A Co-training Approach for Multi-view Spectral Clustering

Abhishek Kumar ABHISHEK@CS.UMD.EDU
Hal Daumé III HAL@UMIACS.UMD.EDU
Department of Computer Science, University of Maryland, College Park, MD 20742, USA

Abstract

We propose a spectral clustering algorithm for the multi-view setting where we have access to multiple views of the data, each of which can be independently used for clustering. Our spectral clustering algorithm has a flavor of *co-training*, which is already a widely used idea in semi-supervised learning. We work on the assumption that the true underlying clustering would assign a point to the same cluster irrespective of the view. Hence, we constrain our approach to only search for the clusterings that *agree* across the views. Our algorithm does not have any hyperparameters to set, which is a major advantage in unsupervised learning. We empirically compare with a number of baseline methods on synthetic and real-world datasets to show the efficacy of the proposed algorithm.

1. Introduction

Unlabeled data is plentiful, and increasing quantities of it come in multiple views from diverse sources (Blum & Mitchell, 1998; Chaudhuri et al., 2009). Our goal is to derive clustering algorithms that can leverage these multiple views. The central idea to our work is that the clustering from one view should *agree* with the clustering from another view. We present a mathematically clean extension of standard spectral clustering approaches (Shi & Malik, 2000) to multiple views based on the co-training idea (Blum & Mitchell, 1998). Our approach is based on the assumption that the true underlying clustering would assign corresponding points in each view to the same cluster.

Multi-view data is common in a wide variety of application domains. In natural language tasks, we can have a document or a corpus available in multiple languages (Amini et al., 2009). Internet webpages can be represented as page-text as well as the hyperlinks pointing to them, giving rise to two views of the entity it represents. In computer vision problems, we can have an object or a scene captured from multiple viewing angles. In automatic speech recognition, we can also have access to the image sequence of lip-movements along with the speech sounds.

In the context of clustering, we seek to partition our data based on a similarity measure between the examples. Spectral clustering algorithms have gained attention in the recent past due to their good performance on arbitrary shaped clusters, and due to their well-defined mathematical framework (von Luxburg, 2007). Spectral clustering operates on a graph that is constructed from the data points as nodes, with edges between them representing the similarities. Because the input to a spectral clustering method is a similarity graph, in the rest of the paper we use the terms *graph* and *view* interchangeably. Our algorithm uses the idea of co-training (Blum & Mitchell, 1998) that was originally introduced (and is still mostly used) in the setting of semi-supervised learning. We take this idea to unsupervised learning setting, specifically in the framework of spectral clustering. We bootstrap the clusterings of different views using information from one another. In particular, we use the spectral embedding from one view to *constrain* the similarity graph used for the other view. By iteratively applying this approach, the clusterings of the two views tend to each other. We evaluate the proposed approach on four real-world datasets against several competitive baselines and observe consistent improvements in performance. The graph Laplacians obtained during the course of the algorithm are low rank, which is an advantage in large scale clustering problems. Moreover, our algorithm has *no* hyperparameters to set, which is especially encouraging in an unsupervised setting.

2. Spectral Clustering

Spectral clustering is a technique that exploits the properties of the Laplacian of the graph, whose edges denote the similarities between the data points. The

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

top k eigenvectors of the normalized graph Laplacian are relaxations of the indicator vectors that assign each node in the graph to one of the k clusters. Apart from being theoretically well-motivated, spectral clustering has the advantage of performing well on arbitrary shaped clusters, which is otherwise a shortcoming with several other clustering algorithms such as the k-means algorithm. Here we briefly outline the spectral clustering algorithm due to (Ng et al., 2002):

- Construct an $n \times n$ positive semi-definite similarity matrix (or kernel) $\mathbf{K}$, where $\mathbf{K}_{ij}$ quantifies the similarity between samples i and j.
- Compute the normalized graph Laplacian $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{K}_{ij}$.
- Let $\mathbf{U}$ denote a $n \times k$ matrix with columns as the top k eigenvectors of $\mathcal{L}$
- Normalize each row of $\mathbf{U}$ to obtain $\mathbf{V}$.
- Run the k-means algorithm to cluster the row vectors of $\mathbf{V}$.
- Assign example i to cluster c if the i-th row of $\mathbf{V}$ is assigned to cluster c by the k-means algorithm.

For a detailed introduction to both theoretical and practical aspects of spectral clustering, the reader is referred to (von Luxburg, 2007).

3. Co-training

We provide a brief overview of co-training in this section. It was originally proposed for the problem of semi-supervised learning, where we have access to labeled as well as unlabeled data (Blum & Mitchell, 1998). It considers a setting in which each example can be partitioned into two distinct views, and makes three main assumptions for its success: (a) *Sufficiency:* Each view is sufficient for classification on its own, (b) *Compatibility:* the target functions in both views predict same labels for co-occurring features with high probability, and (c) *Conditional independence:* the views are conditionally independent given the class label.

