

Error-Robust Multi-View Clustering

Mehrnaz Najafi
Department of Computer Science
University of Illinois at Chicago
Chicago, IL, USA
mnajaf2@uic.edu

Lifang He
Weill Cornell Medicine College
Cornell University
New York, NY, USA
lifanghescut@gmail.com

Philip S. Yu
Department of Computer Science
University of Illinois at Chicago
Chicago, IL, USA
psyu@uic.edu

Abstract—In the era of big data, data may come from multiple sources, known as multi-view data. Multi-view clustering aims at generating better clusters by exploiting complementary and consistent information from multiple views rather than relying on the individual view. Due to inevitable system errors caused by data-captured sensors or others, the data in each view may be erroneous. Various types of errors behave differently and inconsistently in each view. More precisely, error could exhibit as noise and corruptions in reality. Unfortunately, none of the existing multi-view clustering approaches handle all of these error types. Consequently, their clustering performance is dramatically degraded. In this paper, we propose a novel Markov chain method for Error-Robust Multi-View Clustering (EMVC). By decomposing each view into a shared transition probability matrix and error matrix and imposing structured sparsity-inducing norms on error matrices, we characterize and handle typical types of errors explicitly. To solve the challenging optimization problem, we propose a new efficient algorithm based on Augmented Lagrangian Multipliers and prove its convergence rigorously. Experimental results on various synthetic and real-world datasets show the superiority of the proposed EMVC method over the baseline methods and its robustness against different types of errors.

Keywords—Multi-view learning; Robust; Clustering; Noise; Corruptions

I. INTRODUCTION

In the era of big data, data may have different views (i.e., variety), where observations are represented by multiple sources, known as *multi-view data*. For instance, specific news may be available on different broadcasting websites such as BBC and CNN so that each website represents a view of the same news. As another example, in Wikipedia, concept of dog may have multiple representations in the form of image and text. Fig. 1 shows the mentioned examples of multi-view data.



Figure 1: Multi-View data

Multi-view data commonly have the following properties:

- Each view may be represented by an arbitrary type and number of features. It may be also collected from diverse domains (variety of big data). Different views often contain *complementary* and *compatible* information to each other [1]. For instance, in Wikipedia, one view (image) consists of vision features, while another view (text) has textual features.
- Due to inevitable system errors caused by data extractors, each view may be *erroneous* (veracity of big data). Generally, error refers to the deviation between model assumption and data. It could exhibit as *noise* and *corruptions* in reality [2]. Fig. 2 illustrates three types of errors. Noise refers to *slight* perturbation of random subset of entries in data. Random corruptions indicate that a fraction of random entries are *grossly* perturbed, while sample-specific corruptions (or outliers) represent the phenomena that a fraction of the samples (or data points) in each view are far away from the real values. Real-world multi-view data can encounter any or combination of these error types.

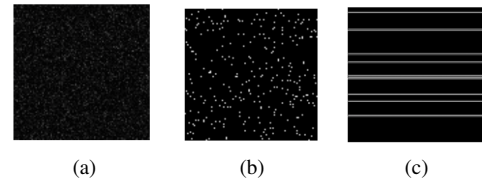


Figure 2: Three types of errors (what we show is a perturbed data matrix whose rows are samples and columns are features). (a) Noise (b) Random Corruptions (c) Sample-Specific Corruptions (or Outliers)

Due to ever increasing need of learning from multi-view data and lack of label information in many applications, clustering on multiple views has received considerable attention recently [3], [4], [1]. The problem is referred to as *multi-view clustering* aiming at finding compatible clusters of data points across all views. One of the most common algorithms used for multi-view clustering is *spectral clustering* [5].

Spectral clustering models the data which is in the form of a graph where nodes are data points and edges represent similarities between data points [5]. First, it projects all the

data points to a new low-dimensional space where they are easily separable. The new space is built with the eigen-decomposition of the Laplacian matrix of the graph. Then, it finds the clusters by applying another clustering algorithm like k -means. It is shown that spectral clustering can find the normalized min-cut of the graph [5]. Also, there is a hidden relation between spectral clustering and *Markov chains* [6]. In single-view clustering, Laplacian of the graph can be obtained by real relaxation of the combinatorial normalized cut [7]. Then, it is converted to a transition probability matrix which generates a Markov chain on the graph [7]. In the context of multi-view clustering, the transition probability matrix would need to get built across all views.

A challenging problem may arise when the views are erroneous, which makes the corresponding transition probability matrices being perturbed. It could then result in portion of data points being assigned to wrong clusters. To address multi-view clustering with noisy views, Xia *et al* proposed a method, named as RMSC [7]. This approach decomposes a transition probability matrix of each view into two parts: a shared transition probability matrix across all views and an error matrix which encodes the noise in the transition probability matrix in each view. The error matrix of each view captures the difference between the transition probabilities of that view and their correspondings in the shared transition probability matrix. RMSC assumes sparse representation error matrix via ℓ_1 norm.

One of the shortcomings of RMSC is that it *only* handles noise in data. Specifically, since it only imposes ℓ_1 norm on the error matrices, it cannot deal with sample-specific corruptions well. This is because error matrix with sample-specific corruptions has sparse row supports. Also, RMSC treats each error matrix independently. However, data may come from various sources, which could result in error matrices with inconsistent magnitude values, and thus degradation of clustering performance, when error matrices are treated independently [2].

To handle typical types of errors in multi-view data, we propose a novel Error-Robust Multi-View Clustering (EMVC) method based on Markov chains. Different from RMSC [7], EMVC is based on integration of low-rank decomposition and *group* ℓ_1 [8] and $\ell_{2,1}$ regularization terms, aiming to learn a shared transition probability matrix where the transition probability matrices of different views will be co-regularized to a common consensus, while improving the robustness of clustering.

In some cases, features of a certain view are more or less discriminative for clusters. As a result, error in more discriminative features could substantially decrease clustering performance. To improve robustness of clustering on this error, we impose *group* ℓ_1 norm on error matrix. In this way, in contrast to ℓ_1 norm, *group* ℓ_1 norm learns group-wise features importance of one view on each cluster and thus improves robustness of clustering against erroneous dis-

criminative features. By various experiments, we also claim that by using *group* ℓ_1 rather than ℓ_1 norm, the proposed EMVC method achieves better clustering performance both on non-erroneous and erroneous datasets.

To deal with sample-specific corruptions, EMVC imposes $\ell_{2,1}$ norm on error matrix because similar to [2], error matrix with this error type has sparse row supports. Furthermore, since data may come from multiple heterogeneous sources, error matrices could have inconsistent magnitude values. In contrast to RMSC, as suggested in [2], with the aim of increasing clustering performance, we enforce the column of each error matrix with respect to each view to have jointly consistent magnitude values by vertical concatenation of error matrices of all views.

The $\ell_{2,1}$ and *group* ℓ_1 norms are two non-smooth structured sparsity-inducing norms, which make the corresponding objective function of EMVC challenging to optimize. We present a reformulation of the objective function and propose a new efficient optimization algorithm based on the Augmented Lagrangian Multiplier [9] to optimize it. We also present a rigorous proof of convergence for the optimization technique. Our contributions can be summarized as follows:

- 1) To the best of our knowledge, EMVC is the first work that can address any or combination of the typical types of errors in multi-view clustering via combination of $\ell_{2,1}$ and *group* ℓ_1 norms. Since it is generally hard to know which type of error incurred in each view, it is important to have an all-encompassing approach that can handle any or combination of the typical error types.
- 2) EMVC is the first Markov chains method that does not treat error matrices independently. Independent treatment of error matrices could decrease clustering performance [2]. The proposed EMVC method enforces the error matrices in each view to have jointly consistent magnitude values.
- 3) We propose a new efficient optimization algorithm to solve the EMVC optimization problem, along with rigorous proof of convergence.
- 4) Through extensive experiments on synthetic and real-world datasets, we show that EMVC is superior to several state-of-the-art methods in the multi-view clustering and robust against typical error types.

II. PRELIMINARIES

In this section, we introduce some related concepts and notations. The mathematical notations used in the rest of the paper are summarized in Table I.

