-space characters were deleted from the lyrics. Then, tokenization was performed by segmenting the lyrics into tokens. The list of Greek stop words in [23] was used in order to get rid of common words that do not bear any discriminating power. Next, a stem vocabulary was created, including 4,553 stems. In [21], [22], the tf-idf weights were used to represent quantitatively the song lyrics. The tf-idf weight is a numerical statistic, quantifying the importance of a term in a document (i.e., song lyrics) [24]. It is the product of two statistics, namely the term frequency and the inverse document frequency. The term frequency weighs more heavily the most frequent terms in a specific document. On the other hand, the inverse document frequency down-weighs the terms, which tend to appear many times in several documents in the corpus. By doing so, the terms that are truly representative of a document are given higher weights.

T-distributed stochastic neighbor embedding (t-SNE)[1] was used to visualize the high-dimensional tf-idf weights by giving each descriptor a location in a two-dimensional map [25]. t-SNE is a variation of stochastic neighbor embedding (SNE) [26]. It employs a symmetric version of the SNE cost function and a Student-t distribution rather than a Gaussian one in order to compute the similarity between two points in the two-dimensional map. Thus, the tendency to overcrowd the patterns in the center of the map is reduced. It is seen that lyrics descriptors from the two classes under study are easily discriminated in Fig. 2(a).

Monophonic wav audio recordings sampled at 22.050 KHz are available for the time being. A 30 s long excerpt was extracted from each audio recording. The OpenSMILE Toolkit [27][2] was used to extract 28 MFCCs without any delta and delta-delta coefficients from each 30 ms long frame. The frames were 30 ms long and overlapped by 50%, resulting in 2000 MFCC vectors for each recording in total. Finally, an audio descriptor of size 28×1 was used to represent each recording by averaging the 2000 MFCC vectors. Fig. 2(b) visualizes the audio descriptors for recordings originated from Asia Minor and Pontus. The two-classes are not easily discriminating using the aforementioned audio descriptors.

[1] http://lvdmaaten.github.io/tsne/

[2] http://www.audeering.com/research/opensmile

TABLE I. Confusion matrices for lyrics descriptor classification

	Predicted Class			
Ground Truth	Class 1	Class 2	Class 1	Class 2
Class 1	24	0	23	1
Class 2	1	22	0	23
Class 1	23	1	24	0
Class 2	1	22	4	19
Class 1	24	0		
Class 2	1	22		

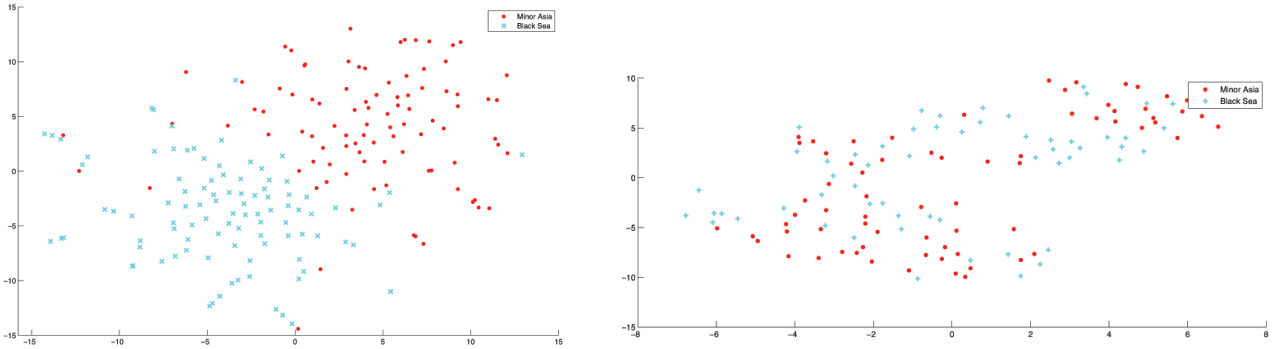
IV. Experiments

Three sets of experiments are described for two-class classification problems, employing lyrics and audio descriptors from songs that are originated from Pontus and Asia Minor, which are referred to as Class 1 and Class 2, hereafter.

