KERNELS AND BEYOND FOR DATA SIMILARITY LEARNING IN DATA MINING

by

AKSHAY MALHOTRA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2019

To my parents Subir and Manisha Malhotra

who have helped and guided me at every moment in my life.

ACKNOWLEDGEMENTS

A lot of people have contributed to the success and completion of my dissertation, from whom I have learnt a lot of lessons which will be invaluable to me in my future pursuits.

Firstly, I would like to thank my supervising Professor, Dr. Ioannis. D. Schizas, for his mentorship, constant motivation to explore, for having faith in my capabilities and the invaluable pieces of advice during the course of my research. Dr. Schizas was always there to encourage me and guide me and always had my back over any issue throughout the course of my research at UTA. I will deeply cherish the long discussions that I had with him and how he mentored me and helped me mathematically formulate my ideas and shape them into meaningful and mature algorithms and theorems.

I am also thankful to Dr. Michael T. Manry, Dr. Ramtin Madani, Dr Vangelis Metsis and Dr Vassilis Athitsos for taking out time out of their busy schedules to serve on my PhD defense committee.

I would also like to extend my appreciation to my co-authors and other researchers in our group; Kazi T. Shahid, Jia Chen and Guohua Ren for their support, encouragement and for the helpful discussions we had about our work.

I am thankful to all my friends and in particular to Pratik Ghatte, Yashas Prasad, Ankush Jagushte, Kanishka Tyagi, Sumedh Datar, Snehal Kumar, Pooja HP, Aditya Mayya, Pragati MHM, Srinivasan Ekambaram, Vidyapriya P, Vasudha Hegde, Arjun Sharma, Amith Devaramani, Allu Praveen, Rakshith Shetty and Klaara Nieminen. They all played a big role in my PhD by encouraging and motivating me and distracting me whenever I was down (and needed a break). Also, a special thanks to Dr. Prabhakar Mishra and Rakshith

Shetty for getting me involved with research during my B.E. and for motivating me to do my doctorate studies.

Through my graduate studies at UT-Arlington I have also been very lucky to have met my fiance Uma Sagar who has been amazing during this journey, through thick and thin, helping me out whenever I was down or confused and celebrating with me during all the amazing times. Another person who has played a really big role and emotionally helped me at every stage during my PhD has been Paresh Naik. Thanks for taking out the time to meet me every single week (without fail). Finally, I would like to thank my mom and dad for helping me out at every stage of my life and for supporting me in all my decisions. You both have inspired me all my life and I hope I have made you proud. Beyond the research related aspects of a PhD, emotional strength has a very big role to play in the completion of the program. I am very lucky to have had such supportive family and friends who were always there for me and helped at every step of this program. Without the help of my parents, Uma and Paresh, I don't think it would have been possible.

November 8, 2019

ABSTRACT

KERNELS AND BEYOND FOR DATA SIMILARITY LEARNING IN DATA MINING

AKSHAY MALHOTRA, Ph. D.

The University of Texas at Arlington, 2019

Supervising Professor: Ioannis D. Schizas

This work discusses the problem of unsupervised clustering of signals/data vectors based on their information content. A correlation based perspective to the clustering problem has been considered, thus relying on the high correlation between data vectors from the same class rather than on the position of the vectors in the data space. In the past, correlation based clustering has been formulated using a canonical correlation framework or as a matrix factorization problem and has been solved with different variants of gradient descent. This work focuses on improving the clustering performance by modifying the framework to utilize non-linear associations or correlations. To this end, kernelized variants for both the correlation based frameworks have been presented. We also propose an unsupervised kernel learning framework that performs at par with the state of the art supervised kernel learning methods. The proposed method uses a novel eigenvalue maximization framework to learn a convex combination of a dictionary of kernels that will be most suited for the correlation based clustering approach. A joint non-negative matrix factorization based clustering and kernel learning framework has also been proposed. Under certain assumptions, the joint formulation is guaranteed to find the ideal combination of

kernels for correlation based clustering. We also establish the convergence of the proposed formulation to a stationary point.

Going beyond kernel based non-linear maps/associations, we propose two unsupervised deep learning methods to map the data vectors from the data space to a feature space wherein the within class vectors are highly correlated while the vectors across classes are uncorrelated.

As part of this work we have utilized different optimization approaches like mixed integer programming (MIP) and majorize-minimize (MM) algorithms towards solving the resulting non-convex problems. The different methods developed as part of this research, have been applied to a variety of datasets including data from wireless sensor networks (WSN), remote sensing, human activity classification, etc., and the results have been compared to the state of the art algorithms.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

With the advancements in sensor technology and increase in computational resources, sensors in various forms (eg. mobile phones, cameras, satellites etc.) produce large amounts of data every second. Analyzing the data and extracting representative information from the data is of great importance, and the fields of machine learning, signal processing and statistics have heavily contributed towards these tasks. From the data analysis stand point, the objective of identifying meaningful groupings or relationships in the data have been areas of high research activity in the past decades and they are more commonly referred to as clustering in an unsupervised setting [7], and classification in a supervised setting [8].

Popular schemes like K-means [9] and support vector machines (SVM) [10] visualize a position based perspective towards clustering and classification respectively. For these methods, the position of the data vector in the $\mathbb{R}^{N_s}$ data space (where, N_s is the dimensionality of the vector) dictates the cluster/class they belong to. As can be inferred from the example in Fig. 1.1, for the K-means approach, the clustering is dependent on the distance between the cluster centroids and the data points. Similarly, for the SVM approach, the position of the data vector with respect to the decision boundary decides the classification for the vector. Thus, the clustering/classification of the data vector is dependent purely on its position.

For clustering applications relating to sensors measurements where the objective revolves around clustering sensors (or sensor signals) based on the commonality in their observations, a correlation perspective to clustering is more meaningful. Consider the example of hyperspectral images obtained in a remote sensing setting over a field with multi-

ple crops. A hyperspectral image is a 3D image block where each pixel represents a vector containing values representing the energy in different visible, infrared or ultraviolet frequency bands. These values are dependent on the material the pixel observes. Therefore, the set of pixels observing the same crop, for instance wheat, will have similar signatures along different spectral bands and thus will be correlated. Since these pixels (or sensors, in a generic setting) are affected by the same source, second order statistics like correlation or covariances can be used as a metric of similarity in identifying the cluster of pixels (or sensors) that have similar information content or observe the same crop (or source).

In correlation based clustering approaches, the position of the data vector in the data space has no bearing on the clustering, rather, the correlation between data vectors across its dimensions is the measure of similarity. Consider an exaggerated example as seen in Fig. 1.2. Here, 8 different 10 dimensional data vectors have been considered. In the figure, the dimension index has been represented along the y-axis and the magnitude of each dimension is represented along the x-axis. In Fig. 1.2(a), a position based method like K-means has been used for clustering the 8 data vectors. As can be expected, the vectors in red are clustered together as they will be close to each other in R^{10} space. Similarly, the vectors in blue are clustered together too. In the correlation based clustering approach seen in, Fig. 1.2(b), the 4 vectors in red, though far apart from a positional stand point, are clustered together. This is because if the 4 vectors are closely observed, their magnitudes change in a similar fashion across the dimensions and are essentially the slightly scaled and displaced versions of each other but have a high correlation/covariance. The same can be observed for the vectors in blue.

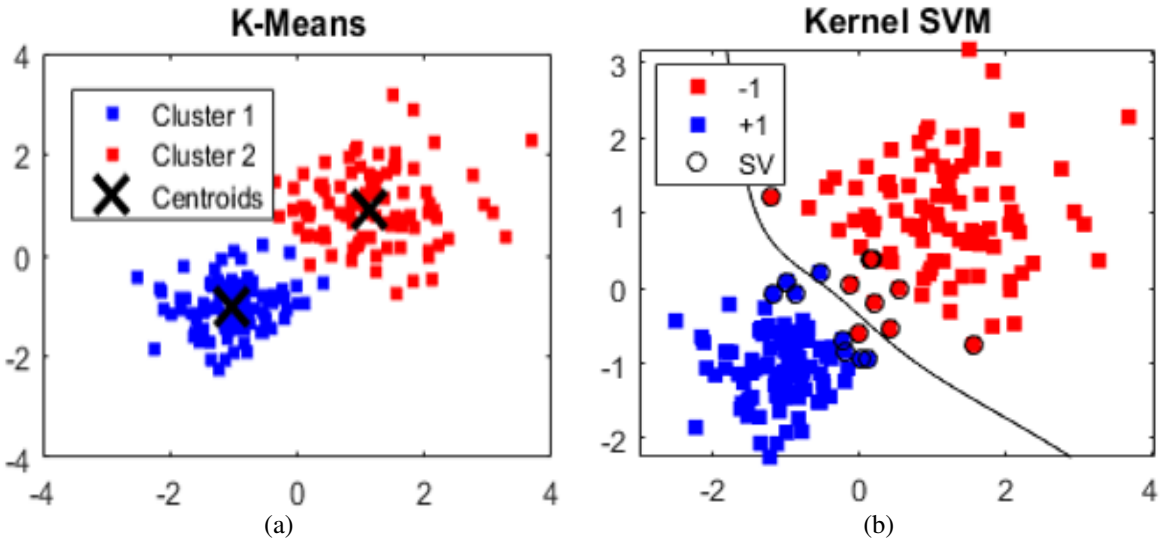


Figure 1.1. Example of position based classification and clustering approaches like K-Means and SVM. The example uses the data from [1] .

1.1 Correlation Based Clustering

Clustering data vectors based on the underlying correlation or by utilizing different similarity metrics (like cosine similarity, Gaussian similarity and many more) has been extensively studied in the literature by modeling the clustering as different optimization problems. In [11] the canonical correlation analysis (CCA) based clustering framework was proposed and was solved in an online, distributed setting. The CCA based formulation proposed in [11] was mainly suitable for linear data models, and was further extended to convolution based data models in [12]. The work also proposed an alternating CCA-PCA based clustering scheme with theoretical guarantees towards perfect clustering in linear and convolution based data settings.

In addition to the CCA based frameworks, matrix factorization approaches have also been extensively explored for clustering. The connection between clustering and non-negative matrix factorization (NMF) was explicitly defined in [13, 14]. An augmented Lagrangian approach towards solving an orthogonal NMF problem has been presented

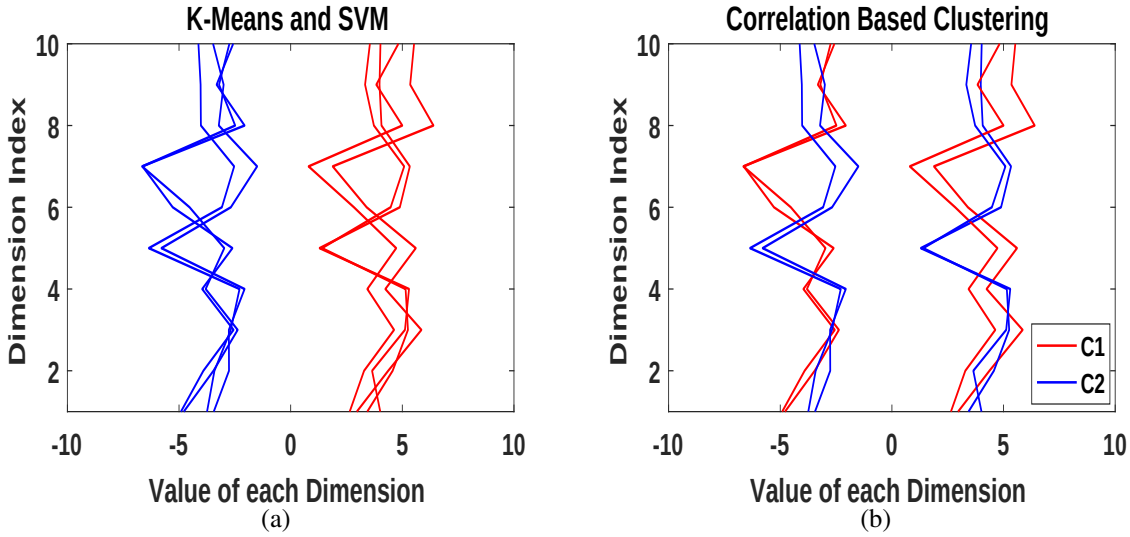


Figure 1.2. A model example showcasing the difference between position based and correlation based clustering..

in [15]. In [16] an alternating approach based on the Procrustes rotation and projection towards symmetric NMF has been proposed. From the correlation/covariance perspective, the clustering problem was modeled as a matrix factorization problem and solved under norm-1 sparsity constraints to estimate the support of the sparse covariance factors in [17]. A comprehensive review of different NMF algorithms can be found in [18, 19].

Standard NMF and CCA based methods depend on the data to be linearly separable or linearly uncorrelated across different classes, many different formulations have been proposed in the last decade to handle non-linear sensor-source relationships. A deep NMF approach which factorizes the matrix into more than two factors and thus obtains low dimensional models more suitable for clustering is presented in [20]. Kernelized or graph-based approaches tackle non-linear data models by either mapping the data into a higher dimensional space or by penalizing the cost function with graph regularizers. In Kernelized CCA based clustering [21, 22], the data covariance matrix is replaced with a RBF kernel covariance matrix. A graph based approach to utilize the non linear relationships in data

towards clustering are explored by graph based approaches like [23] where the standard NMF cost function is supplemented by a graph regularizer thus supplementing the formulation with a non-linear relations in the data. A similar regularizer has been used in [24] to establish a graph perspective of the CCA framework. In [25] another approach towards utilizing the graph information by having symmetric factors has been explored. A mixed integer formulation for a kernelized NMF problem has been explored in [26]. Though some of these techniques take into account the non linearity in the data correlation, a major challenge remains in identifying the correct kernel family and parameters to effectively extract the non linear relationships in data. Majority of the kernel or graph based approaches still depend upon supervised models or prior information for finding the appropriate kernels or graphs.

1.2 Kernel Learning

There are various different families of kernels that can be employed, and each of them have multiple tunable parameters that need to be carefully selected, to effectively express the non-linear relationship present within a particular dataset. Selection of the correct parameters is usually referred to as 'kernel learning', and is a well-documented facet of research being conducted at present. One popular method of supervised kernel learning is Kernel Target Alignment (KTA) [27], wherein the suitable kernel matrices are identified by finding the alignment or the normalized inner product between the kernel correlation matrices and the correlation of the class labels. This has further been modified to make the kernel matrices centered and applied towards the kernel learning problem [28]. A computationally efficient supervised kernel learning method for SVMs (Support Vector Machines) has been presented in [29]. A different approach was considered in [30], where the genetic algorithm [31] was used in tandem with SVM for kernel learning based classification. An

online kernel learning scheme using random features based estimation has been proposed in [32], the scheme can also efficiently adapt the kernel in dynamic environments. A detailed survey on supervised kernel learning methods has been given in [33]. All of the aforementioned approaches require training data and they are not particularly suited for unsupervised techniques.

There has been some research conducted in unsupervised methods on kernel learning as well. One such method seeks to use the distribution of the data to find the optimal kernel, choosing a linear combination of a set of predefined kernels that will minimize the distortion over the given data [34]. An unsupervised approach using principal component analysis (PCA) to reconstruct the kernel matrix has been described in [35]. Another unsupervised method attempts to find the best combination of kernels that minimizes the intra-class variance among data points that are projected in a higher-dimensional space [36]. Finally, [37] reformulates the Maximum Margin Clustering [38] approach to find a sub-optimal but computationally simpler solution using a linear combination of given kernels in an unsupervised manner.

1.3 Contributions of This Work

As part of this work, different methods have been proposed to make the CCA and NMF formulations more robust to variations in the data, $i)$ to account for the non linearities in data in an unsupervised manner and $ii)$ to attain sparse factors that accurately indicate the underlying data clustering. The contributions of the current work are as follows,

1. A kernelized variant of sparsity regularized canonical correlation analysis based clustering framework has been proposed.

2. The non convex, NMF based clustering problem is reformulated and posed as a mixed integer linear programming (MILP) problem with sparsity imposed as a hard constraint.
3. An unsupervised multiple kernel learning (MKL) scheme has been proposed that requires no parameter tuning and performs at par with supervised schemes in many datasets. The work presents a different perspective towards kernel learning using the eigenvalues of kernel covariance matrices. It is shown that under certain assumptions on the data, the proposed approach identifies the best linear combination of kernels covariance matrices.
4. The objectives of multiple kernel learning and non negative matrix factorization based clustering have been coupled into a joint robust formulation.
5. A new sparsity metric based on $\ell_1 - \ell_2$ norm has been applied towards matrix factorization.
6. The highly non-convex, joint MKL and NMF based clustering formulation is solved using the difference-of-convex algorithm (DCA) which is based on the majorization-minimization principle. A proof of convergence to a stationary point has also been established.
7. The eignevalue based MKL scheme is extended to learn a deep neural network to transform the data from the data space to a feature space to improve the efficiency of correlation based clustering.
8. Another deep learning based transformation that works on linearizing the data using an autoencoder type formulation has also been proposed.

The rest of the dissertation is organized as follows. Chapter 2 introduces the linear CCA based clustering framework and the formulations is then extended to its kernelized variant. In Chapter 3, the MILP reformulation of the NMF based clustering has been explained. Chapter 4 explores the unsupervised kernel learning scheme and details the differ-

ence of convex reformulation to maximize a specific eigenvalue of a convex combination of matrices. In Chapter 5 the joint MKL and NMF based clustering formulation has been presented and the convergence results have been derived. Chapters 6 and 7 explore the unsupervised deep learning based methods for mapping data into a feature space appropriate for correlation based clustering. The last section provides the concluding remarks and the scope for future work.

CHAPTER 2

CANONICAL CORRELATIONS FOR CLUSTERING

The canonical correlation analysis (CCA) based clustering framework is an unsupervised algorithm where the objective is to cluster together the data vectors that exhibit high correlations. In this chapter the correlation based clustering problem is formally introduced, we then explore the data observation model for which a CCA based scheme is most suited for, and present the algorithm for CCA based clustering. Finally, we explore the kernelized variants of CCA for applications to non-linearly related data.

2.1 Data Observation Model

Consider a set of P signals/data vectors $x_p(n)$ where $p \in 1, ..., P$ is the signal/vector index and n is the sample or the dimension index such that $n \in 1, ..., N_s$. These signals or vectors can be considered to be the output of a sensor or a set of sensors that are observing a field with Q stationary sources $s_q(n)$ such that $q \in \{1, ..., Q\}$ and $Q < P$.

Such a sensor observation setting can potentially represent several different scenarios. In a remote sensing setting, the signal $x_p(n)$ represents a pixel of an hyperspectral image captured over agricultural fields with different crops [21, 22]. The image can be observing a field with Q varieties of crops with each pixel representing a part of the field with a specific crop. In a human activity recognition application using data from smartphone sensors, the signals $x_p(n)$ can be the time-series data received from accelerometer sensors in the smartphones, and the Q groups may refer to the different activities done by the user (e.g. walking, sleeping, climbing up stairs, etc.), see e.g., [4, 39]. Another potential exam-

ple is with respect to a wireless sensor network (WSN) wherein P sensors can be assumed to be monitoring a physical quantity in the environment with Q sources [12, 17, 40].

Each of the P sensors acquire local scalar measurements and they are assumed to be affected by a single underlying source. Also, its is assumed that the underlying source signals are uncorrelated. The objective here is to find signals which are affected by the same source. The sensor observation model can thus be described as:

$$x_p(n) = \sum_{j=1}^{Q} c_j^p f_j^p(s_j(n)) + w_j^p(n) \quad \forall p \in 1, ..., P \tag{2.1}$$

where, f_j^p represents the transfer function affecting the measurement of the source signal $s_j(n)$ and the p-th sensor and is considered to be unknown, while $w_j^p(n)$ represents the additive observation noise at sensor p. The variable c_j^p represents the class membership of the p-th sensor. Since we assume that each signal/ data vector corresponds only to a single class, the set $\{c_1^p, c_2^p, ..., c_Q^p\}$ $\forall p$ has only one non zero element. Therefore the objective is essentially to identify this membership variable and cluster the P signals into Q classes

Since we assume that the source signals $s_j(n) \forall j \in \{1, ..., Q\}$ are uncorrelated to each other, if the observation model $f_j^p(\cdot)$ is linear, the cross-correlation between signals can be utilized to identify signals observing the same source.

2.2 CCA-based Clustering

The CCA based clustering framework aims at utilizing the correlations between data vectors to identify data clusters without the need for any training data or any priors about the data distribution. The only information required is the number of classes, Q, in which the data should be segregated.

Consider the $P \times N_s$ data matrix, $\mathbf{X}$, containing the data from all the P data vectors stacked row wise. We split the rows in two non-overlapping groups of data vectors $\mathbf{x} \in \mathbb{R}^{P^x \times N_s}$ and $\mathbf{y} \in \mathbb{R}^{P^y \times N_s}$ where $P^x + P^y = P$. This formulation makes the following underlying assumptions: 1) the number of classes Q, is lesser than the number data vectors, N_s; and 2) each group of data vectors, $\mathbf{x}$ and $\mathbf{y}$, have at least one representative vector from each class. These assumptions are easily fulfilled for any real-world application, as the number of data vectors considered for clustering are always much larger than Q. Also, for forming the non-overlapping sets $\mathbf{x}$ and $\mathbf{y}$, vectors/signals are alternatingly allocated to the 2 groups. Thus the cardinality of the two sets $\mathbf{x}$ and $\mathbf{y}$, (i.e. P^x and P^y) is approximately the same ($\approx P/2$).

To cluster the P data vectors/signals according to the underlying source signal (or the class they represent), we make use of the statistical correlations that vectors in $\mathbf{x}$ and $\mathbf{y}$ groups exhibit when representing the same activity. Using CCA we can identify the entries of the two vectors, $\mathbf{x}$, and $\mathbf{y}$, that are maximally correlated and by imposing a ℓ_1 norm regularization, as shown in [11], we can utilize the CCA framework as a clustering algorithm. The modified CCA framework with the ℓ_1 norm regularization is given as:

$$\begin{aligned}(\hat{\mathbf{E}}, \hat{\mathbf{D}}) = \arg\min_{E,D} {N_s}^{-1} \textstyle\sum_{\tau=1}^{N_s} ||\mathbf{y}_\tau - \mathbf{E}\mathbf{D}\mathbf{x}_\tau||_2^2 \\ + \textstyle\sum_{\rho=1}^{Q} \lambda_\rho^{\mathbf{E}} ||\mathbf{E}_{:\rho}||_1 + \textstyle\sum_{\rho=1}^{Q} \lambda_\rho^{\mathbf{D}} ||\mathbf{D}_{\rho:}||_1,\end{aligned} \tag{2.2}$$

where the subscript τ indicates the time index/dimension of the vectors. $\mathbf{x}_\tau$ and $\mathbf{y}_\tau$ are $\mathbb{R}^{P^x \times 1}$ and $\mathbb{R}^{P^y \times 1}$ dimensional vectors containing the τ-th column of the matrices $\mathbf{x}$ and $\mathbf{y}$ respectively. Matrices $\hat{\mathbf{D}} \in \mathbb{R}^{Q \times P^x}$ and $\hat{\mathbf{E}} \in \mathbb{R}^{P^y \times Q}$ are the sparse matrices indicating the clustering. The sparsity is introduced as a result of the ℓ_1 norm regularization part in the modified CCA framework. The support (non-zero entries) of each row of $\mathbf{D}$ in Eq. (2.2) will indicate which entries in $\mathbf{x}_\tau$ contain information about the same activity. Note that the rows of $\mathbf{D}$ are expected to be sparse since not all entries of $\mathbf{x}_\tau$ correspond to the same

activity. Thus, it is pertinent to impose sparsity across each row of $\mathbf{D}$ that represents a different activity. This enables activity clustering among the entries of $\mathbf{x}_\tau$. Similarly, the columns of $\mathbf{E}$ can be forced to be sparse and their support will point to these entries of $\mathbf{y}_\tau$ that represent the same physical activity. Ideally, the matrix $\hat{\mathbf{D}}$ should have a single non-zero entry in each of the N^x columns and similarly for the matrix $\hat{\mathbf{E}}$ there should be only one non-zero entry in each of the N^y rows. However, in practice the row-entry with the strongest amplitude is treated as the non-zero entry pointing to the activity, whereas the entries of negligible amplitude are treated as zeros. The position of the strongest in amplitude non-zero entry indicates which of the Q activities does the signal/data vector relate to. It should be noted that $\mathbf{x}_\tau$ and $\mathbf{y}_\tau$ are not used to represent the signal /data vectors but are a means to divide the entire set of vectors into two sets of vectors, which is required for applying CCA. The operator $\|\cdot\|_1$ refers to ℓ_1 norm, while the parameters $\lambda_\rho^{\mathbf{D}}$ and $\lambda_\rho^{\mathbf{E}}$ correspond to sparsity controlling coefficients in $\mathbf{D}$ and $\mathbf{E}$, respectively. Interestingly, sparsity across rows for $\mathbf{D}$ (columns for $\mathbf{E}$) can be viewed as sparsity across columns for $\mathbf{D}$ (rows for $\mathbf{E}$) after rearranging terms.

2.3 Regularized Kernel CCA

To improve the performance and to take into account any non-linearities in the vectors, we will utilize a nonlinear kernel mapping in the formulation in (2.2). Such a formulation offers more robustness to minor changes in the data while keeping the computation cost under control by employing the kernel trick [10].

Thus, the non-linear mapping $\boldsymbol{\phi}(\cdot)$ (where, $\boldsymbol{\phi} : \mathbb{R}^{P^x} \rightarrow \mathbb{R}^{P^x \times F}$ and F represents the dimensionality of a higher dimensional feature space) is applied independently across each dimension of the input vectors $\boldsymbol{x}_\tau \rightarrow \boldsymbol{\phi}(\boldsymbol{x}_\tau)$ and $\boldsymbol{y}_\tau \rightarrow \boldsymbol{\phi}(\boldsymbol{y}_\tau)$, respectively. Thus,

$\boldsymbol{\phi}(\boldsymbol{x}_\tau) = [\hat{\phi}(\boldsymbol{x}_{\boldsymbol{\tau}}^{\mathbf{1}}), ..., \hat{\phi}(\boldsymbol{x}_{\boldsymbol{\tau}}^{\boldsymbol{P^x}})]^T$, where $\hat{\phi} : \mathbb{R} \rightarrow \mathbb{R}^F$. Therefore, the modified form of equation (2.2) is given as

$$(\hat{\mathbf{E}}, \hat{\mathbf{D}}) = \arg\min_{E,D} N_s^{-1} \sum_{\tau=1}^{N_s} ||\boldsymbol{\phi}(\mathbf{y}_\tau) - \mathbf{E}\mathbf{D}\boldsymbol{\phi}(\mathbf{x}_\tau)||_2^2 + \sum_{\rho=1}^{Q} \lambda_\rho^{\mathbf{E}} ||\mathbf{E}_{:\rho}||_1 + \sum_{\rho=1}^{Q} \lambda_\rho^{\mathbf{D}} ||\mathbf{D}_{\rho:}||_1. \tag{2.3}$$

The cost function in (2.3) consists of two parts, the first one representing the CCA framework (henceforth mentioned as $J^s(\mathbf{E}, \mathbf{D})$) and the second being the ℓ_1-norm regularization part, denote as $J^{reg}(\mathbf{E}, \mathbf{D})$). To minimize the cost in (2.3) we utilize the gradient descent method to update the clustering matrices $\mathbf{E}$ and $\mathbf{D}$. Since it is an iterative approach the k-th iteration for the update is found by utilizing the following recursive update rule:

$$\hat{\mathbf{E}}_k = \hat{\mathbf{E}}_{k-1} - c\nabla^{\mathbf{E}}_{\hat{\mathbf{E}}_{k-1}, \hat{\mathbf{D}}_{k-1}} J(\mathbf{E}, \mathbf{D}), \tag{2.4a}$$

$$\hat{\mathbf{D}}_k = \hat{\mathbf{D}}_{k-1} - c\nabla^{\mathbf{D}}_{\hat{\mathbf{E}}_{k-1}, \hat{\mathbf{D}}_{k-1}} J(\mathbf{E}, \mathbf{D}) \tag{2.4b}$$

where $c > 0$ is the step-size. Note that $\nabla^{\mathbf{E}}_{\hat{\mathbf{E}}_{k-1}, \hat{\mathbf{D}}_{k-1}} J(\mathbf{E}, \mathbf{D})$ and $\nabla^{\mathbf{D}}_{\hat{\mathbf{E}}_{k-1}, \hat{\mathbf{D}}_{k-1}} J(\mathbf{E}, \mathbf{D})$ refer to the partial derivatives of $J(\mathbf{E}, \mathbf{D})$ with respect to $\mathbf{E}$ and $\mathbf{D}$, respectively. It should be noted that since the update equations (2.4a) and (2.4b) are dependent on $\mathbf{E}$ and $\mathbf{D}$, at the k-th iteration we utilize the matrix values obtained for these matrices at the $(k-1)$-th iteration. Here we have utilized a Jacobi type update where the new value is dependent only on the previous state. A Gauss-Seidel type update, where if $\hat{\mathbf{E}}_k$ is evaluated first, then for evaluating $\hat{\mathbf{D}}_k$ the current value of $\hat{\mathbf{E}}_k$ can be used instead of using $\hat{\mathbf{E}}_{k-1}$. Nonetheless, the performance was essentially the same for both types of updates.