The central idea of co-training algorithms is to limit the search for target hypothesis to the set of "compatible hypotheses" that predict same labels for co-occurring patterns in each view. Unlabeled data allows us to do this pruning of the hypothesis space. In the original co-training algorithm (Blum & Mitchell, 1998), two initial hypotheses h_1 and h_2 are trained in the individual views using the labeled data. Both hypotheses then label a certain number of unlabeled examples on which they are most confident. These examples are added to the labeled pool, and h_1, h_2 are retrained. This process is repeated for a pre-chosen number of iterations. The intuition behind co-training algorithm is that h_1 adds examples to the labeled set that are used for training h_2, and vice versa. This process should slowly drive h_1 and h_2 to agree with each other on labels.

Variants of the original co-training algorithm were also proposed later and evaluated on different datasets. We specifically mention the *co-EM* algorithm (Nigam & Ghani, 2000). It differs with the original co-training algorithm in a couple of places. Firstly, it is not incremental in nature, i.e., all of unlabeled data is labeled in each iteration for further use. Secondly, only the data labeled by h_2 is used to retrain h_1 (and vice versa), unlike the original co-training algorithm that uses data labeled by both h_1 and h_2 in retraining each of these. Nigam and Ghani (2000) observe that co-EM is a closer match to the theoretical argument of Blum and Mitchel (1998) than the original co-training algorithm.

4. Co-training for Spectral Clustering

In this section, we apply the idea of co-training to the problem of multi-view spectral clustering. There is no labeled data in unsupervised learning problems, so semi-supervised co-training cannot be applied directly. However, the motivation still remains the same as in semi-supervised problems: to limit our search to hypotheses (in our problem, clusterings) that agree with those in other views. Specifically, we want the relationship within a pair of points to be consistent across the views. If two points are assigned in same cluster in one view, it should be so in all the views. On the other hand, if two points belong to different clusters, it should be so in all the views. This is a reasonable approach to take in the light of compatibility assumption of co-training.

We know that the first k eigenvectors of a graph Laplacian with exactly k number of connected components are the component (or cluster) indicator vectors, i.e., each vector is associated to a cluster and has non-zero values only at positions that correspond to points in the cluster. In another words, these eigenvectors only contain discriminative information about the clusters, ignoring the within-cluster details. For a fully connected graph (one connected component), spectral clustering solves a relaxed version of the min-cut problem (normalized or unnormalized). The eigenvectors in this case are not the cluster indicator vectors, yet they still contain discriminative information which is used in spectral clustering. In the multi-view setting, we can make use of eigenvectors obtained from one view to "label" the points in other view, and vice versa. Our proposed multi-view algorithm aims to work along

the lines of Figure 1.

1. Solve spectral clustering on individual graphs to get the discriminative eigenvectors in each view, say $\mathbf{U}_1$ and $\mathbf{U}_2$.
2. Cluster points using $\mathbf{U}_1$ and use this clustering to *modify* the graph structure in view 2.
3. Cluster points using $\mathbf{U}_2$ and use this clustering to *modify* the graph structure in view 1.
4. Go to Step 1 and repeat for a number of iterations.

Figure 1. General framework for co-training based clustering

Now, the question remains of how to *modify* the graph structure using clustering information from the other view. One naïve way could be to reduce the edge-weight of a pair in a graph if its points belong to different clusters according to the other view. Alternatively, we can amplify the edge-weight of a pair if the other view puts it in same cluster. A similar idea could be applied for the other graph. However, this would require us to cluster points at each step, which may not be computationally efficient. In addition, there is also a question of how to decide the amounts or factors by which to reduce the different edge weights.

Instead of completely solving the clustering at each iteration and then "labeling" the other graph, we take an indirect approach that results in extra computational savings, and is also more elegant. For a similarity matrix $\mathbf{K}_{n\times n}$, we can consider each column $\mathbf{k}_i$ of it as an n-dimensional vector that indicates the similarities of ith point with all the points in the graph. The eigenvectors of the graph Laplacian are vectors in the n-dimensional space. Since we know that the first k eigenvectors have the discriminative information for clustering, we can project the similarity vectors along these *directions* to retain the information needed for clustering and *throw away* the within cluster details that might confuse us in clustering. We back-project to the original n-dimensional space to get back the modified graph. Since the projection matrix is orthogonal, the inverse projection can be done using its transpose. This process is equivalent to steps 2 and 3 in Figure 1. Algorithm 1 gives a detailed description of the algorithm. We perform the symmetrization step on $\mathbf{S}_1$ and $\mathbf{S}_2$ in the algorithm since the projection of similarity matrix $\mathbf{K}$ on the eigenvectors does not yield a symmetric matrix. Symmetrization operator on a matrix $\mathbf{S}$ is defined as $sym(\mathbf{S}) = (\mathbf{S} + \mathbf{S}^T)/2$.

