2014 22nd International Conference on Pattern Recognition

Multiple Kernel Learning Based Multi-view Spectral Clustering

Dongyan Guo*†, Jian Zhang†, Xinwang Liu‡, Ying Cui*†, Chunxia Zhao*
*School of Computer Science and Engineering, Nanjing University of Science and Technology, China
†Advanced Analytics Institute & School of Software, University of Technology, Sydney, Australia
‡ School of Computer, National University of Defense Technology, China

Abstract—**For a given data set, exploring their multi-view instances under a clustering framework is a practical way to boost the clustering performance. This is because that each view might reflect partial information for the existing data. Furthermore, due to the noise and other impact factors, exploring these instances from different views will enhance the mining of the real structure and feature information within the data set. In this paper, we propose a multiple kernel spectral clustering algorithm through the multi-view instances on the given data set. By combining the kernel matrix learning and the spectral clustering optimization into one process framework, the algorithm can determine the kernel weights and cluster the multi-view data simultaneously. We compare the proposed algorithm with some recent published methods on real-world datasets to show the efficiency of the proposed algorithm.**

I. Introduction

Multi-view data is common in wide applications including image processing, computer vision, internet webpage processing and natural language processing. In video surveillance applications, an object or a scene can be captured from multi-views with different angles. For a color image, it can be viewed from different modalities such as color texture and shape. In a multimedia application, synchronize human speech and their lips is a typical multi-view data problem. In proteomics, protein folding is driven by various factors, such as physico-chemical properties, geometrical and evolutionary constraints. Different views form different feature spaces, which have particular statistical properties, will be possible to enhance the object classification and feature clustering.

The Multi-view spectral clustering (MVSC) has been paid great attention in machine learning research because of its effectiveness and reinforcement on single-view clustering methods. In this paper, we propose a new algorithm in the space of MVSC through an unsupervised multi-kennel learning. Specifically, we integrate and formulate the kernel matrix learning and special clustering into an optimization framework.

Many references on multi-view clustering methods have been published [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. One of their weak points is that, sometimes, it may lead to performance degradation in the case where the information on some views is corrupted due to the noise or irrelevant. This has generated a certain gap between the motivations on expectation of using the information from multiple views and the real performance. One typical example is for multimedia data where the image tags might have relatively accurate information for image objects (e.g. the Sydney harbour bridge, the New York Time Square and the Great Wall) while the corresponding image features such as color, texture and local key interest points are low level which have the gap to contribute to the semantic level of object information. Therefore, to efficiently combine the information from these different views, it involves how to weight these views. Inspired by the Kernel-based weighted multi-view clustering [14], [15], we adopt spectral clustering method through a kernel matrix learning. For each single view, different kernel functions such as Gaussian and polynomial kernels can be considered to build a single view kernel matrix. Theoretically, an efficient kernel matrix would be learned based on the information for each cluster in the data set if the clusters are available. While the cluster information could also be obtained through a clustering process based on the learned kernel matrix if the kernel matrix are generated. Based on this point of view, the learning and clustering processes are correlated and depend on each other. To address such important issue, we propose to combine the above two processes: learning the kernel matrix and optimizing the spectral clustering into a common framework. Our experiments have demonstrate that the proposed algorithm has achieved the best performance of clustering on major public datasets.

This paper is structured as follows. In section 2, a short review on related works is provided. In section 3, the proposed Multiple Kernel Spectral Clustering (MKSC) and the solution are presented. Experimental results are shown in section 4, and concludes is in section 5.

II. Related work

In recent publications, many clustering methods have been proposed to learn the data from multi-views of the data. In general, some typical methods can be divided into two major steps. At the first step a set of combined features from the multi-views are obtained and then at the second step a popular clustering method such as k-Means are applied on these features. The Canonical Correlation Analysis (CCA) [1], [2] based approach is a typical example of this approach. Alternatively, [3] formulats the multiple views of the data as part of the clustering algorithm. A Co-EM based framework for multi-view clustering in mixture models is proposed. As the name of the Co-EM, it computes expected values of hidden variables in one view with given parameters in the E-step and uses these hidden values in the M-step (Maximize the likelihood) for finding the model parameters in the other view , and vice versa. This process is repeated until a suitable stopping criterion is met. However, the algorithm cannot guarantee its convergence.