For ease of notation we replace $\hat{\mathbf{E}}_{k-1}$ and $\hat{\mathbf{D}}_{k-1}$ with $\mathbf{E}$ and $\mathbf{D}$ to get the following:

$$\nabla^{\mathbf{E}} J(\mathbf{E}, \mathbf{D}) = \frac{\delta J^s(\mathbf{E}, \mathbf{D})}{\delta \mathbf{E}} + \frac{\delta J^{reg}(\mathbf{E}, \mathbf{D})}{\delta \mathbf{E}}, \tag{2.5a}$$

$$\nabla^{\mathbf{D}} J(\mathbf{E}, \mathbf{D}) = \frac{\delta J^s(\mathbf{E}, \mathbf{D})}{\delta \mathbf{D}} + \frac{\delta J^{reg}(\mathbf{E}, \mathbf{D})}{\delta \mathbf{D}}. \tag{2.5b}$$

The second term in eqs. (2.5a) and (2.5b) represent the sub-gradient of the ℓ_1 norm regularization part of equation (2.3). The sub-gradients are given as:

$$\frac{\delta J^{reg}(\mathbf{E},\mathbf{D})}{\delta \mathbf{E}} = \text{sgn}(\mathbf{E})\text{diag}(\boldsymbol{\lambda}^{\mathbf{E}}), \tag{2.6a}$$

$$\frac{\delta J^{reg}(\mathbf{E},\mathbf{D})}{\delta \mathbf{D}} = \text{diag}(\boldsymbol{\lambda}^{\mathbf{D}})\text{sgn}(\mathbf{D}). \tag{2.6b}$$

where the matrices $\text{diag}(\boldsymbol{\lambda}^{\mathbf{D}})$ and $\text{diag}(\boldsymbol{\lambda}^{\mathbf{E}})$ are diagonal matrices whose ρ-th elements are $\lambda_{\rho}^{\mathbf{D}}$ and $\lambda_{\rho}^{\mathbf{E}}$, which are the sparsity controlling coefficients. The operator $\text{sgn}(\cdot)$ is the element wise sign operator.

The first term in (2.5a) and (2.5b), $J^s(\mathbf{E},\mathbf{D})$ can be written as:

$$J^s(\mathbf{E},\mathbf{D}) = tr(\hat{\mathbf{C}}_{\mathbf{y}} - 2{\cdot}\mathbf{E}{\cdot}\mathbf{D}{\cdot}\hat{\mathbf{C}}_{\mathbf{xy}} + \mathbf{E}{\cdot}\mathbf{D}{\cdot}\hat{\mathbf{C}}_{\mathbf{x}}{\cdot}\mathbf{D}^T{\cdot}\mathbf{E}^T) \tag{2.7}$$

and thus correspondingly we have the non linearly mapped expression:

$$J^s(\mathbf{E},\mathbf{D}) = tr(\hat{\mathbf{K}}_{\mathbf{y}} - 2{\cdot}\mathbf{E}{\cdot}\mathbf{D}{\cdot}\hat{\mathbf{K}}_{\mathbf{xy}} + \mathbf{E}{\cdot}\mathbf{D}{\cdot}\hat{\mathbf{K}}_{\mathbf{x}}{\cdot}\mathbf{D}^T{\cdot}\mathbf{E}^T) \tag{2.8}$$

where $\hat{\mathbf{K}}_x, \hat{\mathbf{K}}_y, \hat{\mathbf{K}}_{xy}$ denote the cross covariance matrix after the non-linear mapping $\boldsymbol{x}_\tau \rightarrow \phi(\boldsymbol{x}_\tau)$ and $\boldsymbol{y}_\tau \rightarrow \phi(\boldsymbol{y}_\tau)$.

$$\hat{\mathbf{K}}_{\mathbf{y}} = {N_s}^{-1} \sum_{\tau=1}^{N_s} \boldsymbol{\phi}(\mathbf{y}_\tau)\boldsymbol{\phi}^T(\mathbf{y}_\tau), \tag{2.9a}$$

$$\hat{\mathbf{K}}_{\mathbf{x}} = {N_s}^{-1} \sum_{\tau=1}^{N_s} \boldsymbol{\phi}(\mathbf{x}_\tau)\boldsymbol{\phi}^T(\mathbf{x}_\tau), \tag{2.9b}$$

$$\hat{\mathbf{K}}_{\mathbf{xy}} = {N_s}^{-1} \sum_{\tau=1}^{N_s} \boldsymbol{\phi}(\mathbf{x}_\tau)\boldsymbol{\phi}^T(\mathbf{y}_\tau). \tag{2.9c}$$

2.4 Kernel CCA Performance

In this section we test the performance of the proposed kernel regularized canonical correlation analysis (CCA) based approach on two different datasets; 1) In a remote sensing

setting where the objective is to classify the every pixel of a hyperspectral image based on the material it observes and; 2) An activity classification problem where using the data from the accelerometer of a phone placed on a human subject, the activity (walking, running, climbing, etc.) being performed by the subject has to be identified.

2.4.1 Hyperspectral Images

The numerical tests are performed on a hyperspectral image gathered by an AVIRIS sensor [41] over the Indian Pines test site in North-western Indiana and consists of 145×145 pixels and 224 spectral reflectance bands ($N_s = 200$ after removing the bands corresponding to regions of water absorption) in the wavelength range of $0.4 - 2.5 \times 10^{-6}$ meters. In our test we cluster pixels randomly selected to contain information about $Q = 4$ materials of interest, where $p^x = p^y = 60$. An equal number of pixels equal to 30 is selected to represent each of the four materials of interest.

In the hyperspectral setting, we compare the clustering performance of our novel approach with i) the supervised kernel SVM approach in [42] (K-SVM); and ii) K-means clustering approach. The figure of merit used here will be the probability of correctly clustering the hyperspectral pixels according to the material they observe.

When applying kernel CCA, the step-size is set as $c = 10^{-4}$, whereas the λ parameters are set equal to 0.1 each. The gradient descent recursions for updating $\mathbf{D}$ and $\mathbf{E}$ are run for a number of iterations until the updating error of the entries in these matrices drops below 10^{-5}. The variance for the Gaussian RBF kernels was chosen to be $\sigma^2 = 1000$. Clustering performance comparisons will be performed for different numbers of used spectral bands per pixel varying from $N_s = 140$ to $N_s = 200$. In addition testing will be performed for different percentages of dead pixels across $N_s = 200$ spectral bands. Dead pixels may originate due to sensor malfunctioning at certain pixel intensities, thus some pixels may have a magnitude of 0 for some frequency bands. This behavior is modeled by randomly

placing dead pixels (according to a uniform distribution) in $\mathbf{x}_\tau$ and $\mathbf{y}_\tau$ for $\tau \in \{1, ..., N_s\}$. The probability of correct clustering in all tests is obtained by averaging the correct clustering rates obtained on 100 independent trials in kernel CCA, K-SVM and K-Means each, with random initialization in each trial. For kernel CCA, this implies random initialization of the $\mathbf{D}$ and $\mathbf{E}$ sets, for K-SVM, this implies random selection of training pixels (equal to 12) from each source, and for K-means, this implies random selection of cluster centroids.

Fig. 1 (top) depicts the probability of correct clustering versus number of utilized spectral bands without dead pixels (0% of dead pixels). As it can be seen, at $N_s = 200$ bands, the proposed kernel CCA scheme achieves a clustering accuracy close to 90%, near the supervised K-SVM which achieves just over 90% accuracy and significantly better than the unsupervised K-Means approach, which trails at near 75% accuracy. Kernel CCA achieves clustering performance close to the one from K-SVM, without the need of training pixels, when a sufficient number of bands in used.

Fig. 1 (bottom) displays the clustering performance of the aforementioned three methods versus a varying percentage of dead pixels in $\mathbf{x}_\tau$ and $\mathbf{y}_\tau$ for $\tau \in \{1, ..., N_s\}$. The percentage varies from 0.1 to 10, while the probability of correct clustering was measured using $F_s = 200$ spectral bands. Further, these results were averaged over 20 independent different trials with a different set of pixels picked on every different trial. It can be seen clearly, that for 0.1% up to 0.3% of dead pixels the supervised K-SVM will have a marginal advantage over kernel CCA which does not utilize training data. However, further introduction of dead pixels clearly can deteriorate the performance on both K-SVM and K-Means, whereas kernel CCA exhibits robustness and outperforms the existing approaches. The robustness and consistency of kernel CCA in the presence of faulty data acquisition is evident. Note further that after 1.1% dead pixels and above, the faulty entries will interfere with the training part of K-SVM to the point that it will even perform worse than the unsupervised K-Means.

Additionally, further tests revealed that dead pixels reaching even as high as 30% give almost similar clustering accuracy, which deteriorates beyond that number. This is far better than K-SVM and K-Means, which clearly suffer considerably even with dead pixels as low as 1%.

2.4.2 Human activity

For the human activity classification application, we have presented the results on 3 different datasets: a) SBHAR [43], UniMiB [4] and Texas State datasets [39]. Here, the objective is to utilize the accelerometer signals from a smartphone (which is mounted on the user) to identify the activity being performed by the user at different time periods. The signals are first pre-processed to split the time series accelerometer signals into frames/epochs during which only a single activity is being performed.

Consider the signal in Fig. 2.2(a) corresponding to the accelerometer output of a smartphone while the user is performing different activities. Fig. 2.3(a) and 2.3(b) show the zoomed in versions of the signal where the user is walking and climbing-up the stairs, respectively. As it can be seen, the data has repetitive patterns or epochs during each of these activities which are distinctively different from the epochs corresponding to the other activity. Lets say there are P frames of relevance. The signal can thus be considered as a set of P vectors each having N_s dimensions/samples. Thus, we can utilize the statistical correlation between these vectors corresponding to the same activity to identify the vectors pertaining to the same activity.

More details on pre-processing the signal to obtain the individual frames/epochs can be found in appendix A

A Gaussian kernel with a variance of $\sigma^2 = 10^{-1.5}$, 10^{-2} and 10^{1} for the SBHAR [43], UniMiB [4] and Texas State datasets, respectively, has been utilized. The $\lambda_i^{\mathbf{D}}$ and $\lambda_i^{\mathbf{E}}$ value were kept fixed at 0.1 and a step size of $c = 5 \times 10^{-4}$ was used. The values should be

appropriately selected such that each column of $\mathbf{D}$ and each row of $\mathbf{E}$ have at least one non zero element, or the scheme suggested in [11] can also be used. The gradient descent iterations were executed until the error drops below a factor of 10^{-6}. For each trial of CCA, the $\mathbf{D}$ and $\mathbf{E}$ clustering matrices are randomly initialized.

For the simulations, in the case of CCA, the algorithm is applied towards each user signal separately and the results presented are averaged across all the users for each of the data-sets being considered.

2.4.3 SBHAR dataset

The first data-set is the Smartphone-based Human Activity Recognition (SBHAR) dataset [43]. The data-set contains a set of 3 major activities (walking, walking-up the stairs and walking down the stairs). In addition to this, there is data for other stationary events like standing, sitting and laying, which we don't consider as events of interest for this study. In our simulations we utilize the XYZ-axis of the accelerometer data to recognize the three major activities in this data. The signals are recorded at a sampling frequency of 50Hz.

For a few of the data samples from this dataset the average value of the signal shifts significantly with time and this interferes with the thresholds used by the pre-processing stage, thus a few of the epochs were not picked up by the pre-processing stage and were not used for the classification stage. The epoch structure in these signals is perfectly fine and thus CCA based techniques suggested in this paper are still valid to these signals by adapting the pre-processing part or by using a different scheme for epoch detection. The files are around 6.4 minutes in length on an average. Thus a total of almost 5400 epochs across all the signals have been considered. The confusion matrix for the CCA case is given in Table 2.1.

Another parameter that impacts the performance of the algorithm is the epoch length or the data vector length. As seen in Fig. 2.4, with the epoch length being small, the error

Table 2.1. Confusion matrix for CCA using the SBHAR dataset. W=walking, U=Upstairs, D=Downstairs.

		Predicted					
		W	**U**	**D**	$\mathbf{P_C}$	$\mathbf{R_C}$	$\mathbf{F1_C}$
Actual	**W**	1649	355	352	90.1	70.0	78.8
	U	59	975	77	60.2	87.8	71.4
	D	123	289	1506	77.8	78.5	78.2
		Accuracy: 76.7			76.0	78.7	76.1

is high since the correlations cannot be accurately observed. As the length of the epoch is increased the accuracy increases. Once the epoch length is increased beyond a threshold, in this case 210, the error starts to increase, as mentioned before this is due to the epoch containing the samples from the neighboring epochs.

2.4.4 UniMiB dataset

The second data set is the University of Milano Bicocca Smartphone-based Human Activity Recognition (UniMiB) dataset [4]. This set has data from 30 users performing a much wider range of activities, a total of 17 activities including 9 daily activities like walking, running, climbing the stairs. The data is pre-split to represent the individual epochs from each of the activities and thus no pre-processing has been done on this data. To showcase the performance over a wider set of activities, we use data pertaining to all the 9 daily activities for this data set. The signals are recorded at a sampling frequency of 50Hz and the dataset for these 9 activities amounts to a total of 7565 epochs. The data from XYZ-axis of the accelerometer is utilized towards classification.

For the CCA based unsupervised clustering approach, we have only presented the clustering accuracies for 3 classes. We are currently improving the approach to reduce computational complexity and enable data clustering in the presence of larger number of clusters. The CCA based approach gives an overall accuracy and a MAA of 71.1% each.

Table 2.2. Confusion matrix for CCA using the UniMiB data-set. W=walking, R=Running, J=Jumping.

		Predicted					
		W	**R**	**J**	$\mathbf{P_C}$	$\mathbf{R_C}$	$\mathbf{F1_C}$
Actual	**W**	515	46	99	64.0	78.0	70.4
	R	144	446	70	79.6	67.6	73.1
	J	145	68	447	72.6	67.7	70.1
		Accuracy: 71.1			72.1	71.1	71.2

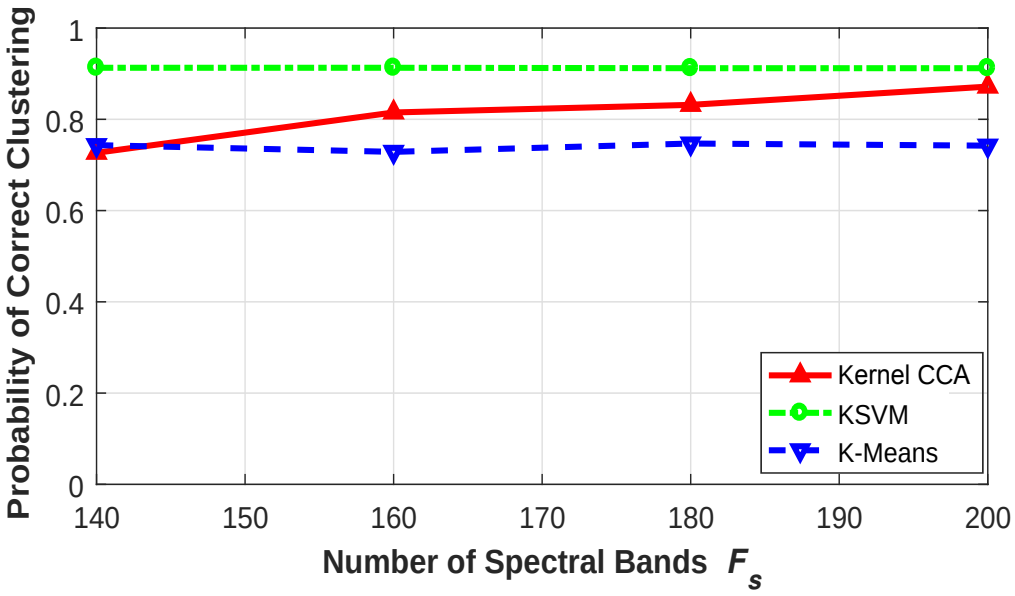
The accuracies for the CCA based scheme are given in the form of a confusion matrix in Tables 2.2.

2.4.5 Texas State dataset

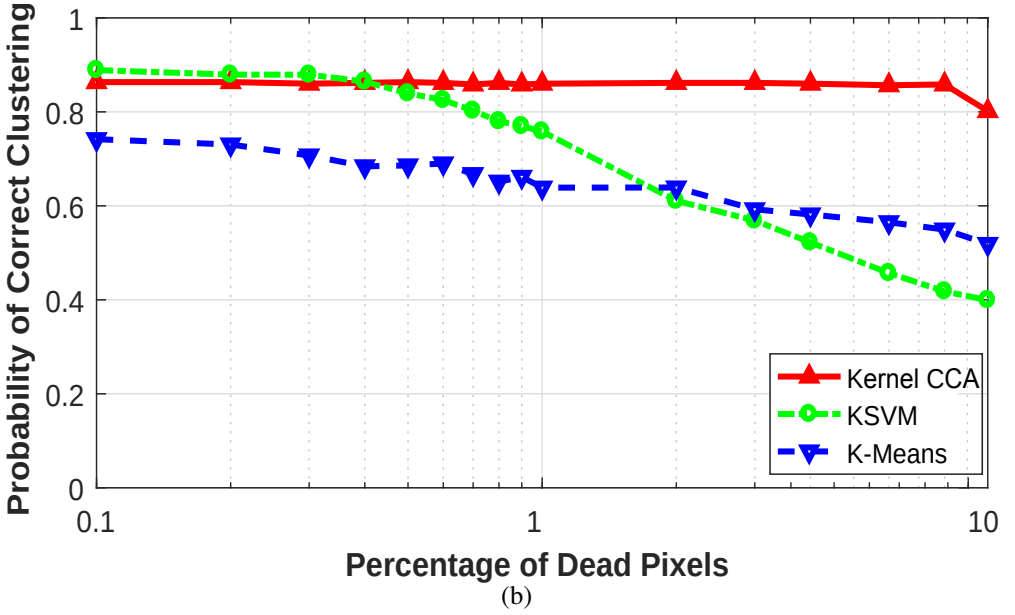
The third dataset that was used was collected by Dr. Vangelis Metsis and his team at Texas State University. It is a relatively small data-set, including 3 users performing 5 activities (i.e. walking on a level surface, going downstairs, going upstairs, standing and sitting). The user repeats the activity sequence five times. The total duration of each run was about 2 minutes and 40 seconds. 3-dimensional acceleration data at a sampling rate of 250 Hz were collected using a BioRadio device [44] mounted on the waist of each subject (see figure in [39]). This data-set was mainly created to facilitate the development stages of our methods. Nonetheless, our experimental results are illustrated here as an extra evaluation resource. The accuracies for the CCA based scheme are given in the form of a confusion matrix in Tables 2.3.

Table 2.3. Confusion matrix for CCA using the Texas State data-set.

		Predicted					
		W	**D**	**U**	$\mathbf{P_C}$	$\mathbf{R_C}$	$\mathbf{F1_C}$
Actual	**W**	460	134	190	93.9	58.7	72.2
	D	18	132	44	44.6	68	53.9
	U	12	30	126	35.0	75.0	47.7
		Accuracy: 62.6			57.8	67.2	57.9

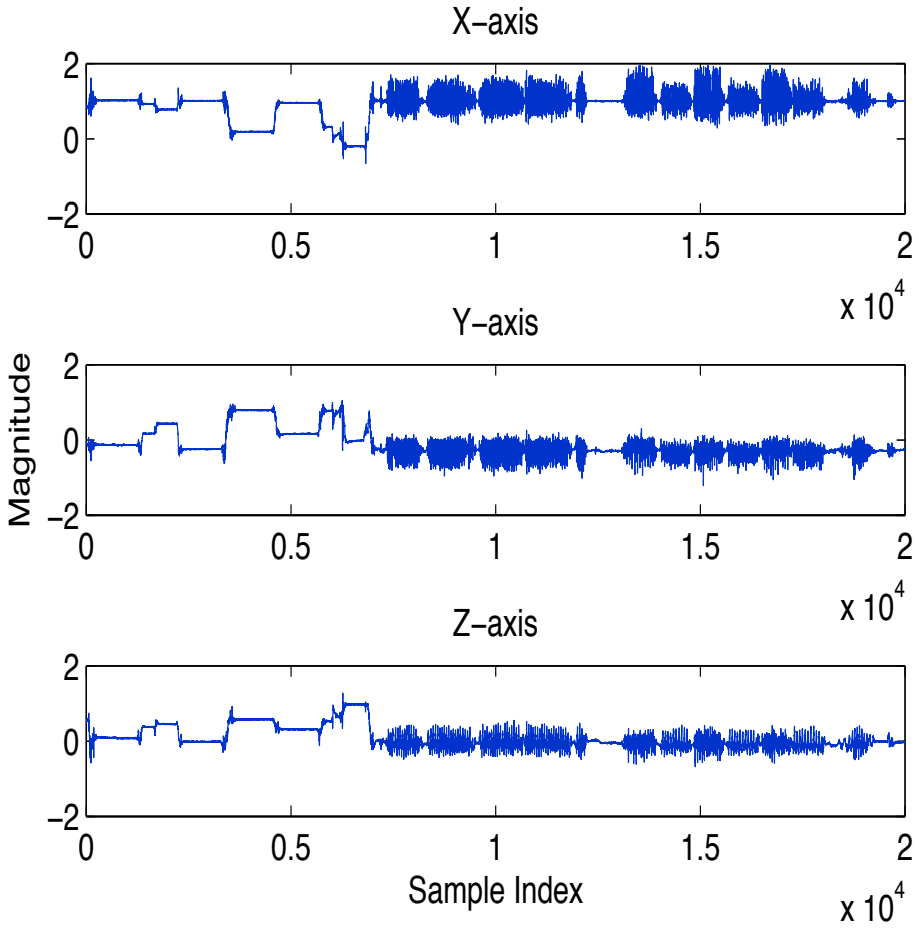


(a)

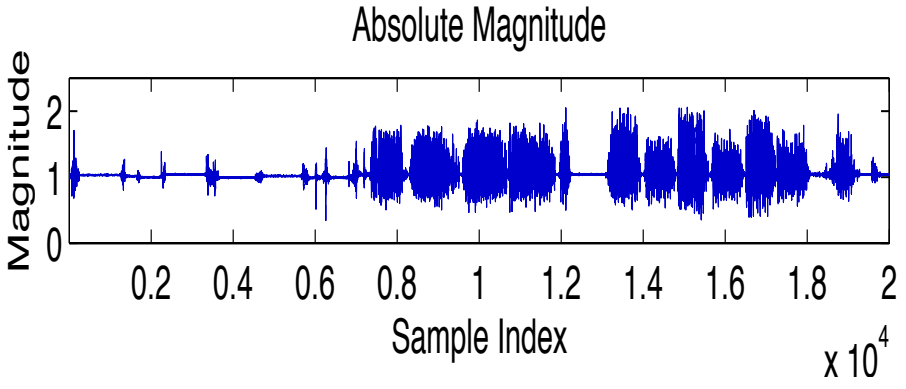


(b)

Figure 2.1. Probability of correct clustering versus number of spectral bands used (top); and percentage of missing (dead) pixel intensities (bottom)..

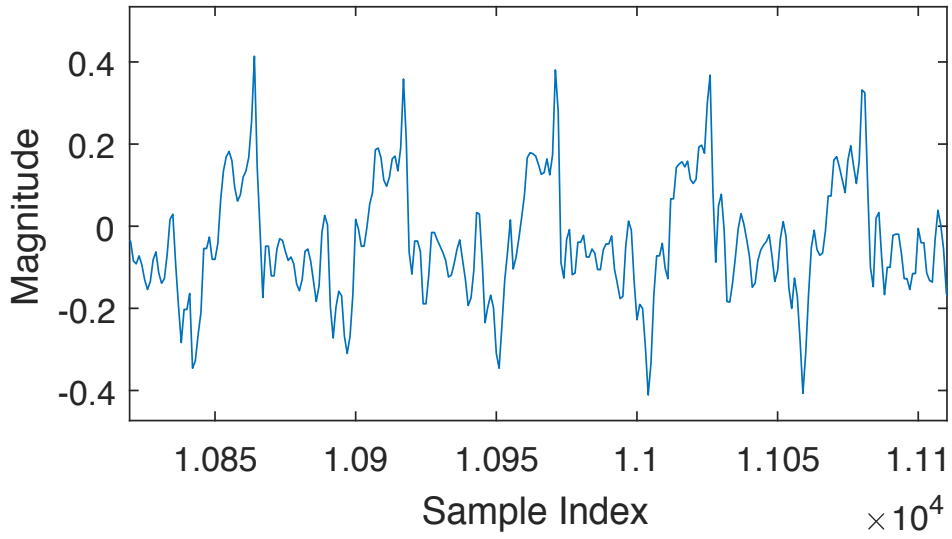


(a)

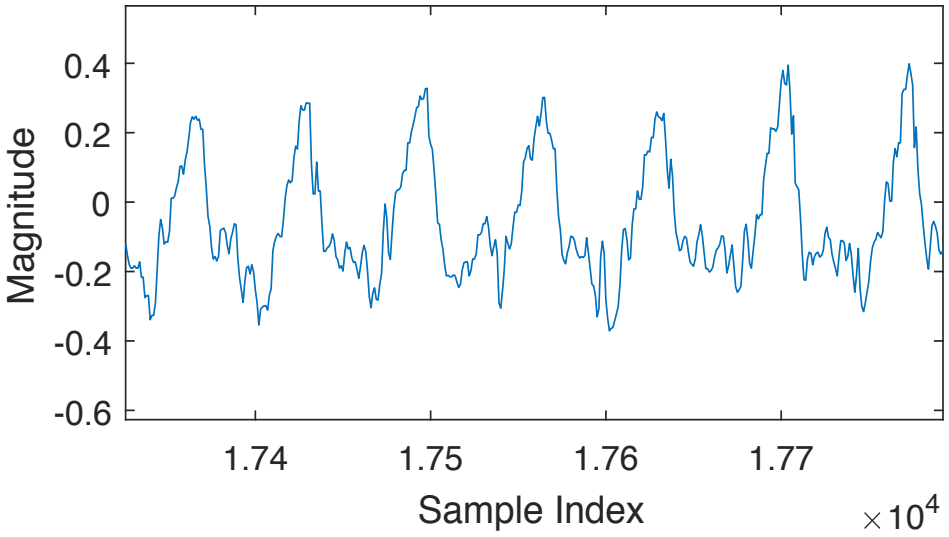


(b)

Figure 2.2. Example of a sample acceleration signal over multiple activities. Fig. (a) shows the signal from the three accelerometer channels (X, Y, Z) separately, Fig. (b) shows the absolute value (magnitude) of the signal vector across the XYZ components.



(a)



(b)

Figure 2.3. Example of a sample acceleration signal over multiple activities. Fig. (a) zooms in on the portion of the Z-axis signal in 2.2 with walking as the activity, and Fig (b) zooms in on the portion of the Z-axis signal with climbing up-stairs as the activity. The repetitive epochs for the two activities can be clearly seen in the signal. The epochs from the two activities have different structure, this structural differences are utilized by CCA for classification.