To further reinforce the idea of projection along the eigenvectors, let us consider a simple case where the first graph has exactly k components in it, i.e., the

Algorithm 1 Co-trained Multi-view Spectral Clustering

Input:
Similarity matrix for both views: $\mathbf{K}_1$, $\mathbf{K}_2$
Output: Assignments to k clusters
Initialize: $\mathbf{L}_v = \mathbf{D}_v^{-1/2}\mathbf{K}_v\mathbf{D}_v^{-1/2}$ for $v = 1, 2$
$\mathbf{U}_v^0 = \underset{\mathbf{U}\in\mathbb{R}^{n\times k}}{\operatorname{argmax}} tr(\mathbf{U}^T\mathbf{L}_v\mathbf{U})$, s.t. $\mathbf{U}^T\mathbf{U} = I$ for $v = 1, 2$
for $i = 1$ to *iter* **do**
1: $\mathbf{S}_1 = sym\left(\mathbf{U}_2^{i-1}\mathbf{U}_2^{i-1^T}\mathbf{K}_1\right)$
2: $\mathbf{S}_2 = sym\left(\mathbf{U}_1^{i-1}\mathbf{U}_1^{i-1^T}\mathbf{K}_2\right)$
3: Use $\mathbf{S}_1$ and $\mathbf{S}_2$ as the new graph similarities and compute the Laplacians. Solve for the largest k eigenvectors to obtain $\mathbf{U}_1^i$ and $\mathbf{U}_2^i$.
end for
4: Row-normalize $\mathbf{U}_1^i$ and $\mathbf{U}_2^i$.
5: Form matrix $\mathbf{V} = \mathbf{U}_v^i$, where v is believed to be the most informative view a priori. If there is no prior knowledge on the view informativeness, matrix $\mathbf{V}$ can also be set to be column-wise concatenation of the two $\mathbf{U}_v^i$s.
6: Assign example j to cluster c if the j-th row of $\mathbf{V}$ is assigned to cluster c by the k-means algorithm.

weights of across cluster edges are 0. As we know, the Laplacian of this graph would have the top k eigenvectors as the cluster indicator vectors (von Luxburg, 2007). Let us assume that the second view has a fully connected graph as follows:

$$\mathbf{K}_2 = \begin{pmatrix} 0 & b & c & d \\ b & 0 & e & f \\ c & e & 0 & g \\ d & f & g & 0 \end{pmatrix} \tag{1}$$

The self-similarities of all points are assumed to be same and equal to 0, since they do not affect the min-cut solution. Suppose the second graph gives $\mathbf{u}_1^1 = \frac{1}{\sqrt{3}}(1\,1\,1\,0)^T$ and $\mathbf{u}_1^2 = (0\,0\,0\,1)^T$ as the top two eigenvectors. This implies that first 3 points are in one cluster and the fourth point is in the second cluster. Let $\mathbf{U}_1 = \left(\mathbf{u}_1^1\,\mathbf{u}_1^2\right)$. The projection of $\mathbf{K}_2$ onto the subspace spanned by $\mathbf{U}_1$ and the subsequent symmetrization yields the following modified graph in view 2:

$$\frac{1}{3}\begin{pmatrix} b+c & b+(c+e)/2 & c+(b+e)/2 & 2d+(f+g)/2 \\ b+(c+e)/2 & b+e & e+(b+c)/2 & 2f+(d+g)/2 \\ c+(b+e)/2 & e+(b+c)/2 & c+e & 2g+(d+f)/2 \\ 2d+(f+g)/2 & 2f+(d+g)/2 & 2g+(d+f)/2 & 0 \end{pmatrix}$$

Let us pay attention to the shaded sub-matrix in the above matrix. The new weight of edge (i, j) is obtained by averaging out the edges within the cluster.

The across cluster edges are also averaged out in the new graph. This implies that the projection in the subspace of discriminative eigenvectors makes edges within a cluster close to each other, throwing away the intra-cluster information that is irrelevant for clustering. As the number of iterations increase, the edges within a cluster diffuse to one another. The across cluster edges also diffuse to one another. All points within a cluster are treated in a similar way, and different from points in other clusters. In other words, this process "glues" all the points in a cluster, and they tend to appear together in the subsequent spectral clustering solution.