[4], [5] both propose a spectral clustering framework for

the multi-view clustering task. In [4], a graph cut is employed to achieve the best average cut over the multiple graphs built by multi-view for the clustering. However, this might not be the best for each single graph (single-view feature). To overcome this problem, a random walk based formulation is proposed. The algorithm allows users to set up weights for each view. In [5], a bipartite graph from nodes of both views is built to address the two-view clustering problem. In their method, the edges of the bipartite graph connect nodes from one view to those in the other view. A standard spectral clustering method is applied to this bipartite graph. In [6], a fusion process known as Linked Matrix Factorization is applied to fuse the information from those multiple graphs generated by the multi-views. In [7], a consensus clustering method is proposed to handle the multi-view clustering. However, such method does not generally work with original features. Instead, for a given dataset, they take different clustering results which come from different views as the input. A reconciliation process is then applied on the input to find a final clustering. In [8], a multi-view convex mixture model is proposed that extends convex mixture models to the multi-view setting. In [9], two co-regularization schemes are proposed for multi-view spectral clustering. The method enforces the possibility of the clustering hypotheses on different views if they can agree with each other. An objective function based on the graph Laplacians from all views is proposed. They apply the regularizations on the eigenvactors of the Laplacians which results in a possible consistence on the clustering structures. However, their method cannot determine the view weights automatically and is inefficient on computation. In [11], a method takes the views which are independently clustered and a final partitioning of the data is derived by minimizing an objective function. Based on all views, the object function measures how close to the final cluster of each single view with the help of a mapping function. In a similar approach, instead of using the mapping functions, [12] adopts a matrix factorization to reconcile the groups arising from the individual views.

In [14], it models each view through a convex mixture distribution and a weighted combination. The weights are learned through an EM method. The weighted combination reflects the view‘s importance. They further explore the weight impacts in [15] by introducing weights to the views which are automatically learned by minimizing the intra-cluster variance.

Overall, despite the wide variety of multi-view clustering methods in many references, most of them treat equally with all views, regardless of the conveyed information (except [4],[14] and [15]).

In this paper, we propose a systemic approach to automatically select weights for learning the kernel matrix on each view through an optimization process in spectral clustering. In our work, the kernel matrix learning is based on the kernel alignment to measure the similarity between two kernel matrices. As a starting point, we give a short review on spectral clustering for a single-view data followed by the discussion of kernel alignment.

A. Spectral Clustering

Spectral clustering technique is to exploit properties of the graph in which edges denote the similarities between data points. The spectrum referred as top K eigenvectors of the kernel matrix are approximation of the indicator matrix Y that assigns each node in the graph to one of the K clusters.

Suppose a given dataset $X = \{x_1, x_2, \dots, x_n\}$ that needs to be clustered into K subsets.

After mapping the data to a higher dimensional feature space (Hilbert space) via a nonlinear transformation $\phi : X \to H$, we can minimize the intra-cluster variance ε_H in the feature space:

$$\varepsilon_H = \sum_{i=1}^{n}\sum_{j=1}^{k} \delta_{ij}\|\phi(x_i) - m_j\|^2, \quad m_j = \frac{\sum_{i=1}^{n}\delta_{ij}\phi(x_i)}{\sum_{i=1}^{n}\delta_{ij}}. \quad (1)$$

Where δ_{ij} is an indicator variable with $\delta_{ij} = 1$ if x_i belongs to the j-th cluster and 0 otherwise. m_j is the j-th cluster center.

According to [16], equation (1) can be equivalently posed as a trace difference:

$$\varepsilon_H = tr(K) - tr(Y^T K Y). \quad (2)$$

where $Y \in \mathbb{R}^{n\times k}$ is an indicator matrix with $Y_{ij} = \frac{\delta_{ij}}{\sqrt{\sum_{l=1}^{n}\delta_{lj}}}$, and K is a kernel matrix with $K_{ij} = \phi(x_i)^T\phi(x_j)$.

However, the discrete nature of Y results in a hard optimization problem on Eq (2). One solution is to relax Y to be an arbitrary orthonormal matrix (i.e. $U^T U = I$). Based on linear algebra, the optimal Y can be composed of the top k eigenvectors of the kernel matrix K. Hence, the Eq. (2) is relaxed to solve the following optimization problem:

$$\max_{U\in\mathbb{R}^{n\times k}} tr(U^T K U), \qquad s.t. U^T U = I. \quad (3)$$

The rows of matrix U are the embeddings of the data points that can be given to the k-means algorithm to obtain the final cluster memberships.

