Partial Multi-View Clustering Using Graph Regularized NMF

Nishant Rai
Indian Institute of Technology
Kanpur

Sumit Negi
Xerox Research Centre
India

Santanu Chaudhury
Indian Institute of Technology
Delhi

Om Deshmukh
Xerox Research Centre
India

***Abstract*—Real-world datasets consist of data representations (*views*) from different sources which often provide information complementary to each other. *Multi-view* learning algorithms aim at exploiting the complementary information present in different views for clustering and classification tasks. Several *multi-view clustering* methods that aim at partitioning objects into clusters based on multiple representations of the object have been proposed. Almost all of the proposed methods assume that each example appears in all views or at least there is one view containing all examples. In real-world settings this assumption might be too restrictive. Recent work on *P*artial *V*iew *C*lustering addresses this limitation by proposing a Non-negative Matrix Factorization based approach called PVC. Our work extends the PVC work in two directions. First, the current PVC algorithm is designed specifically for two-view datasets. We extend this algorithm for the k partial-view scenario. Second, we extend our k partial-view algorithm to include *view specific* graph laplacian regularization. This enables the proposed algorithm to exploit the intrinsic geometry of the data distribution in each view. The proposed method, which is referred to as GPMVC (Graph Regularized Partial Multi-View Clustering), is compared against 7 baseline methods (including PVC) on 5 publicly available text and image datasets. In all settings the proposed GPMVC method outperforms all baselines. For the purpose of reproducibility, we provide access to our code.**

I. Introduction

Many real world datasets naturally comprise of different (heterogenous) representations or *views*. For instance, image dataset can be represented using a collection of heterogenous features (or views) e.g. colour descriptor, local binary patterns, local shape descriptor etc. Similarly, a corpus of scientific documents can be represented using words in the documents, document metadata (e.g. title, author and journal) and the co-citation network graph. Observing that these multiple representations or views often provide compatible and complementary information, *multi-view* learning methods have been proposed that integrate the information present in the different views for tasks such as clustering and classification.

Considering its practical applicability, the problem of unsupervised learning from multiple-views of unlabeled data (referred to as *multi-view clustering*) has attracted a lot of attention in the past. The goal of multi-view clustering is to partition objects into clusters based on multiple representations of the object. A number of approaches which are either *spectral* [6], [7] or *subspace* [5], [3] based have been proposed. There has also been efforts to enable these techniques for challenges which real-world data poses. Some prominent examples include the work on scale-up [9], handling corrupted/noisy views [12], missing and unmapped views [13] etc. Recently authors investigated the problem of *partial-views* [8]. Most previous studies on multi-view clustering either assumed that all examples have full information in all views, or that there exists at least one view which contains all the examples. The author's [8] investigate the case where every view suffers from some missing information, which results in what the author's refer to as *partial examples*. To illustrate this point, consider bi-lingual documents (where the two languages can be seen as two views of a document) - in the *partial view* setup many documents might have only view available i.e for some documents only a single language translation of the document is available. The proposed **P**artial **V**iew **C**lustering (PVC) algorithm was shown to be effective in scenarios with partial examples.

Our work extends the partial-view work in two concrete ways. First, the current PVC algorithm is designed specifically for two-view datasets. We extend this algorithm for the k partial-view scenario. Second, we extend our k partial-view algorithm to include *view-specific* graph laplacian regularization. This enables the proposed algorithm to exploit the intrinsic geometry of the data distribution in each view. We compare our method against 7 state-of-art multi-view clustering methods on both text and image datasets. The baseline methods and datasets used in this work are more exhaustive than what was used in the PVC work. Our experiments show that the proposed **GPMVC** method outperforms PVC and other competitive baseline methods on all the 5 datasets. We also provide insights into how our algorithm performs when there is a *skew* in the distribution of partial examples across views[1] something not discussed in earlier work.

Paper Organization: Prior work is covered in Section II. Our proposed method is described in Section III and Section IV. Experiments and results are discussed in Section V. Conclusion and future work is presented in Section VI.

II. Prior and Related Work

For multi-view clustering a number of approaches which are either spectral or subspace based have been proposed. We briefly review a few of these approaches.