A. Transition Probability Matrix

Given a graph G with N nodes, a square matrix called transition probability matrix is defined over G that contains the transitions of a Markov chain. It is denoted as $\mathbf{P} \in \mathbb{R}^{N \times N}$. Each element of \mathbf{P} denotes a probability (i.e.,

Table I: List of basic symbols

Symbol	Definition and description
x	each lowercase letter represents a scale
\mathbf{x}	each boldface lowercase letter represents a vector
\mathbf{X}	each boldface uppercase letter represents a matrix
$\langle \cdot, \cdot \rangle$	denotes inner product
$\text{rank}(\cdot)$	denotes the rank of the matrix
$\text{Tr}(\cdot)$	denotes the trace of the matrix

$p_{i,j} \geq 0$), and all outgoing transitions from a specific state have to sum to one (i.e., $\sum_{j=1} p_{i,j} = 1$). Each row in \mathbf{P} is a distribution probability over the transitions of the corresponding state.

B. Spectral Clustering via Markov chains

Spectral clustering seeks clusters of data points in a weighted graph G where vertices are data points and edges represent similarity between two connecting data points. There is a relationship between spectral clustering and transition probability matrix [6]. Spectral clustering on graph G is equivalent to finding clusters on G such that the Markov random walk remains long within the same cluster and jumps infrequently between clusters [7].

In the context of clustering, a natural way to construct a transition probability matrix is to first build a similarity matrix \mathbf{S} between pairs of data points and then calculate the corresponding transition probability matrix \mathbf{P} by $\mathbf{P} = (\mathbf{D}^{-1}\mathbf{S})$, where \mathbf{D} denotes the degree matrix of graph G . One way to build a similarity matrix \mathbf{S} is to use Gaussian kernels [7]. Let $s_{i,j}$ denotes the similarity on a pair of data points \mathbf{x}_i and \mathbf{x}_j . It can be calculated as follows:

$$s_{i,j} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / \sigma^2) \quad (1)$$

where $\|\cdot\|_2$ denotes the l_2 -norm and σ^2 indicates the standard deviation (e.g., it could be set to median of Euclidean distance over all pairs of data points). Algorithm 1 summarizes the overall scheme for computing transition probability matrix.

Algorithm 1 Transition Probability Matrix Construction

Input: Data matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$

Output: Transition probability matrix $\mathbf{P} \in \mathbb{R}^{N \times N}$

```

1: for  $i = 1, \dots, N$  do
2:   for  $j = 1, \dots, N$  do
3:      $s_{i,j} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / \sigma^2)$ 
4:   end for
5: end for
6: for  $i = 1, \dots, N$  do
7:    $d_{i,i} = \sum_{j=1}^N s_{i,j}$ 
8: end for
9:  $\mathbf{P} = \mathbf{D}^{-1}\mathbf{S}$ 

```

The steps of spectral clustering via Markov chains is described in Algorithm 2 [10]. To perform clustering using

Markov chains, a crucial step is to build the transition probability matrix \mathbf{P} over graph G in Line 1. A stationary distribution of \mathbf{P} is obtained in Line 2. Two matrices \mathbf{L} and $\hat{\mathbf{D}}$ are computed in Lines 3 and 4. Finally, k -means must be performed on eigenvectors of the generalized eigenproblem $\mathbf{L}\mathbf{u} = \lambda\hat{\mathbf{D}}\mathbf{u}$ (Lines 5 and 6).

Algorithm 2 Spectral Clustering via Markov Chains

Input: Graph G

Output: Clustering results

- 1: Define a random walk over G with a transition probability matrix \mathbf{P} constructed by Algorithm 1
 - 2: Compute stationary distribution π satisfying $\pi = \mathbf{P}\pi$
 - 3: Build diagonal matrix $\hat{\mathbf{D}}$ with the i -th diagonal element as $\hat{d}_{i,i}$, such that $\hat{d}_{i,i} = \pi(i)$
 - 4: $\mathbf{L} = \hat{\mathbf{D}} - ((\hat{\mathbf{D}}\mathbf{P} + \mathbf{P}^T\hat{\mathbf{D}})/2)$
 - 5: Calculate R smallest generalized eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_r$ of the generalized eigenproblem $\mathbf{L}\mathbf{u} = \lambda\hat{\mathbf{D}}\mathbf{u}$
 - 6: Cluster \mathbf{U} by k -means to obtain clustering results, where \mathbf{U} is the matrix consisting of the vectors $\mathbf{u}_1, \dots, \mathbf{u}_r$.
-

III. ERROR-ROBUST MULTI-VIEW CLUSTERING

In this section, we will first systematically propose a novel error-robust multi-view algorithm for clustering, followed by a new efficient iterative algorithm to solve the formulated non-smooth objective function.

Problem. In the setting of clustering, given N distinct data points or samples with K related views, their views are denoted as $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \dots, \mathbf{X}^{(K)}$. The goal is to derive a clustering solution across all views. We assume that the features in each individual view are sufficient for obtaining most of the clustering information and each individual view might be erroneous.

A. Transition Probability Matrix Construction

Transition probability matrices have been used to model multiple views [7], [11]. There are several ways to build a transition probability matrix with respect to each view. We use Algorithm 1 to construct transition probability matrices with respect to each individual view.

B. Problem Formulation

Assuming that each individual view might be erroneous so that it causes wrong assignment of data points to clusters, each transition probability matrix $\mathbf{P}^{(k)}$ can be decomposed into two terms: a shared transition probability matrix $\hat{\mathbf{P}}$ and the error matrix $\mathbf{E}^{(k)}$ that indicates the error in the transition probabilities in view k .

$$\forall k, \mathbf{P}^{(k)} = \hat{\mathbf{P}} + \mathbf{E}^{(k)} \quad (2)$$

We use $\hat{\mathbf{P}}$ as the input transition probability matrix to the Markov chains method (i.e., Algorithm 2) to obtain clustering solution. Using the transition probability construction

method described in Algorithm 1, we get initial transition probability matrices with respect to each individual view.

In order to approximate the shared transition probability matrix while reducing its complexity, we minimize $\text{rank}(\hat{\mathbf{P}})$ (i.e., low rankness criterion). Since data may come from different sources, error matrices could have inconsistent magnitude values, which adversarially affects clustering performance [2]. To enforce the column of $\mathbf{E}^{(k)}$ in each view to have jointly consistent magnitude values, we vertically concatenate error matrices of views along their columns (i.e., $\mathbf{E} = [\mathbf{E}^{(1)}; \mathbf{E}^{(2)}; \dots; \mathbf{E}^{(K)}]$).

In some cases, error may appear in features of a specific view which are more or less discriminative for clustering. Compared to error in less discriminative features, error in more discriminative features degrade clustering performance significantly. For example, color features substantially affects the detection of traffic light and trees whereas they are irrelevant for finding cars in the context of image clustering. Thus, error in color features would substantially decrease clustering performance. To improve robustness of clustering against error in these features, we add group ℓ_1 norm [8] on \mathbf{E} . The group ℓ_1 norm can be defined as follows:

$$\|\mathbf{E}\|_{G1} = \sum_{i=1}^N \sum_{j=1}^K \|\mathbf{e}_i^j\|_2 \quad \text{s.t.} \quad \mathbf{E} = [\mathbf{E}^{(1)}; \mathbf{E}^{(2)}; \dots; \mathbf{E}^{(K)}] \quad (3)$$

where \mathbf{e}_i^j denotes $\mathbf{E}((j-1) \times N + 1 : j \times N, i)$ (i.e., the segment $((j-1) \times N + 1 : j \times N)$ of i -th column of \mathbf{E}). This norm uses ℓ_2 norm within each view and ℓ_1 norm between views. Thus, it enforces the sparsity between different views, i.e., if features of one view are not discriminative for clustering, Eq. (3) will assigns zero to them.