Least squares regression defined by (7) and (8) was applied to the lyrics descriptors extracted from the songs of the aforementioned classes. There were 98 songs from Pontus and another 94 songs from Asia Minor. A training set was created, including 75% of the lyrics descriptors extracted from the just mentioned songs, i.e., 74 lyrics descriptors from songs of Pontus and 71 lyrics descriptors from songs of Asia Minor. The remaining 24 lyrics descriptors from songs of Pontus and 23 ones from songs of Asia Minor built the test set. The solution of (7) was a projection vector $\mathbf{w}_{LS} \in \mathbb{R}^{4553\times 1}$. If the predicted class label by (8) is the same with the actual label, a correct classification will occur. An average accuracy of 97.02% was measured using 5-fold stratified cross validation. The confusion matrices in the 5 folds are listed in Table I.

The aforementioned least squares regression was also applied to the audio descriptors extracted from the songs of the aforementioned classes. There were 57 audio recordings from Pontus and another 70 from Asia Minor. A training set was created, including 75% of the audio descriptors extracted from these audio recordings, i.e., 43 audio descriptors from songs of Pontus and 53 audio descriptors from songs of Asia Minor. The remaining 14 descriptors from songs of Pontus and 17 ones from songs of Asia Minor built the test set. The solution of (7) was a projection vector $\mathbf{w}_{LS} \in \mathbb{R}^{28\times 1}$. If the predicted class label by (8) is the same with the actual label a correct classification will occur. An accuracy of 72.90% was measured in 5-fold stratified cross validation. The confusion matrices in the 5 folds are listed in Table II. For the choices of class indicator matrices $\mathbf{Y}$, $\mathbf{Y}'$, and $\mathbf{Y}''$ defined in Section II-A, accuracies ranging from 65% to 69% were measured in the same 5-fold stratified cross-validation setting.

DCCA was applied to running 127 MFCC descriptors corresponding to the 28 MFCCs of 9 overlapped frames across each recording, because it is pointless to explicitly average the MFCCs within the DCCA. Both networks had $c_1 = c_2 = 28$ nodes in the input layer (i.e., equal to the number of feature attributes) and $o = 2$ nodes in the output layer (i.e., equal to the



(a) (b)

Fig. 2. Visualization using t-SNE of the (a) lyrics descriptors and (b) audio descriptors for Greek folk songs originated from Asia Minor and Pontus (Black Sea).

TABLE II. CONFUSION MATRICES FOR AUDIO DESCRIPTOR CLASSIFICATION

Ground Truth	Predicted Class			
	Class 1	Class 2	Class 1	Class 2
Class 1	9	5	11	3
Class 2	4	13	7	10
Class 1	9	5	9	5
Class 2	3	14	6	11
Class 1	11	3		
Class 2	1	16		

number of classes). The number of hidden layers L was also chosen to be the same for both networks. For simplicity, each hidden layer had the same number of nodes. The input data $\mathbf{X}$ in the first network were the MFCC descriptors extracted for each song. In the second network, the class membership indicator matrix was fed as input. This matrix had for each data point a vector having 1 in the entry associated to the class it belonged to and 0 in all other entries.

The *dcca* C++ code[3] provided with [8] was compiled and run on a Linux machine. The code relies on Boost libraries (headers only) and the Intel Math Kernel Library. An attempt was made to tune the hyperparameters on a validation set (e.g., 15% the size of the whole dataset), but the results obtained were comparable to the results obtained when the default hyperaparameter values provided with the code distribution were used. Thus, all the experiments were run using the distributed hyperparameter values, i.e.: a) the regularization parameter λ_a for the input, hidden and output layers pre-training for the first network was set to values 4.711×10^{-4}, 0.0052, and 2.424×10^{-4}, respectively; b) the corresponding values for the second network were $3.153 \times 10{-}4$, 5.504×10^{-4}, and 2.125×10^{-4}; c) in the first network, the variance σ_a of the Gaussian noise in the denoising autoencoder pre-training of input and hidden layers was set to values 0.1538 and 0.0264, respectively, while for the second network these values were

[3]https://homes.cs.washington.edu/~galen/files/dcca.tgz

TABLE III. AVERAGE TOTAL DCCA CORRELATION AND CLASSIFICATION ACCURACY ACROSS ALL THE FOLDS IN THE TEST SET WHEN THE SVM CLASSIFIER IS APPLIED TO THE OUTPUT DATA OBTAINED UNDER DIFFERENT DEEP NEURAL NETWORK STRUCTURE.

