A General Model for Multiple View Unsupervised Learning

Bo Long* Philip S. Yu† Zhongfei (Mark) Zhang ‡

Abstract

Multiple view data, which have multiple representations from different feature spaces or graph spaces, arise in various data mining applications such as information retrieval, bioinformatics and social network analysis. Since different representations could have very different statistical properties, how to learn a consensus pattern from multiple representations is a challenging problem. In this paper, we propose a general model for multiple view unsupervised learning. The proposed model introduces the concept of mapping function to make the different patterns from different pattern spaces comparable and hence an optimal pattern can be learned from the multiple patterns of multiple representations. Under this model, we formulate two specific models for two important cases of unsupervised learning, clustering and spectral dimensionality reduction; we derive an iterating algorithm for multiple view clustering, and a simple algorithm providing a global optimum to multiple spectral dimensionality reduction. We also extend the proposed model and algorithms to evolutionary clustering and unsupervised learning with side information. Empirical evaluations on both synthetic and real data sets demonstrate the effectiveness of the proposed model and algorithms.

1 Introduction.

In many important data mining applications, the same instances have multiple representations from different spaces (views). These multiple representations could be from different feature vector spaces; for example, in bioinformatics, genes can be represented in the expression vector space (corresponding to the genetic activity under the biological conditions) and also in the term vector space (corresponding to the text information related to the genes) [18]. Multiple representations could also be from different graph spaces; for example, in social network analysis, the same instances could be related to each other in multiple networks (graphs), such as email networks, collaboration networks and organization hierarchy. Finally, multiple presentations could be from mixed views of vector spaces and graph spaces; for example, Web pages can have following multiple representations: the term vector corresponding to words occurring in the pages themselves, the graph of hyper-links between the pages, and the term vectors corresponding to words contained in anchor text of links pointing to the pages.

Multiple view data raise a natural, yet non-standard new problem: how to learn a consensus pattern based multiple representations, which is more accurate and robust than patterns based on a single view. Due to the phenomenal impact of multiple view data on many applications, multiple view learning is attracting more and more attention [29]. In this work, we consider the problem of multiple view unsupervised learning, i.e., mining hidden patterns, such as clustering and low-dimension embedding, from multiple view data without supervised information (such as class labels).

There are two directions for seeking solutions to multiple view unsupervised learning. One is to design centralized algorithms, which make use of multiple representations simultaneously to mine hidden patterns from the data. The top challenge for the centralized algorithms is the diversity of multiple representations, i.e., different representations not only could be of different formulations (vectors and graphs), but also could have very different statistical properties (continuous values with various distributions and discrete values with various distributions). Let us consider the scenario of clustering. A large number of clustering algorithms have been proposed in the literature for various types of data with various statistical properties [19, 4]. For example, Gaussian EM algorithm is a classic algorithm for data with spherical shape clusters; Subspace clustering approaches are designed for high dimensional data; spectral graph clustering approaches are popular for weighted undirected graphs. Hence, in many situations, designing a centralized multiple view clustering algorithm is close to designing a single algorithm to accomplish several different difficult tasks which are originally handled by several different clustering algorithms. This direction is so difficult that the existing efforts usually have to restrict themselves to special cases of multiple view data with strong assumptions. For example, [6] focuses on the data of two sets of features, which are assumed to be independent of each other and can be handled by the same algorithm such as k-means or

*Dept. of Computer Science, SUNY Binghamton
†Dept. of Computer Science, University of Illinois at Chicago
‡Dept. of Computer Science, SUNY Binghamton

multinomial EM; [36] deals with the data of two graphs which all are assumed to be well explained by a mixture of Markov chains. Above examples also imply that for different types of multiple view data, we need to design different centralized algorithms.

Another direction to solve multiple view unsupervised clustering is distributed, which has rarely been addressed in the literature of multiple view unsupervised learning. The idea of the distributed approaches is to learn hidden patterns individually from each representation of a multiple view data and then learn the optimal hidden patterns from those multiple patterns. Compared with the centralized approaches, the distributed approaches have the following advantages. First, they take base unsupervised learning methods for different representations as a black box, allowing practitioners to select the most appropriate method for each representation. Second, under a distributed framework, an algorithm which learns an optimal pattern from multiple patterns can be used for various types of multiple view data, since it does not work on the data directly. Third, for the multiple view data under the privacy-preserving scenario, a distributed algorithm is still a feasible solution while a centralized algorithm is not. For example, for a set of customers of different companies, one representation is from a bank and another is from a credit union; due to the privacy issue, the companies cannot share the information (representations), though they want to use both representations to learn hidden patterns for the customers. In such a scenario, the distributed approach provides a natural solution, since the companies could just mine patterns based on their own data and share the patterns (such as clusterings) instead of the original data.

Under a distributed framework, choosing the most appropriate unsupervised clustering for each representation is left for domain experts. Hence, the key problem is how to learn the optimal pattern from multiple patterns. The main challenge of the problem is that since different patterns from different representations exist in different spaces and they cannot be compared directly (Section 2.2).

In this paper, we address the problem of multiple view unsupervised learning under a distributed framework. Our main contributions can be summarized as follows:

1. We propose a general model for multiple view unsupervised learning, which is applicable to various types of unsupervised learning on various types of multiple view data (Section 2.2). We show how to formulate the problems of multiple view clustering (Section 2.3) and multiple view spectral embedding (dimensionality reduction) (Section 2.3).

2. We derive an iterating algorithm to solve the constrained non-convex problem of multiple view clustering (Section 3.1), which iteratively updates the optimal clustering pattern matrix and the mapping matrices until convergence. We prove the objective function is non-increasing under the updating rules and hence the convergence of the algorithm is guaranteed.

3. We derive a simple algorithm for multiple spectral embedding, which provides a global optimal solution to the problem (Section 3.2). To the best of our knowledge, our algorithm is the first one to address the problem of multiple view dimensionality reduction.

4. We also introduce extensions to our algorithms to handel two important learning settings: unsupervised learning with side information and evolutionary clustering (Section 4).

2 Background and Model Formulation

2.1 Notations and Background First, a word about notation. Capital letters, such as A and P, represent matrices. We use A_{ij} or $[A]_{ij}$ to represent the (i,j)th element of A. Bold lower case letters, such as $\mathbf{x}$ and $\mathbf{a}$, represent vectors. Calligraphic letters, such as $\mathcal{X}$ and $\mathcal{Y}$, represent sets. we use $\mathbb{R}$ to denote the set of real numbers, $\mathbb{R}_+$ to denote the set of nonnegative real numbers, and $\mathbb{R}_{++}$ to denote the set of positive real numbers.

No matter in unsupervised learning or supervised learning, we have input data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. However, in unsupervised learning, we have no supervised target output, which is given in supervised learning, such as the class labels given in a classification task. The goal of unsupervised learning is to build "concise representations" (patterns) from the data that can be used for reasoning, decision making, and communications, etc. Two classic examples of unsupervised learning are clustering and dimensionality reduction.