In sample-specific corruptions, \mathbf{E} has sparse row supports [2]. Thus, to handle error in specific samples, we add $\ell_{2,1}$ penalty [12] on \mathbf{E} . The $\ell_{2,1}$ norm is defined as follows:

$$\|\mathbf{E}\|_{2,1} = \sum_{i=1}^{K \times N} \|\mathbf{e}^i\|_2 \quad (4)$$

where \mathbf{e}^i denotes $\mathbf{E}(i, :)$ (i.e., the i -th row of \mathbf{E}).

Based on the above consideration, the objective function for obtaining a shared transition probability matrix is formulated as follows:

$$\begin{aligned} \min_{\hat{\mathbf{P}}, \mathbf{E}} \quad & \text{rank}(\hat{\mathbf{P}}) + \beta \|\mathbf{E}\|_{2,1} + \lambda \|\mathbf{E}\|_{G1} \\ \text{s.t.} \quad & \mathbf{E} = [\mathbf{E}^{(1)}; \mathbf{E}^{(2)}; \dots; \mathbf{E}^{(K)}], \\ & \mathbf{P}^{(i)} = \hat{\mathbf{P}} + \mathbf{E}^{(i)}, \hat{\mathbf{P}} \geq 0, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1} \end{aligned} \quad (5)$$

where $\text{rank}(\hat{\mathbf{P}})$ is the rank of $\hat{\mathbf{P}}$. $\mathbf{1}$ denotes the vector with all ones, β and λ are non-negative trade-off parameters. \mathbf{E} is obtained by concatenation of error matrices of views vertically. To enforce $\hat{\mathbf{P}}$ to be a transition probability matrix, two constraints $\hat{\mathbf{P}} \geq 0$ and $\hat{\mathbf{P}}\mathbf{1} = \mathbf{1}$ have been considered.

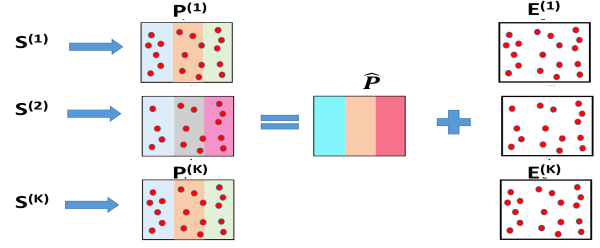


Figure 3: Overview of the proposed EMVC method

Since $\text{rank}(\hat{\mathbf{P}})$ is non-convex, the objective function in Eq. (5) is an instance of NP-hard problem. One natural way is to replace $\text{rank}(\hat{\mathbf{P}})$ with the trace norm $\|\hat{\mathbf{P}}\|_*$. The resulted objective function is as follows:

$$\begin{aligned} \min_{\hat{\mathbf{P}}, \mathbf{E}} \quad & \|\hat{\mathbf{P}}\|_* + \beta \|\mathbf{E}\|_{2,1} + \lambda \|\mathbf{E}\|_{G1} \\ \text{s.t.} \quad & \mathbf{E} = [\mathbf{E}^{(1)}; \mathbf{E}^{(2)}; \dots; \mathbf{E}^{(K)}], \\ & \mathbf{P}^{(i)} = \hat{\mathbf{P}} + \mathbf{E}^{(i)}, \hat{\mathbf{P}} \geq 0, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1} \end{aligned} \quad (6)$$

The trace norm is the convex envelope of the rank. Therefore, minimizing the trace norm of a matrix often applies the low-rank structure on that [13], [14]. Fig. 3 visualizes the proposed EMVC method. We first build initial transition probability matrices that might be erroneous. Then, we use decomposition via low-rankness and regularization to obtain a shared transition probability matrix.

C. Optimization Procedure

The objective function in Eq. (6) imposes a probabilistic simplex constraint on each of rows of $\hat{\mathbf{P}}$. We use an *Augmented Lagrangian Multiplier* scheme [9] to solve the optimization problem. By introducing an auxiliary variable \mathbf{Q} , the objective function in Eq. (6) can be stated equivalently as follows:

$$\begin{aligned} \min_{\hat{\mathbf{P}}, \mathbf{E}, \mathbf{Q}} \quad & \|\mathbf{Q}\|_* + \beta \|\mathbf{E}\|_{2,1} + \lambda \|\mathbf{E}\|_{G1} \\ \text{s.t.} \quad & \mathbf{P}^{(i)} = \hat{\mathbf{P}} + \mathbf{E}^{(i)}, i = 1, \dots, K. \\ & \hat{\mathbf{P}} \geq 0, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1}, \hat{\mathbf{P}} = \mathbf{Q} \end{aligned} \quad (7)$$

where $\mathbf{Q} \in \mathbb{R}^{N \times N}$. The corresponding augmented Lagrangian function of Eq. (7) is:

$$\begin{aligned} L(\hat{\mathbf{P}}, \mathbf{E}, \mathbf{Q}) = & \|\mathbf{Q}\|_* + \beta \|\mathbf{E}\|_{2,1} + \lambda \|\mathbf{E}\|_{G1} \\ & + (\mu/2) \|\hat{\mathbf{P}} - \mathbf{Q}\|_F^2 + \langle \mathbf{Z}, \hat{\mathbf{P}} - \mathbf{Q} \rangle \\ & + (\mu/2) \sum_{i=1}^K \|\hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)}\|_F^2 \\ & + \sum_{i=1}^K \langle \mathbf{Y}^{(i)}, \hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)} \rangle \end{aligned} \quad (8)$$

where $\mathbf{Z} \in \mathbb{R}^{N \times N}$ and $\mathbf{Y}^{(i)} \in \mathbb{R}^{N \times N}$ are the Lagrange multipliers. μ is an adaptive penalty parameter which can be adjusted efficiently according to [15].

Algorithm 3 Error Robust Multi-View Clustering (EMVC)

Input: $\lambda, \beta, \mathbf{P}^{(i)} \in \mathbb{R}^{N \times N} (i = 1, 2, \dots, K)$

Output: Clustering results

```

1:  $\hat{\mathbf{P}} = \mathbf{0}, \mathbf{Q} = \mathbf{0}, \mathbf{Z} = \mathbf{0}, \mathbf{Y}^{(i)} = \mathbf{0}, \mathbf{E}^{(i)} = rand,$ 
    $\mu = 10^{-6}, \rho = 1.9, max_{\mu} = 10^{10}, \epsilon = 10^{-8}$ 
2: while not converge do
3:    $\mathbf{C} = (1/(K+1))(\mathbf{Q} - \mathbf{Z}/\mu + \sum_{i=1}^K (\mathbf{P}^{(i)} - \mathbf{E}^{(i)} - \mathbf{Y}^{(i)}/\mu))$ 
4:   for  $i = 1, \dots, N$  do
5:     Run Algorithm 4 using  $\mathbf{c}^i$  as input to update  $\hat{\mathbf{p}}^i$ 
     Where  $\mathbf{c}^i/\hat{\mathbf{p}}^i$  is the  $i$ -th row of  $\mathbf{C}/\hat{\mathbf{P}}$ , respectively.
6:   end for
7:   for  $l = 1, \dots, N$  do
8:     Update  $\mathbf{e}_l$  by Eq. (15)
9:   end for
10:  Update  $\mathbf{Q}$  by Eq. (11)
11:  Update  $\mathbf{Z}$  by Eq. (21)
12:  for  $i = 1, \dots, K$  do
13:    Update  $\mathbf{Y}^{(i)}$  by Eq. (22)
14:  end for
15:   $\mu \leftarrow \min(\rho\mu, max_{\mu})$ 
16: end while
17: Apply Algorithm 2 on  $\hat{\mathbf{P}}$  to get clustering results

```

The iterative algorithm for solving Eq. (8) is shown in Algorithm 3. We detail each step as follows.