Number of hidden layers	Number of nodes per hidden layer	Test set correlation	Accuracy
2	256	0.43	66%
4	128	0.42	59%
4	256	0.48	**69%**
4	1024	0.44	64%
10	256	0.34	63%

0.0096 and 0.1566; d) the regularization parameters λ_b, r_X and r_Y were set to values 0.045, 41.67, and 59.06, respectively. The convergence tolerance of the L-BFGS algorithm was set to 10^{-4} and 10^{-3} for the first and second network, respectively. The activation function for all the layers was a sigmoid function based on the cubic root.

The experiments were run with 10-fold stratified cross validation with 75% of the data used for training and the remaining 25% used for testing. The classification decision on the output data of the first network was conducted by means of a Support Vector Classifier, a multi-layer perceptron, and a naive classifier that assigns the data point to the class that has the maximum value. The difference in classification performance in terms of accuracy between the three classification methods is negligible. Experiments were conducted for different number of hidden layers and different number of nodes in hidden layers. Some representative results on the average classification accuracy across all folds for the SVM classifier are listed in Table III. According to Table III, the best classification result was obtained with a deep neural network with 4 hidden layers and 256 nodes in each layer. It is also worth mentioning that experiments with deeper networks have been conducted (i.e., $50 - 80$ hidden layers), yielding unsatisfactory classification results. This can be attributed to the fact that the data size is not adequately big for really deep networks.

In order to check whether the accuracy differences are statistically significant, we apply the approximate analysis in [28]. Let us assume that the accuracies ϖ_1 and ϖ_2 are

binomially distributed random variables. If $\hat{\varpi}_1, \hat{\varpi}_2$ denote the empirical accuracies, and $\overline{\varpi} = \frac{\hat{\varpi}_1+\hat{\varpi}_2}{2}$, the hypothesis $H_0 : \varpi_1 = \varpi_2 = \overline{\varpi}$ is tested at 95% level of significance. The accuracy difference has variance $\beta = 2\frac{\overline{\varpi}(1-\overline{\varpi})}{n_t}$, where n_t is the number of test samples (i.e., 47 for lyrics descriptors and 31 for audio descriptors). For $\zeta = 1.65\sqrt{\beta}$, if $\hat{\varpi}_1 - \hat{\varpi}_2 \geq \zeta$, we reject H_0 with risk 5% of being wrong. The aforementioned analysis certifies that the accuracy difference of 3.9% between the least squares regression and the DCCA is not statistically significant, because ζ=19.03%. Moreover, for lyrics descriptor classification, the accuracy differences across the folds can be shown to be statistically insignificant. However, for audio descriptor classification, the accuracy differences in certain folds are shown to be statistically significant (e.g., between the fold corresponding to the top left confusion matrix and the bottom left confusion matrix in Table II).

V. Conclusions

Experimental evidence has been disclosed for classifiers resorting to CCA and DCCA in a two-class problem, employing lyrics and audio descriptors extracted from Greek folk songs. For CCA, we have exploited its equivalence with least squares regression. Moreover, we have verified experimentally that there are not any statistically significant accuracy differences between the CCA and the DCCA. Future research could exploit the auditory spectrotemporal modulations that were shown to yield very good results in Western music genre recognition [29]. An accuracy of 91% has been achieved in preliminary experiments with the auditory spectrotemporal modulations in the two-class problem studied in this paper.

Acknowledgment

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operation Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: THALIS-UOA-ERASITECHNIS MIS 375435.