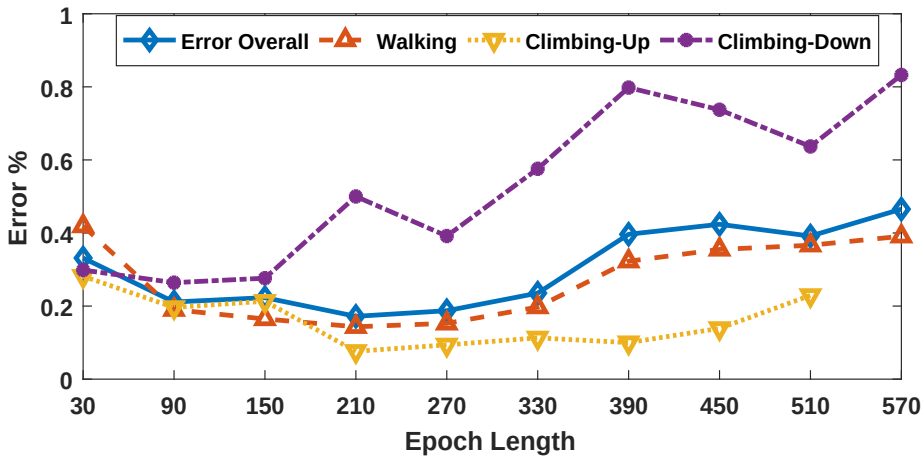


Figure 2.4. Percentage error with CCA for different epoch lengths in the SBHAR data-set..

CHAPTER 3

NMF BASED CLUSTERING

Upto this point we have looked at posing the correlation based clustering problem using the CCA framework. Another possibility is to pose the correlation based clustering as a matrix factorization problem. From Chapter 2, we know that transfer function $f_j^p(.)$, can be linear in some scenarios and in such cases, the data covariance matrix can be directly utilized for clustering data vectors. The data covariance matrix can be expressed as,

$$\mathbf{C}_{ij} = {N_s}^{-1} \sum_{n=1}^{N_s} (x_i(n) - \bar{x}_i)(x_j(n) - \bar{x}_j) \tag{3.1}$$

$$\mathbf{C} = \mathbf{B}\mathbf{D_s}\mathbf{B} \tag{3.2}$$

where, $\bar{x}_i = {N_s}^{-1} \sum_{n=1}^{N_s} x_i(n)$ is the mean vector, $\mathbf{C} \in \mathbb{R}^{P \times P}$ is the data covariance matrix, the matrix $\mathbf{D_s} \in \mathbb{R}^{Q \times Q}$ is the source covariance matrix. Since the sources are assumed to be uncorrelated in nature, the matrix $\mathbf{D}_s$ will be a diagonal matrix. The entries of matrix $\mathbf{B} \in \mathbb{R}^{P \times Q}$ are such that $\mathbf{B}_{pq} \neq \{0\}$ if the p-th signal observes the q-th source and is $\mathbf{B}_{pq} = \{0\}$ otherwise. Also, since the signals observing the same source are correlated together and the ones observing different sources are uncorrelated, the covariance matrix can be considered to be block diagonal under finite permutations of row and equivalent column exchange operations. This is true since performing a row and column exchange operation is equivalent to modifying the order of signals before finding the covariance matrix. Thus, a block diagonal covariance matrix essentially appears when all the signals belonging to the same class are ordered together.

Since the goal is to perform signal/data vector clustering according to the underlying information, the problem can be reformulated to finding a factor $\mathbf{M}$ of the block diagonal

covariance matrix $\mathbf{C}$ such that $\mathbf{M} = \mathbf{B}\mathbf{D}^{1/2}$. As we do not have access to the source signal and the data model, but only the sensor measurements, we solve the matrix factorization problem $||\mathbf{C} - \mathbf{M}\mathbf{M}^T||_F$ under sparsity constraints to recover $\mathbf{M}$ such that $supp(\mathbf{M}) = supp(\mathbf{B}\mathbf{D}^{1/2})$, thus obtaining the underlying clustering. The support of columns of $\mathbf{B}\mathbf{D}^{1/2}$ points to the signal's class-membership information contained in the columns of $\mathbf{B}$, since the source covariance matrix $\mathbf{D}_s$ is diagonal.

3.1 Inducing Sparsity

There are infinite solutions to the $||\mathbf{C} - \mathbf{M}\mathbf{M}^T||_F$ factorization problem, thus to recover a sparse $\mathbf{M}$ whose column's support is indicative of the clustering within the acquired signals, the optimization problem has to be supplemented with sparsity inducing norms. Ideally we would want to impose a ℓ_0 penalty, but since ℓ_0 would be non convex and non differentiable, a relaxed variant in the form of ℓ_1 penalty is most commonly used. The ℓ_1 regularization is pretty effective in the generic case, though it may not provide the sparsest solution. Different sparsity inducing formulations have been proposed in the past, in [45] a re-weighted ℓ_1 formulation has been explored. Regularization with ℓ_p where, $0 < p < 1$ has also been explored in [46, 47].

The sparsity regularized matrix factorization problem can then be rewritten as,

$$\begin{aligned} &||\mathbf{C} - \mathbf{M}\mathbf{M}^T||_F^2 + \lambda\big(Sparsity\big) \\ &\text{s. to} \qquad \mathbf{M} \geq 0 \end{aligned} \tag{3.3}$$

where, λ is the regularization parameter controlling the sparsity. $\mathbf{M} \geq 0$ are the entry-wise nonnegativity constraints on $\mathbf{M}$.

3.1.1 Kernelized Framework and Kernel Selection

As was discussed in the previous chapter, for most real world applications the source-to-sensor relationship, $f_j^p(\cdot)$, is non-linear. Thus, linear correlations can not find meaningful relationships between signals. To address this, the data can be mapped through a non linear mapping, $\phi(\cdot)$, to a higher dimensional space and the kernel trick maybe utilized to efficiently compute the correlations [10]. The mapping can thus be represented as, $\boldsymbol{\phi} : \mathbb{R}^N \rightarrow \mathbb{R}^{N_s \times F_H}$, where F_H represents the dimensionality of a higher dimensional feature space. The non linear mapping $\phi(\cdot)$ is applied across each dimension of the input vectors $\boldsymbol{x}$ as $\boldsymbol{x}_n \rightarrow \phi(\boldsymbol{x}_n)$, where $\boldsymbol{x}_n$ is the vector containing the n-th sample of the sensor measurements $x_i(n)$ across all $i \in \{1, ..., P\}$ sensor signals. Thus, $\boldsymbol{\phi}(\boldsymbol{x}_n) = [\hat{\phi}(x_i(n)), ..., \hat{\phi}(x_N(n))]^T$, where $\hat{\phi} : \mathbb{R} \rightarrow \mathbb{R}^{F_H}$. The kernelized correlations can then be stated as,

$$\hat{\mathbf{K}}_{\mathbf{x}} = N^{-1} \sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}^T(\mathbf{x}_n), \tag{3.4}$$

The objective of the optimization problem for the kernelized matrix factorization can be stated as:

$$F(\mathbf{M}) = ||\hat{\mathbf{K}}_x - \mathbf{M}\mathbf{M}^T||_F^2 + \lambda\big(Sparsity\big)$$
$$\text{s. to} \qquad \mathbf{M} \geq 0 \tag{3.5}$$

Several approaches have been proposed in the literature for solving similar NMF based clustering problems, [16, 17, 19, 23]. The resultant sparse factors are then clustered using K-means iterations or simply by imposing an $argmax$ operation to assign the class based on the dimension corresponding to the largest magnitude. Neither of the sparse formulations impose hard clustering constraints and thus after solving with these techniques, there is either more than one class with a non zero value or the weightage of the sparsity related regularization parameters has to be tuned to achieve the required sparsity. In the

following section we discuss a mixed integer linear programming (MILP) reformulation of the matrix factorization based clustering problem. Here, sparsity is imposed as a hard constraint, thus the clustering obtained from the relaxed MILP formulation post convergence has no ambiguity in the solution as every data vector is assigned only to a single class. Also, no tuning of the regularization parameters is required.

3.2 Mixed Integer Reformulation of NMF

We modify the matrix factorization formulation explained above to model it as an MILP. The problem described in (3.5) has a non linear objective w.r.t to $\mathbf{M}$. This non linearity in (3.5) is essentially due to three components; 1) the square of Frobenius norm, $||.||_F^2$ or essentially the square of the error of factorization; 2) the matrix factorization part, which involves the matrix product $\mathbf{M}\mathbf{M}^T$; and 3) the sparsity inducing regularization. The first cause of non linearity, $||.||_F^2$, appears since the majority of the factorization procedures in literature rely on different variants of gradient schemes to find a stationary point. Since Frobenius norm is differentiable across its domain, it serves as the obvious choice. Since as part of reformulating the problem into a MILP we are not dependent on the differentiability of the cost, the $||\mathbf{K} - \mathbf{M}\mathbf{M}^T||_F^2$ is replaced by $||\mathbf{K} - \mathbf{M}\mathbf{M}^T||_1$. Further, the minimization of ℓ_1 norm $||\mathbf{K} - \mathbf{M}\mathbf{M}^T||_1$ can be replaced with its epigraph, see e.g., [48]:

$$\begin{aligned} \arg\min_{\mathbf{M}} ||\mathbf{K} - \mathbf{M}\mathbf{M}^T||_1 = \arg\min_{\mathbf{M},T} \quad & \sum_{i=1}^{P}\sum_{j=1}^{P} \mathbf{T}_{ij} \\ s.to \;\; & \mathbf{T} \geq \mathbf{K} - \mathbf{M}\mathbf{M}^T, \\ & \mathbf{T} \geq -(\mathbf{K} - \mathbf{M}\mathbf{M}^T). \end{aligned} \tag{3.6}$$

where, $\mathbf{T}_{ij}$ is the element of matrix $\mathbf{T}$ from the i-th row and the j-th column.

Next, we linearize the product of the matrix factors $\mathbf{M}\mathbf{M}^T$. Consider $\mathbf{W}$ as the product of matrices $\mathbf{M}\mathbf{M}^T$, thus we have $\mathbf{W}_{ij} = \sum_{k=1}^{Q} \mathbf{M}_{ik} * \mathbf{M}_{jk}$ where, the subscripts ij represent the ith term of the jth column. Also, let us define $\mathbf{W}_{ijk} = \mathbf{M}_{ik} * \mathbf{M}_{jk}$ as the product of the k th term. This product term can be linearized by applying the popular approach of McCormick's envelope [49] and adding constraints representing the convex and concave envelopes on each of the product terms.

Since the kernel matrix $\mathbf{K}$ is normalized to unit magnitude, the entries of $\mathbf{M}$ can have a maximum magnitude of 1. Thus for $\mathbf{W}_{ijk} = \mathbf{M}_{ik} * \mathbf{M}_{jk}$ where, $0 \leq \mathbf{M}_{ik}, \mathbf{M}_{jk} \leq 1 \;\; \forall i, j \in 1, ..., P, \; k \in 1, ..., Q$, we have the reformulation as $\mathbf{W}_{ijk} \leq \mathbf{M}_{ik}, \mathbf{W}_{ijk} \leq \mathbf{M}_{jk}, \mathbf{W}_{ijk} \geq \mathbf{M}_{ik} + \mathbf{M}_{jk} - 1$.

The sparsity inducing parameter in (3.5) was mainly introduced to ensure that each row of M has only one non zero entry, thus enforcing the set partitioning constraint ensures that every sensor can observe only one source. Such a constraint can easily be enforced by adding a linear constraint of the form $\sum_{k=1}^{Q} \mathbf{M}_{ik} = 1 | \forall i \in \{1, ..., P\}$. Applying the aforementioned steps, the minimization problem in (3.5) can be reformulated as:

$$\hat{\mathbf{M}}, \hat{\mathbf{T}}, \hat{\mathbf{W}}, \hat{\mathbf{U}} \in \arg \min_{\mathbf{M}, \mathbf{T}, \mathbf{W}, \mathbf{Q}} \sum_{i=1}^{P} \sum_{j=1}^{P} \mathbf{T}_{ij} \tag{3.7}$$

$$s.to$$

$$\mathbf{W}_{ijk} \leq \mathbf{M}_{ik} \; \forall \; i, j \in \{1, ..., P\}, \; k \in \{1, ..., Q\}$$

$$\mathbf{W}_{ijk} \leq \mathbf{M}_{jk} \; \forall \; i, j \in \{1, ..., P\}, \; k \in \{1, ..., Q\}$$

$$\mathbf{W}_{ijk} \geq \mathbf{M}_{ik} + \mathbf{M}_{jk} - 1 \; \forall \; i, j \in \{1, ..., P\}, \; k \in \{1, ..., Q\}$$

$$\mathbf{U}_{ij} = \sum_{k=1}^{Q} \mathbf{W}_{ijk} \; \forall \; i, j \in \{1, ..., P\}$$

$$0 \leq \mathbf{U}_{ij} \leq 1 \; \forall \; i, j \in \{1, ..., P\}$$

$$\mathbf{T}_{ij} \geq \mathbf{K}_{ij} - \mathbf{U}_{ij} \; \forall \; i, j \in \{1, ..., P\}$$

$$\mathbf{T}_{ij} \geq \mathbf{U}_{ij} - \mathbf{K}_{ij} \; \forall \; i, j \in \{1, ..., P\}$$

$$\sum_{k=1}^{Q} \mathbf{M}_{ik} = 1 \; \forall \; i \in \{1, ..., P\}$$

$$\mathbf{M} \in \mathbb{B}^{P \times Q}, \mathbf{W} \in \mathbb{B}^{P \times P \times Q}$$

where, $\mathbb{B}$ indicates the set of binary numbers, $\{0, 1\}$.

For solving the mixed integer problem, a branch and bound method with cutting planes has been used. We use the openly available Gurobi MILP solver [50] for this. The Gurobi toolbox allows access to multiple parameters for solving the problem, two of the main parameters for which we define the values are, $'\mathbf{Cuts}'$ and $'\mathbf{MIPFocus}'$. The $'\mathbf{MIPFocus}'$ controls the balance of effort between finding new solutions, and trying to prove that the current solution is optimal. If the value is 1, it encourages the solver to find feasible solutions quickly. $'\mathbf{Cuts}'$ controls the aggression of cuts. We set the value of this variable to 1, which implies moderate cut generation. These parameters internally initialize different branching and cutting plane methods and gave the best results in our simulations.

The solution obtained from the branch and bound method for the problem in (3.7) is globally optimal as the solver has essentially explored all the nodes of the branching tree where a solution with a lower objective value can exist. The solution though may not be unique, for example the columns of $\mathbf{M}$ can be interchanged while resulting in the same matrix $\mathbf{M}\mathbf{M}^T$ and thus the same cost. But both of these solutions result in the same clustering accuracies. For further discussion on optimality in mixed integer problems, please refer to [50, 51].

Next, we showcase the performance of the proposed approach on 2 different datasets, a hyperspectral image dataset, Salinas [2], and an image clustering dataset, COIL20

[3]. We compare the performance of the proposed technique with four different schemes: 1) K-means, 2) kernel SVM with 25% of the data used for training, 3) standard NMF and 4) graph NMF (GNMF) [23]. For GNMF, in the case of Salinas, we have set the value of the variable α to 1, since this gave better clustering accuracies than the suggested value of 100. In the case of COIL20 we have continued to use a value of 100.

3.2.1 Salinas Hyperspectral

For the hyperspectral dataset, the images captured by the AVIRIS sensing system [2] over the Salinas valley in California, with a spatial resolution of 3.7m per pixel have been considered. The dataset contains a total of 224 spectral reflectance bands for each pixel, observing materials from 16 different classes.

The objective here is to consider each of the image pixels as an independent signal and cluster the pixels pertaining to the same class or observing the same material together. For our simulations, we randomly choose 4 materials out of these 16 and select 15 pixels each from these randomly chosen materials. The experiment is repeated over a total of 100 random material and pixel selections.

On this data, the kernel trick was applied with a variance of 10^7. After applying the kernel trick, 60x60 covariance matrices are obtained, pertaining to the covariances of the spectral reflectance values for those 60 pixels. The covariance matrix should be appropriately centered. Fig. 7.2 shows the clustering accuracies for the 5 schemes through a box plot, the red line in the boxes indicates the median of the clustering accuracy across all trials. The proposed scheme outperforms all the 4 methods, including the KSVM, which is a supervised learning scheme. The average accuracy across all the trials has been given in Table 3.1 in the row named $Salinas60$. The clustering accuracies are calculated as explained in [39].

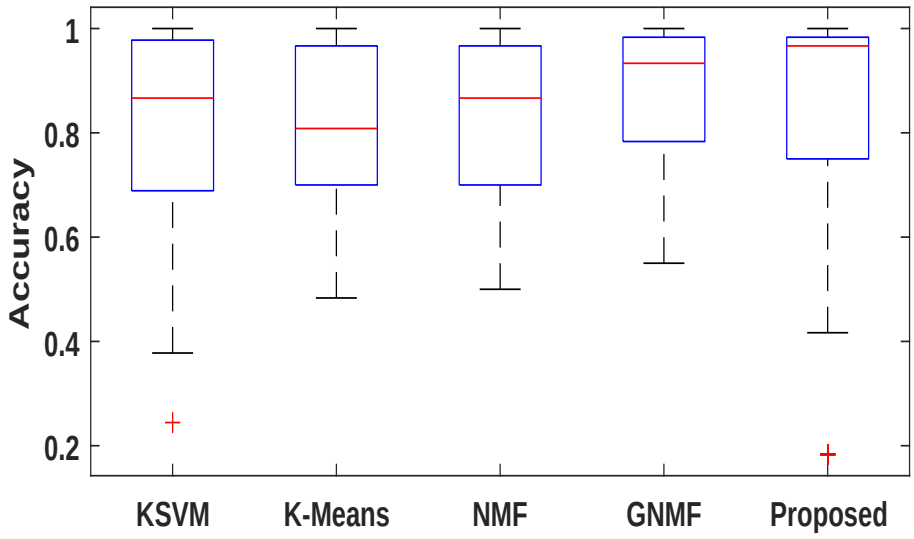


Figure 3.1. Boxplot comparing the accuracies of 4 different schemes with the proposed method for the Salinas hyperspectral image dataset [2]. The central red mark in the box refers to the median accuracy, and the edges of the box mark 25^{th} and 75^{th} percentiles of the accuracy across all trials.

Table 3.1. Mean clustering accuracy of the 5 schemes for the 2 datasets.

	KSVM	**KMeans**	**NMF**	**GNMF**	**Proposed**
Salinas 60	82.8	81.9	83.0	87.9	**93.6**
COIL20	69.1	72.9	81.9	84.2	**85.6**
Salinas 300	95.3	84.4	82.0	88.5	**93.3**

3.2.2 COIL20 dataset

The second dataset is the COIL20 image dataset which contains images of objects taken from 72 different angles. The objective here is to cluster the images based on the object, thus in accordance with the problem description, the pixels of each image can be considered as an independent signal irrespective of the common origin.

In this dataset, across each trial, 3 classes are randomly selected from the 20 available classes, and then 15 images are randomly selected from each of the selected classes. The kernel trick is applied with a variance of $\sigma^2 = 10^{-1}$. After applying the kernel trick, a

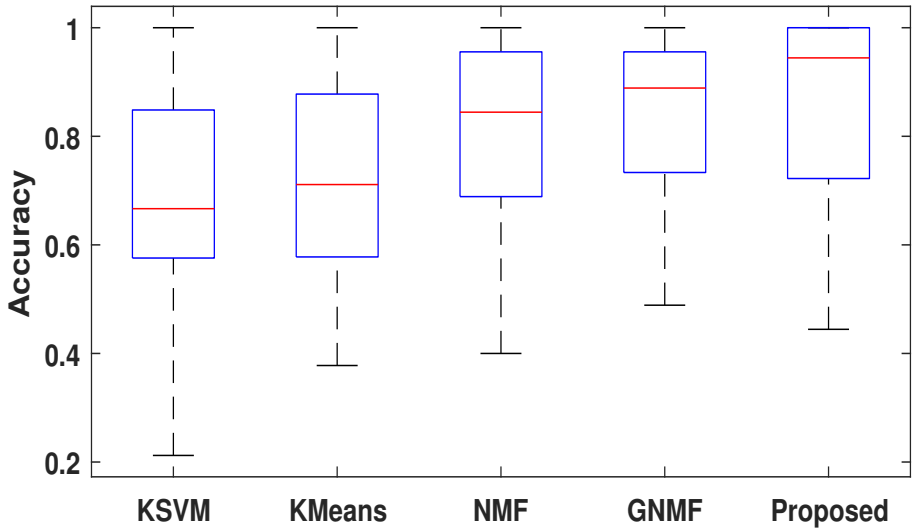


Figure 3.2. Boxplot comparing the accuracies of 4 different schemes with the proposed method for the COIL20 image clustering dataset [3].

45x45 covariance matrix is obtained, pertaining to the covariances of the 45 images. Before calculating the covariance, the data was normalized as was done in the simulations for [23].

In Fig. 3.2 we present the performance of the clustering schemes for this dataset through a box plot. The proposed scheme outperforms all the 4 methods. The mean or average accuracy across all trials can be found in Table 3.1.

3.2.3 Scalability

As evident from the discussion above, the proposed approach gives high clustering accuracy with both the datasets, but as a trade off, the MILP based algorithm scales poorly when a larger chunk of the dataset is considered in a single iteration. In Fig. 3.3, the dotted blue line shows the time taken by the Gurobi solver to cluster the pixels in Salinas dataset for different number of pixels. As evident, the time increases super-linearly and thus beyond 60 pixels the time is outside the scope of the curve. Therefore the original formulation in its raw form may not be feasible for larger number of data points. To overcome this, an iterative solution can be considered.

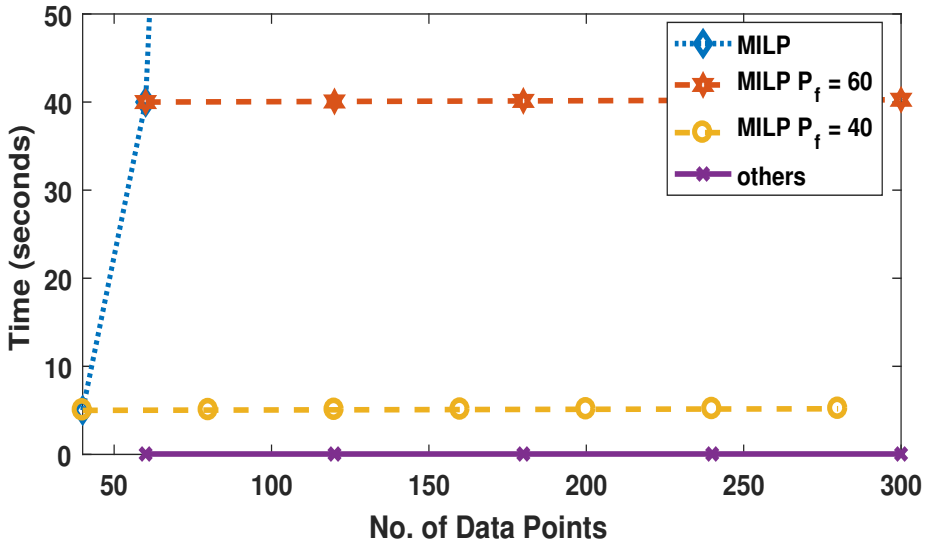


Figure 3.3. Comparison of the execution time as the complexity of the problem is increased for the Salinas hyperspectral image dataset [2].

As a first step, instead of using all the P signals for generating the kernel covariance matrix $\mathbf{K}$, a fraction of the signal, P_f can be used for generating a $P_f \times P_f$ covariance matrix. Lets denote this set of P_f signals as $\mathbb{S}_{P1}$. For this kernel covariance matrix, appropriate factors $\mathbf{M}_1 \in \mathbb{B}^{P_f \times Q}$ should be found by solving (3.7). In step-2, a kernel cross-covariance matrix, $\mathbf{K}_{12}$ between signals $\mathbb{S}_{P1}$ and a new set of P_f signals (say $\mathbb{S}_{P2}$) from the remaining $P - P_f$ signals should be generated. This matrix $\mathbf{K}_{12}$ can then be factorized as $\mathbf{M}_1\mathbf{M}_2^T$, and as we already know $\mathbf{M}_1$ from step-1, evaluating $\mathbf{M}_2$ is trivial and can be evaluated extremely efficiently. Step-2 can be repeated with different sets of signals $\mathbb{S}_{P2}$ to achieve the corresponding matrices $\mathbf{M}_2$ and thus the underlying clustering. The accuracies for Salinas with a total of $P = 300$ pixels and $P_f = 60$ have been reported in Table 3.1 in the row corresponding to $Salinas300$.

In Fig. 3.3, the yellow and the orange curves show the MILP processing times for different values of P_f. As is evident from the figure, the cross covariance based modification makes MILP scale linearly with an added constant value due to the factorization in

step-1. For the NMF and GNMF algorithms the complexity increases linearly too and since the codes are extremely well optimized the processing times are under 1 sec. The step-1 processing time in the MILP scheme can also be brought down significantly by optimizing the way the Gurobi toolbox is queried, but the important aspect here is that with the cross covariance based modification the dataset can be processed in small chunks (each of P_f signals), and the time complexity becomes a linear function of the number of chunks and thus the number of data points.

CHAPTER 4

UNSUPERVISED KERNEL LEARNING

The family of the kernel function and the value of its parameters have a huge impact on the clustering performance and thus kernel selection or kernel learning needs to be performed properly to extract the correct data correlations. In the existing literature, the kernel selection/learning task has predominantly been achieved in a supervised setting [27, 28, 32, 52, 53]. However, an important aspect of the CCA/NMF-based clustering methods is that they do not require any training data. Thus in order to keep the entire clustering process unsupervised, the kernel learning/selection objective has to be accomplished in an unsupervised setting as well.

To accomplish this task in an unsupervised setting, we need to identify the properties of a good kernel and leverage them for kernel selection/learning. In a linear setting, it is trivial to show that if the underlying source signals are uncorrelated, the sensors observing a common source will be correlated with each other and the sensors observing different sources will be uncorrelated. Thus, the covariance matrix, under column and equivalent row exchnage operations, will have a block diagonal structure. The column and row exchanges can be considered without any loss of generality since performing such operations on the covariance matrix is equivalent to modifying the order of sensors before forming the covariance matrix. Each diagonal block in the covariance matrix thus points to the sensors observing a given common source. The presence of this block diagonal structure (owing to the uncorrelated nature of the signals from different classes) is the factor that influences the clustering accuracy of CCA/NMF based approaches. This is true since for a block diagonal matrix with properly selected sparsity coefficients $\lambda_\rho^{\mathbf{D}}$ and $\lambda_\rho^{\mathbf{E}}$, the correct clustering can be

identified with CCA [11]. The same holds true in the NMF setting as well [17]. Therefore, the main feature of an effective kernel for correlation based clustering is its capability to produce a strong block diagonal covariance matrix.

Consider the family of Gaussian kernels, the parameter that controls the shape of a kernel is its variance. To understand the impact of variance on the block diagonal behavior of the kernel covariance, consider 20 pixels from a hyperspectral image from the Indian Pines dataset [2], taken from 4 different classes. 5 pixels are taken from each of the classes and have been arranged sequentially as per their class labels, such that each set of pixels represents the same class. If we choose a kernel variance that is too small, we obtain a kernel covariance matrix that has high magnitude along the diagonal, while the rest of the entries are almost zero, Fig. 4.1(c). On the other hand, if we choose a very high variance value, all the entries of the covariance matrix have a unit magnitude, Fig. 4.1(b).

However, if we choose a suitable value, then we get a covariance matrix like the one shown in Fig. 4.1(a). The block diagonal structure of the matrix essentially points to the set of highly correlated pixels observing the same material. The low magnitude in the off block diagonal elements of the covariance matrix shows that they observe different materials, and are thus uncorrelated. From Fig. 4.1(a) it is easy to observe that the set of the rows corresponding to a given diagonal block are linearly independent to the rows from a different diagonal block, while being similar and thus linearly dependent to each other inside a block. Therefore, if we know that the data stems from Q classes, the objective is to achieve a kernel covariance matrix with Q independent sets of rows or Q block diagonals and thus a kernel covariance matrix with rank Q. A similar analogy will hold true even if the pixels are not ordered before finding the covariance matrix, since the block diagonal behavior can be achieved by row and column permutations. This is true even in the linear setting where the signals from different classes are assumed to be uncorrelated and thus the covariance has a rank Q. The same is emphasized by the following proposition.