B. Optimizing Kernel Alignment

To approximate a kernel matrix with an ideal target matrix $Y^T Y$, where Y is an indicator matrix in (2), we need to consider how to measure the similarity of two matrices. In this paper, we adopt kernel alignment [17] which measures the similarity between two kernel matrices K_1 and K_2:

$$M(K_1, K_2) = \frac{< K_1, K_2 >_F}{\sqrt{< K_1, K_1 >_F < K_2, K_2 >_F}}. \quad (4)$$

Where $< K_i, K_j >_F = tr(K_i K_j^T)$ and $tr(A)$ is the trace of matrix A.

Therefore, based on Eq (4), the similarity measure between kernel 1 (K_1) and the ideal target kernel 2 ($K_2 = Y^T Y$) is defined as follows:

$$M(K_1, K_2) = \frac{< K_1, YY^T >_F}{\sqrt{< K_1, K_1 >_F < YY^T, YY^T >_F}}. \quad (5)$$

III. Multiple Kernel Spectral Clustering Algorithm

In our work, we represent multi-view data with multiple kernel matrices. The key contribution of this paper is to find a set of optimized weights for these multiple kernels to form a better multi-kernel combination (a linear combination in our work) in the purpose of well representing those multi-view features. We apply spectral clustering on the combined kernels . By employing the kernel alignment, we construct our objective functions which can efficiently formulate the kernel matrix learning and spectral clustering into a unified framework. An alternating optimization method is applied to solve the objective function. In the following we will describe our algorithms in details.

A. Model Description

Due to the different scales on different kernel matrices, we need to normalize them first and then combine them linearly. That is, our kernel function has the following form:

$$K_\mu = \sum_{p=1}^{m} \mu_p D_p^{-1/2} K_p D_p^{-1/2}, \quad \mu \in \triangle. \tag{6}$$

Where $\triangle = \{\|\mu\|_1 = 1, \mu \succeq 0\}$ and D is a diagonal matrix with $D_{ii} = \sum_j K_{ij}$.

To obtain an optimize kernel matrix function in Eq (6), we can get μ by maximizing the alignment between the K_μ and YY^T where Y is an indicator matrix as discussed in section II:

$$\max_{\mu \in \triangle} \frac{< K_\mu, YY^T >_F}{\|K_\mu\|_F \|YY^T\|_F}. \tag{7}$$

Note that, in Eq (7), if we set $v = k\mu, k > 0$,then:

$$\frac{< K_V, YY^T >_F}{\|K_V\|_F \|YY^T\|_F} = \frac{< kK_\mu, YY^T >_F}{\|kK_\mu\|_F \|YY^T\|_F} = \frac{< K_\mu, YY^T >_F}{\|K_\mu\|_F \|YY^T\|_F}.$$

So, Eq (7) can be transformed into the following optimization function:

$$\max_{v \succeq 0} \frac{< K_v, YY^T >_F}{\|K_v\|_F \|YY^T\|_F}. \tag{8}$$

and get μ by $\mu = v/\|v\|_1$

Since there is no label data in the process of clustering algorithms, we can use the orthonormal matrix U to approximate indicator matrix Y (see Eq. (3)). Hence, by replacing Y with U, Eq. (8) can be re-written as Eq. (9):

$$\max_{v \succeq 0} \frac{tr(K_v UU^T)}{\|K_v\|_F \|UU^T\|_F}. \tag{9}$$

According to the invariance of the trace under cyclic permutations, we have:

$\|UU^T\|_F = \sqrt{tr(UU^TUU^T)} = \sqrt{tr(U(U^TU)U^T)} = \sqrt{tr(UU^T)} = \sqrt{tr(U^TU)} = \sqrt{tr(I)} = \sqrt{k}.$