It is possible to extend the proposed co-training framework for more than two views. We can take the similarity matrix $\mathbf{K}_v$ of a view, and project it onto the union of subspaces spanned by top k discriminative eigenvectors of the other views. More formally, steps 1 and 2 in Algorithm 1 are replaced by $\mathbf{S}_v = sym\left(\left(\sum_{i \neq v} \mathbf{U}_i \mathbf{U}_i^T\right) \mathbf{K}_v\right)$ for all the views.

4.1. Computational Efficiency

The projection of the similarity matrix $\mathbf{K}$ using the projection matrix $\mathbf{U}\mathbf{U}^T$ gives a matrix that has a rank of k. After symmetrization, each of the new similarity matrices $\mathbf{S}_1$ and $\mathbf{S}_2$ has a maximum rank of $2k$. Hence, the normalized Laplacian $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$ also has a maximum rank of $2k$. This can be of great advantage in large scale problems, since there exist efficient randomized algorithms for doing SVD if the original matrix is low rank and a good upper bound on the rank is known in advance (Liberty et al., 2007). The matrix $\tilde{\mathbf{S}} = \mathbf{U}\mathbf{U}^T K$ is a diagonalizable matrix and has all real non-negative eigenvalues (refer to Theorem 7.6.3 in (Horn & Johnson)). However, it is not necessary for $sym(\mathbf{S}) = (\tilde{\mathbf{S}} + \tilde{\mathbf{S}}^T)/2$ to have non-negative eigenvalues. The individual entries of $sym(\mathbf{S})$ can also be negative, and so the corresponding Laplacian can be non-positive definite. In our experiments, we add a rank-1 matrix to $sym(\mathbf{S})$ that has all its entries equal to the minimum negative entry of $sym(\mathbf{S})$. This makes sure that the corresponding Laplacian is positive semidefinite at each iteration.

5. Experiments

We compare our co-trained multi-view spectral clustering approach with a number of baselines. In particular, we compare with:

- **Single View:** Using the most informative view, i.e., one that achieves the best spectral clustering performance using a single view of the data.
- **Feature Concatenation:** Concatenating the features of each view, and then running standard spectral clustering using the graph Laplacian derived from the joint view representation of the data.
- **Kernel Addition:** Combining different kernels by adding them, and then running standard spectral clustering on the corresponding Laplacian. As suggested in earlier findings (Cortes et al., 2009), even this seemingly simple approach often leads to near optimal results as compared to more sophisticated approaches for classification. It can be noted that kernel addition reduces to feature concatenation for the special case of linear kernel. In general, kernel addition is same as concatenation of features in the Reproducing Kernel Hilbert Space.
- **Kernel Product (element-wise):** Multiplying the corresponding entries of kernels and applying standard spectral clustering on the resultant Laplacian. For the special case of Gaussian kernel, element-wise kernel product would be same as simple feature concatenation if both kernels use same width parameter σ. However, in our experiments, we use different width parameters for different views so the performances of kernel product may not be directly comparable to feature concatenation.
- **CCA based Feature Extraction:** Applying CCA for feature fusion from multiple views of the data (Blaschko & Lampert, 2008), and then running spectral clustering using these extracted features. We apply both standard CCA and kernel CCA for feature extraction and report the clustering results for whichever gives the best performance on test data.
- **Minimizing-Disagreement Spectral Clustering:** Our last baseline is the *minimizing-disagreement* approach to spectral clustering (de Sa, 2005), and is perhaps most closely related to our co-training based approach to spectral clustering. This algorithm is discussed more in Sec. 6.

We report experimental results on one synthetic and three real-world datasets. We give a brief description of each dataset here.

- **Synthetic data:** Our synthetic data consists of three views and is generated as follows. We first choose the cluster c_i each sample belongs to, and then generate each of the views $x_i^{(1)}$, $x_i^{(2)}$ and $x_i^{(3)}$ from a two-component Gaussian mixture model. These views are combined to form the sample $(x_i^{(1)}, x_i^{(2)}, x_i^{(3)}, c_i)$. We sample 1000 points from each view. The cluster means in view 1 are $\mu_1^{(1)} = (1\ 1)$, $\mu_2^{(1)} = (3\ 4)$; in view 2 are

$\mu_1^{(2)} = (1\ 2)$, $\mu_2^{(2)} = (2\ 2)$; and in view 3 are $\mu_1^{(3)} = (1\ 1)$, $\mu_2^{(3)} = (3\ 3)$. The covariances for the three views are given below. The notation $\Sigma_c^{(v)}$ denotes the parameter for cth cluster in vth view.