Proposition 1. In a linear setting as described by (2.1) *and* (3.1)*, each of the block diagonals in the correlation matrix representing the high correlations between the signals from the same class can be considered to be rank-1 blocks and in a dataset with Q classes, there are Q such rank-1 diagonal blocks.*

Proof. From (3.2) we know that, $\mathbf{B}\mathbf{D}_s^{1/2}$ is a factorization for the correlation matrix $\mathbf{C}$. Also, for a given class q let $\mathcal{G}_q$ be the ground truth set containing the indices p of all the signals/data vectors from class q, i.e., $\{\mathbf{B}_{pq} \neq 0 | p \in \mathcal{G}_q\}$ from (3.2). We can then represent the correlation matrix as,

$$\mathbf{C} = \sum_{q=1}^{Q} d_q \mathbf{B}_{:,q} \mathbf{B}_{:,q}^T \tag{4.1}$$

where d_q is the q,q-th diagonal element of the matrix $\mathbf{D}_s$. Without loss of generality we can assume that the indices of vectors from each class have been ordered such that the members of $\mathcal{G}_q$ are consecutive. Thus, for the block diagonal $\mathbf{C}$, each block diagonal is essentially a result of the matrix product $\mathbf{B}_{:,q}\mathbf{B}_{:,q}^T$, and it has rank-1. □

Therefore, in the data with Q possible classes, the objective for effective kernel learning is towards finding kernels with underlying mapping $\boldsymbol{\phi}(\cdot)$ that can linearize the correlations in the higher dimensional space such that there are Q rank-1 blocks along the diagonal, and thus a kernel covariance matrix with a rank Q. The same analogy holds true even when the signals haven't been ordered based on the class labels.

4.1 Eigenvectors-Based Kernel Selection and Kernel Learning

For real world applications, the data is always corrupted by some noise, also the data vectors are finite length and only an estimate of the actual covariance can be evaluated. Thus, the covariance matrices evaluated in practice almost always have a rank greater than Q. To impose rank Q constraints in practice, we can impose constraints on the magnitude of the Q largest eigenvalues. But maximizing the sum of Q eigenvalues is insufficient as

matrices of the form considered in Fig. 4.1(b) have a constant magnitude across all entries, and will result in a feasible matrix which has less than Q strong eigenvalues (one in this case). Thus, its important not just to maximize the sum of Q eigenvalues but also to ensure that matrix has Q strong eigenvalues. To that end we propose maximizing the Q-th largest eigenvalue which ensures that the matrix has a rank of at least Q.

For the kernel selection problem, where the objective is to select the most appropriate kernel for correlation based clustering, a proper $\hat{\mathbf{K}}_x^{j^\star}$ can be selected, from a set of dictionary kernels, $\hat{\mathbf{K}}_x^j$ where $j \in \{1, ..., B\}$ as follows,

$$j^\star \in \underset{j \in \{1,...,B\}}{\text{argmax}} \Lambda^Q(\hat{\mathbf{K}}_x^j). \tag{4.2}$$

where, the function $\Lambda^i(.) : \mathbb{R}^{P \times P} \rightarrow \mathbb{R}$ represents the i-th largest eigenvalue of the input argument matrix and for a given matrix, $\mathbf{K}$, it is defined as:

$$\sum_{i=1}^{Q} \Lambda^i(\mathbf{K}) = \sup_{\mathbf{V}} \ \mathbf{V}^T \mathbf{K} \mathbf{V}$$
$$\text{s. to} \qquad \mathbf{V}\mathbf{V}^T = \mathbf{I}_Q, \tag{4.3}$$

where $\mathbf{I}_Q$ is the identity matrix of size $Q \times Q$ and the columns of matrix $\mathbf{V}$ are the eigenvectors of $\mathbf{K}$. $\mathbf{V}$ is a matrix of size $P \times Q$.

Given that the covariance matrices may be scaled differently for each of the kernel parameters, it is essential to normalize these matrices appropriately before finding the eigenvalues. Since the proposed method focuses on increasing the magnitude of the Q-th eigenvalue, an appropriate normalizing strategy is to scale each matrix by its trace (or sum of all its eigenvalues). Thus, post normalization, maximizing the Q-th eigenvalue is equivalent to maximizing the percentage share of the Q-th eigenvalue with respect to (w.r.t.) the sum of all the eigenvalues.

From the multiple kernel leraning (MKL) perspective, the problem formulates to finding a convex combination of kernels with the Q-th eignevalue maximized. Thus the objective in (4.2) translates to,

$$\begin{aligned} \boldsymbol{\alpha} \in \underset{\boldsymbol{\alpha}}{\text{argmax}} & \Lambda^{Q}(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) \\ \text{s. to} \quad & \boldsymbol{\alpha}_j \geq 0, \quad \textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j = 1. \end{aligned} \tag{4.4}$$

Unlike, the kernel selection problem in (4.2), where a single kernel from a set of B kernels (see [54]) has to be selected, the MKL problem mentioned above is highly non convex and very hard to solve. A possible solution emerges by alternatively expressing (4.4) as a difference of sum of eigenvalues,

$$\begin{aligned} \boldsymbol{\alpha} \in \underset{\boldsymbol{\alpha}}{\text{argmax}} & \sum_{i=1}^{Q} \Lambda^{i}(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q-1} \Lambda^{i}(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) \\ \text{s. to} \quad & \boldsymbol{\alpha}_j \geq 0, \quad \textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j = 1. \end{aligned} \tag{4.5}$$

The constraint $\sum_{j=1}^{B} \boldsymbol{\alpha}_j = 1$ ensures that the weights for all the kernels sum to 1, and $\boldsymbol{\alpha}_j \geq 0$ is required to ensure that the sum $\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j$ is a valid positive semidefinite kernel.

The sum of eigenvalues of a linear combination of matrices is a convex function since all the eigenvalues are continuous functions and the supremum of continuous functions is convex [55]. The problem appears since, the function to be minimized in (4.5) is non-convex. Nonetheless, both the sum of $Q-1$ eigenvalues, and the sum of Q eigenvalues are individually convex in (4.5), resulting in a difference of two convex functions. This enables tackling the problem in (4.5) by utilizing the difference of convex algorithm (DCA) proposed in [56].

4.2 Algorithm

4.2.1 Difference of Convex Formulation

The difference of convex functions algorithm [56] deals with objective functions of the type $\mathbf{F}(\mathbf{X}) = \mathbf{G}(\mathbf{X}) - \mathbf{H}(\mathbf{X})$ where, $\mathbf{G}(\mathbf{X})$ and $\mathbf{H}(\mathbf{X})$ are convex functions. The algorithm is a two step iterative approach, wherein at the $k+1$-th iteration, the concave part of the function [i.e., $-\mathbf{H}(\mathbf{X})$] is approximated by an affine majorization of the $\left\langle \mathbf{X}, \left\{ \frac{\partial -\mathbf{H}(\mathbf{X})}{\partial \mathbf{X}} |_{\mathbf{X}=\mathbf{X}^k} \right\} \right\rangle$, where, $\left\langle \cdot, \cdot \right\rangle$ represents the inner product of the two input arguments. The majorization is used as a surrogate for minimizing the relaxed $\mathbf{F}(\mathbf{X})$. The algorithm can then be represented as:

$$\begin{aligned} \mathbf{Y}^k &\in \partial \mathbf{H}(\mathbf{X}^k) \\ \mathbf{X}^{k+1} &\in \underset{\mathbf{X}}{\operatorname{argmin}} \mathbf{G}(\mathbf{X}) - \langle \mathbf{X}, \mathbf{Y}^k \rangle \end{aligned} \tag{4.6}$$

where, $\partial \mathbf{H}(\mathbf{X}^k)$ is the sub-gradient of $\mathbf{H}(\mathbf{X})$ at $\mathbf{X}^k$. The superscript k represents the k-th iterate of the DCA. For the eigenvalue problem in (4.5), this translates to determining

$$[\tilde{\boldsymbol{\alpha}}^k]_Q \in \partial \Bigg(\sum_{i=1}^{Q} \Lambda^i (\sum_{j=1}^{B} \alpha_j^k \hat{\mathbf{K}}_x^j) \Bigg). \tag{4.7}$$

where, $\partial(.)$ refers to the subgradient, which here is found for the sum of the Q eigenvalues and can be calculated based on the discussion about the subgradient of the maximum eigenvalue [57]. Thus, a possible subgradient for the expression in (4.7) is

$$[\tilde{\boldsymbol{\alpha}}^k]_Q = \begin{bmatrix} \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_Q^k \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_Q^k \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_Q^k \end{bmatrix} \tag{4.8}$$

where, $\mathbf{z}_i^k$ is the i-th eigenvector of matrix $\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j$ evaluated at $\boldsymbol{\alpha} = \boldsymbol{\alpha}^k$. Then, the tresulting convex problem with the majorization approximation is given as,

$$\begin{aligned} \boldsymbol{\alpha}^{k+1} \in \underset{\boldsymbol{\alpha}}{\text{argmin}} \;\; & \sum_{i=1}^{Q-1} \Lambda^i\Big(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j\Big) - \langle \boldsymbol{\alpha}, [\tilde{\boldsymbol{\alpha}}^k]_Q \rangle \\ \text{s. to} \quad & \textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j = 1, \quad \boldsymbol{\alpha}_j \geq 0, \quad \forall j \in \{1, ..., B\} \end{aligned} \tag{4.9}$$

and can be minimized easily using either gradient descent or the interior point method. These two steps are repeated until convergence , i.e. until $\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k < \epsilon$, with ϵ corresponding to a small positive constant (this approach is guaranteed to converge to a stationary point. The proof for a more generic setting of joint NMF based clustering and kernel leraning has been presented in the Appendix B. The proof for the kernel learning can be easily inferred from the same by ignoring the discussions about the NMF related variables).

4.3 CASE STUDIES

In this section the performance of the proposed approaches are compared against two supervised kernel selection/learning schemes relying on kernel alignment. For the analysis presented in this section we acronymize the proposed eigenvalue based unsupervised kernel selection scheme as EV and the unsupervised kernel learning scheme as EVKL. The kernel target alignment based supervised kernel selection scheme [27] is acronymized as KTA and the supervised kernel learning scheme [28] is acronymized as ALIGNF. To quantify their performances we find the clustering accuracy with kernelized CCA (Sec. 2.3) on two different data-sets, while using the kernels as calculated by the four schemes. The comparative results have been presented as follows.

4.3.1 UniMiB dataset

The first data set is the University of Milano Bicocca Smart-phone-based Human Activity Recognition (UniMiB) dataset [4]. This set has data from 30 users performing a wide range of activities, a total of 17 activities including 9 daily activities like walking, running, climbing stairs. To showcase the performance of CCA, over a different set of activities, we use data from 3 classes pertaining to climbing, running and walking. For this data-set, the results are averaged out over signals from the 30 users in the data-set.

The accuracy of the four schemes for the UniMiB data-set are given in Table 4.1. Here we show the detailed results for all the users and the averages for both, the first 5 sets of users, as well as for all the 30 users have been provided. We can observe that EV is able to achieve better accuracy in 7 out of 30 users, despite being unsupervised compared to KTA, which is supervised. Similarly, EVKL performs better than ALIGNF in 11 out of 30 cases. Even on an average, both of the unsupervised approaches perform reasonably close to their supervised counterparts. The superior accuracy illustrates the capabilities of the proposed unsupervised method, even when compared to a supervised scheme.

4.3.2 Hyperspectral dataset

For the second data set, tests are performed on hyperspectral images gathered by the AVIRIS sensing system [2] over the Salinas valley in California, with a spatial resolution of 3.7m per pixel. This dataset contains 224 spectral reflectance bands, encompassing 16 objects. Out of these 16 objects, five random combinations of 4 objects were chosen, and 30 pixels were randomly chosen from each object for five different trials. Salinas was chosen because it is a large image with a massive number of pixels pertaining to each object, thereby ensuring substantial variability in randomly choosing a set of pixels from such large sample sizes. Thus, we had a total of 120 pixels from 4 objects with even distribution.

	EV	KTA [27]	EVKL	ALIGNF [28]
1	55.55%	60.00%	86.66%	82.22%
2	49.33%	40.66%	46.66%	50.67%
3	57.14%	76.19%	60.00%	67.62%
4	58.09%	55.23%	56.19%	59.05%
5	64.44%	75.55%	77.03%	74.81%
6	65.71%	78.09%	79.04%	60.95%
7	50.00%	65.83%	44.16%	57.50%
8	60.00%	70.83%	65.83%	61.66%
9	55.83%	70.83%	45.00%	71.66%
10	65.55%	58.88%	73.33%	60.00%
11	89.33%	70.66%	64.00%	89.33%
12	52.59%	64.44%	45.92%	67.40%
13	60.00%	68.88%	65.55%	63.33%
14	54.28%	64.76%	50.47%	62.85%
15	45.18%	64.44%	46.66%	61.48%
16	59.25%	43.70%	74.07%	74.81%
17	61.66%	80.83%	82.50%	65.00%
18	73.33%	80.00%	66.66%	80.00%
19	89.33%	73.33%	77.33%	85.33%
20	47.61%	56.19%	59.04%	47.61%
21	80.00%	93.33%	80.00%	66.66%
22	42.66%	56.66%	52.66%	48.00%
23	52.38%	58.09%	54.28%	59.04%
24	45.00%	66.66%	69.16%	67.50%
25	46.66%	74.16%	43.33%	64.16%
26	63.70%	63.70%	65.18%	58.51%
27	55.83%	75.00%	55.83%	63.33%
28	47.61%	58.09%	50.47%	57.14%
29	53.33%	73.33%	48.57%	61.90%
30	49.62%	46.66%	42.96%	59.25%
Avg(5)	56.91%	61.53%	65.31%	66.87%
Avg(30)	58.37%	66.17%	60.96%	65.00%

Table 4.1. Clustering accuracies for CCA with four different kernel selection/learning schemes on UniMib dataset.

	EV	KTA [27]	EVKL	ALIGNF [28]
1	88.13%	88.13%	88.07%	87.27%
2	79.80%	79.80%	86.33%	88.47%
3	79.80%	79.80%	80.87%	85.80%
4	81.67%	81.67%	84.93%	83.40%
5	84.80%	84.80%	84.80%	90.07%
Avg(5)	82.84%	82.84%	85.00%	87.00%

Table 4.2. Clustering accuracies for CCA with four different kernel selection/learning schemes on Salinas Hyperspectral dataset.

As before, the accuracy of all the four schemes for the Salinas dataset are provided in Table 4.2. Each row represents the average of five different trials for each combination of four randomly selected objects, where each of the five trials is an average of five separate instances of CCA being conducted with random initialization. Thus, each row is essentially an average of 25 trials. For this dataset, both of the kernel selection schemes always select the same kernel and thus have same accuracies across each of the experiments. This demonstrates the superior performance of the EV approach in selecting the correct kernels, inspite of being unsupervised. Even among the kernel learning approaches, EVKL and ALIGNF have similar accuracies for each of the experiments and thus, similar average accuracies.

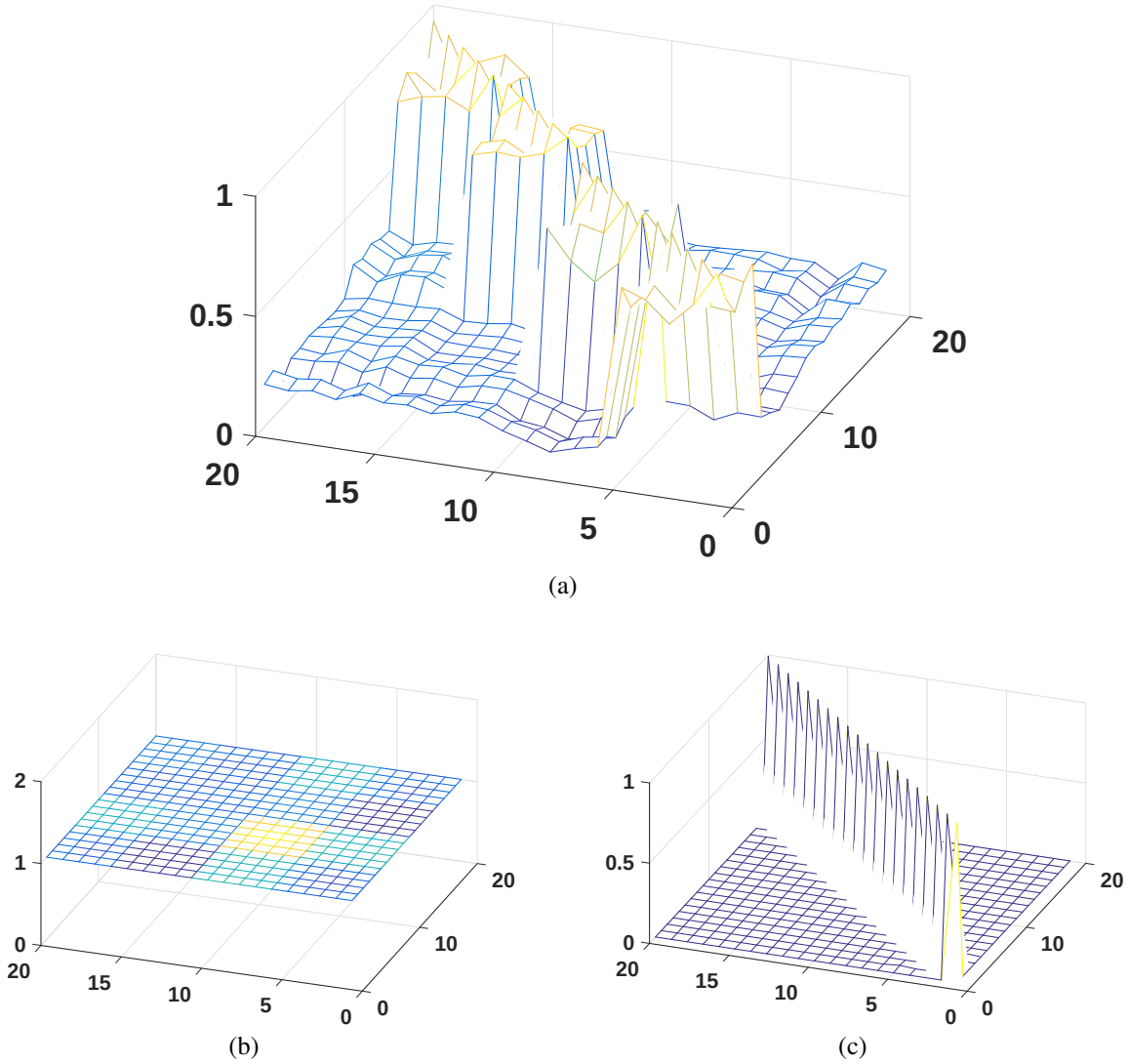


Figure 4.1. Effect of selecting different kernel variance σ^2 in the kernel covariance structure: (a) $\sigma^2 = 10^{3.5}$, (b) $\sigma^2 = 10^{10}$, (c) $\sigma^2 = 10^{-5}$.

CHAPTER 5

JOINT CLUSTERING AND KERNEL LEARNING

Kernel selection or multiple kernel learning methods discussed in the previous chapter improve the performance of the correlation based clustering methods by identifying the correct feature space in which the data correlations/covariance lends the maximum information towards the clustering objective. In this chapter, the eigenvalue maximization based kernel learning method is combined with the NMF based clustering approach into a joint clustering and kernel learning framework. We prove that, under certain assumptions, the joint formulation is guaranteed to select the best convex combination of kernels while simultaneously identifying the underlying clustering. The proof of convergence of the joint formulation to a stationary point has also been established.

In Chapter 3 the NMF based clustering objective was introduced and solved under hard clustering constraints using the branch and bound approach. The branch and bound schemes scale poorly and thus a cross covariance based method was also presented. The cross covariance approach though scales linearly but doesn't utilize the information from the entire dataset simultaneously and the performance is dependent on the accuracy achieved in the first batch of factorization. To utilize a larger section (or the entire data), we have to fall back on the original sparsity regularized formulation such that the framework can be solved in reasonable time utilizing the gradient based approaches.

The problem in (3.5) can be relaxed to a biconvex objective, thus, the block diagonal covariance matrix $\mathbf{C}$ is factorized as two factors $\mathbf{M}$ and $\mathbf{N}$, such that $\mathbf{M} = \mathbf{N} = \mathbf{B}\mathbf{D}_s^{1/2}$. Since we do not have access to the source signals and the data model, but only the sensor measurements, we solve the matrix factorization problem $||\mathbf{C} - \mathbf{M}\mathbf{N}^T||_F$ under sparsity

constraints to recover $\mathbf{M}$ and $\mathbf{N}$ as $\mathbf{B}\mathbf{D}_s^{1/2}$ and thus obtain the underlying clustering. It should be noted that we do not explicitly enforce constraints to make $\mathbf{M} = \mathbf{N}$ since we are not interested in recovering $\mathbf{M}$ and $\mathbf{N}$ exactly as $\mathbf{B}\mathbf{D}_s^{1/2}$ but are interested in the sparse solution wherein $supp(\mathbf{M}) = supp(\mathbf{N}) = supp(\mathbf{B}\mathbf{D}_s^{1/2})$, where $supp(\cdot)$ is the support of the columns of the input matrix argument. The support of the columns of $\mathbf{B}\mathbf{D}_s^{1/2}$ points to the signal's class-membership information contained in columns of $\mathbf{B}$, since the source covariance matrix $\mathbf{D}_s$ is diagonal.

5.1 Inducing Sparsity

In [58, 59], the $\ell_1 - \ell_2$ penalty has been introduced and its advantages over many of the other sparsity metrics have been discussed. Beyond giving a high degree of sparsity, it has the added advantage of being Lipschitz continuous and can be effectively represented by a difference of convex functions formulation. Thus, we utilize the $\ell_1 - \ell_2$ regularization for the non negative matrix factorization task discussed here (more details in [58]).

The sparsity regularized matrix factorization problem can then be rewritten as,

$$\begin{aligned} ||\mathbf{C} - \mathbf{M}\mathbf{N}^T||_F^2 + \lambda\big(\sum_{i=1}^{P} ||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2 \\ + ||\mathbf{N}_{i,:}||_1 - ||\mathbf{N}_{i,:}||_2\big) \\ \text{s. to} \qquad \mathbf{M} \geq 0, \mathbf{N} \geq 0 \end{aligned} \tag{5.1}$$

where, $||\cdot||_1$ and $||\cdot||_2$ represent the ℓ_1 and ℓ_2 norms of a vector and λ is the regularization parameter controlling the sparsity. $\mathbf{M}_{i,:}$ and $\mathbf{N}_{i,:}$, refer to the i-th row of $\mathbf{N}$ and $\mathbf{M}$ matrices respectively. $\mathbf{M} \geq 0, \mathbf{N} \geq 0$ are the entry-wise nonnegativity constraints on $\mathbf{M}$ and $\mathbf{N}$.

It is important to note that beyond the sparsity properties of the $\ell_1 - \ell_2$ norm, it is particularly well suited for the clustering problem because it can potentially enforce the set partitioning or hard clustering constraint. If $||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2 = 0$ is imposed as a

constraint $\forall i \in \{1, ..., P\}$, the feasible set is reduced only to the points where each row of $\mathbf{M}$ can have at most one non-zero entry. This subsequently conforms with our setting wherein every signal belongs only to a single class. Thus, the formulation in (5.1), can also be visualized as a Lagrangian relaxation of the matrix factorization problem under the set partitioning constraint.

Combining the MKL problem with the NMF based matrix clustering discussed in the previous chapter, the objective for the overall clustering problem can thus be stated as:

$$\begin{aligned} F(\mathbf{M}, \mathbf{N}, \boldsymbol{\alpha}) &= || \sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j - \mathbf{M}\mathbf{N}^T ||_F^2 \\ +\lambda\big(\sum_{i=1}^{P} ||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2 + ||\mathbf{N}_{i,:}||_1 - ||\mathbf{N}_{i,:}||_2 \big) \\ +\mu\big(\sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) \big) \end{aligned} \tag{5.2}$$

Proposition 2. If there exists a linear combination of the B dictionary kernels, such that the overall kernel covariance matrix has a rank-Q (and equivalently Q- diagonal blocks), the proposed formulation in the (5.2), at its minima achieves the desired solution.

Proof. The proof can be found in Appendix C. □

5.1.1 Non-Convexity

The objective here is to minimize the cost function $F(\mathbf{M}, \mathbf{N}, \boldsymbol{\alpha})$, (5.2), w.r.t to $\mathbf{M}$, $\mathbf{N}$ and $\boldsymbol{\alpha}$. As is evident, the function in (5.2) is non-convex in all, $\mathbf{M}, \mathbf{N}$ and $\boldsymbol{\alpha}$. The non convexity in $\mathbf{M}$ is due to the square of the Frobenious norm of $\mathbf{M} * \mathbf{N}^T$, the term $||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2$ and the eigenvalue maximization part. Consider the non convexity with respect to $\mathbf{M} * \mathbf{N}^T$, this problem is biconvex in nature w.r.t the two variables $\mathbf{N}$ and $\mathbf{M}$. Thus, $\mathbf{N}$ and $\mathbf{M}$ can be updated alternatively while keeping the other fixed, making this part of the problem convex with respect to any one of the two variables. The other source of

non-convexity in $\mathbf{M}$ or $\mathbf{N}$ is due to the sparsity constraint, $||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2$. The $-||\mathbf{M}||_2$ function is concave in nature and thus gradient descent applied directly to such a problem will not find a stationary point. However, the $\ell_1 - \ell_2$ sparsity function can be minimized using the difference of convex algorithm as described in section 4.2.

The other set of non convexities are with respect to $\boldsymbol{\alpha}$. As explained earlier, maximizing the Q-th eigenvalue helps in selecting proper kernel matrices for clustering. Also, the maximization of the Q-th eigenvalue can be expressed as minimizing $\sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)$. The first half of this expression corresponds to the sum of $Q-1$ eigenvalues, which is a convex function in $\boldsymbol{\alpha}$, but the second half corresponding to $-\sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)$ is concave and similar to the$-||\mathbf{M}_{i,:}||_2$ part of the sparsity function, it cannot be solved with gradient descent iterations. To work around these challenges, we resort to the difference of convex functions algorithm (DCA), explained in section 4.2.

As explained in section 4.2, for an objective function of the type $\mathbf{F}(\mathbf{X}) = \mathbf{G}(\mathbf{X}) - \mathbf{H}(\mathbf{X})$ where, $\mathbf{G}(\mathbf{X})$ and $\mathbf{H}(\mathbf{X})$ are convex functions, the DCA algorithm iteratively computes an affine majorization for the concave part of the function [i.e., $-\mathbf{H}(\mathbf{X})$]. The concave parts are replaced with their majorization to form a relaxed surrogate function which can be minimized using standard gradient based methods.

Parts of the objective function in (5.2) can be expressed as difference of convex functions. Thus comparing (5.2) to the $\mathbf{G}(\mathbf{X}) - \mathbf{H}(\mathbf{X})$ and the variable $\mathbf{X}$ being replaced with $\mathbf{M}, \mathbf{N}, \boldsymbol{\alpha}$, we have that $\mathbf{H}(\mathbf{M}, \mathbf{N}, \boldsymbol{\alpha}) = \lambda||\mathbf{M}_{i,:}||_2 + \lambda||\mathbf{N}_{i,:}||_2 + \mu \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)$ and the affine majorization w.r.t $\boldsymbol{\alpha}$ and $\mathbf{M}$ gives,

$$[\tilde{\boldsymbol{\alpha}}^k]_Q \in \partial\Big(\sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)\Big), \quad \tilde{\mathbf{M}}_{i,:}^k \in \partial\big(||\mathbf{M}_{i,:}||_2\big),$$
$$\tilde{\mathbf{N}}_{i,:}^k \in \partial\big(||\mathbf{N}_{i,:}||_2\big) \tag{5.3}$$

and the relaxed objective function then can be expressed as,

$$\begin{aligned}\{\mathbf{M}^{k+1},\mathbf{N}^{k+1},\boldsymbol{\alpha}^{k+1}\} \in & \underset{\mathbf{M},\mathbf{N},\boldsymbol{\alpha}}{\operatorname{argmin}} \; || \textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j - \mathbf{M}\mathbf{N}^T||_F^2 \\ & + \mu \sum_{i=1}^{Q-1} \Lambda^i (\sum_{j=1}^{B} \boldsymbol{\alpha} \hat{\mathbf{K}}_x^j) + \textstyle\sum_{i=1}^{P} \lambda(||\mathbf{M}_{i,:}||_1 + ||\mathbf{N}_{i,:}||_1) \\ & \quad - \mu \langle \alpha, [\tilde{\boldsymbol{\alpha}}^k]_Q \rangle - \textstyle\sum_{i=1}^{P} \lambda(\langle \mathbf{M}_{i,:}, \tilde{\mathbf{M}}_{i,:}^k \rangle + \langle \mathbf{N}_{i,:}, \tilde{\mathbf{N}}_{i,:}^k \rangle). \\ & \text{s. to } \; \mathbf{M} \geq 0, \mathbf{N} \geq 0 \;\; \boldsymbol{\alpha}_j \geq 0, \quad \textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j = 1 \end{aligned} \tag{5.4}$$

The subgradient $[\tilde{\boldsymbol{\alpha}}^k]_Q$ given in (5.3) corresponds to the derivative of the sum of eigenvalues w.r.t $\boldsymbol{\alpha}$. The sub gradient can be evaluated based on the explanation in [57], where for a matrix $\mathbf{X}$ and eigenvector $\mathbf{v}_Q$ (which corresponds to the Q-th largest eigenvalue λ_Q of the matrix $\mathbf{X}$) the derivative of the eigenvalue λ_Q can be expressed as,

$$\partial \lambda_Q = \mathbf{v}_Q^T \partial \mathbf{X} v_Q. \tag{5.5}$$

Thus for $[\tilde{\boldsymbol{\alpha}}^k]_Q$ we have,

$$[\tilde{\boldsymbol{\alpha}}^k]_Q = \begin{bmatrix} \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_Q^k \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_Q^k \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_1^k + \mathbf{z}_2^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_2^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_Q^k \end{bmatrix} \tag{5.6}$$

where, $\mathbf{z}_i^k$ is the i-th eigenvector of matrix $\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j$ evaluated at $\boldsymbol{\alpha} = \boldsymbol{\alpha}^k$.