Since the optimization objective function (9) is not less than zero, it is equivalent to the following equation:

$$\max_{v \succeq 0} \frac{tr^2(U^T K_v U)}{\|K_v\|_F^2}. \tag{10}$$

In (10), for a given value v, $\|K_v\|_F^2$ is constant. It can be seen that Eq (3) and Eq (10) are the same form of function expression but for different goals of optimization on U and v respectively. By combining the optimization problems of (3) and (10), we can finally derive our objective function as (11), which integrates the kernel matrix learning and kernel based spectral clustering into one function form:

$$\max_{U \in \mathbb{R}^{n \times k}, v \succeq 0} \frac{tr^2(U^T K_v U)}{\|K_v\|_F^2} - \lambda \|v\|_2^2, \quad s.t. U^T U = I. \tag{11}$$

In the above formula, we add regularization term $\lambda\|v\|_2^2$ to prevent over-fitting U caused by v. To facilitate the calculation, we add the regularization term in the denominator to get a revised objective function Eq (12).

$$\max_{U \in \mathbb{R}^{n \times k}, v \succeq 0} \frac{tr^2(U^T K_v U)}{\|K_v\|_F^2 + \lambda \|v\|_2^2}, \quad s.t. U^T U = I. \tag{12}$$

At this stage, we cannot point out yet whether there is any different consequence for the results due to the variations on regulation terms in (11) and (12). However, it is demonstrated in our experiments that the regulation term in (12) can greatly improve the accuracy of data clustering.

B. Alternating Optimization

The optimization problem in Eq (12) is a nonlinearly constrained nonconvex optimization problem. To the best of our knowledge, it is complex and difficult to find its global optimal solution in a direct way.

To solve the problem, we set an initial value for v, for example $v = [1/m, \cdots, 1/m]^T$ and get U by optimization (3). We then updata v by optimizing (12) with the result U. Note that (3) is only to drive the eigenvectors while problem (12) is not that easy.

By plugging equation (6) into (12) with a given U, it transforms into:

$$\max_{v \succeq 0} \frac{v^T a a^T v}{v^T (M + \lambda I) v}. \tag{13}$$

where $M_{ij} = tr(D_i^{-1/2} K_i D_i^{-1/2} D_j^{-1/2} K_j^T D_j^{-1/2}))$ and $a = [tr(U^T D_1^{-1/2} K_1 D_1^{-1/2} U), \cdots, tr(U^T D_m^{-1/2} K_m D_m^{-1/2} U)]^T$.

Cortes [20] has proved that problem (13) can be transformed into the following Quadratic Program:

$$\min_{v \succeq 0} v^T (M + \lambda I) v - 2v^T a. \tag{14}$$

Note that, Eq (14) is equivalent to the following question:

$$\min_{v \succeq 0} v^T M v - 2v^T a + \lambda \|v\|_2^2. \tag{15}$$

The term $\lambda\|v\|_2^2$ is the same form in Ridge Regression [21] to prevent overfitting. Further to refer [18], when optimizing the kernel alignment, kandola et al. did adopt this method as well. Therefore, it is a strong evidence to explain the rationality of constraining v in (12).

Based on the descriptions, an alternating optimization procedure is formed as shown in Algorithm 1, namely MKSC to obtain an optimal solution on (12) locally.

It is hard to prove the convergence of our MKSC algorithm. However, it can be seen that the objective function $\frac{tr^2(U^T K_v U)}{\|K_v\|_F^2+\lambda\|v\|_2^2}$ increases with the increasing of the iteration numbers. In particular, with a fixed v, the optimal U will increase the value of the objective function; verse vice with a fixed U, the optimal v will also increase the value of the objective function as well.

Algorithm 1 MKSC Algorithm

```
Input:
    Multiple Kernels for different views: $K_1, K_2, \cdots, K_m$,
    The clustering number $k$, and regularization parameter $\lambda$.
Output:
    Clustering label for samples.
1:  Initialization: $\hat{K}_p = D_p^{-1/2} K_p D_p^{-1/2} \quad p = 1, 2, \cdots, m$;
    $v = [1/m, \cdots, 1/m]^T$;  $M_{ij} = tr(\hat{K}_i \hat{K}_j^T)$.
2:  Repeat
3:      $\hat{K}_v = \sum_{p=1}^{m} v_p \hat{K}_p$.
4:      $U = \max_{U \in \mathbb{R}^{n \times k}} tr(U^T \hat{K}_v U) \quad s.t. U^T U = I$.
5:      for given $U$, get $v$ by solving Quadratic Program (14).
6:      $v = v/\|v\|_1$
7:  Until convergence
8:  Form the matrix $V$ by normalize each row of $U$ ($V_{ij} = U_{ij}/(\sum_j U_{ij}^2)^{1/2}$).
9:  Cluster each row of V into $k$ clusters via K-means algorithm.
10: Assign data $x_i$ to cluster $c$ if and only if the row $i$ of the matrix V was assigned to cluster $c$.
```