If we represent an input data set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ for unsupervised learning as a matrix $X \in \mathbb{R}^{n\times d}$, then in general the learned pattern can also be formulated as a matrix $A \in \mathcal{A}$ (we call A *pattern matrix* and $\mathcal{A}$ *pattern space*), where the pattern space $\mathcal{A}$ has a lower dimension than the original data space, i.e., $\mathcal{A} \subseteq \mathbb{R}^{n\times k}$ with $k \ll d$. For different types of unsupervised learning, the pattern spaces are different. For example, for clustering, the pattern space is a probabilistic simplex such that $\mathcal{A} = \{A | A \in \mathbb{R}_+^{n\times k}, A\mathbf{1} = \mathbf{1}\}$ where $\mathbf{1}$ denotes a vector consisting of 1's; i.e., in a clustering pattern matrix A, A_{ij} denotes the probability that the ith object is associated with the jth cluster; The clustering pattern

matrix provides the cluster structure of the data and the cluster centers can also be obtained based on clustering pattern matrix and the data matrix. Another example is spectral dimensionality reduction, in which $\mathcal{A}$ is a subset of an eigen-space such that $\mathcal{A} = \{A \in \mathbb{R}^{n\times k}, A^T A = I\}$ where I denotes an identity matrix, i.e., the pattern matrix A is a low-dimensional embedding of X in the eigen-space.

2.2 A General Model for Multiple View Unsupervised Learning In this section, we propose a general model for multiple view unsupervised learning under a distributed framework. Given a multiple view data set consisting of n objects with m representations denoted as a set of matrices $\mathcal{X} = \{X^{(i)} \in \mathbb{R}^{n\times m_i}\}_{i=1}^m$. Note that $X^{(i)}$ could be either a feature matrix or a graph affinity matrix. When $X^{(i)} \in \mathbb{R}^{n\times m_i}$ is a graph affinity matrix, we have $m_i = n$. Our task is to learn the optimal pattern matrix from $\mathcal{X}$.

First, the appropriate unsupervised learning methods for each representation $X^{(i)}$ are chosen by domain experts to learn a pattern matrix $A^{(i)}$ from $X^{(i)}$. This step is not our focus. In this paper, we focus on the core problem: learning an optimal pattern $B \in \mathbb{R}^{n\times k}$ from multiple patterns $\mathcal{A} = \{A^{(i)} \in \mathbb{R}^{n\times k_i}\}_{i=1}^m$.

Intuitively, we expect that the optimal pattern is the consensus pattern which is "shared" by multiple patterns in $\mathcal{A}$ as much as possible. From the view of mathematical optimization, we seek the optimal pattern which is close to all the patterns as much as possible under a certain distance measure. Hence, we need to compare two patterns under a certain distance measure. The problem is that since the patterns are learned in an unsupervised way, they are not directly comparable. Let us have an illustrative example under the clustering scenario. Assume that we have two clustering patterns for six data objects, which are represented as $A^{(1)}$ and $A^{(2)}$,

Example 1

$$A^{(1)} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}, A^{(2)} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \\ 0.7 & 0.3 \\ 0.6 & 0.4 \\ 0.2 & 0.8 \\ 0.1 & 1.9 \end{bmatrix}.$$

$A^{(1)}$ is a hard clustering pattern with three clusters and $A^{(2)}$ is a soft clustering pattern with two clusters. It is difficult to directly compare $A^{(1)}$ with $A^{(2)}$ to decide how close they are. Note that even for different clustering patterns with the same number of clusters, they are not directly comparable. For example, for the following two clustering patterns,

Example 2

$$A^{(3)} = \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.8 & 0.2 & 0 \\ 0 & 0.7 & 0.3 \\ 0 & 0.7 & 0.3 \\ 0 & 0.1 & 0.9 \\ 0 & 0.1 & 0.9 \end{bmatrix}, A^{(4)} = \begin{bmatrix} 0.2 & 0 & 0.8 \\ 0.2 & 0 & 0.8 \\ 0.7 & 0.3 & 0 \\ 0.7 & 0.3 & 0 \\ 0.1 & 0.9 & 0 \\ 0.1 & 0.9 & 0 \end{bmatrix}.$$

they are both a clustering pattern with three clusters. If we directly compare $A^{(3)}$ with $A^{(4)}$, they appear different. However, the two clusterings are actually equivalent; since cluster 1, 2, and 3 of $A^{(3)}$ actually is equivalent to cluster 3, 1 and 2 of $A^{(4)}$, respectively. This example implies that even two pattern matrices with the same dimension, they are not directly comparable. This is true for other types of unsupervised learning such as dimensionality.

To solve this problem, we propose to use a mapping function to map a pattern matrix into the pattern space of another pattern matrix; i.e., instead of comparing the two pattern matrices directly, we compare the mapped pattern matrix with another pattern matrix in the same pattern space. For example, for two pattern matrices $A^{(1)} \in \mathbb{R}^{n\times k_1}$ and $A^{(2)} \in \mathbb{R}^{n\times k_2}$, $f(A^{(1)})$ and $A^{(2)}$ are comparable where $f : \mathbb{R}^{n\times k1} \to \mathbb{R}^{n\times k2}$ is a mapping function.

By using the concept of the mapping function, we are able to obtain an optimal pattern from multiple patterns existing in different pattern spaces. The basic idea is that the optimal pattern should be "close" to each pattern as much as possible after it is mapped into the pattern space for each pattern. Formally, we define the model of multiple view unsupervised learning as follows.

DEFINITION 2.1. *Given a set of pattern matrices $\mathcal{A} = \{A^{(i)} \in \mathbb{R}^{n\times k_i}\}_{i=1}^m$, a positive integer k, a set of non-negative weights $\{w_i \in R_+\}_{i=1}^m$ and a distance function l, the optimal pattern matrix $B \in \mathbb{R}^{n\times k}$ and the optimal mapping functions $\mathcal{F} = \{f_i : \mathbb{R}^{n\times k} \to \mathbb{R}^{n\times k_i}\}_{i=1}^m$ are given by the minimization,*

$$\min_{B,\mathcal{F}} \sum_{i=1}^{m} w_i l(A^{(i)}, f_i(B)). \tag{2.1}$$

In Definition 2.1, we formulate the problem of multiple view unsupervised learning as an optimization problem w.r.t both the optimal pattern matrix and the mapping functions. The mapping functions provide not only a technical convenience, but also desirable information in some applications, since they actually denote the relations between the final optimal pattern and the patterns from different views.