Solving \mathbf{Q} . When other variables are fixed, the objective function w.r.t. \mathbf{Q} can be stated as:

$$\min_{\mathbf{Q}} \|\mathbf{Q}\|_* + (\mu/2) \|\hat{\mathbf{P}} - \mathbf{Q}\|_F^2 + \langle \mathbf{Z}, \hat{\mathbf{P}} - \mathbf{Q} \rangle \quad (9)$$

The optimization problem in Eq. (9) is equivalent to the following objective function:

$$\min_{\mathbf{Q}} \|\mathbf{Q}\|_* + (\mu/2) \|\hat{\mathbf{P}} - \mathbf{Q} + \mathbf{Z}/\mu\|_F^2 \quad (10)$$

To solve Eq. (10), we use the Singular Value Threshold (SVD) method [16]. Let $\mathbf{U}\Sigma\mathbf{V}^T$ denote the SVD form of $(\hat{\mathbf{P}} + \mathbf{Z}/\mu)$, then \mathbf{Q} can be obtained using the following equation:

$$\mathbf{Q} = \mathbf{U}\mathbf{S}_{\sigma}(\sum)\mathbf{V}^T \quad (11)$$

where $\mathbf{S}_{\sigma} = \max(\mathbf{X} - \sigma, 0) + \min(\mathbf{X} + \sigma, 0)$ is the shrinkage operator [9].

Solving \mathbf{E} . When other variables are fixed, the objective function w.r.t. \mathbf{E} can be stated as:

$$\begin{aligned} \min_{\mathbf{E}} \beta \|\mathbf{E}\|_{2,1} + \lambda \|\mathbf{E}\|_{G1} + (\mu/2) \sum_{i=1}^K \|\hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)}\|_F^2 \\ + \sum_{i=1}^K \langle \mathbf{Y}^{(i)}, \hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)} \rangle \end{aligned} \quad (12)$$

The optimization problem in Eq. (12) can be stated equivalently as follows:

$$\min_{\mathbf{E}} (\beta/\mu) \|\mathbf{E}\|_{2,1} + (\lambda/\mu) \|\mathbf{E}\|_{G1} + (1/2) \|\mathbf{E} - \mathbf{B}\|_F^2 \quad (13)$$

where \mathbf{B} is constructed by vertically concatenating the matrices $\mathbf{P}^{(i)} - \hat{\mathbf{P}} - (1/\mu)\mathbf{Y}^{(i)}$ together along column. Taking the derivative of Eq. (13) w.r.t. \mathbf{E} and setting it to zero, we have the following result for $1 \leq l \leq N$:

$$(\beta/\mu)\hat{\mathbf{D}}\mathbf{e}_l + (\lambda/\mu)\mathbf{D}^l\mathbf{e}_l + (\mathbf{e}_l - \mathbf{p}_l^{(i)} + \hat{\mathbf{p}}_l - (1/\mu)\mathbf{y}_l^{(i)}) = \mathbf{0} \quad (14)$$

where \mathbf{e}_l denotes $\mathbf{E}(:, l)$ (i.e., l -th column of \mathbf{E}). Likewise, $\mathbf{p}_l^{(i)}$, $\hat{\mathbf{p}}_l$ and $\mathbf{y}_l^{(i)}$ indicate l -th column of $\mathbf{p}^{(i)}$, $\hat{\mathbf{p}}$ and $\mathbf{y}^{(i)}$, respectively. \mathbf{D}^l is a block diagonal matrix with the j -th diagonal block as $(1/(2\|\mathbf{e}_l^j\|_2)) \times \mathbf{I}$, \mathbf{I} is an identity matrix with size of N , \mathbf{e}_l^j is the j -th segment of \mathbf{e}_l and includes the representation errors in j -th view. $\hat{\mathbf{D}}$ is a diagonal matrix with the i -th diagonal element as $1/(2\|\mathbf{e}^i\|_2)$ where \mathbf{e}^i represents i -th row of \mathbf{E} . \mathbf{e}_l can be obtained by:

$$\mathbf{e}_l = ((\beta/\mu)\hat{\mathbf{D}} + (\lambda/\mu)\mathbf{D}^l + \mathbf{I})^{-1}(\mathbf{P}^{(i)} - \hat{\mathbf{P}} + (1/\mu)\mathbf{Y}^{(i)}) \quad (15)$$

Solving $\hat{\mathbf{P}}$. Fixing other variables, we need to solve the following objective function to obtain $\hat{\mathbf{P}}$:

$$\begin{aligned} \min_{\hat{\mathbf{P}}} \sum_{i=1}^K \langle \mathbf{Y}^{(i)}, \hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)} \rangle \\ + (\mu/2) \sum_{i=1}^K \|\hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)}\|_F^2 \\ + \langle \mathbf{Z}, \hat{\mathbf{P}} - \mathbf{Q} \rangle + (\mu/2) \|\hat{\mathbf{P}} - \mathbf{Q}\|_F^2 \quad \text{s.t.} \quad \hat{\mathbf{P}} \geq \mathbf{0}, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1} \end{aligned} \quad (16)$$

The objective function in Eq. (16) can be converted to the following equivalent form:

$$\begin{aligned} \min_{\hat{\mathbf{P}}} (\mu/2) \sum_{i=1}^K \|\hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)} + \mathbf{Y}^{(i)}/\mu\|_F^2 \\ + (\mu/2) \|\hat{\mathbf{P}} - \mathbf{Q} + \mathbf{Z}/\mu\|_F^2 \quad \text{s.t.} \quad \hat{\mathbf{P}} \geq \mathbf{0}, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1} \end{aligned} \quad (17)$$

For ease of presentation, we define a new variable \mathbf{C} as follows:

$$\mathbf{C} = (1/(K+1))(\mathbf{Q} - \mathbf{Z}/\mu + \sum_{i=1}^K (\mathbf{P}^{(i)} - \mathbf{E}^{(i)} - \mathbf{Y}^{(i)}/\mu)) \quad (18)$$

Then the objective function in Eq. (17) can be converted into the following equivalent form:

$$\min_{\hat{\mathbf{P}}} (1/2) \|\hat{\mathbf{P}} - \mathbf{C}\|_F^2 \quad \text{s.t.} \quad \hat{\mathbf{P}} \geq \mathbf{0}, \hat{\mathbf{P}}\mathbf{1} = \mathbf{1} \quad (19)$$

Algorithm 4 Proximal Operator with Simplex Constraint

Input: $\mathbf{c}^i \in \mathbb{R}^N$

Output: $\hat{\mathbf{p}}^i \in \mathbb{R}^N$

- 1: $\mathbf{u} = \text{Sort}(\mathbf{c}^i, 'descend')$
 - 2: $\hat{j} = \max\{j : 1 - \sum_{r=1}^j (u_r - u_j) \geq 0\}$
 - 3: $\sigma = (1/\hat{j})(\sum_{i=1}^{\hat{j}} u_i - 1)$
 - 4: **for** $j = 1, \dots, N$ **do**
 - 5: $\hat{p}_{i,j} = \max(c_{i,j} - \sigma, 0)$
 - 6: **end for**
-

Eq. (19) can be further rewritten as:

$$\min_{\hat{\mathbf{p}}^1, \dots, \hat{\mathbf{p}}^N} (1/2) \sum_{i=1}^N \|\hat{\mathbf{p}}^i - \mathbf{c}^i\|_F^2 \quad \text{s.t.} \quad \sum_{i=1}^K \hat{p}_{i,j} = 1, \hat{p}_{i,j} \geq 0 \quad (20)$$

where \mathbf{c}^i indicates the i -th row of matrix \mathbf{C} and $\hat{\mathbf{p}}^i$ denotes the i -th row of matrix $\hat{\mathbf{P}}$, respectively. The optimization problem in Eq. (20) has N independent subproblems. Each problem is a proximal operator problem with a probabilistic simplex constraint that can be efficiently solved by the projection algorithm [17]. The algorithm for this optimization procedure is shown in Algorithm 4.