C. Time-Complexity Analysis

There are three parts form the time complexity of MKSC. The first is the initialization part (line 1). The time complexity of this part is $O(n \times m)^2$ where n is the number of samples and m is the number of kernel matrices as mentioned in section III. The second part (line 2 - line 7) is for the alternating optimization. The update of U has the time complexity of the eigenvalue decomposition of an $n \times n$ matrix. It is $O(n^3)$. The update of v has the time complexity of $O(m^c + (m+k) \times n^2)$ where c is constant and k is the clustering number. The third part (line 8 - line 10) is $k-means$ clustering. The time complexity of this part is $O(nk^2)$. Therefore, the entire time complexity of MKSC is $O((m^c+n^3+(m+k)\times n^2)\times T+(n\times m)^2+nk^2)$, where T is the number of training iterations and T is around five (always less than eight) in all experiments. Note that the kernel matrix number m and clustering number k are far less than samples number n. So the computation cost focus on the eigenvalue decomposition in the second part (line 4). When there are many kernel matrices, calculate matrix M in the first part (line 1) will also need more computation cost.

IV. Experiment

We have compared our multiple kernel spectral clustering algorithm with a number of baseline methods. In particular, they are:

- **Single View:** As mentioned in [10], we apply the spectral clustering on every view of the data. The most informative view can be achieved by the best spectral clustering performance.
- **Feature Concatenation:** As mentioned in [10], we concatenate the features from each view, and then apply the spectral clustering on all these features using the graph Laplacian derived from the join view representation (View1, View2,) of the data.
- **Kernel Addition:** As mentioned in [10], we combine different kernels by adding them and apply spectral clustering on the combined kernel. Cortes [19] reported that this looks to be a very simple method, but it can often achieve very good or near optimal clustering results, even better than many other complex algorithms. [10] indicated that when using a linear kernel, under normal circumstances, kernel addition in Reproducing Kernel Hilbert Space and the feature concatenation are equivalent.
- **Kernel Product (element-wise):** As mentioned in [10], we apply the dot operation on all corresponding elements of the kernel matrices and then apply the standard spectral clustering on the resultant Laplacian. As a special case where all the kernel matrices are constructed by the Gaussian kernel with the same width σ, the kernel product and feature concatenation will have the same kernel element-wise product. However, if there are different σ parameters for different views, the kernel product and feature concatenation results are different.

We also compared with some state-of-the-art algorithms, such as: Co-trained spectral[10], Co-regularized[9] and MVSpec[15]. In the next, we will give a detailed description on the real-world datasets, experiment setting and results analysis.

A. Real-world datasets

The experimental results are reported on three real-world datasets. A brief description of each dataset is given as follows:

- **Protein Fold Prediction data:** Our first dataset [22] is a multi-kernel learning set on protein fold prediction. This dataset contains 694 labeled instances with 27 classes. Each instance is described with 12 different feature representations. Referring to [23], we construct second order polynomial kernels for feature sets 1 to 10 and inner product kernels for sets 11 and 12.
- **Pendigits data:** Our second real-world dataset [24] is from pen-based digit (0-9) recognition [25] from the UCI machine learning repository. This dataset contains four different feature representations. We construct gaussian kernel, ploynomial kernel and ployplus kernel for each feature representation. The experiment was done on all the constructed kernels.
- **YouTube Multiview Video Games data:** Our last dataset [26] is also taken from UCI machine learning repository [27]. This dataset contains feature values and Corresponded labels for about 120K videos(instances). Each instance has 13 feature types from 3 different high level feature families: textual,

TABLE I. CLUSTER PERFORMANCE ON PROTEIN FOLD PREDICTION DATA. NUMBERS IN PARENTHESES ARE THE STD. DEVIATIONS.