$$\Sigma_1^{(1)} = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{pmatrix}, \quad \Sigma_2^{(1)} = \begin{pmatrix} 0.3 & 0.2 \\ 0.2 & 0.6 \end{pmatrix}$$

$$\Sigma_1^{(2)} = \begin{pmatrix} 1 & -0.2 \\ -0.2 & 1 \end{pmatrix}, \quad \Sigma_2^{(2)} = \begin{pmatrix} 0.6 & 0.1 \\ 0.1 & 0.5 \end{pmatrix}$$

$$\Sigma_1^{(3)} = \begin{pmatrix} 1.2 & 0.2 \\ 0.2 & 1 \end{pmatrix}, \quad \Sigma_2^{(3)} = \begin{pmatrix} 1 & 0.4 \\ 0.4 & 0.7 \end{pmatrix}$$

- **Reuters Multilingual data:** The test collection contains feature characteristics of documents originally written in five different languages (English, French, German, Spanish and Italian), and their translations, over a common set of 6 categories (Amini et al., 2009). We use documents originally in English as the first view, and their French and German translations as the second and third views. We randomly sample 1200 documents from this collection in a balanced manner, with each of the 6 clusters having 200 documents. The documents are in bag-of-words representation which implies that the features are extremely sparse and high-dimensional. The standard similarity measures (like Gaussian kernel) in very high dimensions are often unreliable. Since spectral clustering essentially works with similarities of the data, we first project the data using Latent Semantic Analysis (LSA) (Hofmann, 1999) to a 100-dimensional space and compute similarities in this lower dimensional space. This is akin to a computing topic based similarity of documents (Blei et al., 2003).
- **UCI Handwritten digits data:** Our second real-world dataset is taken from the handwritten digits (0-9) data from the UCI repository. The dataset consists of 2000 examples, with view-1 being the 76 Fourier coefficients, and view-2 being the 216 profile correlations of each example image.
- **BBC and BBCSPORTS data:** These datasets consist of news articles from the BBC (Greene & Cunningham, 2005). BBC data contains 2225 complete news articles corresponding to stories in five topical areas (business, entertainment, politics, sport, tech). BBCSPORTS data consists of 737 sports news articles in five classes (athletics, cricket, football, rugby, tennis). These are synthetic multi-view datasets, wherein each document is segmented and segments are randomly assigned to the two views (Greene & Cunningham, 2009).

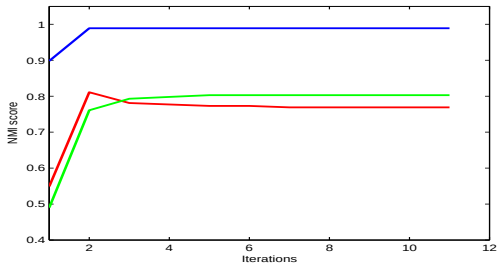

Figure 2. NMI scores in different views vs number of iterations of co-trained spectral clustering for **Synthetic data**

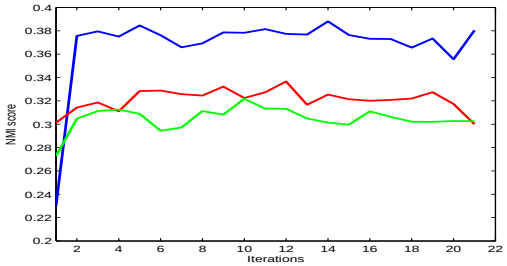

Figure 3. NMI scores in different views vs number of iterations of co-trained spectral clustering for **Reuters multilingual data**

We compare all the approaches on a number of evaluation measures. Here, we report precision, recall, F-score, normalized mutual information (NMI), average entropy, and adjusted rand index (Manning et al., 2008; Hubert & Arabie, 1985). For all these measures, the higher value indicates better clustering quality, except for average cluster entropy, for which lower value signifies better clustering quality. Each evaluation measure penalizes or favors different properties in the clustering, hence we report results on these diverse measures to do a comprehensive evaluation.

We use Gaussian kernel for computing the graph similarities in all the experiments. The standard deviation of the kernel is taken equal to the median of the pair-wise Euclidean distances between the data points, except for the BBC data for which it gives extremely low performance. We use a kernel std. dev. of 100 for both BBC datasets. In all the result tables, the numbers in the brackets are the standard deviations of the performance measures obtained with 20 different runs of k-means with random initializations.

The results for synthetic data are shown in Table 1. As it can be seen, the proposed approach outperforms all the baselines. Baselines are run using first using two views and then using all three views, and the best results are reported here. The closest performing approach is kernel addition. For synthetic data, order-2 polynomial kernel based kernel-CCA gives best perfor-

Table 1. Clustering performance on **synthetic data**. Number (2) or (3) indicates the number of views used in the approach. Std. deviations of all performance metrics are zero for this synthetic data.