A natural question about the model in Definition 2.1 is why we do not use mapping functions to map $A^{(i)}$ into the space of B to have the following model,

$$\min_{B,\mathcal{F}} \sum_{i=1}^{m} w_i l(f_i(A^{(i)}), B), \tag{2.2}$$

where $\mathcal{F} = \{f_i : \mathbb{R}^{n\times k_i} \to \mathbb{R}^{n\times k}\}_{i=1}^m$. The above model looks similar to the model in Definition 2.1. However, it has a serious problem that there always exists a useless global optimal solution to (2.2), $B^* = \{0\}^{n\times k_i}$ and $\mathcal{F}^* = \{f_i : \mathbb{R}^{n\times k_i} \to \{0\}^{n\times k}\}_{i=1}^m$; i.e., if we let B be a zero matrix and f_i be a mapping function which maps $A^{(i)}$ to a zero matrix, we always have a minimum distance. Obviously, this solution is useless in real applications. Hence, to design an algorithm to solve the model in (2.2), we need extra constraints to ensure the algorithm not to converge to the region near the useless solution. On the other hand, we do not have this problem with the model in Definition 2.1.

2.3 Two Specific Models: Multiple View Clustering and Multiple View Spectral Embedding

In this section, we show how to apply the general model of multiple view learning in Definition 2.1 to two of the most important unsupervised learning settings, clustering and spectral dimensionality reduction.

Definition 2.1 provides a general model which is applicable to different types of unsupervised learning on various multiple view data. To derive a specific model for a certain type of unsupervised learning, first we need to select the function space for the mapping functions. In this paper, we adopt the popular linear transformation function as the mapping function, since it provides a computational convenience and has an intuitive meaning in many applications. With mapping functions in the linear function space, i.e., $f_i \in \{f|f(X) = BX\}$, the model is reduced to the following optimization,

$$\min_{B\in\mathbb{R}^{n\times k},\{P^{(i)}\in\mathbb{R}^{k\times k_i}\}_{i=1}^m} \sum_{i=1}^{m} w_i l(A^{(i)}, BP^{(i)}). \tag{2.3}$$

In the rest of the paper, we call $P^{(i)}$ mapping matrix.

Next, based on the properties of pattern spaces in different unsupervised learning tasks, we select appropriate distance functions and constraints on the optimal pattern matrix and mapping matrices.

For multiple view clustering, each clustering pattern matrix lies in a probability simplex such that $A^{(i)} \in R_+^{n\times k_i}$ and $A\mathbf{1} = \mathbf{1}$. Hence, the optimal clustering pattern matrix B is also in a probabilistic simplex such that $B \in R_+^{n\times k}$ and $B\mathbf{1} = \mathbf{1}$. Since both $A^{(i)}$ and B are non-negative, we restrict the mapping matrix $P^{(i)}$ to be non-negative such that $P^{(i)} \in R_+^{k\times k_i}$. This restriction also gives the mapping matrix $P^{(i)}$ an intuitive interpretation, i.e., $P_{pq}^{(i)}$ denotes the weight of the pth cluster of the optimal clustering related to the qth cluster of the ith clustering. For example, the optimal mapping matrix for the two clustering patterns in Example 2 of Section 2.2 is

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

We observe that weights in P provides the equivalence of the clusters from the two clustering patterns.

For the distance function, since the element of the pattern matrices denotes probabilities, an intuitive choice is KL-divergence or Generalized I-divergence. Since generalized I-divergence is more general than KL-divergence in the sense that it does not require that the sum of the elements in a matrix equal to 1, we elect to use generalized I-divergence. Formally, we define the problem of multiple view clustering as follows.

Definition 2.2. *Given a set of clustering membership matrices $\mathcal{A} = \{A^{(i)} \in \mathbb{R}_+^{n\times k_i}\}_{i=1}^m$, a positive integer k, and a set of non-negative weights $\{w_i \in R_+\}_{i=1}^m$, the optimal clustering membership matrix $B \in \mathbb{R}_+^{n\times k}$ and the optimal mapping matrices $\mathcal{P} = \{P^{(i)} \in \mathbb{R}_+^{k\times k_i}\}_{i=1}^m$ are given by the minimization,*

$$\min_{\substack{B,\mathcal{P} s.t. \\ B\mathbf{1}=\mathbf{1}}} \sum_{i=1}^{m} w_i GI(A^{(i)}||BP^{(i)}), \tag{2.4}$$

where GI is generalized I-divergence function such that $GI(X||Y) = \sum_{ij}(\log X_{ij} \log \frac{X_{ij}}{Y_{ij}} - X_{ij} + Y_{ij})$.

One of the most popular dimensionality reduction approach is to embed data objects from high dimension feature space or a large graph into its low dimension eigen-space. We call it spectral embedding in this paper. In multiple view spectral embedding, pattern matrices from multiple views are spectral embedding matrices such that $A^{(i)} \in \mathbb{R}^{n\times k_i}$ and $A^TA = I$, where I is an identity matrix. Hence, the optimal pattern matrix is also a spectral embedding matrix such that $B \in \mathbb{R}^{n\times k}$ and $B^TB = I$. There are no constraints on the mapping matrices since they serve the purpose of general linear mapping. For the distance function, we elect to use the most popular one, Euclidean distance function. Formally, we define the problem of multiple view spectral embedding as follows.

Definition 2.3. *Given a set of spectral embedding matrices $\mathcal{A} = \{A^{(i)} \in \mathbb{R}^{n\times k_i}\}_{i=1}^m$, a positive integer k, and*

a set of non-negative weights $\{w_i \in R_+\}_{i=1}^m$*, the optimal spectral embedding matrix* $B \in \mathbb{R}^{n\times k}$ *and the optimal mapping matrices* $\mathcal{P} = \{P^{(i)} \in \mathbb{R}^{k\times k_i}\}_{i=1}^m$ *are given by the minimization,*

$$\min_{\substack{B,\mathcal{P} s.t.\\ B^TB=I}} \sum_{i=1}^m w_i ||A^{(i)} - BP^{(i)}||^2, \tag{2.5}$$

where $||\cdot||$ *denotes Frobenius norm such that* $||X||^2 = \sum_{ij} X_{ij}^2$.

3 Algorithm Derivation

In this section, we derive algorithms for multiple view clustering and multiple view spectral embedding. To avoid the clutter, in the following algorithm derivations we omit the weights w_i. However, all the derivation can be extended to weighted cases. Based on Definition 2.2 and Definition 2.3, Our task is to solve the following two optimizations: multiple view clustering,

$$\min_{\substack{B\geq 0, B\mathbf{1}=\mathbf{1}\\ \{P^{(i)}\geq 0\}_{i=1}^m}} \sum_{i=1}^m GI(A^{(i)}||BP^{(i)}), \tag{3.6}$$

where $B \geq 0$ and $P^{(i)} \geq 0$ denote that B and $P^{(i)}$ are non-negative, and multiple view spectral embedding,

$$\min_{\substack{B^TB=I,\\ \{P^{(i)}\}_{i=1}^m}} \sum_{i=1}^m ||A^{(i)} - BP^{(i)}||^2, \tag{3.7}$$

3.1 Multiple View Clustering Algorithm Since Generalized I-divergence is a separable distance function, by letting $A = [A^{(1)}, \ldots, A^{(m)}]$ and $P = [P^{(1)}, \ldots, P^{(m)}]$, we write the objective function in (3.6) in a pure matrix form as follows,

$$\min_{\substack{B\geq 0, B\mathbf{1}=\mathbf{1}\\ P\geq 0}} GI(A||BP). \tag{3.8}$$

Note that $A \in \mathbb{R}^{n\times r}$ and $P \in \mathbb{R}^{k\times r}$ with $r = \sum_{i=1}^m k_i$.