Method	F-score	Precision	Recall	NMI	Adj-RI	Entropy
Best Single View	0.2322 (0.0336)	0.2480 (0.0142)	0.2213 (0.1084)	0.5172 (0.0308)	0.1962 (0.0252)	2.3045 (0.0601)
Kernel Addition	0.1934 (0.0173)	0.2236 (0.0195)	0.1708 (0.0156)	0.4898 (0.0136)	0.1579 (0.0180)	2.4277 (0.0484)
Kernel Product	0.0777 (0.0102)	0.0526 (0.0065)	**0.2854 (0.2074)**	0.2294 (0.0423)	0.0043 (0.0077)	3.7300 (0.2403)
Co-trained	**0.2467 (0.0177)**	**0.2787 (0.0193)**	0.2221 (0.0163)	**0.5340 (0.0149)**	**0.2127 (0.0189)**	**2.2578 (0.0629)**
Co-regularized	0.2019 (0.0192)	0.2334 (0.0228)	0.1780 (0.0167)	0.4992 (0.0146)	0.1668 (0.0201)	2.3965 (0.0528)
MVSpec	0.2175 (0.0174)	0.2424 (0.0189)	0.1995 (0.0165)	0.5200 (0.0147)	0.1816 (0.0181)	2.2919 (0.0600)
MKSC	**0.2375 (0.0152)**	**0.2548 (0.0185)**	**0.2259 (0.0140)**	**0.5348 (0.0131)**	**0.2010 (0.0160)**	**2.2513 (0.0705)**

TABLE II. CLUSTER PERFORMANCE ON PENDIGITS DATA. NUMBERS IN PARENTHESES ARE THE STD. DEVIATIONS.

Method	F-score	Precision	Recall	NMI	Adj-RI	Entropy
Best Single View	0.6359 (0.0130)	0.6254 (0.0139)	0.6489 (0.0153)	0.7190 (0.0087)	0.5947 (0.0145)	1.1263 (0.0368)
Feature Concatenation	0.5774 (0.0326)	0.5700 (0.0350)	0.5867 (0.0302)	0.6253 (0.0241)	0.5296 (0.0366)	1.4641 (0.0799)
Kernel Addition	0.7125 (0.0254)	**0.7034 (0.0256)**	0.7243 (0.0228)	0.7756 (0.0176)	0.6801 (0.0283)	1.1461 (0.0731)
Kernel Product	0.5237 (0.0310)	0.4766 (0.0382)	0.5968 (0.0243)	0.5947 (0.0247)	0.4646 (0.0361)	2.7886 (0.0811)
Co-trained	0.6820 (0.0204)	0.6734 (0.0210)	0.7021 (0.0204)	0.7308 (0.0154)	0.6440 (0.0228)	1.2427 (0.0424)
Co-regularized	0.6822 (0.0341)	0.6699 (0.0362)	0.6990 (0.0307)	0.7451 (0.0225)	0.6461 (0.0381)	1.2291 (0.1052)
MVSpec	**0.7190 (0.0226)**	0.7032 (0.0230)	**0.7404 (0.0254)**	**0.7852 (0.0184)**	**0.6870 (0.0251)**	**1.0630 (0.0597)**
MKSC	**0.7346 (0.0312)**	**0.7061 (0.0400)**	**0.7750 (0.0202)**	**0.8079 (0.0188)**	**0.7036 (0.0353)**	**1.0410 (0.0616)**

TABLE III. CLUSTER PERFORMANCE ON YOUTUBE MULTIVIEW VIDEO GAMES DATA. NUMBERS IN PARENTHESES ARE THE STD. DEVIATIONS.