Index k is the outermost loop iteration index which counts the update of all the three parameters $\mathbf{M}, \mathbf{N}, \boldsymbol{\alpha}$. Each iteration k consists of two update loops, 1) for updating $\mathbf{M}, \boldsymbol{\alpha}$ while keeping $\mathbf{N} = \mathbf{N}^{k-1}$ constant; and 2) for updating $\mathbf{N}$ keeping $\mathbf{M}, \boldsymbol{\alpha}$ constant. The inner loops are enumerated by an index p. During iteration p, the affine majorizers based on (5.3) are evaluated. The relaxed or the surrogate problem with the majorization approximation is then solved using simple subgradient descent or interior point based methods.

The iterations for the sub gradient descent can further be represented by the index q. For the projected subgradient descent case we have the subgradients with respect to $\mathbf{M}$ and $\boldsymbol{\alpha}$ as,

$$\begin{aligned}
\mathbf{M}^{k,p,q+1} &= \Big[\mathbf{M}^{k,p,q} - c^{q+1}(-2\mathbf{N}^{k-1^T}(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k,p,q}\hat{\mathbf{K}}_x^j \\
&- \mathbf{M}^{k,p,q}\mathbf{N}^{k-1^T}) + \lambda(sgn(\mathbf{M}^{k,p,q}) - \tilde{\mathbf{M}}^{k,p}\Big]_+ \\
\boldsymbol{\alpha}_j^{k,p,q+1} &= \Big[\boldsymbol{\alpha}_j^{k,p,q} - c^{q+1}\Big((\hat{\mathbf{K}}_x^j)^T(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k,p,q}\hat{\mathbf{K}}_x^j \\
&- \mathbf{M}^{k,p,q+1}\mathbf{N}^{k-1^T}) + [\tilde{\boldsymbol{\alpha}}^{k,p,q}]_Q - [\tilde{\boldsymbol{\alpha}}^{k,p}]_{Q-1}\Big)\Big]_{S+}.
\end{aligned} \tag{5.7}$$

where, $[\tilde{\boldsymbol{\alpha}}^{k,p,q}]_Q = \partial\Big(\mu\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j)\Big)|_{\boldsymbol{\alpha}=\boldsymbol{\alpha}^{k,p,q}}$ is the derivative of the sum of Q eigenvalue functions. The operator $[\cdot]_+$ represents the component wise projection of the input vector into the positive real space R_+ and the operator $[\cdot]_{S+}$ represents the projection of the vector onto the unit simplex, and can be evaluated as described in [60]. To ensure convergence of the projected sub gradient descent method, the step size c^q should adhere to the following two conditions, 1) $\sum_{q=1}^{\infty}(c^q)^2 < \infty$, 2) $\sum_{q=1}^{\infty}c^q = \infty$, as noted in [61].

After the projected gradient descent iterations with index q have converged, $|\mathbf{M}^{k,p,q*} - \mathbf{M}^{k,p,q*-1}| \leq thresh_1$ and $|\boldsymbol{\alpha}_j^{k,p,q*} - \boldsymbol{\alpha}_j^{k,p,q*-1}| \leq thresh_1$. Similar gradient descent iterations are carried out to update $\mathbf{N}$ keeping $\mathbf{M} = \mathbf{M}^{k+1,p,q^*}$. Post convergence, all variables are updated as,

$$\mathbf{M}^{k,p+1} = \mathbf{M}^{k,p,q*}, \mathbf{N}^{k,p+1} = \mathbf{N}^{k,p,q*}, \boldsymbol{\alpha}_j^{k,p+1} = \boldsymbol{\alpha}_j^{k,p,q*} \tag{5.8}$$

and the affine majorization in (5.3) is thus iteratively re-evaluated at the $p+1$-th iteration. The overall algorithm has been summarized in Algorithm 1.

5.1.2 Implementation Details

As part of the kernelized framework we use different kernel matrices derived from the two most prominently used family of kernels, the Gaussian radial basis function (RBF) kernels and the polynomial kernels. Although it should be noted that the algorithm is in no way limited to the use of these two type of kernels. In fact the dictionary can be built from different kernel families going beyond RBF and polynomials. The entries of the Gaussian and the polynomial kernels are given as follows:

$$\begin{aligned}\mathbf{k}_x^{i,j}(n,n') &= \exp\left(-\frac{||\mathbf{x}_i(n)-\mathbf{x}_j(n')||^2}{2\sigma^2}\right),\\ \mathbf{k}_x^{i,j}(n,n') &= (\mathbf{x}_i(n)\mathbf{x}_j(n'))^d \end{aligned} \tag{5.9}$$

where $\mathbf{x}_i(n)$ and $\mathbf{x}_j(n')$ correspond to the i-th and the j-th entries of the vector $\mathbf{x}_n$ and $\mathbf{x}_{n'}$, respectively. For the RBF kernel, the variance σ^2 decides the spread upto which the magnitude of difference, $||\mathbf{x}_i(n) - \mathbf{x}_j(n')||^2$, is relevant towards kernel covariance. In the polynomial case, the degree d is the tunable factor controlling the kernel covariance outcome.

An important step while computing kernel matrices is in centering the data by accounting for the mean. While computing the covariance matrix in the linear case, $\hat{\mathbf{C}}_{\mathbf{x}}$ is evaluated with zero mean data vectors. Similarly in the non linear case, after mapping the data through the function $\phi(\cdot)$ and obtaining $\boldsymbol{\phi}(\mathbf{x}_n)$ and $\boldsymbol{\phi}(\mathbf{y}_n)$, the average quantities $\bar{\boldsymbol{\phi}}(\mathbf{x}) := {N_s}^{-1}\sum_{n=1}^{N_s}\boldsymbol{\phi}(\mathbf{x}_n)$ and $\bar{\boldsymbol{\phi}}(\mathbf{y}) := {N_s}^{-1}\sum_{n=1}^{N_s}\boldsymbol{\phi}(\mathbf{y}_n)$ are respectively subtracted out, [62]. Since the values $\boldsymbol{\phi}(\mathbf{x}_n)$ and $\boldsymbol{\phi}(\mathbf{y}_n)$ are not explicitly available, the centering is achieved as follows:

$$\hat{\mathbf{K}}_x = {N_s}^{-1}\sum_{n=1}^{N_s}[\boldsymbol{\phi}(\mathbf{x}_n) - \bar{\boldsymbol{\phi}}(\mathbf{x})][\boldsymbol{\phi}(\mathbf{x}_n) - \bar{\boldsymbol{\phi}}(\mathbf{x})]^T$$

where the (i,j)th entry of $\hat{\mathbf{K}}_x$, namely $[\hat{\mathbf{K}}_x]_{i,j}$ can be written as $[\hat{\mathbf{K}}_x]_{i,j} = {N_s}^{-1} \sum_{n=1}^{N_s} [\hat{\mathbf{K}}_x(n)]_{i,j}$ with

$$\begin{aligned}[\hat{\mathbf{K}}_x(n)]_{i,j} :&= [[\boldsymbol{\phi}(\mathbf{x}_n) - \bar{\boldsymbol{\phi}}(\mathbf{x})]_i] \cdot [[\boldsymbol{\phi}(\mathbf{x}_n) - \bar{\boldsymbol{\phi}}(\mathbf{x})]_j]^T \\ &= \mathbf{k}_x^{i,j}(n,n) - {N_s}^{-1} \textstyle\sum_{n'=1}^{N_s} \mathbf{k}_x^{i,j}(n,n') \\ &- {N_s}^{-1} \textstyle\sum_{n'=1}^{N_s} \mathbf{k}_x^{j,i}(n,n') + {N_s}^{-2} \textstyle\sum_{n'=1}^{N_s} \sum_{n''=1}^{N_s} \mathbf{k}_x^{i,j}(n',n''),\end{aligned} \tag{5.10}$$

as evident from the above equation, $\mathbf{k}_x^{i,j}(n,n') := [\boldsymbol{\phi}(\mathbf{x}_n)]_i \cdot \boldsymbol{\phi}(\mathbf{x}_{n'})]_j = \hat{\phi}(\mathbf{x}_i(n)) \cdot \hat{\phi}(\mathbf{x}_j(n'))$.

Also, since the kernel learning part of the cost function relies on maximizing the Q-th eigenvalue, it is necessary that the eigenvalues are comparable across different kernel matrices. Therefore, each of the kernels are normalized with respect to their trace. At the first iteration $\boldsymbol{\alpha}$ can be initialized giving equal weight-age to each of the kernels, thus $\boldsymbol{\alpha}_i = B^{-1}$ and $\mathbf{M}$ can be initialized randomly.

The nonzero entries in each row of $\mathbf{M}$ (and each column of $\mathbf{N}$) are then used to assign the corresponding input data vector/signal to the the corresponding cluster. Since for a given row $\mathbf{M}_{i,:}$ there may exist more than one nonzero entries, the i-th data vector/signal is assigned to the cluster $\hat{q}$ where, $\hat{q} \in \underset{q \in \{1,\ldots,Q\}}{\operatorname{argmax}} M_{i,q}$. To further utilize the information from both $\mathbf{M}$ and $\mathbf{N}$, the cluster for the i-th data vector/signal is selected as, $\hat{q} \in \underset{q \in \{1,\ldots,Q\}}{\operatorname{argmax}} \mathbf{M}_{i,q}\mathbf{N}_{q,i}$.

5.1.3 Convergence Analysis

For the non smooth optimization function in (5.2), the convergence analysis for the DCA based formulation in (5.3) and (5.4) is discussed in Appendix B. In line with the analysis in [58], we have established the following results,

Proposition 3, For the proposed DCA based method, the following three results hold true:

1) The iterates at every iteration in Algorithm 1 are nonicreasing.

2) The objective function in (5.2) is lower bounded.

Algorithm 1 DCA based approach

Initialize $\mathbf{M}$ randomly, and $\boldsymbol{\alpha}_j = \frac{1}{B} \forall j \in 1, ..., B$.

The kernel matrices, $\hat{\mathbf{K}}_x^j$ should be trace-normalized $\forall j \in 1, ..., B$.

while $(|\mathbf{M}^k - \mathbf{M}^{k-1}| > thresh_1)$ || $(|\mathbf{N}^k - \mathbf{N}^{k-1}| > thresh_1)$ **do**

  while $|\mathbf{M}^{k,p} - \mathbf{M}^{k,p-1}| > thresh_1$ **do**

    $\mathbf{N} = \mathbf{N}^{k-1,p^*}$

    Compute the majorizations for $-||\mathbf{M}_{i,:}||_2, -\sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)$ as given in (5.3) and (5.6).

    Update $\mathbf{M}$ and $\boldsymbol{\alpha}$ using interior point or projected sub-gradient descent approach (5.7) until convergence.

  end while

  while $|\mathbf{N}^{k,p} - \mathbf{N}^{k,p-1}| > thresh_1$ **do**

    $\mathbf{M} = \mathbf{M}^{k,p^*}$

    Compute the majorizations for $-||\mathbf{N}_{i,:}||_2$ as given in (5.3).

    Update $\mathbf{N}$ using interior point or projected sub-gradient descent approach (5.7) until convergence.

  end while

end while

3) At the limit point of the DCA based algorithm, $\mathbf{M}^*, \mathbf{N}^*, \boldsymbol{\alpha}^*$, the first order optimality condition is satisfied.

Proof. The proof for these three results have been discussed in appendix B.1), B.2) and B.3) respectively. □

5.2 Results

In this section numerical results are presented in with multiple datasets to show the efficacy of the proposed algorithm. First, to further build the intuition and to demonstrate

the capability of the eigenvalue based procedure in learning the desired kernel a simulation in a synthetic setting has been presented. Here we just examine the behavior of the kernel learning scheme. We then look at the performance of the overall kernel learning and clustering framework in 3 different settings, 1) In a hyperspectral image dataset, i.e. Salinas [2]; 2) In a smartphone based human activity detection setting [4] and 3) In a document classification setting [5].

We compare the performance of the proposed scheme against 4 different unsupervised algorithms; 1) standard non negative matrix factorization (NMF); 2) Graph non negative matrix factorization (GNMF) [23]; 3) Deep NMF (DNMF) [20]; and 4) K-Means. We also show the results with kernel support vector machines with 10% and 25% training data, just to give a perspective about the performance of supervised approaches under similar simulation settings. For the GNMF and Deep NMF the simulations were repeated for 200 different sets of parameters for each of the datasets. The results reported here are with the best combination of parameters for each individual dataset. The parameter values have been reported in the sections elaborating the experiments for each of the datasets. In addition, to showcase the robustness of our algorithm to changes in the dataset, for the proposed algorithm we have used the same value of parameters across all the datasets.

For the simulations of GNMF and Deep NMF the codes made available by the authors of the respective papers were utilized. For the kernel SVM, the inbuilt implementation in MATLAB with auto-scaling was used, wherein, the kernel parameters are automatically selected using the training data. For K-Means too, MATLAB's inbuilt implementation was used.

The clustering accuracy is utilized as a metric for comparing the performance of the methods. Since clusters obtained with unsupervised methods do not have an associated class label, the accuracy is calculated by assigning each cluster with the class label in accordance with the approach explained in [63,64]. Where essentially all combinations for

cluster and label pairs are explored and the one that maximizes the accuracy w.r.t ground truth is selected. Further details can be found in [63, 64]

5.2.1 Synthetic Kernel Selection

Here we consider a synthetic example to show the performance of the eigenvalue based kernel learning scheme, discussed in Section 2. We consider the case with $Q = 4$ classes. A set of 15 vectors are present from each of the 4 classes, thus the kernel matrices have a size of 60×60. The vectors from the same class are ordered/grouped together while evaluating the kernel covariance. As can be seen from Fig. 5.1(a), a total of $B = 6$ artificially generated kernels have been considered. Starting from the top left in Fig. 5.1(a) and going clock wise, the first kernel has all zero values. The next one has high covariance for elements from class 1 and 3. The next one represents high covariance in data vectors from class 2 only. The next two kernels represent unsuitable information as the diagonal matrix essentially conveys that all data vectors are independent and a constant value kernel represents that all data vectors are from the same class. These cases are similar to the ones discussed in Fig. 4.1(b) and 4.1(c), representing the low and high variance RBF kernels. The last kernel in Fig. 5.1(a) represents highly correlated vectors from class 4.

For this setting we would expect the kernel learning to have non-zero α_j values for kernel matrices 2,3 and 6 (numbered clockwise, starting from top left in Fig. 5.1(a)). From the eigenvalue based approach we attain the $\alpha = \{0, 0.5, 0.25, 0, 0, 0.25\}$. The resulting kernel matrix $\sum_{j=1}^{B} \alpha_j \hat{\mathbf{K}}_x^j$ is given in Fig. 5.1(b) which has the strong block diagonal structure that is favorable for matrix factorization based clustering.

5.2.2 Activity Detection

The first data set is the University of Milano Bicocca Smartphone-based Human Activity Recognition (UniMiB) dataset [4]. The dataset contains accelerometer readings

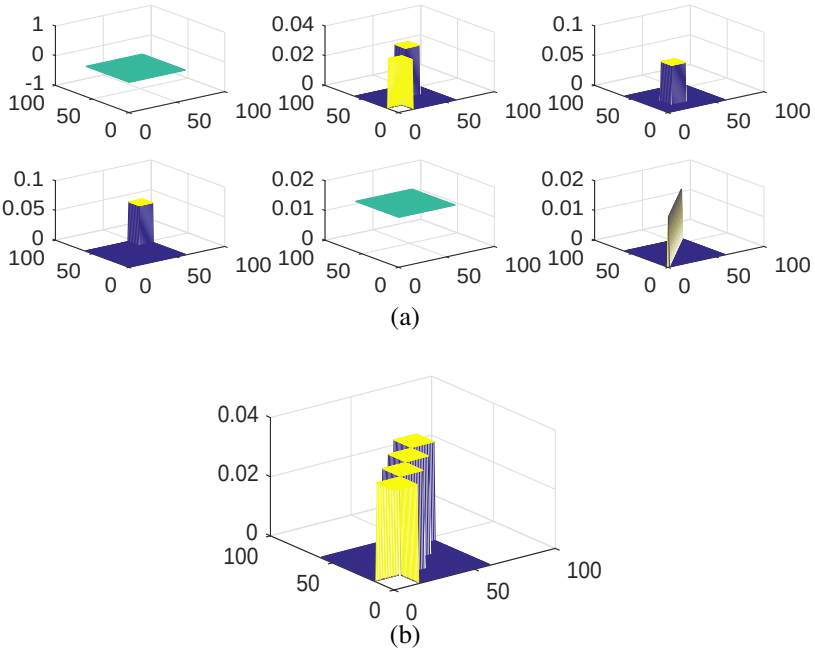


Figure 5.1. A synthetic example of the kernel learning scheme. In (a), the 6 input kernels have been showcased. In (b), the output of the kernel learning scheme obtained as a convex combination of the input kernels can be seen.

from smartphones mounted on 30 different users while they perform a range of activities including walking, running and climbing the stairs. The signals are pre-split into individual epochs with each of them being 51 samples in length and centered around the peak of the epoch. The signal is considered along each of the XYZ axis-es of the accelerometer and thus the concatenated signal is 153 samples long. Each of these 153 length vectors are associated with a single activity and thus the objective is to cluster the signals/vectors based on the underlying source or the class of activity they represent. Further information on the dataset can be found in [4, 39].

We consider the epochs from a set of 3 activities of walking, running and climbing stairs. The results are averaged over all the 30 users and are presented in Fig. 5.2 as a box plot. As can be inferred from the figure, our novel proposed framework outperforms each of the six schemes, including the supervised SVM based approach. For the Deep NMF

case, the configuration with 2 hidden layers with 50 and 4 features, respectively and for the GNMF case an $\alpha = 1.2$, yielded the best accuracy.

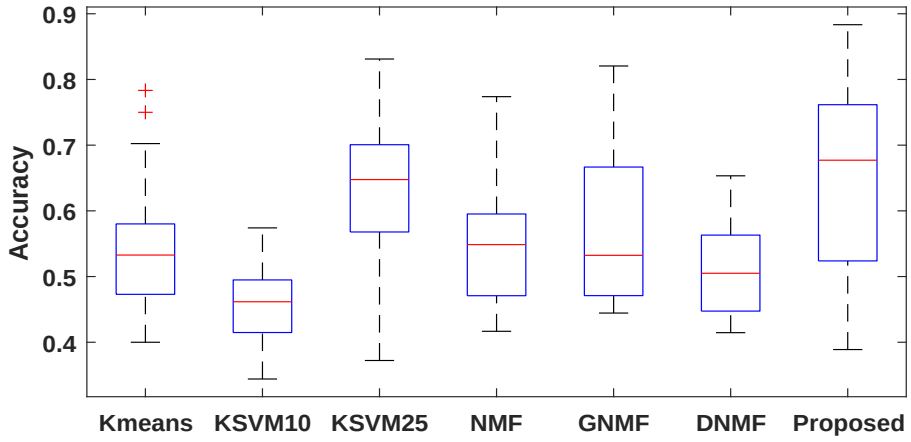


Figure 5.2. Boxplot comparing the accuracies of 5 different schemes with the proposed method for the UniMiB human activity classification dataset [4]. The central red mark in the box refers to the median accuracy, and the edges of the box mark 25^{th} and 75^{th} percentiles of the accuracy across all trials.

5.2.3 Hyperspectral Images

The hyperspectral image dataset represents the application of the proposed approach in a remote sensing setting. The image considered here has been captured by an AVIRIS sensing system over the Salinas valley, California [2]. Each pixel in the image is a 224 dimensional vector where each dimension represents the energy in a specific spectral reflectance band.

The Salinas image primarily consists of farmland where different crops/ materials are present in different parts of the image. Each pixel observes a specific material and there are a total of 16 different types of materials/crops. The objective is to consider each pixel independently and cluster them based on the 224 dimension vector into different classes

based on the material they observe. The underlying assumption is that the pixels observing the same class will have similar spectral reflectance values. For our simulations, during each iteration, we consider a set of $Q = 4$ different randomly selected materials and select 15 random pixels representing each of the 4 classes. The experiment is repeated over a total of 100 random material and pixel selections.

The overall clustering accuracies can be seen in Fig. 7.2. The figure shows a boxplot of the accuracies over the 100 trials, where the red mark inside the box indicates the median and the edges of the box mark the 25 and 75 percentiles of the accuracy across trials. As can be inferred from the figure, the GNMF and the proposed approach perform similar to each other and both perform better than the deep NMF, K-Means and SVM with 10% training. It is important to note that the accuracy for GNMF and Deep semi NMF depend significantly on the value of the tuning parameters. For both the cases we searched over 200 different sets of parameters and finally the accuracies with the best set have been presented in Fig. 7.2. In the GNMF case a value of $\alpha = 1$ was used, whereas a value of $\alpha = 100$ has been suggested in [23] at which the GNMF accuracy was significantly lower.

5.2.4 Document Clustering

The third dataset is the la2 dataset which was compiled using from articles of the Los Angeles Times and was used in TREC [5]. The dataset contains 3075 files from 6 classes, we consider files with at least a total of 100 words and thus the total number of files considered reduces to 2030. For our evaluation we consider a set of 100 files from 4 different classes (25 randomly selected from each of the classes). Similar to the other case studies, the objective here is to cluster the documents pertaining to the same class. A document is represented by a vector where its dimensions represent the frequency of occurrence of a particular word. The document to vectorization is done using the approach in [65].

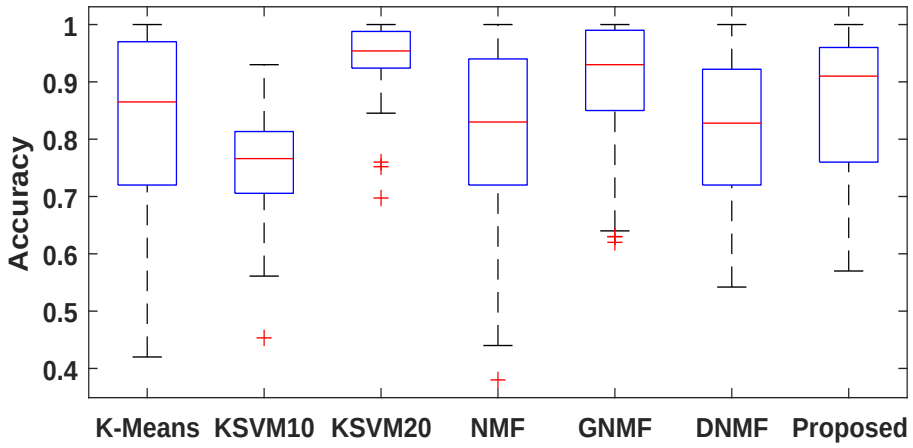


Figure 5.3. Boxplot comparing the accuracies of 5 different schemes with the proposed method for the Salinas hyperspectral image dataset [2]. The central red mark in the box refers to the median accuracy, and the edges of the box mark 25^{th} and 75^{th} percentiles of the accuracy across all trials..

The boxplot giving the clustering accuracies for the document dataset is in Fig. 5.4. At convergence, the α values had significant weightage for kernel matrices corresponding to RBF kernels with $\sigma^2 = 10^7, 10^6, 10^5$ and the polynomial kernel with degree 2. For the Deep NMF case, the configuration with 2 hidden layers with 200 and 4 features, respectively and for the GNMF case an $\alpha = 0.05$, yielded the best accuracy. For the proposed scheme $\lambda = 10^{-2}$ and $\mu = 10$ were used.

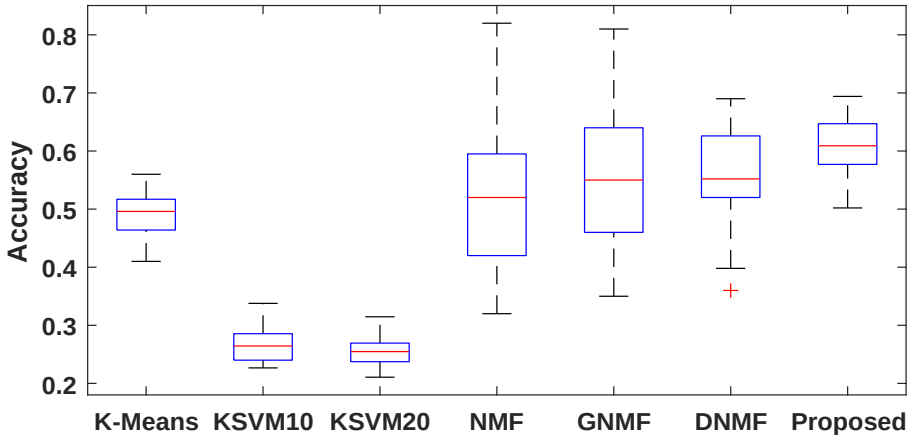


Figure 5.4. Boxplot comparing the accuracies of 5 different schemes with the proposed method for the LA Times document clustering dataset [5]. The central red mark in the box refers to the median accuracy, and the edges of the box mark 25^{th} and 75^{th} percentiles of the accuracy across all trials.

CHAPTER 6

UNSUPERVISED DEEP NEURAL NETWORKS FOR CORRELATION EXTRACTION

So far we have relied on using different kernels (like RBF/Gaussian kernels, polynomial kernels) for computing the data correlations in a mapped feature space. In this chapter we explore a deep neural network based approach to map the data from the data space to the feature space. The method relies on the principle of block diagonalization of the correlation matrix at the ouput of a deep neural network based mapping. The neural network is learnt in an unsupervised setting using the eigenvalue maximization framework detailed in Chapter 4 and is utilized towards improving the performance of the NMF based clustering method.

As was discussed in the previous chapters, the kernel correlation/covariance matrix most suitable for sparse factorization should ideally have a large magnitude for the Q-th eigenvalue. Depending on the data and application, the RBF or polynomial kernels may not be sufficient to linearize the source to sensor observation model. Thus instead of restricting the search of the $\phi(\cdot)$ to just these family of functions which have a closed form expression for evaluating the correlations, we may build a more suitable mapping $\phi(\cdot)$ by utilizing the neural network like function estimators that could satisfy our eigenvalue criterion.

Consider a fully connected neural network as seen in Fig. 6.1. The objective is to utilize the network to transform the data from the input space x to an output space $\phi(x)$ such that the vectors from the same class are correlated and the ones from differing classes are uncorrelated. In line with the earlier discussions, where this objective was achieved via multiple kernel learning by maximizing the Q-th eigen value of the convex sum of kernel

correlation matrices, the neural network (NN) is trained such that the Q-th eigen value of the correlation matrix of the output layer of the NN is maximized.

To formalize the discussion, consider a fully connected network with $H + 2$ layers, one input layer with N_s input nodes (where N_s is the dimension of the input data vector), H hidden layers with the h-th layer having N^h nodes and 1 output layer with N^o output nodes. Since we consider a fully connected model, each node in the input or hidden layer is connected to each node in the next corresponding layer, where each connection corresponds to a weight. The weights of the connections between H-th hidden layer and the output layer are given by the $(N^H + 1) \times N^o$ matrix W^o, where the 1 in $(N^H + 1)$ corresponds to the weight for the bias term. Similarly, the weights between the two consecutive hidden layers $h1$ and $h2$ are given by the $(N^{h1} + 1) \times N^{h2}$ matrix W^{h2}. The output of each node of the hidden and the output layer is the weighted sum of the outputs of the previous layer passed through an activation function. As part of this work we utilize ReLU (rectified linear unit) [66] function as the activation function for our formulation, but it must be noted that the same formulation can be easily extended to different activation functions too. The ReLU function, $\sigma(\cdot)$, can be defined as $\sigma(z) = max\{0, z\}$.