To the best of our knowledge, there is no efficient way to find the global optimum to the constrained non-convex optimization in (3.8). We derive an alternating algorithm for (3.8), which iteratively updates B and P until it converges to a local optimum.

First we fix P to update B. To deal with the constraint $B\mathbf{1} = \mathbf{1}$ efficiently, we transform it to a "soft" constraint by adding a penalty term, $\alpha GI(\mathbf{1}||B\mathbf{1})$, to the objective function, where α is a positive constant. Therefore, we obtain the following optimization.

$$\min_{B\geq 0} GI(A||BP) + \alpha GI(\mathbf{1}||B\mathbf{1}). \tag{3.9}$$

We derive an efficient updating rule for B based on the bound optimization procedure [30, 13]. The basic idea is to construct an auxiliary function which is a convex upper bound for the original objective function based on the solution obtained from the previous iteration. Then, a new solution to the current iteration is obtained by minimizing this upper bound. The definition of an auxiliary function is given as follows.

DEFINITION 3.1. *$h(S, S^t)$ is an auxiliary function for $f(S)$ if $h(S, S^t) \geq f(S)$ and $h(S, S) = f(S)$.*

The auxiliary function is useful due to the following lemma.

LEMMA 3.1. *If h is an auxiliary function, then f is non-increasing under the updating rule*

$$S^{t+1} = \arg\min_S h(S, S^t). \tag{3.10}$$

Proof. $f(S^{t+1}) \leq h(S^{t+1}, S^t) \leq h(S^t, S^t) \leq f(S^t)$.

The key step of the derivation is to design auxiliary functions. We propose an auxiliary function for B in the following theorem.

LEMMA 3.2.

$$\begin{aligned} h(B, \tilde{B}) &= \sum_{ij}(\sum_g (B_{ig}P_{gj}) - A_{ij}\sum_g \frac{\tilde{B}_{ig}P_{gj}}{[\tilde{B}P]_{ij}} \log B_{ig}) \\ &\quad +\alpha\sum_i([B\mathbf{1}]_i - \sum_g(\frac{\tilde{B}_{ig}}{[\tilde{B}\mathbf{1}]_i}\log B_{ig})) + t(\tilde{B}) \end{aligned}$$

is an auxiliary function for

$$f(B) = GI(A||BP) + \alpha GI(\mathbf{1}||B\mathbf{1}), \tag{3.11}$$

where $t(\tilde{B})$ is a function w.r.t $\tilde{B}$ s.t.

$$\begin{aligned} t(\tilde{B}) &= \sum_{ij}(A_{ij}\log A_{ij} - A_{ij} - A_{ij}\sum_g(\frac{\tilde{B}_{ig}P_{gj}}{[\tilde{B}P]_{ij}}\log\frac{[\tilde{B}P]_{ij}}{\tilde{B}_{ig}})) \\ &\quad -n\alpha - \alpha\sum_i\sum_g(\frac{\tilde{B}_{ig}}{[\tilde{B}\mathbf{1}]_i}\log(\frac{[\tilde{B}\mathbf{1}]_i}{\tilde{B}_{ig}})) \end{aligned}$$

Proof.

$$\begin{aligned} f(B) &= \sum_{ij}(A_{ij}\log\frac{A_{ij}}{[BP]_{ij}} - A_{ij} + [BP]_{ij}) \\ &\quad +\alpha\sum_i(-\log[B\mathbf{1}]_i - 1 + [B\mathbf{1}]_i) \\ &\leq \sum_{ij}(A_{ij}\log A_{ij} - A_{ij} + \sum_g(B_{ig}P_{gj}) \\ &\quad -A_{ij}\sum_g(\frac{\tilde{B}_{ig}P_{gj}}{[\tilde{B}P]_{ij}}\log(\frac{[\tilde{B}P]_{ij}}{\tilde{B}_{ig}P_{gj}}B_{ig}P_{gj}))) \\ &\quad +\alpha\sum_i([B\mathbf{1}]_i - 1 - \sum_g(\frac{\tilde{B}_{ig}}{[\tilde{B}\mathbf{1}]_i}\log(\frac{[\tilde{B}\mathbf{1}]_i}{\tilde{B}_{ig}}B_{ig}))) \\ &= h(B, \tilde{B}) \end{aligned}$$

During the above deduction, we use Jensen's inequality and concavity of the log function.

The following theorem provides the updating rule for B.

THEOREM 3.1. *The objective function $f(B)$ in Eq.(3.11) is nonincreasing under the updating rule,*

$$B_{ig} = \tilde{B}_{ig} \frac{\sum_j (A_{ij} \frac{P_{gj}}{[\tilde{B}P]_{ij}}) + \frac{\alpha}{[\tilde{B}\mathbf{1}]_i}}{\sum_j P_{gj} + \alpha} \tag{3.12}$$

where $\tilde{B}$ denotes the solution from the previous iteration.

Proof. Based on Lemma 3.2, take the derivative of $h(B, \tilde{B})$ w.r.t. B_{ig} to obtain

$$\begin{aligned} \frac{\partial h(B,\tilde{B})}{\partial B_{ig}} &= \sum_j (P_{gj} - A_{ij} \frac{P_{gj}\tilde{B}_{ig}}{[\tilde{B}P]_{ij} B_{ig}}) \\ &\quad + \alpha - \alpha \frac{\tilde{B}_{ig}}{[\tilde{B}\mathbf{1}]_i B_{ig}} \end{aligned}$$

Solve $\frac{\partial h(B,\tilde{B})}{\partial B_{ig}} = 0$ to obtain Eq.(3.12). By Lemma 3.1, the proof is completed.

Similarly, we present the following theorems to derive the updating rule for P.

LEMMA 3.3.

$$\begin{aligned} h(P, \tilde{P}) &= \sum_{ij} (A_{ij} \log A_{ij} - A_{ij} + \sum_g (B_{ig} P_{gj}) \\ &\quad - A_{ij} \sum_g (\frac{B_{ig}\tilde{P}_{gj}}{[B\tilde{P}]_{ij}} \log \frac{[B\tilde{P}]_{ij} P_{gj}}{\tilde{P}_{gj}})) \end{aligned}$$

is an auxiliary function for

$$f(P) = GI(A||BP). \tag{3.13}$$

THEOREM 3.2. *The objective function $f(P)$ in Eq.(3.13) is nonincreasing under the updating rule*

$$P_{gj} = \tilde{P}_{gj} \frac{\sum_i \frac{A_{ij} B_{ig}}{[B\tilde{P}]_{ij}}}{\sum_i B_{ig}}. \tag{3.14}$$

Following the way to prove Lemma 3.3 and Theorem 3.2, it is not difficult to prove the above theorems. We omit details here.