Method	F-score	Precision	Recall	NMI	Adj-RI	Entropy
Best Single View	**0.2769 (0.0294)**	**0.4060 (0.0373)**	**0.2191 (0.0331)**	**0.6834 (0.0188)**	**0.2315 (0.0324)**	**1.4528 (0.0857)**
Feature Concatenation	0.1540 (0.0178)	0.1036 (0.0148)	**0.3543 (0.00644)**	0.3006 (0.0295)	0.0242 (0.0171)	3.4494 (0.1426)
Kernel Addition	0.0987 (0.0094)	0.1764 (0.0167)	0.0687 (0.0075)	0.4306 (0.0158)	0.0542 (0.0105)	2.5196 (0.0761)
Kernel Product	0.1123 (0.0333)	0.0997 (0.0097)	0.1917 (0.1694)	0.2778 (0.0369)	0.0184 (0.0086)	3.3461 (0.2362)
Co-trained	0.1647 (0.0204)	0.2790 (0.0316)	0.1179 (0.0161)	0.5335 (0.0245)	0.1209 (0.0224)	2.1265 (0.1125)
Co-regularized	0.1174 (0.0138)	0.2002 (0.0213)	0.0840 (0.0112)	0.4624 (0.0240)	0.0713 (0.0156)	2.3884 (0.0901)
MVSpec	0.1895 (0.0283)	0.2807 (0.0375)	0.1477 (0.0239)	0.5591 (0.0333)	0.1397 (0.0321)	2.0154 (0.1303)
MKSC	**0.2548 (0.0350)**	**0.3780 (0.0477)**	0.2006 (0.0312)	**0.6510 (0.0371)**	**0.2086 (0.0396)**	**1.6618 (0.1578)**

visual, and auditory features. There are 31 class labels. The first 30 labels correspond to different video games. Class 31 is not specific, and means none of the 30. We randomly selected 100 samples from each class. Like in Pendigits data, we also construct gaussian kernel, ploynomial kernel and ployplus kernel for each feature of the samples. The experiments were run on all the constructed kernels.

B. Experiment setting.

In our experiments, three different kernel functions (Gaussian kernel, polynomial kernel and ployplus kernel) were used to construct kernel matrix. The standard deviation σ of kernel is assigned equal to the average of the Euclidean distances based on all paired data points[10]. ploynomial kernel and ployplus kernel function are set to be $(x' * y)^2$ and $(x' * y + 1)^2$.

To obtain comprehensive experimental results, we ran 30 times on each dataset by randomly selecting $2/3$ of the experimental data. We took the mean and standard deviation as the final results.

We used different measures to evaluate the performance of all the methods, including F-score, precision, recall, normalized mutual information (NMI), adjusted rand index and average entropy. By following [10], the higher value indicates better clustering quality, except for average cluster entropy where the lower value signifies better clustering quality. For a comprehensive evaluation, the measures penalize or favor different properties in the clustering. Table I, II, III shows the results on these diverse measures. For each measure, the first two best results are show in bold.

In our experiments, we observed that there is a correlation between the λ and $tr(M)$. Hence, we set up empirical value for λ in different datasets. We set $\lambda = 2 \times tr(M)$ for all of the three data sets.

C. Results analysis.

Table I refers to the results of Protein fold prediction data . Since this dataset contains string features which dimensions are variablewe did not run the experiments on Feature concatenation comparison. It can be seen that our algorithm (MKSC) outperforms all the baselines by a significant advantage except the co-train method. The result of co-train method is close to our MKSC algorithm. Note that the computation cost of co-train method is much more expensive than ours.

Table II shows the results on Pendigits dataset. On this dataset, Our algorithm has achieved the best performance. The kernel addition and MVSpec method are close to our MKSC approach. Note that these three methods are all linear model method to form the target kernel matrix.

As shown in Table IIIDue to the database feature, the text features identifies the most accurate information while the video and audio features contain more noise. For this database, the best single view generated the best result. Nevertheless, even such phenomenon that imbalanced noise nature of the multi-view data, the results of our algorithm compared to others, the advance is still obvious.

V. Conclusion

In this paper, we propose a multiple kernel learning based algorithm for multi-view data clustering. In our algorithm, each view is represented by one or more kernel matrices. To efficiently make use of the information from each view, we use a linear model to combine all the kernel matrices to form the target kernel matrix. By combining the kernel matrix learning and the spectral clustering optimization into one process framework, our algorithm determine the kernel weights and cluster the multi-view data simultaneously. The experimental results on 3 different real-world datasets have shown the effectiveness of our algorithm.

Acknowledgment

This work is supported in part by the NSFC under Grant 61101197, 61272220, in part by the NSF of Jiangsu Province under Grant BK2012399.

References

[1] M.B. Blaschko and C.H. Lampert, Correlational spectral clustering. In Computer Vision and Pattern Recognition,IEEE Conference on (pp. 1-8). IEEE, 2008.