Solving \mathbf{Z} . The Lagrangian multiplier \mathbf{Z} can be obtained using the following update:

$$\mathbf{Z} \leftarrow \mathbf{Z} + \mu(\hat{\mathbf{P}} - \mathbf{Q}) \quad (21)$$

Solving $\mathbf{Y}^{(i)}$. The Lagrangian multiplier $\mathbf{Y}^{(i)}$ can be obtained using the following update:

$$\mathbf{Y}^{(i)} \leftarrow \mathbf{Y}^{(i)} + \mu(\hat{\mathbf{P}} + \mathbf{E}^{(i)} - \mathbf{P}^{(i)}) \quad (22)$$

D. Computational and Convergence analysis

In Algorithm 3, Lines 2-5 update $\hat{\mathbf{P}}$ with quadratic complexity. Lines 6-8 update matrix \mathbf{E} . Instead of computing the matrix inverse with cubic complexity, we can solve a system of linear equations which have quadratic complexity in order to obtain e_l . If sufficient computational resources are available, each e_l ($1 \leq l \leq N$) can be computed in parallel with efficiency. Updating \mathbf{Q} in line 9 requires solving an SVD problem. This part can be computed with cubic complexity. Lagrangian Multipliers can be updated with quadratic complexity. Line 16 applies spectral clustering via Markov chains (i.e., Algorithm 2) on the shared transition probability matrix. This step can be done with cubic complexity. When sufficient computational resources are available and parallel computing is implemented, both SVD and linear equations can be solved efficiently.

For convergence analysis, the following theorem guarantees the convergence of Algorithm 3.

Theorem. Algorithm 3 decreases the objective value of Eq. (6) in each iteration.

Proof. To obtain $\hat{\mathbf{P}}_{t+1}$ (i.e., $\hat{\mathbf{P}}$ in $(t+1)$ -th iteration), according to Algorithm 3, we know that

$$\hat{\mathbf{P}}_{t+1} = \min_{\hat{\mathbf{P}}} \|\hat{\mathbf{P}}\|_* + \lambda \sum_{i=1}^N \mathbf{D}_{t+1}^i \|(\mathbf{E}_t)_i\|_2^2 + \beta \text{Tr}(\mathbf{E}_t^T \hat{\mathbf{D}}_{t+1} \mathbf{E}_t) \quad (23)$$

where \mathbf{E}_t represents \mathbf{E} at t -th iteration and $(\mathbf{E}_t)_i$ indicates i -th column of \mathbf{E}_t . According to Algorithm 3, to obtain \mathbf{E}_{t+1} , the following problem must be solved:

$$\hat{\mathbf{E}}_{t+1} = \min_{\mathbf{E}} \|\hat{\mathbf{P}}_{t+1}\|_* + \lambda \sum_{i=1}^N \mathbf{D}_{t+1}^i \|(\mathbf{E}_t)_i\|_2^2 + \beta \text{Tr}(\mathbf{E}^T \hat{\mathbf{D}}_{t+1} \mathbf{E}) \quad (24)$$

Considering Eq. (23) and Eq. (24), we have the following:

$$\begin{aligned} & \|\hat{\mathbf{P}}_{t+1}\|_* + \lambda \sum_{i=1}^N \mathbf{D}_{t+1}^i \|(\mathbf{E}_t)_i\|_2^2 + \beta \text{Tr}(\mathbf{E}_{t+1}^T \hat{\mathbf{D}}_{t+1} \mathbf{E}_{t+1}) \\ & \leq \|\hat{\mathbf{P}}_t\|_* + \lambda \sum_{i=1}^N \mathbf{D}_{t+1}^i \|(\mathbf{E}_t)_i\|_2^2 + \beta \text{Tr}(\mathbf{E}_t^T \hat{\mathbf{D}}_{t+1} \mathbf{E}_t) \end{aligned} \quad (25)$$

Substituting \mathbf{D} and $\hat{\mathbf{D}}$ by their definitions results in the following:

$$\begin{aligned} & \|\hat{\mathbf{P}}_t\|_* + \lambda \sum_{i=1}^N \sum_{j=1}^K \|(\mathbf{E}_t)_i^j\|_2^2 / (2 \|(\mathbf{E}_t)_i^j\|_2) \\ & + \beta \sum_{i=1}^{N \times K} \|(\mathbf{E}_t)^i\|_2^2 / (2 \|(\mathbf{E}_t)^i\|_2) \end{aligned} \quad (26)$$

where $(\mathbf{E}_t)_i^j$ denotes the segment $((j-1)*N+1 : j*N)$ of i^{th} row of \mathbf{E}_t and $(\mathbf{E}_t)^i$ indicates i^{th} row of \mathbf{E}_t . We can derive the following because if we define $f(x) = x - x^2/2\alpha$, then $f(x) \leq f(\alpha)$:

$$\begin{aligned} & \sum_{j=1}^K \|(\mathbf{E}_{t+1})_i^j\|_2 - \sum_{j=1}^K \|(\mathbf{E}_{t+1})_i^j\|_2^2 / (2 \|(\mathbf{E}_t)_i^j\|_2) \\ & \leq \sum_{j=1}^K \|(\mathbf{E}_t)_i^j\|_2 - \sum_{j=1}^K \|(\mathbf{E}_t)_i^j\|_2^2 / (2 \|(\mathbf{E}_t)_i^j\|_2) \end{aligned} \quad (27)$$

and

$$\begin{aligned} & \sum_{i=1}^{K \times N} \|(\mathbf{E}_{t+1})^i\|_2 - \sum_{i=1}^{K \times N} \|(\mathbf{E}_{t+1})^i\|_2^2 / (2 \|(\mathbf{E}_t)^i\|_2) \\ & \leq \sum_{i=1}^{K \times N} \|(\mathbf{E}_t)^i\|_2 - \sum_{i=1}^{K \times N} \|(\mathbf{E}_t)^i\|_2^2 / (2 \|(\mathbf{E}_t)^i\|_2) \end{aligned} \quad (28)$$

If we add all Eq. (25-28) on both sides, we obtain the inequality that objective value at iteration $t+1$ is less than objective value at iteration t . Therefore, we can conclude that the objective value decreases in each iteration and Algorithm 3 converges.

IV. EXPERIMENTAL EVALUATION

To empirically evaluate performance of the proposed EMVC method, we conduct extensive experiments on synthetic and publicly available real-world multi-view datasets and compare with six state-of-the-art methods: (1) **Best Single View (BSV)** performs standard k -means on the most informative view. (2) **Feature Concatenation (Feat. Concat.)** concatenates features of all views and then runs standard k -means clustering on the concatenated feature representations. (3) **Kernel Addition** constructs kernel matrix for each individual view and then obtains the average of these matrices to get a single kernel matrix for spectral clustering. (4) **Co-regularized Spectral Clustering (Co-Reg)** performs centroid based and pairwise co-regularized spectral clustering via Gaussian kernel [18]. The co-regularization parameter λ is tuned by searching a range of $\{0.01, 0.02, 0.03, 0.04, 0.05\}$ as suggested by the authors. (5) **Robust Multi-View Spectral Clustering via Low-Rank and Sparse Decomposition (RMSC)** uses low-rank decomposition and ℓ_1 norm [7]. The regularization parameter λ is tuned by searching the range $\{10^{-3}, 10^{-2}, \dots, 10^2, 10^3\}$ as suggested by the authors (we keep λ the same for all views). The parameter σ^2 is set to the median of all Euclidean distances over all pairs of data points for each individual view as suggested by the authors. (6) **Parameter-Free Auto-Weighted Multiple Graph Learning (AMGL)** finds cluster indicator matrix over all views by applying normalized cut algorithms on the graphs of views [19].

We implement four versions of the proposed EMVC method to investigate the effectiveness of its component terms in multi-view learning: EMVC by only using the first term in Eq. (6) “EMVC(*)”, EMVC by using trace norm and only imposing the group ℓ_1 norm “EMVC(g_1)”, EMVC by using trace norm and only imposing $\ell_{2,1}$ norm “EMVC($\ell_{2,1}$)” and the full version of EMVC based on Eq. (6). We apply grid search to identify optimal values for each regularization hyperparameter from $\{10^{-9}, 10^{-8}, \dots, 10^8, 10^9\}$. The standard deviation is set to the median of all Euclidean distances over all pairs of data points for each individual view.

We use different evaluation metrics including **F-Score**, **Precision**, **Recall**, **Normalized Mutual Information (NMI)**, **Entropy**, **Accuracy**, **Adjusted Rand-Index (AR)** for the purpose of comprehensive evaluation [18], [20]. All of these measure except for Entropy are positive measures, which indicates that larger values stand for better performance. For Entropy, smaller values indicate better performance. Different measurements reveal different properties. Thus, we can obtain comprehensive view from the results.