[1]For comparison with PVC only two-view datasets are considered for the *skew* experiments

Multi-view Spectral Clustering algorithms make use of some similarity measure between objects. For instance, [2] create a bipartite graph based on the nodes co-occurring in both views and find a cut that crosses fewest lines. This is further generalized in [15] where the notion of normalized cut is extended to multiple views. Another interesting work in this space is of integrating multiple information by co-regularizing the clustering hypotheses [7].

Multi-view Subspace Clustering algorithms assume that the multiple views are generated from one common subspace. The subspace approaches aim at learning a latent intrinsic subspace where the representations of instances in each view are close for similar examples. Several approaches have been proposed that factorize each view as a linear combination of shared latent representation – the most popular being Non-negative Matrix Factorization (NMF) based multi-view clustering algorithms [4], etc.

PVC : Multi-view clustering with partial examples. As discussed earlier this algorithm [8] addresses the scenario where every view suffers from some missing information. This results in many *partial examples* i.e. some data instance might not be represented in all the views (Note that PVC only works for two view datasets). The PVC approach uses the NMF framework to establish a latent subspace where (i) instances corresponding to the same example in different views are close to each other and (ii) similar instances (belonging to different examples) in the same view are well grouped.

III. Proposed Approach: GPMVC

In this section, we present our proposed method (referred to as GPMVC : **G**raph regularized **P**artial **M**ulti-**V**iew **C**lustering) for multi-view clustering in presence of partial examples. In Section III-A we present notations that we use throughout the paper. The formulation for GPMVC method is introduced in Section III-B. Section IV provides details of the optimization procedure.

A. Notations

We use $\mathbf{S} = \{(\mathbf{X}, \mathbf{Y})\}$ to denote the dataset. N denotes the total number of instances, M_i is the number of attributes (features) in view i, N_i is the number of instances in view i, k denotes the number of desired latent dimensions and v the number of views.

$\mathbf{X} = \{X_1, X_2, \ldots, X_v\}$ where $X_i \in \mathbb{R}_+^{M_i \times N_i}$ denotes the non-negative data matrix (for view i). Columns in the matrix represent data points and rows represent attributes.

$\mathbf{Y} = \{y^1, y^2, \ldots, y^v\}$ where y_j^i is the cluster label of the j^{th} data instance in the i^{th} view.

$\mathbf{U_i}, \mathbf{V_i}$: For each view NMF factorizes the data matrix X_i into a basis (U_i) and coefficient (V_i) matrix s.t. $X_i = U_i V_i^T$, where $U_i \in \mathbb{R}_+^{M_i \times k}, V_i \in \mathbb{R}_+^{N_i \times k}$.

$\mathbf{P}$, $\mathbf{P^-}$: These notations refer to the internal data structures that are used by our method to identify which data instances are present in which views and vice-versa (i.e which views contain which data instances). This is illustrated in Figure 1 which shows three data instances that are uniquely identified by their ID and four views. Instance A (ID 1) is present in View 1 and View 3 only. Similarly, Instance B (ID 2) is present in View 1, View 2 and View 3 only.

$\mathbf{P} = \{P_1, P_2, \ldots, P_v\}$ where P_i denotes the *view-to-instance* mapping for i^{th} view. Formally, $P_{ij} \in [1, N],\ \forall\ 1 \leq i \leq v, \forall\ 1 \leq j \leq N_i$. P_{ij} gives us the ID of the j^{th} instance in view i. For example, View 1 in Figure 1 contains instances with ID 1 and 2 only hence P_{11}=1, P_{12}=2.

$\mathbf{P^-}$ is the *inverse mapping* which informs which views are present for a given instance. $P^-(i)$ stores all the views which contain the i^{th} instance; in the form of (View, Row) pairs i.e. $P^-(i) = \{\ (l, r) \mid P_{lr} = i\ \}$. Consider Instance A (ID 1) in Figure 1 which is present in View 1 and View 3 only ; $P^-(1)$ = {(1,1), (3,1)} as instance A is the first element in View 1 and View 3.