Method	F-score	Precision	Recall	Entropy	NMI	Adj-RI
Best Single View	0.971	0.975	0.968	0.097	0.898	0.942
Feature Concat	0.980	0.983	0.977	0.068	0.928	0.960
Kernel Addition	0.996	0.996	0.996	0.020	0.973	0.992
Kernel Product	0.990	0.988	0.991	0.041	0.959	0.980
CCA	0.984	0.984	0.984	0.067	0.932	0.968
Min-Disagreement	0.984	0.986	0.983	0.062	0.936	0.968
Co-trained spectral(2)	0.996	0.995	0.996	0.019	0.981	0.992
Co-trained spectral(3)	0.998	0.998	0.997	0.010	0.989	0.996

Table 2. Clustering performance on **Reuters multilingual data**. The languages used are English, French, and German. Number (2) or (3) indicates the number of views used in the approach. Numbers in parentheses are the std. deviations.

Method	F-score	Precision	Recall	Entropy	NMI	Adj-RI
Best Single View	0.342(0.010)	0.296(0.015)	0.407(0.025)	1.878(0.052)	0.287(0.019)	0.186(0.014)
Feature Concat	0.368(0.012)	0.330(0.016)	0.416(0.017)	1.841(0.057)	0.298(0.020)	0.225(0.017)
Kernel Addition	0.386(0.012)	0.358(0.017)	0.420(0.023)	1.770(0.058)	0.323(0.021)	0.252(0.016)
Kernel Product	0.258(0.003)	0.198(0.011)	0.381(0.058)	2.306(0.034)	0.123(0.010)	0.052(0.014)
CCA	0.262(0.007)	0.222(0.005)	0.322(0.034)	2.232(0.009)	0.147(0.003)	0.082(0.003)
Min-Disagreement	0.381(0.014)	0.341(0.004)	0.435(0.035)	1.736(0.052)	0.342(0.024)	0.240(0.012)
Co-trained spectral(2)	0.401(0.009)	0.363(0.007)	0.450(0.030)	1.651(0.024)	0.373(0.012)	0.267(0.007)
Co-trained spectral(3)	0.412(0.001)	0.369(0.001)	0.467(0.003)	1.616(0.017)	0.388(0.007)	0.279(0.001)

mance among all CCA variants, while Gaussian kernel based kernel-CCA performs poorly. We do not report results for Gaussian kernel CCA here. All the baselines outperform the single view case for the synthetic data.

Table 2 shows the document clustering results on Reuters multilingual data with English, French and German documents as the three views. On this dataset too, our approach outperforms all the baselines by a significant margin. The next best performance is attained by minimum-disagreement spectral clustering (de Sa, 2005) approach. It should be noted that CCA and element-wise kernel product performances are worse than that of single view.

Table 3 shows the results on UCI Handwritten digits dataset. On this dataset, quite a few approaches including kernel addition, element-wise kernel multiplication, and minimum-disagreement are close to our co-training based spectral clustering approach. Our approach still manages to perform marginally better than the best of these on all evaluation metrics. It can be noted that feature concatenation performs worse than single view.

Finally, the results on BBC and BBCSPORTS data are shown in Tables 4 and 5. All baselines perform better than single view. Our proposed approach outperforms the closest performing baseline, which is minimum-disagreement approach, by a significant margin. CCA again performs worse than single view as was the case for Reuters multilingual data.

We also show the variation in NMI score as the number of iterations increase in Figures 2 and 3. For the synthetic data, algorithm converges after four iterations for all the three views and remains constant after that. For Reuters data, the major improvement in performance for all views is obtained after the first iteration. It keeps varying around that value in the subsequent iterations. In general, we observe that the algorithm does not converge, as is also the case with semi-supervised co-training algorithm, which is also not guaranteed to converge. In all our experiments, we observed that the biggest increment in performance is obtained in the first iteration. We stop after a fixed number of iterations in our experiments. However, it is possible to apply some heuristic clustering performance measures (e.g. cluster compactness measure) to decide the stopping criterion.

6. Related Work

A number of clustering algorithms have been proposed in the past to learn with multiple views of the data. Some of them first extract a set of shared features from the multiple views and then apply any off-the-shelf clustering algorithm such as k-means on these features. The Canonical Correlation Analysis (CCA) (Chaudhuri et al., 2009; Blaschko & Lampert, 2008) based approach is an example of this. Alternatively, some other approaches exploit the multiple views of the data as part of the clustering algorithm

Table 3. Clustering performance on **Handwritten digits data**. Numbers in parentheses are the std. deviations.