Consider a simple case wherein the data mapping $x \rightarrow \phi(x)$ involves a two layer NN (1 input layer and 1 output layer). Since we consider a fully connected model, each node in the output layer is connected to all the input nodes. Thus, the output of the j-th node of the output layer $\phi_j(x) = \sigma(\mathbf{w}_j^{oT}\mathbf{x})$, where, $\mathbf{w}_j^o$ is the weight vector connecting the j-th node of the output with all the nodes of the preceding layer (input layer in this case).

The correlation matrix entries for the data in the mapped space are then given as,

$$\mathbf{K}_{i_1,i_2} = \sum_{j=1}^{N_o} \phi_j(\mathbf{x}_{i_1})\phi_j(\mathbf{x}_{i_2}) \tag{6.1}$$

where, the matrix $\mathbf{K} \in \mathbb{R}^{P \times P}$ is the correlation matrix across the P data vectors after being mapped through the neural network. The subscript i_1, i_2 represents the corresponding entry of the associated matrix.

The weights $\mathbf{w}_j^o$ are then to be updated in accordance with the objective of maximizing the Q-th eigenvalue of $\mathbf{K}$. In the following section we elaborate on the weight update procedure using a difference of convex formulation. We first develop the theory for the simpler 2 layer case and then later extend it to the multi-layer case. We further discuss the algorithmic and implementation details for achieving the data clustering objective using the sparse matrix factorization of the output correlation matrix $\mathbf{K}$.

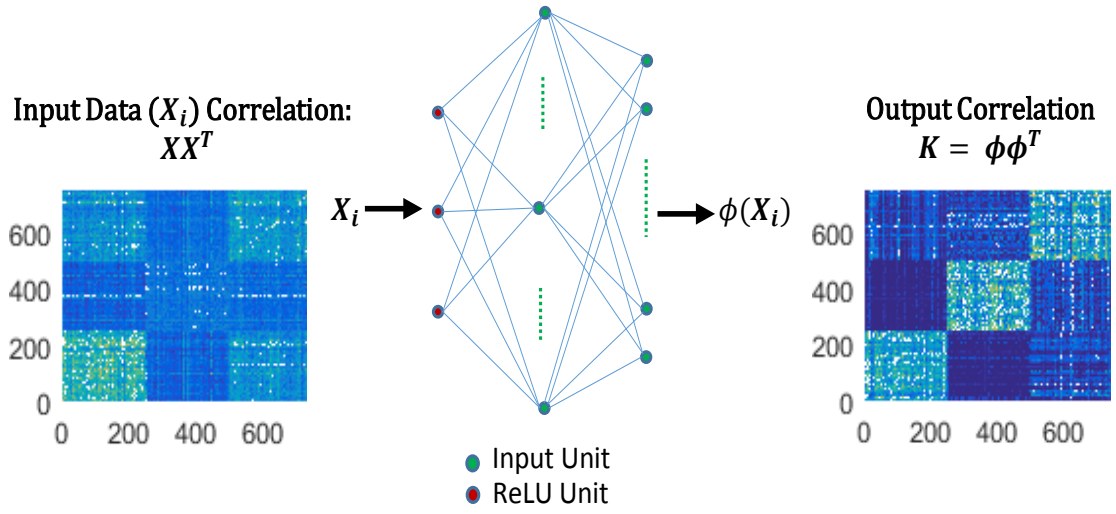


Figure 6.1. Block diagram representation of the unsupervised training method to learn a function mapping that maximizes the Q-th eigenvalue.

6.1 Algorithm

The difference of convex formulation introduced in section 4.2 for the multiple kernel learning application can be further extended towards finding the weights of the neural

network based model described here. For the two layer case, the correlation matrix of the deep kernel can be formulated as:

$$\begin{aligned}\mathbf{K}_{i_1,i_2} =& \sum_{j=1}^{N_o} \phi_j(\mathbf{x}_{i_1})\phi_j(\mathbf{x}_{i_2}) \\ =& \frac{1}{N_o}\sum_{j=1}^{N_o} \sigma(\mathbf{w}_j^{oT}\mathbf{x}_{i_1})\sigma(\mathbf{w}_j^{oT}\mathbf{x}_{i_2}) \\ =& \frac{1}{N_o}\sum_{j=1}^{N_o} \sigma(\mathbf{w}_j^{oT}\mathbf{x}_{i_1})\sigma(\mathbf{w}_j^{oT}\mathbf{x}_{i_2}) \end{aligned} \tag{6.2}$$

where, $\mathbf{w}_j^o$ represents the weights between the input layer and the j-th node of the output layer and σ is the activation function, as defined earlier. Since we consider a ReLU activation function which essentially is a piece-wise linear function with the slope taking the possible value of 1 or 0, the kernel matrix can be represented in a simplified form using element wise dot product as given in,

$$\begin{aligned}\mathbf{K}_{i_1,i_2} =& \frac{1}{N_o}\sum_{j=1}^{N_o} \Big(J_{i_1 j} \cdot (\mathbf{w}_j^{oT}\mathbf{x}_{i_1})\Big)\Big(J_{i_2 j} \cdot (\mathbf{w}_j^{oT}\mathbf{x}_{i_2})\Big) \\ where, \;\; J_{i_1 j} =& \begin{cases} 0 \;\; if \;\; (\mathbf{w}^{oT}_j \mathbf{x}_{i_1}) \leq 0 \\ 1 \;\; if \;\; (\mathbf{w}^{oT}_j \mathbf{x}_{i_1}) > 0 \end{cases} \end{aligned} \tag{6.3}$$

The overall kernel matrix can then be defined as,

$$\mathbf{K} = \frac{1}{N_o}\Big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\Big)\Big(\mathbf{J} \circ (\mathbf{X}\mathbf{W}^o)\Big) \tag{6.4}$$

where, the operator $\circ$ represents element wise dot product. The matrices $\mathbf{W}^o$ and $\mathbf{X}$ represents the weights and the inputs, respectively. $\mathbf{J}$ is the matrix with its entries being either 0 or 1, as given by (6.3).

The optimization problem to learn weights, $\mathbf{W}^o$, to maximize the Q-th eigenvalue can then be represented as,

$$\begin{aligned}\mathbf{W}^o \in & \underset{\mathbf{W}^o}{\text{argmax}} \ \Lambda^Q(\mathbf{K}) \\ \in & \underset{\mathbf{W}^o}{\text{argmin}} \sum_{i=1}^{Q-1} \Lambda^i\Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big(\mathbf{J} \circ (\mathbf{X}\mathbf{W}^o)\big)\Big) \\ & - \sum_{i=1}^{Q} \Lambda^i\Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)\end{aligned} \tag{6.5}$$

Since the optimization problem is not convex but rather a difference of two convex functions (or equivalently, sum of a convex and a concave function). An iterative process with two alternating steps, similar to the one explained in Section 4.2 is utilized. At each iteration, the first step involves relaxing the the concave part of the optimization problem, given by, $\mathbf{H}(\mathbf{W}^o) = -\sum_{i=1}^{Q} \Lambda^i\Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)$. The concave function is replaced it by its linear majorizer, $\mathbf{W}^{oT}\frac{\partial \mathbf{H}(\mathbf{W}^o)}{\partial \mathbf{W}^o}$, where the operator ∂ represents the subgradient of the function. Although the sum of eigenvalue function is convex, it is not smooth across its range, thus subgradients are utilized for forming the majorizer. In line with the discussions in [54, 57], for a matrix $\mathbf{X}$ with eigenvectors $\mathbf{v}_Q$, corresponding to the Q-th largest eigenvalue λ_Q, the sub gradient of the eigenvalue λ_Q is given as,

$$\partial\lambda_Q = \mathbf{v}_Q^T \partial\mathbf{X} v_Q. \tag{6.6}$$

Correspondingly, for the matrix $\Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)$, the subgradient $\frac{\partial \mathbf{H}(\mathbf{W}^o)}{\partial \mathbf{W}^o}$ is given as,

$$\begin{aligned}\frac{\partial \mathbf{H}(\mathbf{W}^o)}{\partial \mathbf{W^o}_{ij}} &= -\sum_{i=1}^{Q} \frac{\partial}{\partial \mathbf{W^o}_{ij}} \mathbf{z}_i^T \Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)\mathbf{z}_i \\ &= -\sum_{i=1}^{Q} \mathbf{z}_i^T \Big(\big(\mathbf{J} \circ \frac{\partial}{\partial \mathbf{W^o}_{ij}}(\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big) \\ &\quad + \big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big(\frac{\partial}{\partial \mathbf{W^o}_{ij}}(\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)\mathbf{z}_i \\ &= -2\sum_{i=1}^{Q} \mathbf{z}_i^T \Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\hat{\mathbf{I}}_{i,j}) \circ \mathbf{J}\big)\Big)\mathbf{z}_i \end{aligned} \tag{6.7}$$

where, $\mathbf{z}_i$ is the eigenvector corresponding to the i-th largest eigenvalue of the matrix $\Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^o) \circ \mathbf{J}\big)\Big)$. Also, $\frac{\partial\, \mathbf{X}\mathbf{W}^o}{\partial \mathbf{W^o}_{ij}} = \mathbf{X}\hat{\mathbf{I}}_{i,j}$ where, $\hat{\mathbf{I}}_{i,j}$ is a matrix with its i,j-th element having a value of 1 and the rest of the elements being 0. Further explanation can be found in [67].

Thus, at the k-th iteration, the overall optimization problem with the majorization relaxation is given as,

$$\begin{aligned}\mathbf{W}^{ok} \in \underset{\mathbf{W}^o}{\text{argmin}} &\sum_{i=1}^{Q-1} \Lambda^i \Big(\big(\mathbf{J} \circ (\mathbf{W}^{oT}\mathbf{X}^T)\big)\Big)\Big(\mathbf{J} \circ (\mathbf{X}\mathbf{W}^o)\big)\Big) \\ &- \Big\langle \mathbf{W}^o, \frac{\partial \mathbf{H}(\mathbf{W}^o)}{\partial \mathbf{W^o}_{ij}}|_{\mathbf{W}^o=\mathbf{W}^{ok-1}} \Big\rangle \end{aligned} \tag{6.8}$$

where, the majorizer has been evaluated at $\mathbf{W}^{ok-1}$ (the value of $\mathbf{W}^o$ at the previous iteration).

At the second step, a sub-gradient descent based scheme is utilized to solve the relaxed problem to a stationary point. Since the sub-gradient descent is an iterative process, we introduce a second index l to indicate the iterations. Thus the weight update process can be expressed as,

$$\begin{aligned}\mathbf{W}^{ok,l+1}_{ij} =& \mathbf{W}^{ok,l}_{ij} - c\Bigg(\sum_{i=1}^{Q-1} 2\mathbf{z}_i^T \Big(\big(\mathbf{J} \circ (\mathbf{W}^{ok,l^T}\mathbf{X}^T)\big) \\ &\big((\mathbf{X}\hat{\mathbf{I}}_{i,j}) \circ \mathbf{J}\big)\Big)\mathbf{z}_i \; - \frac{\partial \mathbf{H}(\mathbf{W}^o)}{\partial \mathbf{W^o}_{ij}}|_{\mathbf{W}^o=\mathbf{W}^{ok-1,0}} \Bigg) \end{aligned} \tag{6.9}$$

where, the eigenvectors $\mathbf{z}_i$ are evaluated at for the the matrix $\Big(\big(\mathbf{J}\circ(\mathbf{W}^{ok,l^T}\mathbf{X}^T)\big)\big((\mathbf{X}\mathbf{W}^{ok,l})\circ \mathbf{J}\big)\Big)$. The inner loop for sub gradient descent (corresponding to the iteration index l) is considered to have converged when $|\mathbf{W}^{ok,l} - \mathbf{W}^{ok,l+1}|_F \leq \epsilon$, where ϵ is a user defined threshold. Towards the next iterate, the majorizer is recalculated at the beginning of $k+1$-th iteration and the relaxed problem is solved to optimality. This process is carried until the stopping criterion given by $|\mathbf{W}^{ok,0} - \mathbf{W}^{ok+1,0}|_F \leq \epsilon$ is met.

6.1.1 Multilayer Case

In the multilayer case with H hidden layers, the gradients have to be propagated from the output node to the input node, through the hidden layers. Thus the output of at the j-th node of the output layer is given as $\sigma(\mathbf{w}_j^{oT}\mathbf{x}_{i1}^{h_H})$, where $\mathbf{w}^o$ indicates the weights of the output layer and $\mathbf{x}_{i1}^{h_H}$ indicates the output of the H-th hidden layer for the input vector $\mathbf{x}_{i1}$.

Since the output of the hidden layer, $\mathbf{x}_{i1}^{h_H}$ is a function of the weights of the remaining $H-1$ hidden-layers, the output of the j-th node of the H-th hidden layer is given as,

$$\mathbf{x}_{i1}^{h_H}{}_j = \sigma(\mathbf{w}_j^{h_H{}^T}\mathbf{x}_{i1}^{h_{H-1}}). \tag{6.10}$$

Thus, the cost function in (6.5) can be expressed as,

$$\begin{aligned}\{\mathbf{W}^o, \mathbf{W}^{h_H}, ..., \mathbf{W}^{h_1}\} \in \\ \underset{\mathbf{W}^o,\mathbf{W}^{h_H},...,\mathbf{W}^{h_1}}{\operatorname{argmin}} & \sum_{i=1}^{Q-1}\Lambda^i\Big(\big(\mathbf{J}\circ(\mathbf{W}^{oT}\mathbf{X}^{h_H{}^T})\big)\big((\mathbf{X}^{h_H}\mathbf{W}^o)\circ\mathbf{J}\big)\Big) \\ & - \sum_{i=1}^{Q}\Lambda^i\Big(\big(\mathbf{J}\circ(\mathbf{W}^{oT}\mathbf{X}^{h_H{}^T})\big)\big((\mathbf{X}^{h_H}\mathbf{W}^o)\circ\mathbf{J}\big)\Big).\end{aligned} \tag{6.11}$$

The derivative of the cost function in (6.11) (say $\mathbf{C}$), w.r.t $\mathbf{W}^{h_H}$ is then given as,

$$\frac{\partial \mathbf{C}}{\partial \mathbf{W}_{mn}^{h_H}} = \sum_j \frac{\partial \mathbf{C}}{\partial \mathbf{X}_j^{h_H}}\frac{\partial \mathbf{X}_j^{h_H}}{\partial \mathbf{W}_{mn}^{h_H}} \tag{6.12}$$

where, $\mathbf{C}$ is the cost function in (6.11). The derivative $\frac{\partial \mathbf{C}}{\partial \mathbf{X}_j^{h_H}}$ can be evaluated using the majorization approach used in (6.7) and (6.8).

6.1.2 Reducing Dimensionality

The complexity of the gradient based update procedure in (6.7) and (6.8) is dependent on the dimensionality of the correlation matrix $\mathbf{K}$ (in (6.4)), where the matrix has a size of $P \times P$ which depends on the number of data vectors being considered (P). Since the update procedure in (6.7) and (6.8) requires evaluating the eigenvectors $\mathbf{z}^Q$ the complexity of the update procedure is a cubic function of P. Thus with the increase in number of vectors being used for evaluating $\phi(\cdot)$, the computation costs can be very high.

Since we are interested only in maximizing the Q-th eigenvalue of K, the correlations can rather be evaluated across dimensions, thus the matrix can be represented as,

$$\hat{\mathbf{K}} = \frac{1}{N_o}\Big(J \circ (\mathbf{X}\mathbf{W})\Big)\Big(J \circ (\mathbf{W}^T\mathbf{X}^T)\Big). \tag{6.13}$$

The eigenvalues of $\hat{\mathbf{K}}$ are same as the eigenvalues of $\mathbf{K}$ and thus the eigenvalue maximization can be performed cost effectively over a matrix of fixed size, $N_o \times N_o$, irrespective of the number of data vectors being considered. Thus if $P > N_o$, $\mathbf{K}$ can be replaced with $\hat{\mathbf{K}}$, and the derivative in (6.7) is modified as,

$$\frac{\partial \mathbf{H}(\mathbf{W})}{\partial \mathbf{W}_{ij}} = -2\sum_{i=1}^{Q} \hat{\mathbf{z}}_i^T\Big(\big((\mathbf{X}\hat{\mathbf{I}}_{i,j}) \circ \mathbf{J}\big)\big(J \circ (\mathbf{W}^T\mathbf{X}^T)\big)\Big)\hat{\mathbf{z}}_i \tag{6.14}$$

where, $\hat{\mathbf{z}}_Q$ represents the eigenvectors of the matrix $\hat{\mathbf{K}}$.

6.1.3 Sparse Matrix Factorization

Similar to the matrix factorization method proposed in the previous chapter, we utilize the $\ell_1 - \ell_2$ penalty to recover sparse matrix factors. The sparsity regularized matrix factorization problem can then be rewritten as,

$$||\mathbf{C} - \mathbf{M}\mathbf{N}^T||_F^2 + \lambda\big(\sum_{i=1}^{P} ||\mathbf{M}_{i,:}||_1 - ||\mathbf{M}_{i,:}||_2 + ||\mathbf{N}_{i,:}||_1 - ||\mathbf{N}_{i,:}||_2\big)$$
$$\text{s. to} \qquad \mathbf{M} \geq 0, \mathbf{N} \geq 0 \tag{6.15}$$

where, $||\cdot||_1$ and $||\cdot||_2$ represent the ℓ_1 and ℓ_2 norms of a vector and λ is the regularization parameter controlling the sparsity. $\mathbf{M}_{i,:}$ and $\mathbf{N}_{i,:}$, refer to the i-th row of $\mathbf{N}$ and $\mathbf{M}$ matrices respectively. $\mathbf{M} \geq 0, \mathbf{N} \geq 0$ are the entry-wise non-negativity constraints on $\mathbf{M}$ and $\mathbf{N}$.

Similar to the discussions in Chapter 5, parts of the objective function in (6.15) can be expressed as difference of convex functions. Thus comparing (6.15) to the $\mathbf{G}(\mathbf{X}) - \mathbf{H}(\mathbf{X})$ and the variable $\mathbf{X}$ being replaced with $\mathbf{M}, \mathbf{N}$, we have that $\mathbf{H}(\mathbf{M}, \mathbf{N}) = \lambda||\mathbf{M}_{i,:}||_2 + \lambda||\mathbf{N}_{i,:}||_2$ and the affine majorization w.r.t$\mathbf{M}$ gives,

$$\tilde{\mathbf{M}}_{i,:}^k \in \partial\big(\lambda||\mathbf{M}_{i,:}||_2\big),$$
$$\tilde{\mathbf{N}}_{i,:}^k \in \partial\big(\lambda||\mathbf{N}_{i,:}||_2\big) \tag{6.16}$$

After applying the DCA based majorization, the relaxed objective function then can be expressed as,

$$\{\mathbf{M}^{k+1}, \mathbf{N}^{k+1}\} \in \underset{\mathbf{M},\mathbf{N}}{\text{argmin}} ||\mathbf{K}_x - \mathbf{M}\mathbf{N}^T||_F^2 + \textstyle\sum_{i=1}^{P} \lambda(||\mathbf{M}_{i,:}||_1 + ||\mathbf{N}_{i,:}||_1) - \sum_{i=1}^{P} \lambda(\langle \mathbf{M}_{i,:}, \tilde{\mathbf{M}}_{i,:}^k \rangle + \langle \mathbf{N}_{i,:}, \tilde{\mathbf{N}}_{i,:}^k \rangle).$$
$$\text{s. to } \mathbf{M} \geq 0, \mathbf{N} \geq 0 \tag{6.17}$$

where,

$$\tilde{\mathbf{M}}_{i,:}^k \in \partial\big(\lambda||\mathbf{M}_{i,:}||_2\big), \;\; \tilde{\mathbf{N}}_{i,:}^k \in \partial\big(\lambda||\mathbf{N}_{i,:}||_2\big) \tag{6.18}$$

The iterative gradient based approach used in the previous chapter can be utilized to find the sparse matrix factors $\mathbf{M}^{k+1}$ and $\mathbf{N}^{k+1}$.

6.2 Results

In this section we present results showing the performance of the proposed aprroach (abbreviated as EVDL NMF) in the hand written digit image database MNIST [6].

We compare the performance of the proposed scheme against two other schemes. 1)the graph non negative matrix factorization (Graph NMF) [23] and 2) kernel learning based NMF (KL NMF).

6.2.1 MNIST Digits Dataset

The MNIST dataset represents the application of the proposed scheme for handwritten digit recognition. The dataset contains images from 10 classes representing 0-9 digits. Each image is gray-scale and has a size of 28x28 (or vectorized as a 784 dimensional vector). The objective here is to cluster images based on the digit each image represents. For our simulations we consider the digits between 0 and 3, and thus $Q = 4$. The experiment is repeated over a total of 100 times, during each iteration, a set of 250 randomly selected images representing each of the 4 classes are utilized for the unsupervised training of the network and 25 images from each class are utilized for the NMF based clustering. A box plot representing the overall clustering accuracies can be seen in Fig. 6.2.

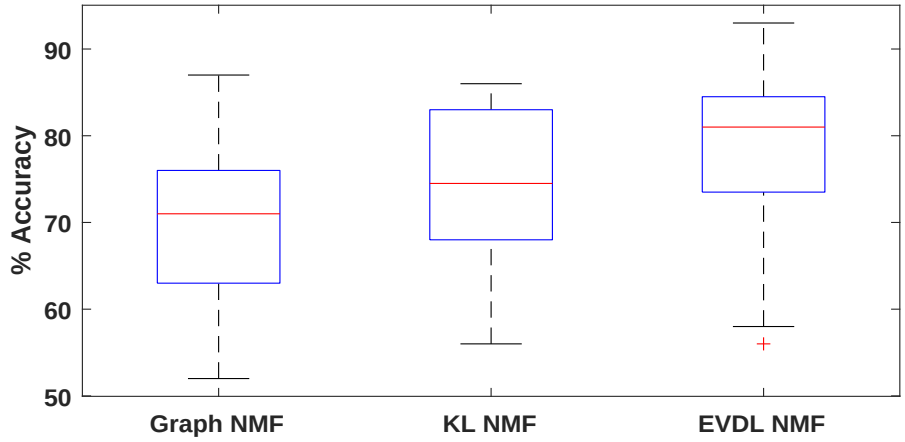


Figure 6.2. Boxplot comparing the accuracies of the proposed eigenvalue based deep learning method with other NMF methods for the MNIST digit image database [6].

CHAPTER 7

DEEP LINEARIZATION MAPPINGS FOR CANONICAL CORRELATION CLUSTERING

In Chapter 2, we explored the CCA based clustering framework and presented its kernelized variant which improved the clustering accuracies in contrast to the standard CCA formulation. The use of kernels improves the performance of CCA based approaches on a case by case basis but provides no guarantees about potentially being able to linearize data or improve performance. Also, utilizing kernels heavily restricts the family of mappings that may be used to only Gaussian and polynomial. Another drawback of kernel methods is with regards to scalability as they scale quadratically with P as compared to a linear CCA approach. This is mainly due to the usage of auto and cross kernel covariance matrices in kernelized CCA, instead of directly using the data as in the standard CCA model in (2.2).

As we know, a linear observation model $f_j^i(\cdot)$ corresponds to a linear relationship between the signal $x_i(n)$ and $s^j(n)$. Since each sensor's observations are derived from one of the Q source signals, the correlation matrix $\mathbf{R} = \mathbf{X}\mathbf{X}^T$ has a rank equal to Q (ignoring the effect of noise). Thus, performing a principal component analysis (PCA) based optimization of the reconstruction MSE, $\sum_{i=1}^{P} \|\mathbf{X}_i - \mathbf{B}\mathbf{C}\mathbf{X}_i\|_2$, the compression matrix $\mathbf{C}$ and the decompression matrix $\mathbf{B}$ need only Q rows and Q columns, respectively. Here, $\mathbf{X}_i \in \mathbb{R}^{N_s}$ represents the i-th row of $\mathbf{X}$

In the case where $f_j^i(\cdot)$ is non-linear, rank$(\mathbf{R}) > Q$ and equivalently there are more than Q principal components in the data. To improve the clustering performance in such applications, a possible solution is to map the signals/data through a non-linear mapping, say $\boldsymbol{\phi}(\cdot)$ (where, $\boldsymbol{\phi} : \mathbb{R}^{N_s} \rightarrow \mathbb{R}^F$), such that the signals/data vectors have linear relationships

in the mapped feature space (Cover's Theorem [68]), here F denotes the dimensionality of the feature space. Thus, ensuring that in the mapped space, the data pertaining to the same class are highly correlated.

Since our underlying assumption states that sensor data $\mathbf{X}$ is dependent on the Q uncorrelated source signals, then the non-linear mapping $\boldsymbol{\phi}(\cdot)$, should linearize the data such that the low-rank reconstruction MSE is minimized, i.e.,

$$\underset{\boldsymbol{\phi}(\cdot),\mathbf{B}\in\mathbb{R}^{F\times Q},\mathbf{C}\in\mathbb{R}^{Q\times F}}{\operatorname{argmin}} \sum_{i=1}^{P} \|\boldsymbol{\phi}(\mathbf{X}_i) - \mathbf{BC}\boldsymbol{\phi}(\mathbf{X}_i)\|_2^2. \tag{7.1}$$

Thus, instead of restricting the search of the $\boldsymbol{\phi}(\cdot)$ to just these family of functions which have a closed form expression for evaluating the correlations, the potential of mapping $\boldsymbol{\phi}(\cdot)$ can be expanded by utilizing neural network based function estimators that could satisfy our rank criterion.

7.1 Deep Linearization Mapping

Since the objective is to linearize the data, we consider a neural network with non-linear activations to represent the mapping $\boldsymbol{\phi}(\cdot)$. The objective is to utilize the network to transform the data from the input space $\mathbf{X}_i \in \mathbb{R}^{N_s}$ to an output space $\boldsymbol{\phi}(X_i) \in \mathbb{R}^F$ such that the rank of the correlation matrix in the projected space is equal to Q.

While solving for (7.1), it is important to ensure that the mapping $\boldsymbol{\phi}(\cdot)$ is invertible and preserves the original information of its argument. For example, if (7.1) is solved unconstrained, a possible mapping that minimizes the cost is $\boldsymbol{\phi}(\mathbf{X}_i) = [0, 0, ..., 0]\ \forall i$. To avoid this, we have to impose additional constraints. Specifically we substitute (7.1) with the following constrained formulation:

$$\begin{aligned} &\underset{\boldsymbol{\phi}(\cdot)}{\operatorname{argmin}} \sum_{i=1}^{P} \|\boldsymbol{\phi}(\mathbf{X}_i) - \mathbf{BC}\boldsymbol{\phi}(\mathbf{X}_i)\|_2 \\ &\text{s. to} \quad \boldsymbol{\phi}^{-1}(\boldsymbol{\phi}(\mathbf{X}_i)) = \mathbf{X}_i, \ \ \forall i \in 1, ..., P. \end{aligned} \tag{7.2}$$

Solving (7.2) is extremely hard due to the highly nonconvex equality constraint. Therefore, the following relaxed reformulation which preserves the original information of the input is adopted instead

$$\underset{\boldsymbol{\phi}(\cdot)}{\operatorname{argmin}} \sum_{i=1}^{P} \left\| \mathbf{X}_i - \boldsymbol{\phi}^{-1}\Big(\mathbf{BC}\boldsymbol{\phi}(\mathbf{X}_i)\Big) \right\|_2 . \tag{7.3}$$

The reformulated unconstrained optimization problem in (7.3) keenly resembels the autoencoder neural network formulation [69] with an added linear transformation $\mathbf{BC}$ to impose linear compression to a Q-dimensional space (with linear PCA applied in feature space).

To formalize the discussion, consider an autoencoder type structure with a fully connected network as seen in Fig. 7.1. The proposed structure can be divided into 3 parts. 1) The nonlinear mapping (encoding) stage, to transform the data as $\mathbf{X}_i \rightarrow \boldsymbol{\phi}(\mathbf{X}_i)$ from $\mathbb{R}^{N_s} \rightarrow \mathbb{R}^{F}$, 2) the linear compression-decompression stage or the PCA stage, where the data is compressed to Q components and decompressed to F components, and 3) the nonlinear decoding or inverse mapping stage that remaps the data from $\mathbb{R}^{F} \rightarrow \mathbb{R}^{N_s}$.