Matrix Access: Matrices are accessed in the following two ways : $(X)_j$ indicates the j^{th} **row** of matrix X; $X_{i,j}$ or $(X)_{i,j}$ indicates the *element* present in the i^{th} row and j^{th} column of matrix X.

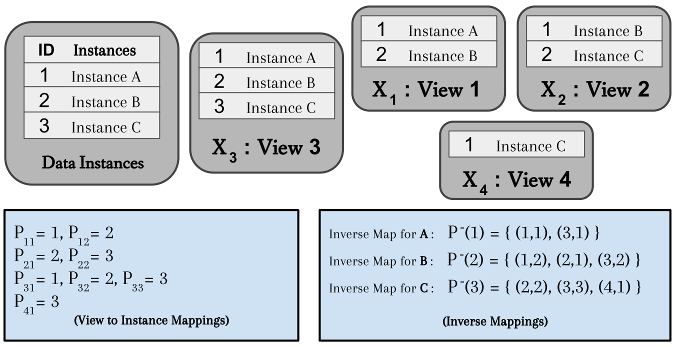

Fig. 1. View-to-Instance Mapping (P) and Inverse Mapping (P^-)

B. GPMVC Formulation

In a multi-view clustering setup one typically assumes that a data point in different views would be assigned to the same cluster with high probability. To enforce this intuition in a NMF setting the coefficient matrices (V_i) learnt from different views are softly regularized towards a common consensus matrix (V^*). This consensus matrix is considered to reflect the latent structure shared by different views [3]. In a multi-view NMF clustering setup disagreement between the i^{th} coefficient matrix and consensus matrix i.e. $\parallel V_i - V^* \parallel_F$ is minimized. Note that as V_i from different views might not be comparable at the same scale one needs to adopt a normalization strategy [3]. Each coefficient matrix V is normalized using matrix Q (where, Q is a diagonal matrix, $Q_{k,k} = \sum_i U_{i,k}$). This gives us the following multi-view NMF-based clustering problem,

$$\min_{U_i, V_i, V^*} \sum_{i=1}^{v} \left(\left\| X_i - U_i V_i^T \right\|_F^2 + \mu_i \left\| V_i Q_i - V^* \right\|_F^2 \right) \quad (1)$$
$$\text{s.t. } U_i \geq 0,\ V_i \geq 0,\ \forall i \ \ s.t.\ 1 \leq i \leq v$$

The above model while learning a joint representation (V^*) of the data completely ignores the intrinsic geometrical structure of each individual view. Prior work [1] has shown that respecting the geometrical/low-dimensional manifold information can improve clustering quality. To enable this we introduce an additional *graph regularization* penalty. Given a similarity matrix W^2 one can define a smoothness penalty term for each view as,

$$\begin{aligned}\text{Penalty}_{GR} &= \frac{1}{2}\sum_{j,l=1}^{N_i} \|(V_i)_j - (V_i)_l\|^2 \times \mathbf{W}_{jl} \\ &= Tr(V_i^T \mathbf{D}_i V_i) - Tr(V_i^T \mathbf{W}_i V_i) \\ &= Tr(V_i^T \mathbf{L}_i V_i)\end{aligned} \tag{2}$$

where, Tr(.) denotes the trace of a matrix and $\mathbf{D}$ is a diagonal matrix such that, $\mathbf{D}_{j,j} = \sum_j \mathbf{W}_{j,l}$. $\mathbf{L} = \mathbf{D} - \mathbf{W}$, is called *graph Laplacian* [3]. Fusing this with the previous model we get the following optimization problem

$$\min_{U_i,V_i,V^*} \sum_{i=1}^{v} \left(\|X_i - U_i V_i^T\|_F^2 + \mu_i \|V_i Q_i - V^*\|_F^2 + \lambda_i Tr(V_i^T L_i V_i) \right)$$
$$s.t.\ \ U_i \geq 0,\ V_i \geq 0,\ \forall i\ \ s.t.\ 1 \leq i \leq v \tag{3}$$