Method	F-score	Precision	Recall	Entropy	NMI	Adj-RI
Best Single View	0.577(0.015)	0.569(0.020)	0.586(0.012)	1.198(0.029)	0.641(0.008)	0.530(0.017)
Feature Concat	0.536(0.027)	0.514(0.026)	0.561(0.032)	1.283(0.050)	0.619(0.015)	0.480(0.026)
Kernel Addition	0.707(0.052)	0.688(0.065)	0.727(0.037)	0.862(0.110)	0.744(0.030)	0.673(0.059)
Kernel Product	0.719(0.049)	0.698(0.064)	0.742(0.032)	0.832(0.102)	0.754(0.026)	0.687(0.055)
CCA	0.638(0.027)	0.616(0.037)	0.662(0.020)	1.073(0.071)	0.682(0.019)	0.596(0.031)
Min-Disagreement	0.693(0.047)	0.663(0.066)	0.729(0.026)	0.870(0.096)	0.745(0.024)	0.658(0.053)
Co-trained spectral	0.726(0.048)	0.709(0.058)	0.745(0.039)	0.793(0.109)	0.765(0.031)	0.695(0.054)

Table 4. Clustering performance on **BBC data**. Numbers in parentheses are the std. deviations.

Method	F-score	Precision	Recall	Entropy	NMI	Adj-RI
Best Single View	0.546(0.001)	0.522(0.001)	0.572(0.001)	1.221(0.003)	0.478(0.001)	0.424(0.001)
Feature Concat	0.559(0.019)	0.526(0.016)	0.598(0.022)	1.152(0.027)	0.512(0.013)	0.439(0.024)
Kernel Addition	0.558(0.013)	0.525(0.013)	0.595(0.013)	1.166(0.016)	0.506(0.008)	0.437(0.017)
Kernel Product	0.572(0.033)	0.536(0.028)	0.614(0.039)	1.132(0.053)	0.522(0.025)	0.455(0.042)
CCA	0.220(0.001)	0.193(0.001)	0.257(0.002)	1.861(0.003)	0.214(0.001)	0.178(0.001)
Min-Disagreement	0.854(0.047)	0.849(0.065)	0.860(0.026)	0.479(0.093)	0.794(0.033)	0.816(0.062)
Co-trained spectral	0.898(0.000)	0.894(0.000)	0.902(0.000)	0.369(0.000)	0.841(0.000)	0.873(0.000)

itself. For example, (Bickel & Scheffer, 2004) proposed an Co-EM based framework for multi-view clustering in mixture models. Co-EM approach computes expected values of hidden variables in one view and uses these in the M-step for other view, and vice versa. This process is repeated until a suitable stopping criteria is met. The algorithm often does not converge.

Multi-view clustering algorithms have also been proposed in the framework of spectral clustering (Zhou & Burges, 2007; de Sa, 2005). In (Zhou & Burges, 2007), the authors obtain a graph cut which is good on average over the multiple graphs but may not be the best for a single graph. They give a random walk based formulation for the problem. (de Sa, 2005) approaches the problem of two-view clustering by constructing a bipartite graph from nodes of both views. Edges of the bipartite graph connect nodes from one view to those in the other view. Subsequently, they solve standard spectral clustering problem on this bipartite graph. In (Tang et al., 2009), the information from multiple graphs are fused using Linked Matrix Factorization.

Consensus clustering approaches can also be applied to the problem of multi-view clustering (Strehl & Ghosh, 2002). These approaches do not generally work with original features. Instead, they take different clusterings of a dataset coming from different sources as input and reconcile them to find a final clustering.

7. Discussion

We proposed a multi-view spectral clustering approach using the idea of co-training, that has been widely used in semi-supervised learning problems. The general framework of our proposed algorithm is to learn the clustering in one view and use it to "label" the data in other view so as to modify the graph structure (similarity matrix). The modification to the graph is dictated by the discriminative eigenvectors and is achieved by projection along these directions.

Our key assumption that the true underlying clustering is same for all views is safe in most scenarios, and is necessary for multi-view algorithms to succeed. This assumption can potentially be violated in situations where the data assumes more than one natural clustering, and different clusterings become prominent in different views. However, in this work, we were not concerned with the problem of multiple clusterings so compatibility of clustering across views was safe to assume.

It is possible to extend the proposed framework to the case where some of the views have missing data. For missing data points, the corresponding entries in the similarity matrices would be unavailable. We can estimate these missing similarities by the corresponding similarities in other views. One possible approach to estimate the missing entry could be to simply average the similarities from views in which the data point is available. Proper normalization of similarities (possibly by Frobenius norm of the whole matrix) might be needed before averaging to make them comparable. Other methods for missing kernel entries estimation can also be used.

Theoretical analysis of the proposed approach can also be pursued as a separate line of work. There has been very little prior work analyzing spectral clustering methods. For instance, there has been some work

Table 5. Clustering performance on **BBCSPORTS data**. Numbers in parentheses are the std. deviations.