We call the algorithm as the Multiple View Clustering (MVC) algorithm, which is summarized in Algorithm 1. The complexity of the algorithm is $O(tnrk)$ for t iterations and it can be further reduced for sparse data. The convergence of the MVC algorithm is guaranteed by Theorems 3.1 and 3.2.

Algorithm 1 Multiple View Clustering

Input: A set of clustering pattern matrices denoted as $A = [A^{(1)} \dots A^{(m)}]$ and a positive integer k.
Output: A clustering pattern matrix B and a set of mapping matrices $P = [P^{(1)} \dots P^{(2)}]$.
Method:

1: Initialize B and P.
2: **repeat**
3:
$$B_{ig} = B_{ig} \frac{\sum_j (A_{ij} \frac{P_{gj}}{[BP]_{ij}}) + \frac{\alpha}{[B\mathbf{1}]_i}}{\sum_j P_{gj} + \alpha}.$$
4:
$$P_{gj} = \tilde{P}_{gj} \frac{\sum_i \frac{A_{ij} B_{ig}}{[B\tilde{P}]_{ij}}}{\sum_i B_{ig}}$$
5: **until** convergence

3.2 Multiple View Spectral Embedding Algorithm Since Frobenius norm is also a separable distance function, by letting $A = [A^{(1)}, \dots, A^{(m)}]$ and $P = [P^{(1)}, \dots, P^{(m)}]$, we can re-write the objective function of multiple view spectral embedding as follows,

$$\min_{B^T B = I} ||A - BP||^2 \tag{3.15}$$

We show how to derive an algorithm which provides the global optimum to the optimization in (3.15). First, we use KKT condition to derive the following lemma.

LEMMA 3.4. *If B and P are the optimal solution to (3.15), then*

$$P = B^T A \tag{3.16}$$

Proof. Let $f(B, P)$ denote the objective function in (3.15). $f(B, P)$ can be expanded as follows,

$$\begin{aligned} f(B,P) &= tr((A - BP)^T (A - BP)) \\ &= tr(A^T A) - 2tr(P^T B^T A) + tr(P^T P) \end{aligned}$$

In the last step of the above deduction, we use $B^T B = I$. Take the derivative of $f(B, P)$ w.r.t P, we obtain

$$\frac{\partial f(B,P)}{\partial P} = -2B^T A + 2P. \tag{3.17}$$

According to KKT condition, we solve $\frac{\partial f(B,P)}{\partial P} = 0$ to obtain Eq.(3.16). The proof is completed.

Based on Lemma 3.4, if we know the optimal B, we can easily obtain the optimal P by Eq.(3.16). Based on the following theorem, we derive the closed form for the optimal B.

Algorithm 2 Multiple View Spectral Embedding

Input: A set of spectral embedding matrices denoted as $A = [A^{(1)}, \ldots, A^{(m)}]$ and a positive integer k.
Output: A spectral embedding matrix B and a set of mapping matrices $P = [P^{(1)}, \ldots, P^{(2)}]$.
Method:

1: $B = [\mathbf{u}_1 \ldots \mathbf{u}_k]$ where $\{\mathbf{u}_i\}_{i=1}^k$ are leading k eigenvectors of the matrix AA^T.
2: $P = B^T A$

THEOREM 3.3. *The minimization problem in* (3.15) *is equivalent to the following maximization problem:*

$$\max_{B^T B = I} tr(B^T A A^T B) \tag{3.18}$$

Proof. Based on Lemma 3.4, we substitute Eq.(3.16) into the objective function in (3.15) to obtain

$$f(B, P) \quad = \quad tr(A^T A) - tr(B^T A A^T B)$$

Since A is a constant matrix, minimization in (3.15) is equivalent to the maximization in (3.18). This completes the proof of the theorem.

It turns out that the maximization in Eq. (3.18) has a closed-form solution.

THEOREM 3.4. *(Ky-Fan thorem [5]) Let M be a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k$, and the corresponding eigenvectors $U = [\mathbf{u}_1, \ldots, \mathbf{u}_k]$. Then $\sum_{i=1}^k \lambda_i = \max_{X^T X = I_k} tr(X^T M X)$. Moreover, the optimal X is given by $[\mathbf{u}_1, \ldots, \mathbf{u}_k]Q$ where Q is an arbitrary orthogonal matrix.*

Based on Theorem 3.3 and Theorem 3.4, the optimal B is given as the leading k eigenvectors of the matrix AA^T.

Hence, we have a simple algorithm for Multiple View Spectral Embedding (MVSE), which is summarized in Algorithm 2. The derivation procedure of Algorithm 2 can be extended to deriving algorithms for other types of multiple view dimensionally reduction, such as multiple view sparse coding [24].

4 Extensions and Discussions

In this section, we discuss how to extend the proposed model and algorithms to two important related fields, unsupervised learning with side information and evolutionary clustering.

4.1 Evolutionary Clustering In the literature of multiple view learning, there is a popular assumption that multiple representations are independent [29, 6, 14], though in real applications this assumption may not always hold true. On the other hand, our model and algorithms do not make this assumption. Hence, they are applicable to a wider range of data. Furthermore, this makes it easy to extend our model and algorithms to a fairly new field, evolutionary unsupervised learning [9, 10].

Evolutionary clustering arise in the dynamic application scenarios, where the objects to be clustered evolve with time. In evolutionary clustering, we need to address two issues at each time step: the current clustering pattern should depend mainly on the current data features; on the other hand, the current clustering pattern should not deviate dramatically from the most recent history, i.e., we expect a certain level of temporal smoothness between clusters in successive time steps [10].

If we treat data features at each time step as a new representation for the data objects, evolutionary data can be formulated as a special case of multiple view data. Clearly, representations are not independent here. Furthermore, our task in evolutionary clustering is to learn a series of clustering patterns with temporal smoothness, instead of one clustering pattern based on all historic data. Our multiple view clustering model and algorithm can be easily extended to the evolutionary case. We describe the procedure as follows.

- At each time step t, the basic clustering pattern $A^{(t)}$ is directly learned from the data features at time t by a specified clustering algorithm.
- A temporal smooth clustering pattern $B^{(t)}$ is given by calling Algorithm 1 s.t. MVC($[B^{(t-1)}, A^{(t)}], k$), where k is the desired number of clusters for $B^{(t)}$.

The above procedure outputs a series of clustering patterns $\{B^{(t)}\}_{t=1}^m$, which depend on the current data features and also consistent with the most recent clustering pattern. This procedure also allows the number of clusters to change with time. Another interesting property about it is that unlike existing efforts focusing on specific type of clustering algorithms, such as evolutionary k-means [9] and evolution spectral clustering [10], it is flexible to adopt any clustering algorithms. The proposed procedure can also be extended to other types of evolutionary unsupervised learning, which have not been touched in the literature.

4.2 Unsupervised Learning with Side Information Unsupervised learning with side information, such as semi-supervised clustering [3, 34], has become a topic of significant interest, which seeks to do unsupervised clustering by incorporating background knowledge, such as partial labeled data or pairwise constraints for data points.