Each experiment is repeated for five times, and the mean and standard deviation of each metric in each dataset are reported. We then use k -means to obtain final clustering solution. Since k -means is sensitive to initial seed selection,

we run k -means 20 times on each dataset.

A. Experiments on Real-World Datasets

We conduct experiments on the following publicly available real-world datasets. Statistics of the real-world datasets are summarized in Table II (Max # features indicates maximum number of features over all views of the dataset).

Table II: Statistics of the real-world multi-view datasets

Dataset	# data points	Max # features	# views	# clusters
webKB	1051	3000	2	2
FOX	1523	2711	2	4
CNN	2107	3695	2	7
Citeseer	3312	6654	2	6
CCV	9317	5000	3	20

WebKB¹: This dataset contains webpages collected from Texas, Cornell, Washington and Wisconsin universities. Each webpage is described by the content view and link view.

FOX²: The dataset is crawled from FOX web news. Each instance is represented in two views: the text view and the image view. Titles, abstracts, and text body contents are extracted as the text view data, and the image included in the article is stored as the image view data.

CNN³: This dataset is crawled from CNN web news. For this dataset, titles, abstracts, and text body contents are extracted as the text view data. Also, the image included in the article is stored as the image view data.

Citeseer⁴: It contains a selection of the Citeseer dataset. The papers were selected in a way that in the final corpus every paper cites or is cited by at least one other paper. The text view consists of title and abstract of a paper; the link view contains inbound and outbound references.

CCV⁵: This high rank dataset contains 9317 videos over 20 semantic categories. Two views contains visual features, while the third view consists of audio features.

Tables III and IV report the performance comparison on the real-world datasets. From these tables, we have several observations. First, EMVC is always better than the baselines by a large margin. Specifically, compared with RMSC, even EMVC(g_1) can do a lot better in some of the datasets like FOX, CNN and Citeseer. This observation is consistent with our analysis in that group ℓ_1 achieves better performance than ℓ_1 . Second, the full version of EMVC is superior to all its three degenerative versions. This validates the correctness of our objective function and demonstrates the importance of having an all-encompassing approach.

B. Experiments on Synthetic Noisy Dataset

Using similar settings in [18], the synthetic dataset consists of two views and the data in each view is partitioned

¹<http://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/>

²<https://sites.google.com/site/qianmingjie/home/datasets/>

³<https://sites.google.com/site/qianmingjie/home/datasets/>

⁴<http://linqs.cs.umd.edu/projects/lbc/index.html>

⁵<http://www.ee.columbia.edu/ln/dvmm/CCV/>

Table III: Comparison results on the real-world datasets - part 1(mean (standard deviation))

Dataset	Method	F-Score \uparrow	Precision \uparrow	Recall \uparrow	NMI \uparrow	Entropy \downarrow	Accuracy \uparrow	AR \uparrow
WebKB	BSV	0.889(0.000)	0.889(0.000)	0.889(0.000)	0.532(0.000)	0.406(0.000)	0.913(0.000)	0.618(0.000)
	Feat. Concat.	0.947(0.000)	0.947(0.000)	0.947(0.000)	0.717(0.000)	0.214(0.000)	0.963(0.000)	0.845(0.000)
	Kernel Addition	0.946(0.000)	0.946(0.000)	0.946(0.000)	0.717(0.000)	0.214(0.000)	0.963(0.000)	0.845(0.000)
	Co-Reg	0.949(0.000)	0.949(0.000)	0.949(0.000)	0.733(0.000)	0.195(0.000)	0.965(0.000)	0.853(0.000)
	AMGL	0.794(0.000)	0.794(0.000)	0.794(0.000)	0.015(0.000)	0.752(0.000)	0.783(0.000)	0.013(0.000)
	RMSC	0.956(0.000)	0.956(0.000)	0.956(0.000)	0.758(0.000)	0.189(0.000)	0.970(0.000)	0.871(0.000)
	EMVC (*)	0.568(0.000)	0.568(0.000)	0.568(0.000)	0.000(0.000)	0.757(0.000)	0.511(0.000)	0.000(0.000)
	EMVC (g_1)	0.949(0.000)	0.949(0.000)	0.949(0.000)	0.731(0.000)	0.199(0.000)	0.965(0.000)	0.853(0.000)
	EMVC ($\ell_{2,1}$)	0.954(0.000)	0.954(0.000)	0.954(0.000)	0.759(0.000)	0.175(0.000)	0.969(0.000)	0.870(0.000)
FOX	EMVC	0.959(0.000)	0.959(0.000)	0.959(0.000)	0.776(0.000)	0.164(0.000)	0.972(0.000)	0.883(0.000)
	BSV	0.718(0.000)	0.718(0.000)	0.718(0.000)	0.672(0.000)	0.626(0.000)	0.758(0.000)	0.599(0.000)
	Feat. Concat.	0.314(0.000)	0.314(0.000)	0.314(0.000)	0.041(0.000)	1.787(0.000)	0.356(0.000)	0.050(0.000)
	Kernel Addition	0.358(0.000)	0.358(0.000)	0.358(0.000)	0.103(0.000)	1.669(0.000)	0.460(0.000)	0.113(0.000)
	Co-Reg	0.477(0.006)	0.477(0.006)	0.477(0.006)	0.242(0.002)	1.410(0.002)	0.547(0.000)	0.262(0.000)
	AMGL	0.456(0.000)	0.456(0.000)	0.456(0.000)	0.010(0.000)	1.857(0.000)	0.419(0.000)	0.001(0.000)
	RMSC	0.364(0.005)	0.364(0.005)	0.364(0.005)	0.141(0.000)	1.593(0.001)	0.401(0.001)	0.127(0.000)
	EMVC (*)	0.270(0.009)	0.270(0.009)	0.270(0.009)	0.002(0.002)	1.862(0.002)	0.267(0.003)	0.000(0.000)
	EMVC (g_1)	0.761(0.005)	0.761(0.005)	0.761(0.005)	0.691(0.003)	0.565(0.002)	0.818(0.004)	0.664(0.002)
CNN	EMVC ($\ell_{2,1}$)	0.761(0.004)	0.761(0.004)	0.761(0.004)	0.691(0.007)	0.565(0.004)	0.818(0.003)	0.664(0.005)
	EMVC	0.761(0.010)	0.761(0.010)	0.761(0.010)	0.691(0.004)	0.565(0.003)	0.818(0.003)	0.664(0.002)
	BSV	0.388(0.001)	0.388(0.001)	0.388(0.001)	0.405(0.008)	1.736(0.012)	0.486(0.007)	0.228(0.008)
	Feat. Concat.	0.171(0.000)	0.171(0.000)	0.171(0.000)	0.037(0.000)	2.621(0.001)	0.219(0.001)	0.023(0.000)
	Kernel Addition	0.175(0.000)	0.175(0.000)	0.175(0.000)	0.046(0.000)	2.597(0.002)	0.233(0.000)	0.026(0.000)
	Co-Reg	0.200(0.005)	0.200(0.005)	0.200(0.005)	0.076(0.002)	2.513(0.003)	0.276(0.002)	0.056(0.004)
	AMGL	0.250(0.002)	0.250(0.002)	0.250(0.002)	0.031(0.001)	2.667(0.003)	0.239(0.004)	0.000(0.001)
	RMSC	0.219(0.010)	0.219(0.010)	0.219(0.010)	0.122(0.000)	2.388(0.001)	0.300(0.000)	0.078(0.000)
	EMVC (*)	0.149(0.003)	0.149(0.003)	0.149(0.003)	0.003(0.004)	2.716(0.004)	0.165(0.002)	0.000(0.000)
	EMVC (g_1)	0.557(0.004)	0.557(0.004)	0.557(0.004)	0.536(0.002)	1.279(0.005)	0.655(0.005)	0.472(0.000)
	EMVC ($\ell_{2,1}$)	0.558(0.004)	0.558(0.004)	0.558(0.004)	0.536(0.003)	1.281(0.004)	0.656(0.001)	0.472(0.001)
	EMVC	0.560(0.013)	0.560(0.013)	0.560(0.013)	0.542(0.005)	1.264(0.003)	0.657(0.002)	0.474(0.002)

into two clusters. Eq. (29) shows cluster means and covariances for each view. In each view, the two clusters overlap, which is the source of noise in the transition probabilities of each view. First, we choose the cluster that each sample belongs to, and then produce the views from a mixture of two bivariate Gaussian distributions. For each view, we sample 500 data points from each of the clusters.

$$\mu_1^{(1)} = (1, 1), \mu_2^{(1)} = (2, 2), \mu_1^{(2)} = (2, 2), \mu_2^{(2)} = (1, 1)$$

$$\sum_* = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{pmatrix}, \sum_{**} = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.6 \end{pmatrix} \quad (29)$$

where $\mu_j^{(i)}$ denotes cluster means for cluster j in view i . \sum_* is covariance for first and second clusters in first and second views, respectively. \sum_{**} indicates covariance for second and first clusters in first and second views, respectively. Table V presents the comparison results on this dataset. With this type of noise, the proposed EMVC method shows superior clustering performance over all the baselines.