In order to support the partial view setup[4] we utilize the consensus matrix V^* that stores the latent structure of every data instance which is present in at least one view. Given the *view-to-instance* mappings P_i for each view, where P_{ij} gives us the ID of the j^{th} instance in view i, we define $V^*_{P_i} \in \mathbb{R}_+^{N_i \times k}$ $s.t.$ $(V^*_{P_i})_j = (V^*)_{P_{ij}}$, $1 \leq j \leq N_i$. Note, we use the notation $(X)_j$ to indicate the j^{th} row of matrix X. Using this notation one can write a modified version of Equation 3 that supports the partial-view setup

$$\min_{U_i,V_i,V^*} \sum_{i=1}^{v} \left(\|X_i - U_i V_i^T\|_F^2 + \mu_i \|V_i Q_i - V^*_{P_i}\|_F^2 + \lambda_i Tr(V_i^T L_i V_i) \right)$$
$$s.t.\ \ U_i \geq 0,\ V_i \geq 0,\ \forall i\ \ s.t.\ 1 \leq i \leq v \tag{4}$$

Equation 4 is a non-convex optimization problem and is thus difficult to optimize. Since the loss is convex in each variable individually, we separately optimize the loss with w.r.t each variable. The details of this step is provided in the next section.

IV. The Algorithm

As mentioned earlier the loss function in Equation 4 is convex in each variable individually. Exploiting this structure we optimize the loss function w.r.t each variable separately. This is done till convergence. The algorithm flow is summarized in Algorithm 1.

1. Initialization: Proper initialization of the *basis* and *coefficient* matrices play an important role in the overall performance of the GPMVC algorithm. We initialize the view specific basis and coefficient matrices (U_i, V_i) using information present in the respective views. To achieve this we apply single-view NMF clustering on examples without partial views. The single-view optimization problem can thus be written as (ignoring view indices),

$$\min_{U,V} \|X - UV^T\|_F^2 + \lambda\, Tr(V^T L V), \qquad s.t.\ \ U, V \geq 0 \tag{5}$$

This single-view, graph regularized loss can be minimized using a multiplicative update procedure as shown in [14]. Note, this step is performed independently for each view. Due to space constraints we skip details of the update steps. Readers are encouraged to refer to [14] for further details.

[2]Computed using Heat, Dot-product or Nearest-Neighbor Kernel
[3]https://en.wikipedia.org/wiki/Laplacian_matrix
[4]Not all instances are represented in all views

Algorithm 1: Graph Regularized **P**artial **M**ulti-**V**iew **C**lustering Algorithm (GPMVC)

```
Input : Nonnegative data matrix $X_1, \ldots, X_v$,
        parameters $\lambda_1, \mu_1, \ldots, \lambda_v, \mu_v$, number of clusters
        $K$, view-to-instance mapping $P$, inverse-mapping
        $P^-$
Output: Basis Matrices $U_1, \ldots, U_v$, Coefficient Matrices
        $V_1, \ldots, V_v$ and Consensus Matrix $V^*$
1  Construct Graph Laplacians $L_i$ for each view;
2  Normalize $X_i$ such that $\|X_i\|_1 = 1$ for each view ;
3  Initialize $U_i$, $V_i$ and $V^*$ by Eq. 5, Eq. 9;
4  repeat
5      for $i \leftarrow 1$ to $v$ do
6          repeat
7              Fix $V^*$ and $V_i$, update $U_i$ by Eq. 7;
8              Fix $V^*$ and $U_i$, update $V_i$ by Eq. 8;
9              Normalize $U_i$, $V_i$ using $Q_i$;
10         until objective function in Eq. 4 converges;
11     end
12     Fix $U$, $V$ update $V^*$ by Eq. 9;
13 until objective function in Eq. 4 converges;
```

2 Fixing V^*, optimize over U and V: Once V^* is fixed, each view can be optimized independently. Ignoring the view indexes for now we use U, V, Q, λ, μ to denote U_i, V_i, Q_i, λ_i, μ_i (i.e. matrices and parameter for the i^{th} view). This gives rise to the following loss function and optimization problem

$$O(U,V) = \|X - UV^T\|_F^2 + \mu \|VQ - V^*\|_F^2 + \lambda\, Tr(V^T L V)$$
$$\min_{U,V} O(U,V)\ \ s.t.\ U, V \geq 0 \tag{6}$$

We next derive the update rules that can be used to minimize the optimization problem in Equation 6.