Existing literature on unsupervised clustering with side information focuses on how to make use of side information inside a certain type of clustering algorithm. We discuss how our model and algorithm provide a different style of a solution. The basic idea is that we treat side information as patterns, which is provided by domain experts instead of learned from data. Specifically, we formulate the side information as pattern matrices and use them with pattern matrices from multiple representations together as input to our algorithms. Then, the side information is incorporated into the final optimal pattern.

We describe an example on how to formulate the side information as pattern matrices. Assume that we have partially labeled data. We can represent this side information by assign the labeled data objects to the clusters with a certain probability and assigning the average probability to the unlabeled data points. In the following side information pattern matrix,

Example 3

$$A^{(1)} = \begin{bmatrix} 1 & 0 & 0 \\ 0.8 & 0.1 & 0.1 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix},$$

we have two labeled data objects such that the first one is labeled as cluster 1 with probability 1 and the second one is labeled as cluster 1 with probability 0.8; the other data objects are unlabeled. For the pairwise constraint information, they can also be formulated as pattern matrix as follows, assigning the data objects with must-links into the same cluster, the data objects with cannot-links into different clusters and unconstrained data objects into each cluster with the average probability.

5 Experimental Studies

In this Section, we perform an extensive study of our algorithms under different problem settings on both synthetic data and real data. We show that by using multiple patterns from multiple views, our algorithms provide more accurate and robust patterns on various data sets.

5.1 Multiple View Clustering In this section, we evaluate the effectiveness of MVC algorithm. For the comparisons, the average performance of all the views (call it AV) and the best performance of all the views (call it BV) are reported. As for other multiple view clustering algorithms, there are limited efforts in the literature on multiple view clustering [6, 14, 36], which focus on special cases with two views or a certain type of representation (Section 6). We use the classic algorithm proposed in [6] (call it Two View Clustering (TVC)) as a comparison when it is applicable. For performance measure, we elect to use the Normalized Mutual Information (NMI) [31] between the resulting cluster labels and the true cluster labels, which is a standard way to measure the cluster quality. The final performance score is the average of ten runs. For the number of clusters k, we simply use the number of true clusters. The k-means algorithm is used as the base clustering algorithm except stated otherwise. Random initializations are used for MVC. For the value of α in MVC, we simply adopt 1 in all experiments, since we find that the results in the experiments are not sensitive to the change of α.

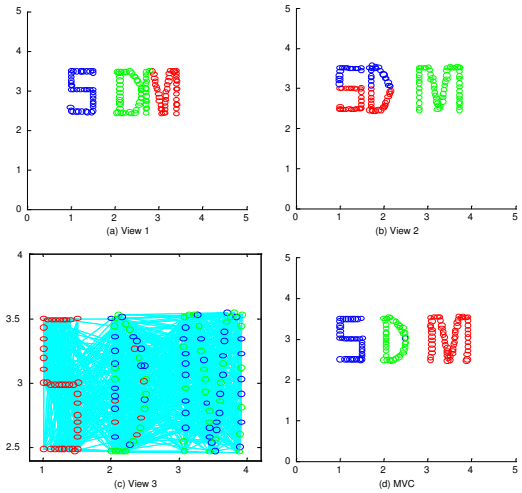

Figure 1: A toy example that demonstrates that our MVC algorithm is able to learn the consensus pattern from multiple views with noise.

	View 1	View 2	View 3	View 4
Cluster 1	N(1,1)	B(1)	P(2)	E(1)
Cluster 2	N(2,1)	B(0.2)	P(6)	E(25)
Cluster 3	N(5,1)	B(0.1)	P(12)	E(50)

Table 1: Distributions and parameters to generate syn2 data.

5.1.1 Synthetic Data The first synthetic data set (syn1) is a toy example consisting of three views for letters "SDM". The first view is in a two dimension space such that "D" and "M" are close; the second view is also in a two dimension space such that "S" and "D" are close; the third view is a graph in which the data points consisting of the letters are linked to each other with certain probabilities, 0.2 for the data points in the same letter and 0.1 for the data points in different

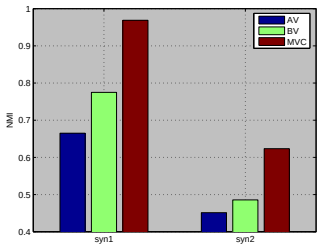

Figure 2: NMI comparison on synthetic data.

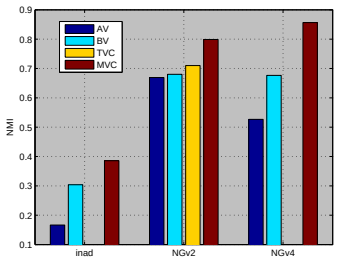

Figure 3: NMI comparison on real data.

letters. Figure 1(a) shows the clustering result of k-means algorithm on view 1, and we observe that "D" and "M" are confused. Figure 1(b) shows the clustering result of k-means algorithm on view 2, and we observe that "S" and "D" are confused. Figure 1(c) shows the clustering result of the graph clustering algorithm, METIS [21], on view 3, and we observe that three letters are confused. Figure 1(d) provides a visualization for the clustering pattern which is learned by MVC algorithm based on the three "confusing" patterns from the three views, and we observe that the three letters are almost perfectly distinguished from each other. The corresponding NMI scores of AV, BV and MVC are shown in Figure 2.

The second synthetic data set (syn2) is to simulate a typical situation: different features from different views have totally different distributions. The distributions and parameters to generate syn2 data are summarized in Table 1, in which "N" denotes normal distribution; "P" denotes Poisson distribution; "B" denotes Bernoulli distribution; "E" denotes exponential distribution. The size of syn2 is 9000 and 3000 for each cluster. We use the appropriate EM algorithms to cluster each view, i.e, Gaussian EM, Bernoulli EM, Poisson EM and Exponential EM [2] are for view 1, 2, 3 and 4, respectively. NMI comparison on syn2 data is shown in Figure 2. We observe that MVC's performances are significantly better than average and best performance based on each single view.

In summary, these experiments based on synthetic data sets demonstrate that the MVC algorithm can effectively learn the consensus clustering pattern, which is more accurate and robust than patterns based on any single view, from multiple representations with very different forms and different statistical properties.