References

[1] Z. Juhász, "Low dimensional visualization of folk music systems using the self organizing cloud," in *Proc. 2011 Int. Conf. Music Information Retrieval*, 2011, pp. 299–304.

[2] J. Six and O. Cornelis, "Tarsos: a platform to explore pitch scales in non-western and western music," in *Proc. 2011 Int. Conf. Music Information Retrieval*, 2011, pp. 169–174.

[3] G. Spyridakis and S. D. Peristeris, "Greek folk songs," in *Folk Music Collection*, A. Polymerou-Kamelake, Ed. Athens: Hellenic Folklore Research Center, Academy of Athens, 1999, vol. 3.

[4] H. Hotelling, "Relations between two sets of variables," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer, 2001.

[6] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

[7] L. Sun, S. Ji, and J. Ye, "Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 194–200, 2011.

[8] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *Proc. 30th Int. Conf. Machine Learning*, vol. JMLR W & CP 28(3), 2013, pp. 1247–1255.

[9] D. R. Hardoon, S. Szedmák, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[10] A. Haghighi, P. Liang, T. Berg-Kirkpatrick, and D. Klein, "Learning bilingual lexicons from monolingual corpora," in *Proc. 46th Annual Conf. Association Computational Linguistics-Human Language Technologies*, 2008, pp. 771–779.

[11] P. Dhillon, D. Foster, and L. Ungar, "Multi-view learning of word embeddings via CCA," in *Advances Neural Information Processing Systems*, vol. 24, 2011.

[12] K. Choukri and G. Chollet, "Adaptation of automatic speech recognizers to new speakers using canonical correlation analysis," *Speech Communication*, vol. 1, no. 2, pp. 95–107, 1986.

[13] R. Arora and K. Livescu, "Multi-view CCA-based acoustic features for phonetic recognition across speakers and domains," in *Proc. 2013 IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 2013, pp. 7135–7139.

[14] A. Katsamanis, G. Papandreou, and P. Maragos, "Face active appearance modeling and speech acoustic information to recover articulation," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 17, no. 3, pp. 411–422, 2009.

[15] F. Bach and M. Jordan, "Kernel independent component analysis," *Journal Machine Learning Research*, vol. (2002), no. 3, pp. 1–48, 2002.

[16] J. Ye, "Least squares discriminant analysis," in *Proc. Int. Conf. Machine Learning*, 2007, pp. 1087–1094.

[17] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY: Springer, 2006.

[18] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proc. 28th Int. Conf. Machine Learning*, 2011.

[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Machine Learning*, 2008.

[20] E. Giouvanakis, C. Kotropoulos, A. Theodoridis, and I. Pitas, "A game with a purpose for annotating Greek folk music in a web content management system," in *Proc. 18th Int. Conf. Digital Signal Processing*, 2013.

[21] M. McVicar, T. Freeman, and T. D. Bie, "Mining the correlation between lyrical and audio features and the emergence of mood," in *Proc. 2011 Int. Conf. Music Information Retrieval*, 2011, pp. 783–788.

[22] X. Wang, X. Chen, D. Yang, and Y. Yu, "Music emotion classification of chinese songs based on lyrics using TF*IDF and rhyme," in *Proc. 2011 Int. Conf. Music Information Retrieval*, 2011, pp. 765–770.

[23] H. Bagola, "Informations utiles à l'intègration de nouvelles langues européennes," Publications of the European Union, Tech. Rep., 2004.

[24] G. Salton, A. Wong, and C. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[25] L. van der Maaten and G. E. Hilton, "Visualizing data using t-SNE," *Journal Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[26] G. E. Hinton and S. Roweis, "Stochastic neighbor embedding," in *Advances Neural Information Processing Systems*, vol. 15, 2002, pp. 883–340.

[27] F. Eyben, F. Weninger, F. Gross, and B. Schuller, "Recent developments in openSMILE, the Munich Open-Source Multimedia Feature Extractor," in *Proc. ACM Multimedia*, 2013, pp. 835–838.

[28] 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, 1998.

[29] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Music genre classification via joint sparse low-rank representation of audio features," *ACM/IEEE Trans. Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1905–1915, 2014.