C. Experiments on Erroneous Real-World Datasets

To evaluate robustness of the proposed EMVC method on noise and random corruptions, we add *white Gaussian noise* with different signal-to-noise ratio $\{0.01, 0.1, 1, 10, 100\}$ on FOX (denoted as NRC-FOX) and CNN (denoted

as NRC-CNN). Fig. 4 shows clustering performance of the methods with various signal-to-noise ratio on the contaminated datasets. We can see that EMVC consistently achieves superior performance over the baselines (we only show the results for full version of EMVC, which is superior to its degenerative versions). This observation demonstrates that EMVC is robust against random noise and corruptions.

We investigate robustness of the proposed EMVC method against sample-specific corruptions on FOX (denoted as SSC-FOX) and CNN (denoted as SSC-CNN). For this experiment, we randomly select a small portion of samples (2%, 6% and 10%) and replace their feature values in all views by random values. This setting is similar to generation of attribute outliers in [21]. Fig. 4 shows clustering performance of the methods on sample-specific corruptions. The proposed EMVC method outperforms the baselines against this type of error (we only show the results for full version of EMVC, which is superior to its degenerative versions). This is mainly because of $\ell_{2,1}$ norm in our objective function, which has sparse row supports.

D. Hyperparameter Analysis

To explore the effects of the hyperparameters on the performance, we run experiments on real-world datasets with different values for $\lambda \in \{10^{-9}, 10^{-8}, \dots, 10^8, 10^9\}$ and $\beta \in \{10^{-9}, 10^{-8}, \dots, 10^8, 10^9\}$ and report the average accuracy in Fig. 5. In this Figure, each grid with different

Table IV: Comparison results on the real-world datasets - part 2(mean (standard deviation))

Dataset	Method	F-Score \uparrow	Precision \uparrow	Recall \uparrow	NMI \uparrow	Entropy \downarrow	Accuracy \uparrow	AR \uparrow
Citeseer	BSV	0.322(0.000)	0.322(0.000)	0.322(0.000)	0.199(0.000)	2.013(0.000)	0.443(0.000)	0.180(0.000)
	Feat. Concat.	0.326(0.001)	0.326(0.001)	0.326(0.001)	0.204(0.002)	2.001(0.001)	0.452(0.001)	0.185(0.000)
	Kernel Addition	0.346(0.002)	0.346(0.002)	0.346(0.002)	0.232(0.003)	1.943(0.002)	0.456(0.001)	0.200(0.001)
	Co-Reg	0.356(0.009)	0.356(0.009)	0.356(0.009)	0.174(0.010)	2.088(0.005)	0.378(0.003)	0.123(0.003)
	AMGL	0.303(0.010)	0.303(0.010)	0.303(0.010)	0.005(0.009)	2.517(0.007)	0.213(0.002)	0.000(0.001)
	RMSC	0.271(0.011)	0.271(0.011)	0.271(0.011)	0.154(0.005)	2.139(0.009)	0.365(0.002)	0.105(0.001)
	EMVC (*)	0.172(0.020)	0.172(0.020)	0.172(0.020)	0.001(0.011)	2.519(0.009)	0.183(0.003)	0.000(0.002)
	EMVC (g_1)	0.386(0.006)	0.386(0.006)	0.386(0.006)	0.283(0.007)	1.800(0.005)	0.532(0.004)	0.251(0.002)
	EMVC ($\ell_{2,1}$)	0.388(0.007)	0.388(0.007)	0.388(0.007)	0.284(0.008)	1.802(0.004)	0.535(0.003)	0.254(0.002)
CCV	EMVC	0.390(0.007)	0.390(0.007)	0.390(0.007)	0.286(0.011)	1.803(0.002)	0.537(0.004)	0.256(0.002)
	BSV	0.119(0.001)	0.119(0.001)	0.119(0.001)	0.177(0.001)	3.466(0.003)	0.181(0.006)	0.069(0.002)
	Feat. Concat.	0.096(0.001)	0.096(0.001)	0.096(0.001)	0.119(0.001)	3.739(0.010)	0.170(0.002)	0.023(0.001)
	Kernel Addition	0.124(0.002)	0.124(0.002)	0.124(0.002)	0.171(0.001)	3.496(0.005)	0.189(0.009)	0.072(0.002)
	Co-Reg	0.119(0.009)	0.119(0.009)	0.119(0.009)	0.176(0.001)	3.473(0.075)	0.180(0.010)	0.068(0.040)
	AMGL	0.080(0.010)	0.080(0.010)	0.080(0.010)	0.089(0.001)	3.901(0.009)	0.165(0.006)	0.019(0.002)
	RMSC	0.130(0.005)	0.130(0.005)	0.130(0.005)	0.203(0.002)	3.225(0.020)	0.196(0.005)	0.082(0.005)
	EMVC (*)	0.070(0.005)	0.070(0.005)	0.070(0.005)	0.085(0.006)	4.001(0.004)	0.152(0.009)	0.012(0.000)
	EMVC (g_1)	0.131(0.004)	0.131(0.004)	0.131(0.004)	0.210(0.007)	3.100(0.003)	0.198(0.004)	0.090(0.004)
	EMVC ($\ell_{2,1}$)	0.131(0.004)	0.131(0.004)	0.131(0.004)	0.211(0.006)	3.090(0.004)	0.198(0.005)	0.090(0.002)
	EMVC	0.141(0.009)	0.141(0.009)	0.141(0.009)	0.300(0.009)	2.987(0.002)	0.203(0.008)	0.091(0.004)

Table V: Comparison results on the synthetic dataset (mean (standard deviation))

Method	F-Score \uparrow	Precision \uparrow	Recall \uparrow	NMI \uparrow	Entropy \downarrow	Accuracy \uparrow	AR \uparrow
BSV	0.655(0.000)	0.655(0.000)	0.655(0.000)	0.246(0.000)	0.758(0.000)	0.771(0.000)	0.293(0.000)
Feat. Concat.	0.748(0.000)	0.748(0.000)	0.748(0.000)	0.424(0.000)	0.581(0.000)	0.849(0.000)	0.486(0.000)
Kernel Addition	0.760(0.000)	0.760(0.000)	0.760(0.000)	0.439(0.000)	0.564(0.000)	0.859(0.000)	0.515(0.000)
Co-Reg	0.750(0.000)	0.750(0.000)	0.750(0.000)	0.437(0.000)	0.569(0.000)	0.850(0.000)	0.489(0.000)
AMGL	0.579(0.000)	0.579(0.000)	0.579(0.000)	0.116(0.003)	0.883(0.003)	0.696(0.002)	0.153(0.004)
RMSC	0.736(0.000)	0.736(0.000)	0.736(0.000)	0.375(0.000)	0.624(0.000)	0.844(0.000)	0.472(0.000)
EMVC (*)	0.499(0.000)	0.499(0.000)	0.499(0.000)	0.000(0.000)	1.000(0.000)	0.501(0.000)	0.000(0.000)
EMVC (g_1)	0.730(0.000)	0.730(0.000)	0.730(0.000)	0.366(0.000)	0.634(0.000)	0.840(0.000)	0.461(0.000)
EMVC ($\ell_{2,1}$)	0.730(0.000)	0.730(0.000)	0.730(0.000)	0.366(0.000)	0.634(0.000)	0.840(0.000)	0.461(0.000)
EMVC	0.762(0.000)	0.762(0.000)	0.762(0.000)	0.449(0.000)	0.555(0.000)	0.860(0.000)	0.517(0.000)