5.1.2 Real Data We evaluate the performance of MVC algorithm on three real data sets. The first data set is internet advertisement data set (INTAD) [22], in which each instance is an image in a Web page and belongs to either of two clusters, ad or non-ad. Each instance is described from six views, geometry of the image, phrases occurring in the URL, the image's URL, alt text, the anchor text, and words occurring near the anchor text. INTAD data set has 3279 instances (2821 nonads and 458 ads) and totally 1558 features (continuous or binary) for 6 views. Multiple views for INTAD data set are naturally formulated and we do not know if the views are independent or not. However, this issue does not affect MVC algorithm

The second and third data sets are based on the 20-newsgroup [23], which was used in [6] to test the multiple view clustering algorithm, TVC. We follow the same way in [6] to generate two multiple view data sets by concatenating TF-IDF term vectors randomly drawn from different newsgroups (please refer to [6] for details). [6] uses this construction to ensure the data to satisfy the independent assumption. The cluster and view structures for two data sets, NGv2 and NGv4 are given as follows.

$$\begin{bmatrix} & cluster1 & cluster2 \\ view1 & windows.misc & pc.hardware \\ view2 & politics.mideast & politics.misc \end{bmatrix}$$

$$\begin{bmatrix} & cluster1 & cluster2 & cluster3 \\ view1 & alt.atheism & comp.graphics & windows.misc \\ view2 & windows.x & misc.forsale & rec.autos \\ view3 & sport.hockey & sci.crypt & sci.electronics \\ view4 & politics.guns & politics.mideast & politics.misc \end{bmatrix}$$

We use 2000 features for each view and 500 documents for each cluster. Hence, NGv2 has 1000 documents with 4000 features and NGv4 has 1500 documents with 8000 features.

The NMI comparison on real data sets is shown in Figure 3. We observe that MVC provides the best performance on all data sets. For the difficult INTAD data set, MVC's performance increases about 100% comparing with the average performance based on all single views and 20% comparing with the best performance of all single views. This shows that MVC is capable of learning the consensus pattern even from very noisy multiple patterns. MVC also performs significantly better than TVC on NGv2 data set (for other data set with more than two views, TVC is not applicable). We do not have definite reason for this, since the theory behind TVC algorithm is still not completely clear and actually TVC's convergence is not guaranteed.

In summary, above experiments show that no matter whether the data satisfy the independence assumption, MVC algorithm always efficiently learns robust

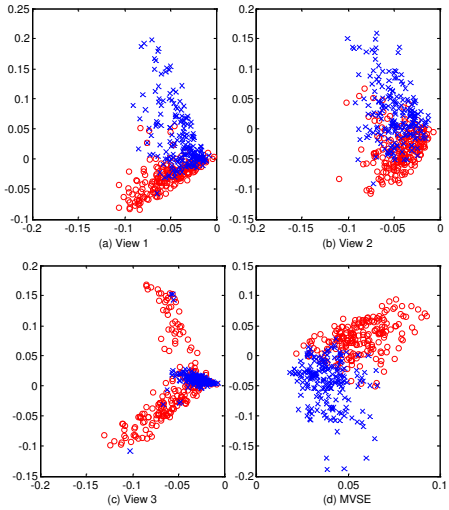

Figure 4: Four embeddings for NGv3 data set.

consensus patterns from multiple view data with different properties and different levels of noise.

5.2 Multiple View Spectral Embedding To test the effectiveness of MVSE algorithm on high dimensional data, we elect to use high dimensional 20-newsgroup data. Similarly, we construct NGv3 data set with the following cluster and view structures.

$$\begin{bmatrix} & cluster1 & cluster2 \\ view1 & windows.misc & pc.hardware \\ view2 & sport.baseball & sport.hockey \\ view3 & politics.mideast & politics.misc \end{bmatrix}$$

Hence NGv3 is a high dimensional data set with 6000 dimensions. We reduce it to two dimensional spaces. Figure 4 shows four embeddings for NGv3 data set. In Figure 4, each embedding is in a two dimension eigen-space corresponding to the two leading eigen-vectors. Figure 4(a), (b) and (c) show the embeddings for view 1, 2 and 3, respectively. Figure 4(d) shows the embedding provided by the MVSE algorithm. We observe that MVSE provides an embedding with the best separation, i.e., the intrinsic cluster patterns of the data are clearer in MVSE embedding than in the embeddings from the three single views.

Since direct observation is not accurate, we use the following method to evaluate which embedding has less noise and provides clearer pattern. We run k-means on each embedding with exactly the same initialization and then compare the clustering results from different embeddings by NMI scores. We expect that the clearer patterns an embedding has, the better the clustering result is. The NMI comparison is shown in Figure 5. We observe that the NMI score for MVSE embedding shows about 200% improvement even compared with the best performance for the single view embeddings. We also conduct experiments on NGv4 data set with

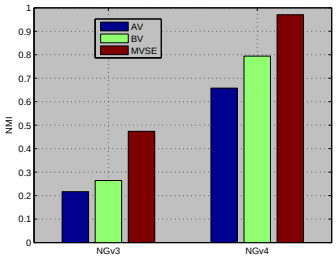

Figure 5: NMI comparison on spectral embedding of NGv3 and NGv4.

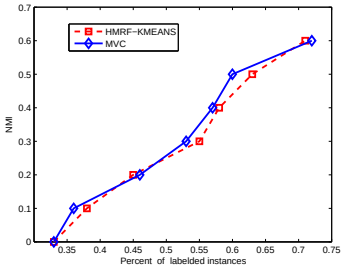

Figure 6: Sumi-supervised clustering results.

8000 dimensions. We reduce it to three dimensional eigen-spaces. The NMI scores are also shown in Figure 4. We observe the similar result to that for NGv3.

In summary, as the first algorithm to address multiple view dimensionality reduction, MVSE demonstrates its effectiveness by providing optimal spectral embeddings, which are more robust to noise and has better separation than the spectral embeddings based on a single view.

5.3 Semi-supervised Clustering As we discussed in Section 4, our model and algorithms can be extended to deal with unsupervised clustering with side information. In this section, we use MVC to perform semi-supervised clustering on INTAD data set to demonstrate the great potential of this extension.

We randomly choose a certain percent of instances from INTAD data set and label them as side information. Then we run MVC on the partially labeled INTAD data set in comparison with a state-of-the-art semi-supervised clustering algorithm, HMRF-KMEANS algorithm [3].

Figure 6 shows the clustering results on INTAD data set with different percents of instances labeled. We observe that both MVC and HMRF-KMEANS's performances increase with the number of labeled instances. MVC's performance is comparable with the classic HMRF-KMEANS algorithm. This demonstrates the great potential of MVC on semi-supervised clustering, since MVC is flexible to adopt different types of clustering algorithm instead of only k-means algorithm.

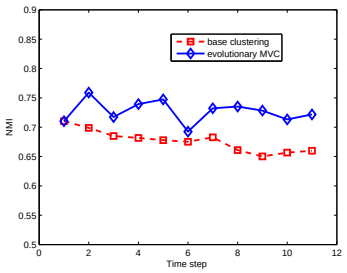

Figure 7: Evolutionary clustering results.

5.4 Evolutionary Clustering As a fairly new filed, there are short of benchmark data with ground truth for evolutionary clustering and performance measures for evolutionary clustering are under development [9]. In this section, we use a synthetic data with true labels available to show the great potential of evolutionary MVC, which provides more accurate clustering patterns at each time step by incorporating temporal smoothness.