Method	F-score	Precision	Recall	Entropy	NMI	Adj-RI
Best Single View	0.387(0.015)	0.405(0.021)	0.370(0.015)	1.565(0.066)	0.286(0.028)	0.210(0.022)
Feature Concat	0.609(0.040)	0.636(0.019)	0.585(0.059)	0.919(0.009)	0.575(0.004)	0.497(0.047)
Kernel Addition	0.604(0.038)	0.634(0.020)	0.578(0.054)	0.898(0.014)	0.584(0.004)	0.491(0.045)
Kernel Product	0.603(0.036)	0.635(0.018)	0.575(0.053)	0.910(0.011)	0.578(0.003)	0.490(0.043)
CCA	0.173(0.008)	0.187(0.010)	0.161(0.006)	1.89(0.074)	0.115(0.011)	0.089(0.005)
Min-Disagreement	0.718(0.082)	0.751(0.051)	0.690(0.109)	0.646(0.080)	0.697(0.045)	0.638(0.102)
Co-trained spectral	0.850(0.078)	0.866(0.057)	0.836(0.095)	0.392(0.091)	0.817(0.047)	0.807(0.099)

on consistency analysis of single view spectral clustering (von Luxburg et al., 2008), which provides results about the rate of convergence as the sample size increases, using tools from theory of linear operators and empirical processes. Similar convergence properties could be studied for multi-view spectral clustering. We can expect the convergence to be faster for multi-view case. Co-training reduces the size of hypothesis space by limiting the search for compatible clusterings, and hence less number of examples should be needed to converge to the solution.

Acknowledgments

This work was partially funded by NSF grant IIS 0712764.

References

Amini, Massih-Reza, Usunier, Nicolas, and Goutte, Cyril. Learning from multiple partially observed views - an application to multilingual text categorization. In *Advances in Neural Information Processing Systems*, 2009.

Bickel, Steffen and Scheffer, Tobias. Multi-View Clustering. In *IEEE International Conference on Data Mining*, 2004.

Blaschko, Matthew B. and Lampert, Christoph H. Correlational Spectral Clustering. In *Computer Vision and Pattern Recognition*, 2008.

Blei, David M., Ng, Andreq Y., and Jordan, Michael I. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, pp. 993–1022, 2003.

Blum, A. and Mitchell, T. Combining labeled and unlabeled data with co-training. In *Conference on Learning Theory*, 1998.

Chaudhuri, Kamalika, Kakade, Sham M., Livescu, Karen, and Sridharan, Karthik. Multi-view Clustering via Canonical Correlation Analysis. In *International Conference on Machine Learning*, 2009.

Cortes, Corinna, Mohri, Mehryar, and Rostamizadeh, Afshin. Learning non-linear combination of kernels. In *Advances in Neural Information Processing Systems*, 2009.

de Sa, Virginia R. Spectral Clustering with two views. In *Proceedings of the Workshop on Learning with Multiple Views, International Conference on Machine Learning*, 2005.

Greene, D. and Cunningham, P. Producing accurate interpretable clusters from high-dimensional data. In *PKDD*, 2005.

Greene, Derek and Cunningham, Pádraig. A matrix factorization approach for integrating multiple data views. In *European Conference on Machine learning*, 2009.

Hofmann, Thomas. Probabilistic latent semantic analysis. In *Uncertainty in Artificial Intelligence*, 1999.

Horn, Roger A. and Johnson, Charles R. *Matrix Analysis*.

Hubert, Lawrence and Arabie, Phipps. Comparing Partitions. *Journal of Classification*, pp. 193–218, 1985.

Liberty, Edo, Woolfe, Franco, Martinsson, Per-Gunnar, Rokhlin, Vladimir, and Tygert, Mark. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 2007.

Manning, Christopher D., Raghavan, Prabhakar, and Schtze, Hinrich. *Introduction to Information Retrieval*. 2008.

Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: analysis and an algorithm. In *Advances in Neural Information Processing Systems*, 2002.

Nigam, Kamal and Ghani, Rayid. Analyzing the Effectiveness and Applicability of Co-training. In *International Conference on Information and Knowledge Management*, 2000.

Shi, J. and Malik, J. Normalized cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

Strehl, Alexander and Ghosh, Joydeep. Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, pp. 583–617, 2002.

Tang, Wei, Lu, Zhengdong, and Dhillon, Inderjit S. Clustering with Multiple Graphs. In *IEEE International Conference on Data Mining*, 2009.

von Luxburg, Ulrike. A Tutorial on Spectral Clustering. *Statistics and Computing*, 2007.

von Luxburg, Ulrike, Belkin, Mikhail, and Bousquet, Olivier. Consistency of Spectral Clustering. *Annals of Statistics*, 36(2):555–586, 2008.

Zhou, Dengyong and Burges, Christopher J. C. Spectral Clustering and Transductive Learning with Multiple Views. In *International Conference on Machine Learning*, 2007.