2.1 Fixing V^* and V, update U: Let ψ be the Lagrange multiplier matrix for the constraint $U \geq 0$, and L be the Lagrange $L(U,V) = O(U,V) + Tr(\psi U)$, where $O(U,V)$ is the loss function in Equation 6. Only considering terms that involve U one can rewrite $L(U,V)$ as

$$L'(U) = \|X - UV^T\|_F^2 + \mu \|VQ - V^*\|_F^2 + Tr(\psi U)$$

Note that Q is dependent on U and hence cannot be ignored. The partial derivatives of L' with respect to U is

$$\frac{\partial L'}{\partial U} = 2UV^TV + 2\mu P - 2XV + \psi$$

where, $P = \left(\sum_{t=1}^{M_i} U_{t,k} \sum_{s=1}^{N_i} V_{s,j}^2 - \sum_{t=1}^{N_i} V_{t,j} V^*_{P_{ij},j} \right)$

Using the Karush-Kuhn-Tucker (KKT) conditions i.e. $\psi_{ik} U_{ik} = 0$, we can derive the update rule

$$U_{i,j} = U_{i,j} \times \left(\frac{(XV)_{i,j} + \mu \sum_{t=1}^{N_i} V_{t,j} V^*_{P_{ij},j}}{(UV^TV)_{i,j} + \mu \sum_{t=1}^{M_i} U_{t,k} \sum_{s=1}^{N_i} V_{s,j}^2} \right) \tag{7}$$

2.2 Fixing V^* and U, update V: Using the same steps as above and using the Lagrange multiplier matrix ϕ for the constraint $V \geq 0$, the objective function becomes

$$L'(V) = \|X - UV^T\|_F^2 + \mu \|VQ - V^*\|_F^2 + \lambda\, Tr(V^T L V) + Tr(\phi V)$$

Following similar steps as before, we get the following update rule for V

$$V_{i,j} \quad = \quad V_{i,j} \times \left(\frac{(X^T U)_{i,j} + \mu V^*_{P_{ij},j} + \lambda \, (WV)_{i,j}}{(VU^T U)_{i,j} + \mu V_{i,j} + \lambda \, (DV)_{i,j}} \right) \tag{8}$$

3 Fixing U and V, update V^*: We take the derivative of $O(U, V)$ w.r.t V^*

$$\frac{\partial O}{\partial V^*} = \frac{\partial \sum_{i=1}^{v} \mu_i \left\| V_i - V^*_{P_i} \right\|_F^2}{\partial V^*} = \sum_{i=1}^{v} \mu_i \left(-2V_i + V^*_{P_i} \right) = 0$$

Solving this gives us a closed form solution

$$(V^*)_i = \frac{\sum_{(l,r) \in P^-(i)} \mu_l \, (V_l)_r}{\sum_{(l,r) \in P^-(i)} \mu_l} \tag{9}$$

V^* provides us with a latent representation of the original data points. Computation of the clusters is done by applying a clustering algorithm (such as k-means) on V^*.

Convergence As the GPMVC algorithm uses modifications to the multiplicative update rules for NMF we borrow the same convergence proof and guarantees as proposed in earlier work [10].

V. Experiment

Datasets We use five publicly available text and image datasets for our experiments (Table I).

Datasets	Size	# Views	# Clusters
3Sources	169	3	6
Digit	2000	5	10
ORL	400	2	40
BBCSports	282	3	5
Cora	2708	2	7

TABLE I: Details of the datasets

- **ORL**: This two-view image dataset contains a set of 400 face images. We construct two views one based on raw pixel values and the other comprising of HOG features.
- **3Sources**: This three-view text dataset is collected from three online news sources. In total there are 948 news articles covering 416 distinct news stories. Of these stories, 169 were reported in all three sources. For our multi-view experiments the dataset containing 169 articles was used.
- **BBCSports**: This text dataset is a collection of sports news articles from the BBC Sport web site. For our multi-view experiments we choose the 3 view dataset which containing 282 reports.
- **Digit**: This image dataset is from the UCI repository and consists of 2000 hand-written digits (0-9). This is a 5-view dataset. Similar to [3], [7] for our two-view experiments we consider the following two views: 216 profile correlations, 240 pixel averages in 2 x 3 windows.
- **Cora**: This dataset consists of 2708 scientific publications. We consider the following two views for our experiments: number of citations between documents and the term-document matrix.