We construct a mixture of Gaussian distributions evolving with time. At the start time, three components of 6000 data points have means 1, 4, 7 and the same variance 1. At each time t, the means of the three cluster centers move a small distance with distribution $N(0, 0.1)$ and the variance increases by 0.1. Hence the cluster centers are drifting and expanding over time. Figure 7 shows the clustering results of the evolutionary MVC and the base clustering algorithm, k-means. We observe that MVC performs significantly better than the base algorithm at most time steps. We also observe that the base clustering patterns degenerate over time and on the other hand, the clustering patterns from evolutionary MVC do not show clear degenerating trend. In fact, the clustering patten for evolutionary MVC at last time step is still significantly better than the start pattern. This is because by efficiently combining the patterns at time $t-1$ and time t, evolutionary MVC is more roust to noise at each time step and by incorporating temporal smoothness, the intrinsic pattern of the data is passed on over time by evolutionary MVC to avoid fast degeneration of the patterns.

6 Related Work

Multiple view learning was introduced by [7] and [20] in the semi-supervised setting. They propose the co-training approach to train a classifier from two representations with both labeled and unlabeled instances. The idea of the co-training approach is to train one learner on each view of the labeled instances and then to iteratively let each learner label the unlabeled instances it predicts with the highest confidence. Given independence between the learners, newly labeled examples from one learner may give the other learner new information to improve its model. [11] extends co-training to explicitly measure the degree of agreement between the rules in different views. [12] proposes PAC bounds for co-training. [1] explains the co-training based on Bootstrapping theory by showing that given a certain type of independence between the learners, the disagreement of two learners gives an upper bound on the error rate; unlabeled data can be used to minimize the disagreement between learners, and hence improves their combined accuracy. A number of approaches have followed and extended the original co-training idea [17, 28, 27, 8].

Multiple view unsupervised learning is a fairly new topic. In fact, the subfields such as multiple view dimensionality reduction has not been touched in literature, though there is limited work on multiple view clustering [6, 14, 36]. As discussed in Section 1, these approaches are centralized and focus on the simple case of two views with strong assumptions. [6] proposes clustering algorithms for the multiple view data with two independent views, which can be viewed as an extension to purely unsupervised setting. [14] also assumes two independent views for a multiple view data set and proposes a spectral clustering algorithm which creates a bipartite graph and is based on the minimizing-disagreement idea. [36] proposes a spectral clustering algorithm for multiple graphs, which generalizes the normalized cut from a single view to multiple views.

Ensemble clustering [31, 33, 15, 26] is a related filed, though its focus is on how to generate and combine different clusterings for a single view data set. Clustering multi-type objects is also close to multiple view clustering [32, 16, 25, 35]. [32] extends the the probabilistic relational model to the clustering multi-type relational data by introducing latent variables into the model; these models focus on using attribute information for clustering. [16] formulates star-structured relational data as a star-structured m-partite graph and develops an algorithm based on semi-definite programming to partition the graph. [35] presents an approach to improve the cluster quality of multi-type interrelated data objects through an iterative reinforcement clustering process.

7 Conclusions

In this paper, we propose a general model for multiple view unsupervised learning, which is applicable to various types of unsupervised learning on various types of multiple view data. We formulate two specific models for clustering and spectral dimensionality reduction; we derive an iterating algorithm for multiple view clustering, and a simple algorithm providing a global optimum to multiple spectral dimensionality reduction. We also extend the proposed model and algorithms to evolutionary clustering and semi-supervised learning with side in-

formation. Empirical evaluations on both synthetic and real data sets demonstrate the effectiveness and great potential of the proposed model and algorithms.

Acknowledgement

This work is supported in part by NSF (IIS-0535162), AFRL (FA8750-05-2-0284), AFOSR (FA9550-06-1-0327), and a research internship at IBM Watson Research Center.

References

[1] S. Abney. Bootstrapping. In *ACL '02*, pages 360–367, 2002.

[2] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *J. Mach. Learn. Res.*, 6:1705–1749, 2005.

[3] S. Basu, M. Bilenko, and R. J. Mooney. A probabilistic framework for semi-supervised clustering. pages 59–68, Seattle, WA, Aug. 2004.

[4] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[5] R. Bhatia. *Matrix analysis*. Springeer-Cerlag, New York, 1997.

[6] S. Bickel and T. Scheffer. Multi-view clustering. In *ICDM '04*, pages 19–26, 2004.

[7] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT' 98*, pages 92–100, 1998.

[8] U. Brefeld and T. Scheffer. Co-em support vector learning. In *ICML '04*, page 16, 2004.

[9] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *KDD '06*, pages 554–560, 2006.

[10] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *KDD '07*, pages 153–162, 2007.

[11] M. Collins and Y. Singer. Unsupervised models for named entity classification, 1999.

[12] S. Dasgupta, M. L. Littman, and D. A. McAllester. Pac generalization bounds for co-training. In *NIPS*, 2001.

[13] D.D.Lee and H.S.Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[14] V. R. de Sa. Spectral clustering with two views. In *ICML Workshop on Learning with Multiple Views*, 2005.

[15] X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *ICML '04*.

[16] B. Gao, T.-Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *KDD '05*, pages 41–50, 2005.

[17] R. Ghani. Combining labeled and unlabeled data for text classification with a large number of categories. In *ICDM '01*, pages 597–598, 2001.

[18] P. Glenisson, J. Mathys, and B. D. Moor. Meta-clustering of gene expression data and literature-based information. *SIGKDD Explor. Newsl.*, 5(2):101–112, 2003.

[19] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[20] H. Kaji and Y. Morimoto. Unsupervised word sense disambiguation using bilingual comparable corpora. In *Proceedings of the 19th COLIN*, pages 1–7, 2002.

[21] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998.

[22] N. Kushmerick. Learning to remove internet advertisements. In *AGENTS '99*, pages 175–181, 1999.

[23] K. Lang. News weeder: Learning to filter netnews. In *ICML*, 1995.

[24] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *NIPS'07*, pages 801–808. 2007.

[25] B. Long, X. Wu, Z. M. Zhang, and P. S. Yu. Unsupervised learning on k-partite graphs. In *KDD-2006*, 2006.

[26] B. Long, Z. M. Zhang, and P. S. Yu. Combining multiple clusterings by soft correspondence. In *ICDM '05*, pages 282–289, 2005.

[27] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM '00*, pages 86–93, 2000.

[28] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

[29] S. Rüping and T. Scheffer. Learning with multiple views. In *ICML Workshop on Learning with Multiple Views*, 2005.

[30] R. Salakhutdinov and S. Roweis. Adaptive overrelaxed bound optimization methods. In *ICML'03*, 2003.

[31] A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining partitionings. In *AAAI 2002*.

[32] B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceeding of IJCAI-01*, 2001.

[33] A. Topchy, A. K. Jain, and W. Punch. Combining multiple weak clusterings. In *Proceedings of the Third IEEE International Conference on Data Mining*, page 331, 2003.

[34] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *ICML '01*, pages 577–584, 2001.

[35] J. Wang, H. Zeng, Z. Chen, H. Lu, L. Tao, and W.-Y. Ma. Recom: reinforcement clustering of multi-type interrelated data objects. In *SIGIR '03*, pages 274–281, 2003.

[36] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *ICML '07*, pages 1159–1166, 2007.