[2] K. Chaudhuri, S.M. Kakade, K. Livescu and K. Sridharan, (2009, June). Multi-view clustering via canonical correlation analysis. In Proceedings of the 26th annual international conference on machine learning (pp. 129-136). ACM.

[3] S. Bickel and T.Scheffer, Multi-View Clustering. In ICDM (Vol. 4, pp. 19-26), 2004(November)

[4] D. Zhou and C.J. Burges, Spectral clustering and transductive learning with multiple views. In Proceedings of the 24th international conference on Machine learning (pp. 1159-1166), ACM, 2007.

[5] V.R. de Sa, Spectral clustering with two views. In ICML workshop on learning with multiple views, 2005(October).

[6] W. Tang, Z. Lu and I.S.Dhillon, Clustering with multiple graphs. In Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on (pp. 1016-1021). IEEE, 2009(December).

[7] A. Strehl and J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions. The Journal of Machine Learning Research, 3, 583-617, 2003.

[8] G. Tzortzis and A. Likas, Convex mixture models for multi-view clustering. In Artificial Neural Networks, Springer Berlin Heidelberg (pp. 205-214), 2009.

[9] A. Kumar, P. Rai and H.D. Iii, Co-regularized multi-view spectral clustering. In Advances in Neural Information Processing Systems (pp. 1413-1421), 2011.

[10] A. Kumar and H.D. Iii, A co-training approach for multi-view spectral clustering. In Proceedings of the 28th International Conference on Machine Learning (ICML-11) (pp. 393-400), 2011.

[11] B. Long, S.Y. Philip and Zhongfei (Mark) Zhang, A General Model for Multiple View Unsupervised Learning. In SDM (pp. 822-833), 2008.

[12] D. Greene and P. Cunningham, A matrix factorization approach for integrating multiple data views. In Machine Learning and Knowledge Discovery in Databases (pp. 423-438). Springer Berlin Heidelberg, 2009.

[13] T. Xia, D. Tao, T. Mei and Y. Zhang, Multiview spectral embedding. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 40(6), 1438-1446. 2010

[14] G.F. Tzortzis and C.L. Likas,. Multiple view clustering using a weighted combination of exemplar-based mixture models. Neural Networks, IEEE Transactions on, 21(12), 1925-1938, 2010.

[15] G. Tzortzis and A. Likas, (2012, December). Kernel-based Weighted Multi-view Clustering. In Data Mining (ICDM), 2012 IEEE 12th International Conference on (pp. 675-684). IEEE.

[16] I.S. Dhillon, Y. Guan and B. Kulis, Weighted graph cuts without eigenvectors a multilevel approach. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(11), 1944-1957, 2007

[17] N. Shawe-Taylor and A. Kandola, On kernel target alignment. Advances in neural information processing systems, 14, 367, 2002

[18] J. Kandola, J. Shawe-Taylor and N. Cristianini, Optimizing kernel alignment over combinations of kernel, 2002.

[19] C. Cortes, M. Mohri and A. Rostamizadeh, Learning non-linear combinations of kernels. In Advances in Neural Information Processing Systems (pp. 396-404), 2009.

[20] C. Cortes, M. Mohri and A. Rostamizadeh, Algorithms for learning kernels based on centered alignment. The Journal of Machine Learning Research, 13, 795-828, 2012

[21] A.E. Hoerl and R.W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12(1), 55-67, 1970.

[22] The UCSD Multiple Kernel Learning Repository: Protein Fold Prediction dataset. (http://mkl.ucsd.edu/dataset/protein-fold-prediction)

[23] T. Damoulas and M.A. Girolami, Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection.Bioinformatics, 24(10), 1264-1270, 2008.

[24] The UCSD Multiple Kernel Learning Repository: Pendigits dataset. (http://mkl.ucsd.edu/dataset/pendigits).

[25] F. Alimoglu and E. Alpaydin, Combining multiple representations and classifiers for pen-based handwritten digit recognition. InDocument Analysis and Recognition, Proceedings of the Fourth International Conference on (Vol. 2, pp. 637-640). IEEE, 1997(August).

[26] Machine Learning Repository: YouTube Multiview Video Games. (http://archive.ics.uci.edu/ml/machine-learning-databases/00269/).

[27] O. Madani, M. Georg and D. Ross, On using nearly-independent feature families for high precision and confidence. Machine Learning, 1-21, 2012.