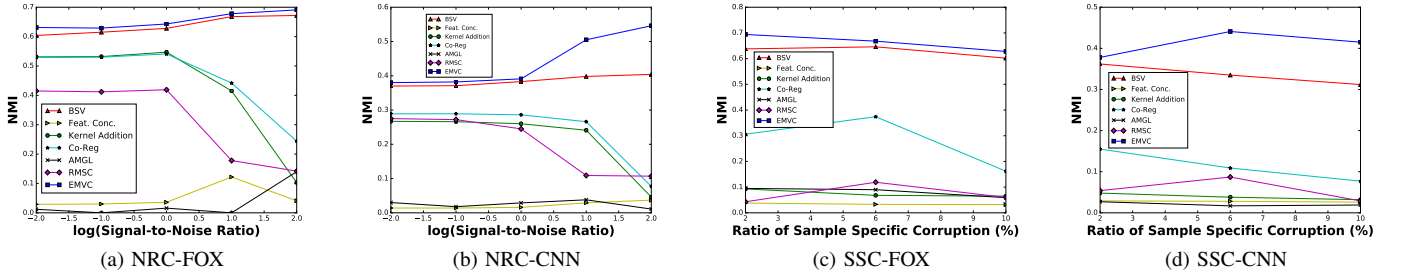


Figure 4: Clustering performance of erroneous real-world datasets

shades of colors reflects the clustering quality, where yellow means excellent quality. We can see that the performance is fairly stable. EMVC enjoys more promising results when $\lambda > 10^{-3}$ and $\beta > 10^{-3}$, while it is almost insensitive to the hyperparameters in that range.

V. RELATED WORK

Existing methods for multi-view clustering can be classified into two categories: 1) centralized approaches; 2) distributed approaches [22]. The centralized approaches constructs a new shared representation (i.e., common consensus) across all views [3], [11], [18], [7], [19]. For example, Bickel

and Scheffer presented an algorithm that interchanges the cluster information among different views [3]. Xia et al. proposed a multi-view clustering method, named as RMSC, that recovers shared transition probability matrix, in favor of low-rank and ℓ_1 regularization [7]. The proposed EMVC method belongs to this category. Different from RMSC, EMVC builds a shared transition probability matrix by integrating decomposition and group ℓ_1 and $\ell_{2,1}$ norms. EMVC also handles typical error types well.

The distributed approaches often build separate learners for each individual view and use the information in each

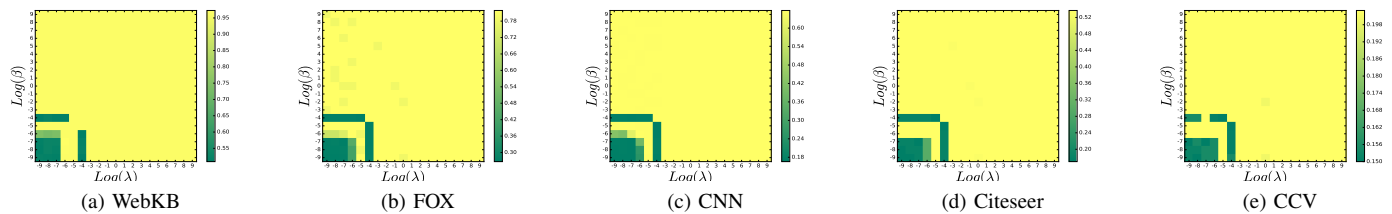


Figure 5: Sensitivity analysis of regularization hyperparameters (Accuracy)

learner to apply constraints on other views [18]. Kumar et al., proposed an approach to combine graphs of each individual view by pairwise co-regularization to achieve better clustering solution [18]. EMVC differs from this category of approaches in a way that it does not construct separate learners. Instead, it recovers a shared transition probability matrix across all views.

VI. CONCLUSION

In this paper, we developed a Markov chains method named EMVC for multi-view clustering via a low rank decomposition and two regularization terms. EMVC has several advantages over existing multi-view clustering methods. First, it handles typical types of error well. Second, an iterative optimization framework is proposed for EMVC which is proved to converge. Compared to the existing state-of-the-art multi-view clustering approaches, EMVC showed better performance on five real-world datasets.

REFERENCES

- [1] J. Zhao, X. Xie, X. Xu, and S. Sun, “Multi-view learning overview: Recent progress and new challenges,” *Information Fusion*, vol. 38, pp. 43–54, 2017.
- [2] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, “Robust recovery of subspace structures by low-rank representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, Jan 2013.
- [3] S. Bickel and T. Scheffer, “Multi-view clustering,” in *ICDM*, vol. 4, 2004, pp. 19–26.
- [4] V. R. De Sa, “Spectral clustering with two views,” in *ICML workshop on learning with multiple views*, 2005, pp. 20–27.
- [5] A. Y. Ng, M. I. Jordan, Y. Weiss *et al.*, “On spectral clustering: Analysis and an algorithm,” in *NIPS*, vol. 14, no. 2, 2001, pp. 849–856.
- [6] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
- [7] R. Xia, Y. Pan, L. Du, and J. Yin, “Robust multi-view spectral clustering via low-rank and sparse decomposition,” in *AAAI*, 2014, pp. 2149–2155.
- [8] J. Huang and T. Zhang, “The benefits of group sparsity,” *arXiv preprint arXiv:0901.2962*, 2009.
- [9] Z. Lin, M. Chen, and Y. Ma, “The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices,” *arXiv preprint arXiv:1009.5055*, 2010.
- [10] D. Zhou, J. Huang, and B. Schölkopf, “Learning from labeled and unlabeled data on a directed graph,” in *ICML*. ACM, 2005, pp. 1036–1043.
- [11] D. Zhou and C. J. Burges, “Spectral clustering and transductive learning with multiple views,” in *ICML*. ACM, 2007, pp. 1159–1166.
- [12] F. Nie, H. Huang, X. Cai, and C. H. Ding, “Efficient and robust feature selection via joint $\ell_{2,1}$ -norms minimization,” in *NIPS*, 2010, pp. 1813–1821.
- [13] M. Fazel, H. Hindi, and S. P. Boyd, “A rank minimization heuristic with application to minimum order system approximation,” in *American Control Conference, 2001. Proceedings of the 2001*, vol. 6. IEEE, 2001, pp. 4734–4739.
- [14] N. Srebro, J. D. Rennie, and T. S. Jaakkola, “Maximum-margin matrix factorization,” in *NIPS*, vol. 17, 2004, pp. 1329–1336.
- [15] Z. Lin, R. Liu, and Z. Su, “Linearized alternating direction method with adaptive penalty for low-rank representation,” in *NIPS*, 2011, pp. 612–620.
- [16] J.-F. Cai, E. J. Candès, and Z. Shen, “A singular value thresholding algorithm for matrix completion,” *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.
- [17] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, “Efficient projections onto the ℓ_1 -ball for learning in high dimensions,” in *ICML*. ACM, 2008, pp. 272–279.
- [18] A. Kumar, P. Rai, and H. Daume, “Co-regularized multi-view spectral clustering,” in *NIPS*, 2011, pp. 1413–1421.
- [19] F. Nie, J. Li, X. Li *et al.*, “Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification,” *IJCAI*, 2016.
- [20] X. Cao, C. Zhang, H. Fu, S. Liu, and H. Zhang, “Diversity-induced multi-view subspace clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 586–594.
- [21] H. Zhao and Y. Fu, “Dual-regularized multi-view outlier detection,” in *IJCAI*, 2015, pp. 4077–4083.
- [22] B. Long, P. S. Yu, and Z. Zhang, “A general model for multiple view unsupervised learning,” in *SIAM international conference on data mining*. SIAM, 2008, pp. 822–833.