At the first and third stage, any type of neural network layers may be utilized (eg., fully connected, convolutional neural networks and so on [70]) and there is no inherent restriction on the number or structure of these layers. As is common in autoencoder architectures, the encoding and decoding layers are usually selected as a mirror image of each other (i.e. same architecture is utilized in the decoding as in encoding but the layers are arranged in the reverse order). The second stage corresponding to the linear compression-decompression (standard linear PCA) forms a linear bottleneck in the neural network architecture. As, $\mathbf{C}$ is a linear projection matrix of size $Q \times F$, the data is essentially compressed to Q principal components. Similarly, $\mathbf{B} \in \mathbb{R}^{F \times Q}$ re-projects the data to the mapping space. From an implementation perspective, $\mathbf{B}$ and $\mathbf{C}$ can be represented as weights between fully connected layers with linear activation functions at nodes, as is seen in Fig. 7.1.

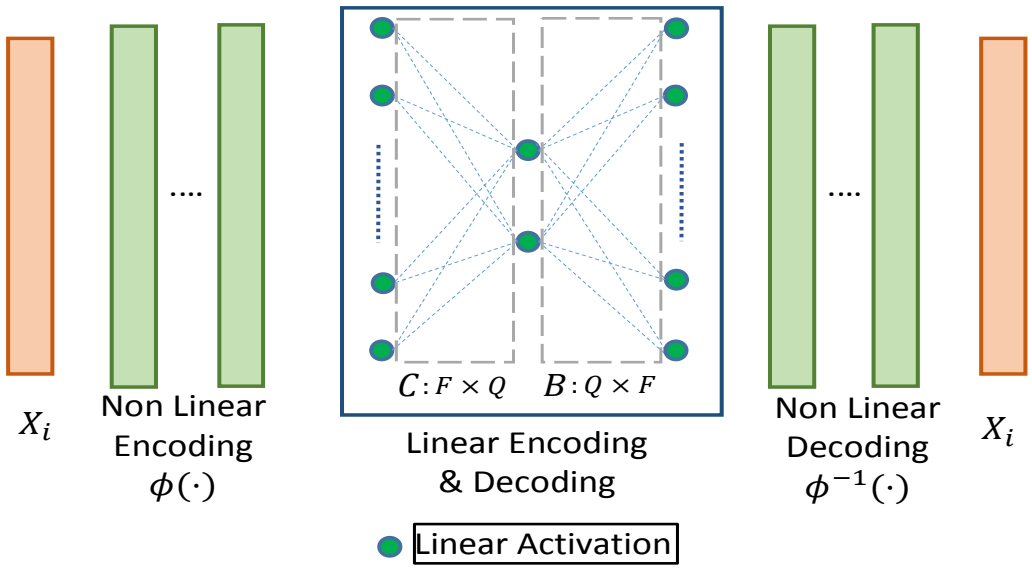


Figure 7.1. Block diagram representation of the unsupervised training method to learn a function mapping that compresses the data vector down to Q principal components.

A gradient descent approach in (7.3) is utilized to learn the weights of each layer, namely $\phi(\cdot), \phi^{-1}(\cdot)$ and $\mathbf{B}$ and $\mathbf{C}$ matrices. An important thing to remember here is that the optimization process mentioned here to learn the mapping $\phi(\cdot)$ is unsupervised and does not require any training labels. Post convergenence, the output of the nonlinear mapping $\phi(\mathbf{X}_i)$ is utilized as the input for the CCA based clustering process discussed in Section 2.3 and [11].

7.2 Results

In this section we present results showing the performance of the proposed deep linearization based CCA (abbreviated as Deep CCA) in two different settings, 1) In a hyperspectral image dataset, i.e. Salinas [2]; 2) In the hand written digit image database MNIST [6].

We compare the performance of the proposed scheme against two other schemes. 1) The standard CCA [11] and 2) the kernelized CCA (KCCA) [21, 22]. For the kernel CCA,

the best kernel was selected from a group of 15 Gaussian and polynomial kernels using the alignment based kernel selection scheme suggested in [27].

7.2.1 Hyperspectral Images

In this dataset we apply the proposed approach for clustering in a remote sensing setting. The hyperspectral image has been captured by the AVIRIS sensing system over the crop fields in Salinas valley, California [2], where different crops/ materials are present in different parts of the image. The pixels in the image have 224 dimensions representing the energy in a specific spectral reflectance band. Each pixel observes a specific crop and there are a total of 16 different types of materials/crops. The objective is to cluster pixels into different classes based on the material they observe while considering each pixel independently. The underlying assumption is that the pixels observing the same class will have similar spectral reflectance values. For our simulations, during each iteration, we consider a set of $Q = 4$ different randomly selected materials and select 250 random pixels representing each of the 4 classes. The experimental results are averaged over a total of 100 random materials and pixel selections.

We utilize fully connected layers to form the nonlinear compression stage in the autoencoder network. The 224 dimension vector is the input to two fully connected layers with 256 nodes each. These two layers form the non linear mapping stage of the autoencoder. Rectified linear units (ReLU) [66] based nonlinear activation was employed. The 256 node output is constrained down to $Q = 4$ nodes (with linear activation) and scaled back up to 256 nodes. This is followed by 2 more layers with nonlinear activation to produce the output. As the second fully connected layer in the nonlinear mapping stage has 256 nodes, the output dimension of the mapping is $F = 256$.

The overall clustering accuracies can be seen in Fig. 7.2. The figure shows a boxplot of the accuracies over the 100 trials, where the red mark inside the box indicates the median

and the edges of the box mark the 25 and 75 percentiles of the accuracy across trials. As can be inferred from the figure, the proposed deep CCA approach performs better than both of the existing schemes.

As was explained in the previous section, if the observation model $f(\cdot)$ is non linear, the PCA reconstruction MSE cost into Q components $\sum_{i=1}^{P} \|\mathbf{X}_i - \mathbf{BCX}_i\|_2$ will be high and can be reduced by utilizing the mapping $\boldsymbol{\phi}(\cdot)$. The same can be inferred from Fig. 7.3.

In Fig. 7.4 we explore the impact of the output vector length F of the mapping $\boldsymbol{\phi}(\cdot)$ on the clustering accuracy. As the vector length is increased, the CCA accuracy increases up to a point (256/512 in this case). Ideally, beyond this value, with increase in vector length, the accuracy increase should flatten out and should remain constant. But as seen from the figure, the accuracy slowly decreases as the vector length increases. This can be attributed to poor generalization or over-fitting that may occur as increasing F essentially results in increasing the number of weights/nodes in the network and hence increasing the overall complexity of the network, while the number of input data vectors available for learning is fixed to a total of $P = 1000$.

7.2.2 MNIST Digits Dataset

The MNIST dataset represents the application of the proposed scheme for handwritten digit recognition. The dataset contains images from 10 classes representing 0-9 digits. Each image is gray-scale and has a size of 28x28 (or vectorized as a 784 dimensional vector). The objective here is to cluster images based on the digit each image represents. For our simulations we consider the digits between 0 and 3, and thus $Q = 4$. The experiment is repeated over a total of 100 times, during each iteration, a set of 250 randomly selected images representing each of the 4 classes is considered. Similar to the hyperspectral case the mapping $\phi(\cdot)$ has 2 fully connected layers with 512 nodes each. A box plot representing the overall clustering accuracies can be seen in Fig. 7.5.

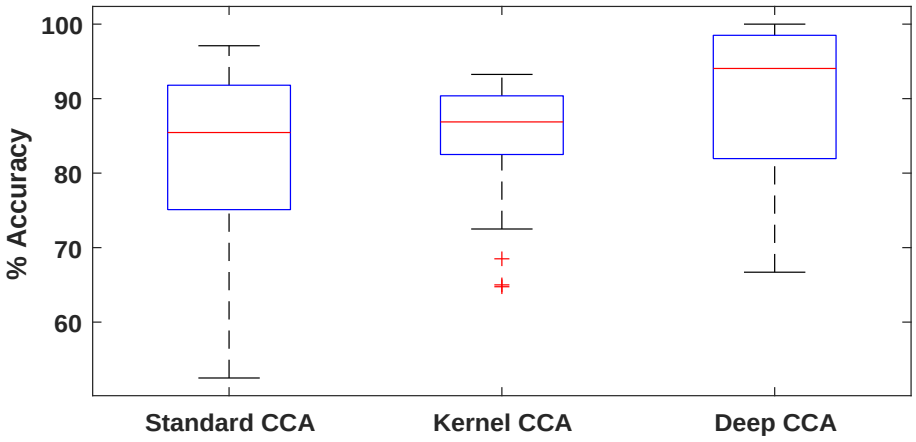


Figure 7.2. Boxplot comparing the accuracies of 3 different variants of CCA based clustering for the Salinas hyperspectral image dataset [2]. The central red mark in the box refers to the median accuracy, and the edges of the box mark 25^{th} and 75^{th} percentiles of the accuracy across all trials.

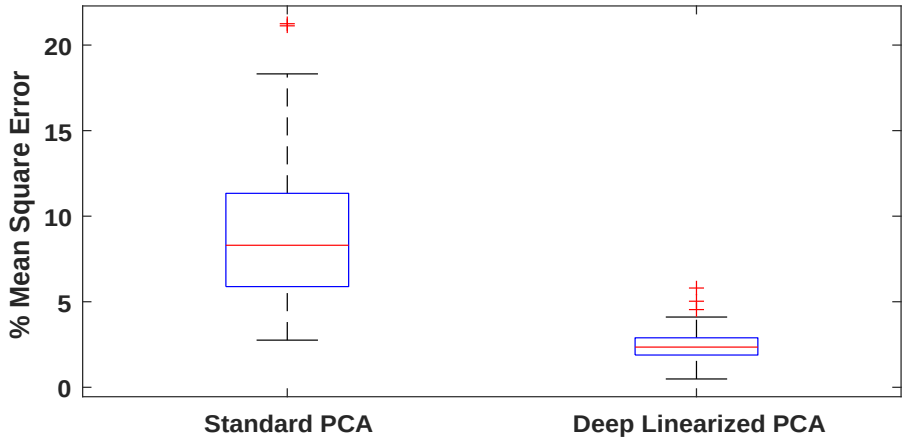


Figure 7.3. Mean square reconstruction loss obtained by the standard PCA based approach and as obtained by the proposed deep kernel linearized method for the Salinas hyperspectral image dataset [2] have been presented.

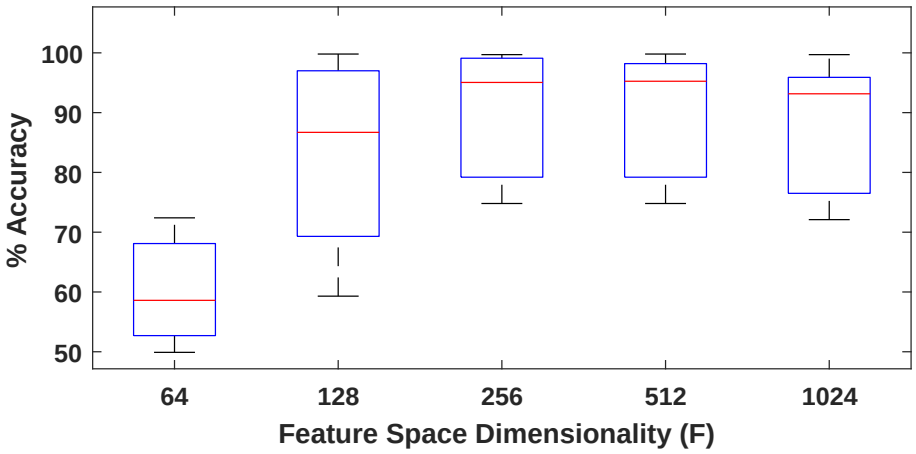


Figure 7.4. Comparison of accuracy plotted against the output dimension F of the mapping $\phi(\cdot)$.

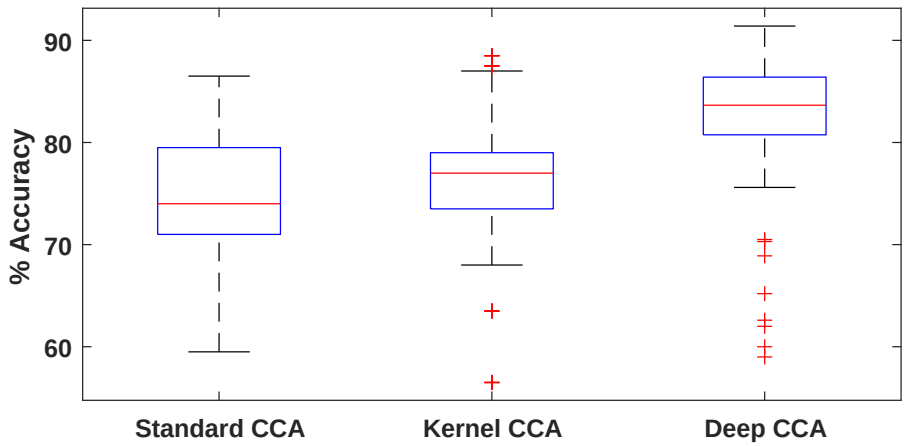


Figure 7.5. Boxplot comparing the accuracies of 3 different variants of CCA based clustering for the MNIST digit image database [6].

CHAPTER 8

CONCLUSION

As part of this work we have explored different methods for improving the accuracies of conventional CCA and NMF based approaches towards the clustering objective. Improvements were made across several different fronts, 1) By kernelizing the formulations to account for non-linear correlations in the data; 2) Exploring a framework for unsupervised kernel selection and multiple kernel learning; 3) Utilizing deep neural networks based mappings for extracting correlations and linearizng the data in an unsupervised setting; 4) And finally by reformulating the non-convex cost functions using the difference of convex functions algorithm to ensure convergence to a stationary point.

To achieve sparse matrix factors, indicative of the clustering, sparsity inducing techniques like $\ell_1 - \ell_2$ regularization have been explored. The non convex sparsity regularization was also relaxed using the difference of convex functions approach. We also proposed the MILP reformulation for the NMF based clustering, thus ensuring that the hard clustering constraints can be enforced. Extensive numerical tests have been conducted in varying datasets to demonstrate the advantage of the novel unsupervised methods, whose clustering performance comes close to the one achieved by supervised alternatives that rely on training data, while outperforming existing unsupervised techniques that rely heavily on different forms of parameter fine-tuning.

8.1 Future Directions

Although the proposed solutions achieve better accuracy than existing methods, scalability remains a major bottleneck. Going forward we want to explore schemes to make the solution linearly scalable while preserving the existing accuracy.

A big part of this work was dedicated towards constructing and utilizing the the Q-th eigenvalue function, $\Lambda^{Q}(\cdot)$. This function can essentially be understood to be as a distance metric for a given input matrix. An important direction of work that we want to explore is regarding the quality and shape of this metric compared to other standard distance metrics like ℓ_p norms where, $0 \leq p \leq \infty$, cosine distance and so on.

APPENDIX A

Human Activity Classification Data : Pre-Processing

The objective of the pre processing step is to find the frames representing the repetitive structures or the epochs and use them as vectors for the CCA based classification. It is important to understand that the epochs or the frames need to be synced along a common reference point to effectively apply the correlation analysis-based algorithms. Since all the epochs have a distinguishable peak, Fig. 2.3(a) and 2.3(b), we center the data around this peak and select a set of samples around it. There are multiple schemes present in the literature (eg. [71, 72]) to isolate epochs. In this work, since our primary objective and the contribution is not the epoch-based classification, we use a rather simplistic approach to isolate the epochs to form the frames with a nominal accuracy and a rather low computational complexity.

The first step is to remove the parts of the signal which are devoid of any activity. As seen in Fig. 2.2(a), the first 7000 samples consists of approximately constant valued signals, corresponding to user standing or sitting states. During this period the variance of the signal, computed over a window, is small in magnitude.

$$\bar{w}(n) = \begin{cases} w(n), & if \;\; \varrho^2(w(n)) \geq thresh_1 \\ 0, & \text{otherwise} \end{cases} \tag{A.1}$$

where, the function $\varrho^2(.)$ represents the variance of the signal calculated over a time window $[n-L/2, ..., n, ...n+L/2-1]$ with L being the window length and $thresh_1$ indicating a predetermined threshold.

Next, a simple peak detection algorithm is employed that finds the peaks and isolates P samples around it to form the frame. The signal magnitude is first compared against a predetermined threshold, $thresh_2$. The parts of the signal that are greater than the threshold are checked for local peaks (i.e neighboring samples have a magnitude lesser than the current sample). Also, since epochs extend over at least a few samples, two consecutive epochs (or two consecutive peaks) have to be separated by at least a certain number of

samples, ϑ_1. Thus, all the peaks which are under ϑ_1 samples away are discarded. The value of L used for evaluating the variance cannot be too small since it should be able to observe the signal statistics over a period comparable to the epoch and again it cannot be too large so that a large number of consecutive samples, devoid of any activity, get coupled with an epoch and are considered for the analysis. Although many combinations of L and $thresh_1$ may give reasonable results, we suggest the use of values in the range $P/2 < L < 2P$. The frames starting at, or around, the peak of the epoch are then used as vectors for the CCA. Thus the n-th frame is given by, $\boldsymbol{\omega}_n = \{\bar{w}(n_t - \hat{P} + 1), ..., \bar{w}(n_t), ..., \bar{w}(n_t + (P - \hat{P}))\}^T$, where n_t is the time instant corresponding to the peak of the t-th epoch and $\hat{P}$ is an integer such that $\hat{P} \in [0, P - 1]$. The above mentioned approach performs reasonably well. Since the focus of the paper is not the epoch detection and frame construction algorithm, we move our attention to the two classification approaches proposed here. The pre-processing stage for vectorization can be easily replaced with any other approach from the literature (like, [71, 72]) and can be used as an input to the algorithms presented later on.

APPENDIX B

Convergence of the Joint Kernel Leraning and NMF Based Clustering Formulation

In the following section we discuss the proof of convergence for the DCA based algorithm. For the convergence proofs, we have used some properties of the functions involved (and their derivatives) and we have also used some simplified notations. Before starting the discussion about the proofs, we discuss some of these properties and notations.

Property 1:

$$\left(\boldsymbol{\alpha}^k\right)^T[\tilde{\boldsymbol{\alpha}}^k]_Q = \begin{bmatrix} \boldsymbol{\alpha}_1^k \\ \boldsymbol{\alpha}_2^k \\ \dots \\ \boldsymbol{\alpha}_B^k \end{bmatrix}^T \begin{bmatrix} \mathbf{z}_1^k\hat{\mathbf{K}}_x^1\mathbf{z}_1^k + \cdots + \mathbf{z}_Q^k\hat{\mathbf{K}}_x^1\mathbf{z}_Q^k \\ \mathbf{z}_1^k\hat{\mathbf{K}}_x^2\mathbf{z}_1^k + \cdots + \mathbf{z}_Q^k\hat{\mathbf{K}}_x^2\mathbf{z}_Q^k \\ \dots\dots\dots\dots\dots\dots\dots\dots \\ \mathbf{z}_1^k\hat{\mathbf{K}}_x^B\mathbf{z}_1^k + \cdots + \mathbf{z}_Q^k\hat{\mathbf{K}}_x^B\mathbf{z}_Q^k \end{bmatrix} = \sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) \tag{B.1}$$

where, $[\tilde{\boldsymbol{\alpha}}^k]_Q$ is as defined in (5.6) and $\mathbf{z}_i^k$ is the eigenvector corresponding to the i-th largest eigenvalue of $\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j$.

Property 2:

At any iteration k of the algorithm,

$$\begin{aligned} &\Big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - {\boldsymbol{\alpha}^k}^T[\tilde{\boldsymbol{\alpha}}^k]_{Q-1} \geq 0 \Leftrightarrow \\ &\Big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - {\boldsymbol{\alpha}^k}^T\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k+1}\hat{\mathbf{K}}_x^j)\Big) \geq 0 \end{aligned} \tag{B.2}$$

Proof: By Property 1, we have:

$$\begin{aligned} &\Big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - {\boldsymbol{\alpha}^k}^T\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k+1}\hat{\mathbf{K}}_x^j)\Big) \\ &= \sum_{i=1}^{Q-1}{\mathbf{z}_i^k}^T\Big(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j\Big)\mathbf{z}_i^k - \sum_{i=1}^{Q-1}{\mathbf{z}_i^{k+1}}^T\Big(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j\Big)\mathbf{z}_i^{k+1} \end{aligned} \tag{B.3}$$

where, $\mathbf{z}_i^k$ is the eigenvector corresponding to the $\boldsymbol{\alpha}_j^k$ term and $\mathbf{z}_i^{k+1}$ is the eigenvector corresponding to $\boldsymbol{\alpha}_j^{k+1}$ term. Consequently it is evident that $\sum_{i=1}^{Q-1}{\mathbf{z}_i^k}^T\Big(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j\Big)\mathbf{z}_i^k \geq \sum_{i=1}^{Q-1}{\mathbf{z}_i^{k+1}}^T\Big(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j\Big)\mathbf{z}_i^{k+1}$ since $\mathbf{z}_i^k$ are eigenvectors of the kernel matrix combination related to $\boldsymbol{\alpha}_j^k$, whereas $\mathbf{z}_i^{k+1}$ are the eigenvectors for a different convex combination.

Simplified Notation:

At the k-th iteration, without loss of generality we can consider the updates w.r.t $\mathbf{M}$ and $\boldsymbol{\alpha}$, thus $\mathbf{N}$ is assumed to be constant. The function in (5.2) can then be written in a simplified form as

$$F(\mathbf{X}) = ||\mathbf{A}\mathbf{X}||_2^2 + \lambda \sum_{i=1}^{P} \big(||\mathbf{J}^i\mathbf{X}||_1 - ||\mathbf{J}^i\mathbf{X}||_2\big) + \mu\big(\sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j)\big) \tag{B.4}$$

where $\mathbf{A} = [\mathbf{C} \ -\boldsymbol{\mathcal{N}}]$ is a matrix of size $P^2 \times (B + PQ)$, $\mathbf{X}$ is of size $(B + PQ) \times 1$ and $\boldsymbol{\mathcal{N}}$ is of size $P^2 \times PQ$.

$$\mathbf{A} = \begin{bmatrix} \mathbf{C} & -\boldsymbol{\mathcal{N}} \end{bmatrix}, \mathbf{X} = \begin{bmatrix} \boldsymbol{\alpha} \\ \mathbf{vec}(\mathbf{M^T}) \end{bmatrix} \mathbf{C} = \begin{bmatrix} \mathbf{vec}(\hat{\mathbf{K}}_x^1) & \dots & \mathbf{vec}(\hat{\mathbf{K}}_x^B) \end{bmatrix} \tag{B.5}$$

$$\boldsymbol{\mathcal{N}} = \begin{bmatrix} \mathbf{N} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{N} & \dots & \mathbf{0} \\ \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{N} \end{bmatrix} \mathbf{J}^i = \begin{bmatrix} \mathbf{0}_{B\times B} & & & \mathbf{0} \\ & \mathbf{0}_{Q(i-1)\times Q(i-1)} & & \\ & & \mathbf{I}_{Q\times Q} & \\ \mathbf{0} & & & \mathbf{0}_{Q(P-i)\times Q(P-i)} \end{bmatrix}$$

For the rest of the section, we use the simplified notations discussed here and the actual notations interchangeably.

B.0.1 Proposition:

At every iteration p, the DCA iterates monotonically decrease, i.e. $F(\mathbf{X}^{k,p}) - F(\mathbf{X}^{k,p+1}) \geq 0$.

Proof. From the simplified form of (5.2), as expressed in (B.4), let

$$h(\mathbf{X}) = \lambda \sum_{i=1}^{P} ||\mathbf{J}^i\mathbf{X}||_2 + \mu \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j). \tag{B.6}$$

We can express the difference between the objective values between any two consecutive iterations $F(\mathbf{X}^{k,p}) - F(\mathbf{X}^{k,p+1})$ as,

$$F(\mathbf{X}^{k,p}) - F(\mathbf{X}^{k,p+1}) = ||\mathbf{A}(\mathbf{X}^{k,p} - \mathbf{X}^{k,p+1})||_2^2 + \langle 2\mathbf{A}(\mathbf{X}^{k,p} - \mathbf{X}^{k,p+1}, \mathbf{A}\mathbf{X}^{k,p+1})\rangle$$
$$+ \lambda\big(\sum_{i=1}^{P} ||\mathbf{J}^i\mathbf{X}^{k,p}||_1 - \lambda||\mathbf{J}^i\mathbf{X}^{k,p+1}||_1\big)$$
$$+ \mu\big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k,p}\hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k,p+1}\hat{\mathbf{K}}_x^j)\big) + \big(h(\mathbf{X}^{k,p+1}) - h(\mathbf{X}^{k,p})\big) \quad \text{(B.7)}$$

where, $\langle\cdot,\cdot\rangle$ represents the inner product between the two input arguments. For simplicity we drop k from the superscript of all the variables. Thus $\mathbf{X}^{k,p}$ is represented as $\mathbf{X}^p$, and so on.

Since $\mathbf{X}^{p+1}$ is a stationary point of the relaxed objective in (5.4) (after applying the majorization w.r.t concave terms of $\mathbf{M}$ and $\boldsymbol{\alpha}$ and keeping $\mathbf{N}$ constant),

$$\left.\frac{\partial F(\mathbf{X})}{\partial \mathbf{X}}\right|_{\mathbf{X}=\mathbf{X}^{p+1}} = 2\mathbf{A}^T(\mathbf{A}\mathbf{X}^{p+1}) + \lambda\mathbf{w}^{p+1} + \mu\mathbf{v}^{p+1} - \mathbf{y}^p = 0 \quad \text{(B.8)}$$

where, $\mathbf{v}^{p+1}$ is a $B + PQ$ dimensional vector, such that the first B components are given by $\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{p+1}\hat{\mathbf{K}}_x^j)\big)$ and the rest PQ components are 0. Similarly, the first B components of the vector $\mathbf{w}^{p+1}$ are 0, and the rest PQ components are $\partial||\mathbf{J}^i\mathbf{X}^{p+1}||_1$.

After multiplying (B.8) by $(\mathbf{X}^p - \mathbf{X}^{p+1})^T$ we obtain

$$\langle 2\mathbf{A}(\mathbf{X}^p - \mathbf{X}^{p+1}), (\mathbf{A}\mathbf{X}^{p+1})\rangle - \lambda||\mathbf{J}^i\mathbf{X}^{p+1}||_1 - \mu\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{p+1}\hat{\mathbf{K}}_x^j)\big)$$
$$+ \lambda\langle\mathbf{w}^{p+1}, \mathbf{X}^p\rangle + \mu\langle\mathbf{v}^{p+1}, \mathbf{X}^p\rangle + \langle\mathbf{y}^p, \mathbf{X}^{p+1} - \mathbf{X}^p\rangle = 0 \quad \text{(B.9)}$$

From (B.9) and (B.7), we have,

$$F(\mathbf{X}^p) - F(\mathbf{X}^{p+1}) = ||\mathbf{A}(\mathbf{X}^p - \mathbf{X}^{p+1})||_2^2 + \lambda\big(||\mathbf{J}^i\mathbf{X}^p||_1 - \langle\mathbf{w}^{p+1}, \mathbf{X}^p\rangle\big)$$
$$+ \mu\big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^p\hat{\mathbf{K}}_x^j) - \langle\mathbf{v}^{p+1}, \mathbf{X}^p\rangle + \big(h(\mathbf{X}^{p+1}) - h(\mathbf{X}^p)\big) - \langle\mathbf{y}^p, \mathbf{X}^{p+1} - \mathbf{X}^p\rangle \quad \text{(B.10)}$$

Lets analyze each term of the above equation individually. The first term $||\mathbf{A}(\mathbf{X}^p - \mathbf{X}^{p+1})||_2^2 \geq 0$. The second term, $\big(||\mathbf{J}^i\mathbf{X}^p||_1 - \langle \mathbf{w}^{p+1}, \mathbf{X}^p \rangle\big)$, where, $\mathbf{w}^{p+1}$ is the $\partial||\mathbf{J}^i\mathbf{X}^p||_1$ and thus is equivalent to $sign(\mathbf{J}^i\mathbf{X}^p)$. Therefore, $\big(||\mathbf{J}^i\mathbf{X}^p||_1 - \langle \mathbf{w}^{p+1}, \mathbf{X}^p \rangle\big) = 0$.