Metrics: The clustering results are evaluated using the following three metrics : Accuracy (AC), Normalized Mutual Information (NMI) and Purity (PUR). Due to space constraints we only show NMI plots in this paper. Accuracy and Purity plots along with the code for GPMVC is available on the GitHub web site[5].

Partial View Dataset Construction To construct a partial view dataset we randomly select a fraction of the instances to be *partial examples* i.e., these instances are described in only one of the views, remaining data instances are complete i.e. they appear in all views. For simplicity, we assume that the incomplete instances (*partial examples*) are equally shared amongst all the views. We later change this assumption and report our results. The PER : **P**artial **E**xample **R**atio dictates the fraction of partial examples in a given dataset. For our experiments PER ranges from 10% to 90% with 20% as interval. Experiments are also conducted at PER=0% which indicates all the views are complete (classic multi-view clustering setup). In order to remove any bias in the results due to the dataset construction process we construct 10 versions of the dataset for each value of PER. We evaluate the performance of all the methods on these 10 datasets and report mean values.

Baseline Algorithms We next describe the different two-view and multi-view baseline algorithms against which we benchmark our method. Except for PVC, no other algorithm can be directly applied in the partial view setting (i.e. PER $>$ 0%). In order to support such comparison we pre-process the partial examples by first filling in missing information. This is done using the ALM[11] matrix completion method, which is similar to what was used in the PVC paper.

Common Baseline Algorithms: These algorithms are used in both two-view and multi-view experiments

- **CentroidSC**: Centroid MV spectral clustering [6].
- **PairwiseSC**: The pairwise MV spectral clustering [7].
- **BestViewNMF**: Best single view results with NMF
- **ConcatNMF**: Concatenating the features of all the views, and then running NMF directly on this concatenated view representation.
- **BestViewGNMF**: Best single view results with Graph Regularized NMF (GNMF) [1].
- **ConcatGNMF**: Concatenating the features of all the views and running GNMF.

Apart from this we use **PVC** *Partial View Clustering* [8] algorithm for our two-view experiments and **MultiNMF**: *Multi View clustering via NMF* [3] for the multi-view experiments.

To achieve the final clustering of the examples (data instances) the latent representations generated by GPMVC needs to be clustered for which we use the *k*-means algorithm. Since k-means is known to be sensitive to initialization we run it 20 times and report mean results. In our experiments we run $75 - 100$ iterations of the GPMVC algorithm. The number of iterations is chosen empirically (convergence plots are shown in Section V-B). The parameter values for the baseline methods are tuned to achieve best performance on the datasets used.

[5]https://github.com/GPMVCDummy/GPMVC

SF (%)	10		30		70		90	
PER (%)	GPMVC	PVC	GPMVC	PVC	GPMVC	PVC	GPMVC	PVC
10	**0.810**	0.708	**0.810**	0.729	**0.807**	0.762	**0.801**	0.773
30	**0.792**	0.598	**0.794**	0.641	**0.785**	0.727	**0.788**	0.763
50	**0.777**	0.498	**0.774**	0.555	**0.774**	0.670	**0.774**	0.738
70	**0.779**	0.390	**0.734**	0.467	**0.730**	0.634	**0.754**	0.704
90	**0.772**	0.317	**0.702**	0.425	**0.684**	0.606	**0.738**	0.724

TABLE II: NMI numbers on ORL (2 view)