The third term is positive by Property 2, since,

$$\begin{aligned}&\big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^p\hat{\mathbf{K}}_x^j) - \langle \mathbf{v}^{p+1}, \mathbf{X}^p\rangle \\ &= \big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^p\hat{\mathbf{K}}_x^j) - \boldsymbol{\alpha}^{pT}\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{p+1}\hat{\mathbf{K}}_x^j)\big). \qquad \text{(B.11)}\end{aligned}$$

For the fourth term, by the definition of subgradient, $h(\mathbf{X}^{p+1}) - h(\mathbf{X}^p) \geq \langle \mathbf{y}^p, \mathbf{X}^{p+1} - \mathbf{X}^p \rangle$, where, $\mathbf{y}^p \in \partial h(\mathbf{X}^p)$. Thus it is positive and we conclude that the iterates $\mathbf{X}^{p+1}$ lead to a non-increasing cost function $F(\mathbf{X}^{p+1})$.

Using similar arguments it can be shown that for the DCA iterations w.r.t update of $\mathbf{N}$, the function value goes down at every iteration. □

B.0.2 Proposition:

The overall objective in (5.2) is lower bounded.

Proof. Examining (B.4) (which is the simplified form of (5.2)), it is evident that the first part of the function, $||\mathbf{AX}||_2^2 \geq 0$ is lower bounded to 0. The next part of the function corresponds to the $\ell_1 - \ell_2$ norm of the columns of $\mathbf{M}$. As we know, $||\mathbf{M}_{i,:}||_1 \geq ||\mathbf{M}_{i,:}||_2$, and the equality is achieved when at most one component of $\mathbf{M}_{i,:}$ is non-zero. Thus this part of $F(\mathbf{X})$ is lower bounded by zero too.

The third part of the function related to the difference of sum of eigenvalues, $\big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j) - \sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j)\big)$. Since we normalize all the kernel matrices $\hat{\mathbf{K}}_x^j$ to unit trace, the

maximum value that $\Lambda^Q(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j)$ can achieve is $\frac{1}{P}$. Therfore, $\Big|\big(\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j)-\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j\hat{\mathbf{K}}_x^j)\big)\Big| \leq \frac{1}{P}$ and is also lower bounded. □

B.0.3 Proposition:

As $k \to \infty$, the limit point $\{\mathbf{M}^*, \mathbf{N}^*, \boldsymbol{\alpha}^*\}$ of the DCA method is a stationary point of the cost in (5.2) and thus satisfies the following three first-order optimality conditions:

$$\begin{aligned}
&\mathbf{0} \in \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^* - \mathbf{M}^*\mathbf{N}^{*T}) + \mu\big(\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^*\hat{\mathbf{K}}_x^j) - \partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^*\hat{\mathbf{K}}_x^j)\big)\\
&\mathbf{0} \in (\sum_{j=1}^{B}\boldsymbol{\alpha}_j^*\hat{\mathbf{K}}_x^j - \mathbf{M}^*\mathbf{N}^{*T})\mathbf{M}^* + \lambda\big(\sum_{i=1}^{P}\partial||\mathbf{M}_{i,:}^*||_1 - \partial||\mathbf{M}_{i,:}^*||_2\big)\\
&\mathbf{0} \in (\sum_{j=1}^{B}\boldsymbol{\alpha}_j^*\hat{\mathbf{K}}_x^j - \mathbf{M}^*\mathbf{N}^{*T})\mathbf{N}^* + \lambda\big(\sum_{i=1}^{P}\partial||\mathbf{N}_{i,:}^*||_1 - \partial||\mathbf{N}_{i,:}^*||_2
\end{aligned} \tag{B.12}$$

Proof. 1) Consider the optimality condition w.r.t $\boldsymbol{\alpha}$. At the k-th iteration of the DCA based algorithm, the optimality condition for the relaxed cost in (5.4) is given as,

$$0 \in \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^k - \mathbf{M}^k\mathbf{N}^{k-1^T}) + \mu\big(\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - \partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k-1}\hat{\mathbf{K}}_x^j)\big). \tag{B.13}$$

Considering this first order optimality condition under the limit $\lim_{k\to\infty}$ and considering only the eigenvalue related terms, we show next the existence of the following limit,

$$\begin{aligned}
\lim_{k\to\infty}\big(\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - \partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k-1}\hat{\mathbf{K}}_x^j)\big) = \lim_{k\to\infty}\Big(\partial\sum_{i=1}^{Q-1}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j)\\
- \partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) + \big(\partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^k\hat{\mathbf{K}}_x^j) - \partial\sum_{i=1}^{Q}\Lambda^i(\sum_{j=1}^{B}\boldsymbol{\alpha}_j^{k-1}\hat{\mathbf{K}}_x^j)\big)\Big)
\end{aligned} \tag{B.14}$$

Since we have already proved that $F(\mathbf{X}^k) - F(\mathbf{X}^{k+1}) \geq 0$ and since $F(\mathbf{X})$ is lower bounded and continuous, we have that as $k \to \infty$, $F(\mathbf{X}^k) - F(\mathbf{X}^{k+1}) \to 0$ and consequently $|\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}|_2 \to 0$.

For a continuous matrix function $C_f(\boldsymbol{\alpha})$ with the i-th largest eigenvalue given by $e_i(\boldsymbol{\alpha})$, the eigenvalues are continuous functions in $\boldsymbol{\alpha}$ if the eigenvalues have a multiplicity

of 1 [73]. In such a case, the eigenvectors $z_i(\boldsymbol{\alpha})$ corresponding to $e_i(\boldsymbol{\alpha})$ are continuous in $\boldsymbol{\alpha}$. If the multiplicity is not 1 then the eigenvectors can crossover and different eigenvectors will correspond to a given $e_i(\boldsymbol{\alpha})$ before and after the crossover point.

In the case of convex sum of kernel matrices as indicated by $C_f(\boldsymbol{\alpha}) = \sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j$, the eigenvalues do not strictly have a multiplicity of 1 and thus eigenvectors may crossover, resulting in a scenario where different eigenvectors correspond to a particular eigenvalue for different values of $\boldsymbol{\alpha}$.

From, (5.6), we know that

$$\partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j) = \begin{bmatrix} \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_1^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^1 \mathbf{z}_Q^k \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_1^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^2 \mathbf{z}_Q^k \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \mathbf{z}_1^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_1^k + \cdots + \mathbf{z}_Q^{k\,T} \hat{\mathbf{K}}^B \mathbf{z}_Q^k \end{bmatrix}$$

where, $\mathbf{z}_i^k$ is the eigenvector corresponding to the i-th largest eigenvalue e_i^k of the matrix $\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j$. Due to lack of guarantees for multiplicity to be 1 for all possible values $\boldsymbol{\alpha}$, the eigenvectors crossover and $\mathbf{z}_i^k$ maybe non-smooth.

Consider the update equation w.r.t $\boldsymbol{\alpha}$ in (5.7). Here $\boldsymbol{\alpha}$ is evaluated at a fixed value of $\mathbf{M}, \mathbf{N}$ so as to minimize, $|| \sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j - \mathbf{M}\mathbf{N}^T||_F^2 + \mu\Big(\sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha} \hat{\mathbf{K}}_x^j) - \langle \boldsymbol{\alpha}, [\tilde{\boldsymbol{\alpha}}^k]_{Q-1} \rangle \Big)$. The first part corresponding to minimizing $|| \sum_{j=1}^{B} \boldsymbol{\alpha}_j \hat{\mathbf{K}}_x^j - \mathbf{M}\mathbf{N}^T||_F^2$ selects an $\boldsymbol{\alpha}$ such that, $rank\Big(C_f(\boldsymbol{\alpha})\Big) = rank\Big(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j \Big) \leq Q$ as $\mathbf{M}$ and $\mathbf{N}$ have only Q columns. The second part by maximizing the Q-th eigenvalue enforces the rank to be Q. Thus, under the assumption that B is sufficiently large so as to ensure that there exists a convex combination of the kernel matrices with rank Q, then by the $\boldsymbol{\alpha}$ update in (5.7) as $k \to \infty$, $rank\Big(C_f(\boldsymbol{\alpha^k})\Big) = Q$.

Thus $e_i^k = 0 \ \forall i > Q$. Since $|\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}|_2 \to 0$, $e_i^k = 0 \ \forall i > Q$. Consequently if eigenvector crossover happens between $\boldsymbol{\alpha}^k$ and $\boldsymbol{\alpha}^{k-1}$, the eigenvectors will be

exchanged only amongst the set of Q eigenvectors corresponding to $\{e_i^k | i \leq Q\}$. Therefore, $\sum_{i=1}^{Q} \mathbf{z}_i^{k^T} \hat{\mathbf{K}}^j \mathbf{z}_i^k = \sum_{i=1}^{Q} \mathbf{z}_i^{k-1^T} \hat{\mathbf{K}}^j \mathbf{z}_j^{k-1} \forall j \in \{1, ..., B\}$ as $k \to \infty$.

Hence, $|\partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j) - \partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^{k-1} \hat{\mathbf{K}}_x^j)|_2 \to 0$ as $k \to \infty$. Therefore from (B.14) we have,

$$\begin{aligned}
&\lim_{k\to\infty} \big(\partial \sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j) - \partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^{k-1} \hat{\mathbf{K}}_x^j)\big) \\
&= \lim_{k\to\infty} \Big(\partial \sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j) - \partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^k \hat{\mathbf{K}}_x^j)\Big) \\
&= \Big(\partial \sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^* \hat{\mathbf{K}}_x^j) - \partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^* \hat{\mathbf{K}}_x^j)\Big)
\end{aligned} \tag{B.15}$$

Considering the rest of the terms,

$$\lim_{k\to\infty} \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^k - \mathbf{M}^k \mathbf{N}^{k-1^T}) \tag{B.16}$$

Since the overall function (5.2) is monotonically non-increasing at every iteration and lower bounded, then, as $k \to \infty$, $||\mathbf{N}^k - \mathbf{N}^{k-1}||_F \to 0$. Therefore, since the product $\mathbf{MN}$ is a continuous function $\mathbf{M}$ and $\mathbf{N}$, $||\mathbf{M}^{k+1}\mathbf{N}^{k+1} - \mathbf{M}^k \mathbf{N}^{k-1}||_F \to 0$. Thus we have,

$$\lim_{k\to\infty} \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^k - \mathbf{M}^k \mathbf{N}^{k-1^T}) = \lim_{k\to\infty} \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^k - \mathbf{M}^k \mathbf{N}^{k^T}) = \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^* - \mathbf{M}^* \mathbf{N}^{*T}) \tag{B.17}$$

and combining this with the eigenvalue terms considered before we get the overall first order optimality condition in (B.12).

$$0 \in \mathbf{C}^T(\mathbf{C}\boldsymbol{\alpha}^* - \mathbf{M}^* \mathbf{N}^{*T}) + \mu\big(\partial \textstyle\sum_{i=1}^{Q-1} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^* \hat{\mathbf{K}}_x^j) - \partial \sum_{i=1}^{Q} \Lambda^i(\sum_{j=1}^{B} \boldsymbol{\alpha}_j^* \hat{\mathbf{K}}_x^j)\big) \tag{B.18}$$

2) Consider the optimality condition w.r.t $\mathbf{M}$. At k-th iteration of the DCA based algorithm, the optimality condition for the relaxed cost in (5.4) is given as,

$$0 \in (\textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}_j^{k-1} \hat{\mathbf{K}}_x^j - \mathbf{M}^k \mathbf{N}^{k-1^T})\mathbf{M}^k + \lambda\big(\sum_{i=1}^{P} \partial||\mathbf{M}_{i,:}^k||_1 - \partial||\mathbf{M}_{i,:}^{k-1}||_2\big)$$

Based on a similar explanation as provided above for (B.16) and (B.17), the terms corresponding to the matrix factorization go to the limit point as $k \to \infty$. Next consider the ℓ_1 term, since we solve for $\mathbf{M}$ under the non negativity constraint, $\mathbf{M} \geq 0$. The value of each component of $\mathbf{M}$ is always positive or zero, therefore, the sub gradient, $\partial||\mathbf{M}^k_{i,:}||_1$ is a vector of ones for all values of k. Thus we have on the $\lim_{k\to\infty}$,

$$0 \in (\textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}^*_j \hat{\mathbf{K}}^j_x - \mathbf{M}^*\mathbf{N}^{*T})\mathbf{M}^* + \lambda\big(\textstyle\sum_{i=1}^{P} \partial||\mathbf{M}^*_{i,:}||_1\big) \\ - \lim_{k\to\infty} \lambda\big(\textstyle\sum_{i=1}^{P} \partial||\mathbf{M}^{k-1}_{i,:}||_2\big). \tag{B.19}$$

The last limit term in (B.19) gives,

$$\lambda \lim_{k\to\infty} \big(\textstyle\sum_{i=1}^{P} \frac{\mathbf{M}^{k-1}_{i,:}}{||\mathbf{M}^{k-1}_{i,:}||_2}\big) \tag{B.20}$$
$$= \lambda \lim_{k\to\infty} \big(\textstyle\sum_{i=1}^{P} \frac{\mathbf{M}^{k}_{i,:}}{||\mathbf{M}^{k}_{i,:}||_2}\big) + \lambda\big(\textstyle\sum_{i=1}^{P} \frac{\mathbf{M}^{k}_{i,:}}{||\mathbf{M}^{k}_{i,:}||_2} - \frac{\mathbf{M}^{k-1}_{i,:}}{||\mathbf{M}^{k-1}_{i,:}||_2}\big) = \lambda\big(\textstyle\sum_{i=1}^{P} \frac{\mathbf{M}^{*}_{i,:}}{||\mathbf{M}^{*}_{i,:}||_2}\big)$$

where, $||\frac{\mathbf{M}^{k}_{i,:}}{||\mathbf{M}^{k}_{i,:}||_2} - \frac{\mathbf{M}^{k-1}_{i,:}}{||\mathbf{M}^{k-1}_{i,:}||_2}||_2 \to 0$ as $||\mathbf{M}^{k}_{i,:} - \mathbf{M}^{k-1}_{i,:}||_2 \to 0$. Thus, overall we have that

$$0 \in (\textstyle\sum_{j=1}^{B} \boldsymbol{\alpha}^*_j \hat{\mathbf{K}}^j_x - \mathbf{M}^*\mathbf{N}^{*T})\mathbf{M}^* + \lambda\big(\textstyle\sum_{i=1}^{P} \partial||\mathbf{M}^*_{i,:}||_1\big) - \lambda\big(\textstyle\sum_{i=1}^{P} \frac{\mathbf{M}^{*}_{i,:}}{||\mathbf{M}^{*}_{i,:}||_2}\big). \tag{B.21}$$

3) The proof for the third part (for the derivative w.r.t $\mathbf{N}$) can be achieved in a similar manner to the above proof for the second part. □

APPENDIX C

Selection of Optimal α

Without loss of generality, we consider the case where the data vectors have been ordered based on the class they belong to. Thus, the ideal kernel covariance matrix will have a strong block diagonal structure.

Since we assume that the data sources representing different classes are independent, the covariance between elements of different classes, i.e., $\sum_n s^{j1}(n)s^{j2}(n) = 0$. From [74] we know that the covariance of any function of independent random variables is 0, therefore for $j_1, j_2 \in \{1, ..., Q | j_1 \neq j_2\}$

$$\hat{\mathbf{K}}^j(x_{i_1}^{j_1}, x_{i_2}^{j_2}) = \sum_n \hat{\phi}(x_{i_1}^{j_1}(n)), \hat{\phi}(x_{i_2}^{j_2}(n)) = 0 \ \ \forall i_1, i_2. \tag{C.1}$$

Thus, irrespective of the mapping $\hat{\phi}(\cdot)$, the off block diagonal elements of the kernel covariance matrix are zero.

As per the primary problem formulation in (5.2), we minimize the reconstruction error $|| \sum_{j=1}^{B} \alpha_j \hat{\mathbf{K}}_x^j - \mathbf{M}\mathbf{M}^T||_F^2$. Since $\mathbf{M}$ has Q columns, $max\ (rank(\mathbf{M}\mathbf{M}^T)) = Q$. Therefore, to minimize the reconstruction error, only a linear combination of kernel matrices with rank of at most Q will be selected.

On the other hand, maximizing the Q-th eigenvalue ensures a nonzero Q-th eigenvalue, thus removing the possibility of having a linear combination $rank(\sum_{j=1}^{B} \alpha_j \hat{\mathbf{K}}_x^j) < Q$.

Thus, at optimality, the solution to (5.2) minimizes the reconstruction error while maximizing the Q-th eigenvalue, thus ensuring that a kernel with a rank of exactly Q is selected through the linear combination. Under the condition wherein there is more than 1 convex combination of kernels with a rank-Q, the solution which maximizes the Q-th eigenvalue or that gives equal weightage to all the Q classes is selected.

REFERENCES

[1] "k-means matlab documentation," Available:https://www.mathworks.com/help/stats/kmeans.html.

[2] "Hyperspectral remote sensing scenes," Available: http://www.ehu.eus/ccwintco/index.php?title= Hyperspectral_Remote_Sensing_Scenes.

[3] S. A. Nene, S. K. Nayar, H. Murase, *et al.*, "Columbia object image library (coil-20)," 1996.

[4] D. Micucci, M. Mobilio, and P. Napoletano, "Unimib shar: a new dataset for human activity recognition using acceleration data from smartphones," *arXiv preprint arXiv:1611.07688v2*, 2017.

[5] "Trec: Text retrieval conference," Available: http://www.trec.nist.gov, 1999.

[6] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[8] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[9] J. Blömer, C. Lammersen, M. Schmidt, and C. Sohler, "Theoretical analysis of the k-means algorithm–a survey," in *Algorithm Engineering*. Springer, 2016, pp. 81–116.

[10] A. J. Smola and B. Schölkopf, *Learning with kernels*. Citeseer, 1998.

[11] J. Chen and I. D. Schizas, "Online distributed sparsity-aware canonical correlation analysis," *IEEE Transactions on Signal Processing*, vol. 64, no. 3, pp. 688–703, Feb 2016.

[12] J. Chen, A. Malhotra, and I. D. Schizas, "Data-driven sensors clustering and filtering for communication efficient field reconstruction," *Signal Processing*, vol. 133, pp. 156 – 168, 2017.

[13] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 2003, pp. 267–273.

[14] D. Cai, X. He, X. Wu, and J. Han, "Non-negative matrix factorization on manifold," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 63–72.

[15] F. Pompili, N. Gillis, P.-A. Absil, and F. Glineur, "Two algorithms for orthogonal non-negative matrix factorization with application to clustering," *Neurocomputing*, vol. 141, pp. 15 – 25, 2014.

[16] K. Huang, N. D. Sidiropoulos, and A. Swami, "Non-negative matrix factorization revisited: Uniqueness and algorithm for symmetric decomposition," *IEEE Transactions on Signal Processing*, vol. 62, no. 1, pp. 211–224, Jan 2014.

[17] I. D. Schizas, "Distributed informative-sensor identification via sparsity-aware matrix decomposition," *IEEE Transactions on Signal Processing*, vol. 61, no. 18, 2013.

[18] Y. Wang and Y. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1336–1353, June 2013.

[19] A. C. Türkmen, "A review of nonnegative matrix factorization methods for clustering," *arXiv preprint arXiv:1507.03194*, 2015.

[20] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A deep matrix factorization method for learning attribute representations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 3, pp. 417–429, 2017.

[21] K. T. Shahid, A. Malhotra, I. D. Schizas, and S. Tjuatja, "Unsupervised kernel correlations based hyperspectral clustering with missing pixels," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, pp. 1–12, 2018.

[22] A. Malhotra, K. T. Shahid, I. D. Schizas, and S. Tjuatja, "Fault tolerant unsupervised kernel-based information clustering in hyperspectral images," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2017, pp. 2191–2194.

[23] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2011.

[24] J. Chen, G. Wang, Y. Shen, and G. B. Giannakis, "Canonical correlation analysis of datasets with a common source graph," *arXiv preprint arXiv:1803.10309*, 2018.

[25] D. Kuang, C. Ding, and H. Park, *Symmetric Nonnegative Matrix Factorization for Graph Clustering*, pp. 106–117.

[26] A. Malhotra and I. D. Schizas, "Milp-based unsupervised clustering," *IEEE Signal Processing Letters*, vol. 25, no. 12, pp. 1825–1829, Dec 2018.

[27] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola, "On kernel-target alignment," in *Advances in Neural Information Processing Systems 14*. MIT Press, 2002, pp. 367–373.

[28] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 795–828, 2012.

[29] Y. Gu, C. Wang, D. You, Y. Zhang, S. Wang, and Y. Zhang, "Representative multiple kernel learning for classification in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 7, pp. 2852–2865, July 2012.

[30] L. Dioş, M. Oltean, A. Rogozan, and J.-P. Pecuchet, "Improving svm performance using a linear combination of kernels," in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2007, pp. 218–227.

[31] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning, 1989," *Reading: Addison-Wesley*, 1989.

[32] Y. Shen, T. Chen, and G. B. Giannakis, "Online ensemble multi-kernel learning adaptive to non-stationary and adversarial environments," *arXiv preprint arXiv:1712.09983*, 2017.

[33] M. Gonen and E. Alpaydin, "Multiple kernel learning algorithms," *J. Mach. Learn. Res.*, vol. 12, pp. 2211–2268, July 2011. [Online]. Available: http://dl.acm.org/citation.cfm?id=1953048.2021071

[34] J. Zhuang, J. Wang, S. Hoi, and X. Lan, "Unsupervised multiple kernel learning," in *Proceedings of the Asian Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, C.-N. Hsu and W. S. Lee, Eds., vol. 20. PMLR, 14–15 Nov 2011, pp. 129–144.

[35] D. Zhang, Z. H. Zhou, and S. Chen, "Adaptive kernel principal component analysis with unsupervised learning of kernels," in *Sixth International Conference on Data Mining (ICDM'06)*, Dec 2006, pp. 1178–1182.

[36] G. Tzortzis and A. Likas, "Greedy unsupervised multiple kernel learning," in *Hellenic Conference on Artificial Intelligence*. Springer, 2012, pp. 73–80.

[37] H. Valizadegan and R. Jin, "Generalized maximum margin clustering and unsupervised kernel learning," in *Advances in neural information processing systems*, 2007, pp. 1417–1424.

[38] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," in *Advances in Neural Information Processing Systems 17*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. MIT Press, 2005, pp. 1537–1544. [Online]. Available: http://papers.nips.cc/paper/2602-maximum-margin-clustering.pdf

[39] A. Malhotra, I. D. Schizas, and V. Metsis, "Correlation analysis-based classification of human activity time series," *IEEE Sensors Journal*, vol. 18, no. 19, pp. 8085–8095, Oct 2018.

[40] J. Chen, A. Malhotra, and I. D. Schizas, "Information-based clustering and filtering for field reconstruction," in *2015 49th Asilomar Conference on Signals, Systems and Computers*, Nov 2015, pp. 576–580.

[41] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe, "220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3," Sep 2015. [Online]. Available: https://purr.purdue.edu/publications/1947/1

[42] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 6, pp. 1351–1362, 2005.

[43] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomput.*, vol. 171, no. C, pp. 754–767, Jan. 2016. [Online]. Available: https://doi.org/10.1016/j.neucom.2015.07.085

[44] GLNT BioRadio physiological monitor. https://glneurotech.com/bioradio/.

[45] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted 1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 877–905, Dec 2008.

[46] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007.

[47] M.-J. Lai and J. Wang, "An unconstrained ℓ_q minimization with $0 < q \leq 1$ for sparse solution of underdetermined linear systems," *SIAM Journal on Optimization*, vol. 21, no. 1, pp. 82–101, 2011.

[48] S. Boyd and L. Vandenberghe, "Convex optimization." Cambridge university press, 2004, ch. 3.1.7.

[49] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part i — convex underestimating problems," *Mathematical Programming*, vol. 10, no. 1, pp. 147–175, Dec 1976. [Online]. Available: https://doi.org/10.1007/BF01580665

[50] Gurobi, "Gurobi optimizer reference manual," 2018. [Online]. Available: http://www.gurobi.com

[51] G. L. Nemhauser and L. A. Wolsey, "Integer programming and combinatorial optimization," *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, vol. 20, pp. 8–12, 1988.

[52] R. Jin, S. C. Hoi, and T. Yang, "Online multiple kernel learning: Algorithms and mistake bounds," in *International Conference on Algorithmic Learning Theory*. Springer, 2010, pp. 390–404.

[53] S. C. Hoi, R. Jin, P. Zhao, and T. Yang, "Online multiple kernel classification," *Machine Learning*, vol. 90, no. 2, pp. 289–316, 2013.

[54] A. Malhotra, K. T. Shahid, and I. D. Schizas, "Unsupervised kernel learning for correlation based clustering," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Oct 2018, pp. 2007–2011.

[55] M. L. Overton and R. S. Womersley, "Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices," *Mathematical Programming*, vol. 62, no. 1, pp. 321–357, Feb 1993.

[56] T. Pham Dinh and H. A. Le Thi, "Convex analysis approach to d.c. programming: Theory, Algorithm and Applications," *Acta Mathematica Vietnamica*, vol. 22, no. 1, pp. 289–355, 1997.

[57] J. R. Magnus, "On differentiating eigenvalues and eigenvectors," *Econometric Theory*, vol. 1, no. 2, pp. 179–191, 1985.

[58] P. Yin, Y. Lou, Q. He, and J. Xin, "Minimization of ℓ_{1-2} for compressed sensing," *SIAM Journal on Scientific Computing*, vol. 37, no. 1, pp. A536–A563, 2015.

[59] E. Esser, Y. Lou, and J. Xin, "A method for finding structured sparse solutions to nonnegative least squares problems with applications," *SIAM Journal on Imaging Sciences*, vol. 6, no. 4, pp. 2010–2046, 2013.

[60] Y. Chen and X. Ye, "Projection onto a simplex," *arXiv preprint arXiv:1101.6081*, 2011.

[61] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods," *lecture notes of EE392o, Stanford University, Autumn Quarter*, vol. 2004, pp. 2004–2005, 2003.

[62] D. Chu, L. Liao, M. Ng, and X. Zhang, "Sparse kernel canonical correlation analysis," in *Proceedings of International Multiconference of Engineers and Computer Scientists, Hong Kong*, 2013.

[63] H. Zha, X. He, C. Ding, M. Gu, and H. D. Simon, "Spectral relaxation for k-means clustering," in *Advances in neural information processing systems*, 2002, pp. 1057–1064.

[64] D. Cai, X. He, and J. Han, "Document clustering using locality preserving indexing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 12, pp. 1624–1637, 2005.

[65] Y. Zhao and G. Karypis, "Topic-driven clustering for document datasets," in *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 2005, pp. 358–369.

[66] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[67] K. B. Petersen, M. S. Pedersen, *et al.*, "The matrix cookbook," *Technical University of Denmark*, vol. 7, no. 15, p. 510, 2008.

[68] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, vol. EC-14, no. 3, pp. 326–334, June 1965.

[69] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[70] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[71] J. L. Navarro-Mesa, E. Lleida-Solano, and A. Moreno-Bilbao, "A new method for epoch detection based on the cohen's class of time frequency representations," *IEEE Signal Processing Letters*, vol. 8, no. 8, pp. 225–227, Aug 2001.

[72] P. Xanthopoulos, S. Rebennack, C. C. Liu, J. Zhang, G. L. Holmes, B. M. Uthman, and P. M. Pardalos, "A novel wavelet based algorithm for spike and wave detection in absence epilepsy," in *2010 IEEE International Conference on BioInformatics and BioEngineering*, May 2010, pp. 14–19.

[73] P. Lax, *Linear Algebra and Its Applications*, ser. Linear algebra and its applications. Wiley, 2007, no. v. 10.

[74] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, ser. McGraw-Hill Series in Electrical Engineering. McGraw-Hill, 1984.

BIOGRAPHICAL STATEMENT

Akshay Malhotra received his B.Eng in Electronics and Communication Engineering from the P.E.S Institute of Technology, Bangalore, India in 2011 and his M.Sc in Electrical Engineering from the University of Texas at Arlington, TX, USA. From July 2011 to December 2013 he worked at Ittiam Systems, Bangalore, India and at Cirrus Logic, Phoenix, AZ, USA between January - June 2016. Since August 2016, he has been working towards his Ph.D in the Department of Electrical Engineering at the University of Texas at Arlington, TX, USA. His research focuses on machine learning, non-convex optimization and signal processing.