SF(%)	10		30		70		90	
PER (%)	GPMVC	PVC	GPMVC	PVC	GPMVC	PVC	GPMVC	PVC
10	**0.900**	0.632	**0.886**	0.632	**0.875**	0.608	**0.882**	0.604
30	**0.880**	0.630	**0.866**	0.629	**0.825**	0.527	**0.808**	0.507
50	**0.828**	0.614	**0.789**	0.614	**0.728**	0.482	**0.733**	0.453
70	**0.811**	0.582	**0.688**	0.582	**0.655**	0.446	**0.679**	0.445
90	**0.748**	0.555	**0.637**	0.555	**0.588**	0.460	**0.638**	0.493

TABLE III: NMI numbers on Digit (2 view)

Graph Laplacian Construction We employ the Dot-product, Nearest Neighbor (NN) and Heat Kernel for the construction of the graph laplacian that is used in the GPMVC method. The dot-product kernel is used for the **BBCSports** and **3Sources** dataset. We employ the heat kernel for the **ORL** dataset and the nearest neighbor kernel for the **Digit** and **Cora** dataset. In our experiments a neighborhood of 5 is used for the NN Kernel and the standard deviation of the heat kernel is set to the median of the pair-wise Euclidean distance between data points.

A. Results

Figure 2 and Figure 3 displays the results for the two-view and multi-view experiments. All results presented in this section are averaged across 10 runs of each method. From both plots we see that the GPMVC method outperforms all the baseline methods including PVC across the entire PER range. At PER=0% which is the classical multi-view setting (i.e all views are complete) the proposed GPMVC method either matches or outperforms the baseline methods in all datasets. The only exception to this is in the Digit (multi-view) experiment, where BestViewGNMF performs marginally better than GPMVC. This is due to the fact that BestViewGNMF considers only the (single) best performing view whereas GPMVC considers all views (good or bad). As PER increases (from 10% to 90%) the GPMVC method outperforms PVC and other baseline methods. The exception here is the **ORL** two-view dataset where at PER=0% the **CentroidSC** and **PairwiseSC** methods do marginally better than GPMVC. These methods also match GPMVC performance @PER=10%. However, as PER increases GPMVC starts outperforming these and other baseline methods. Another observation, which is consistent with what was shown in the PVC work, is that matrix completion methods like ALM might be less effective in the partial view data setting due to the block-wise nature of missing data as compared to missing at random, which is typically expected by matrix completion methods. Another key observation is that even at very high values of PER such as PER=90% GPMVC is still able to get better results than PVC. We also note that PVC performs better than existing baselines only for text datasets - performance of PVC on image datasets is fairly poor. One reasons for poor performance of GNMF is that the method uses the laplacian, which is not very reliable in case of matrix completed views. In fact, we noticed an improvement when the graph regularization parameter was set to zero.

Skew in Distribution of Partial Examples Previous two-view experiments assumed that for any given PER the partial examples are equally distributed in both views. We change this assumption by introducing a *skew-factor* (SF) which controls how the *partial examples* are split between the two views. For example, a skew-factor of 70% indicates that the first view contains 70% of the partial examples, the rest 30% are assigned to the second view. We monitor how our system performs vis-a-vis PVC at different PER, skew-factor combinations and report these numbers in Table 2,3. Due to space constraints we only show results for the Digit and ORL dataset[6]. Our observations, which we present next, holds true for the Cora dataset also. (i) GPMVC outperforms PVC on all the two view datasets across the entire PER, skew-factor range. (ii) For lower values of PER i.e. from PER=10% to PER=50% (inclusive) NMI numbers for GPMVC vary moderately with change in skew-factor (maximum variation on ORL is 1% and on Digit 12%). On the other hand PVC displays large variations in NMI for the same PER, skew-factor range (maximum variation on ORL is 49% and on Digit 33%). This hints that GPMVC is less sensitive to the skew in distribution of partial examples across views. (iii) For values of PER above 50% even though GPMVC dominates we observer large variations in the NMI numbers as skew-factor changes.

B. Convergence Study

Figure 4 shows convergence plots on the three two-view datasets. As the plots indicate the objective function value monotonically decreases as the iterations increase. The objective function and NMI metric value plateaus typically after 40 iterations of the GPMVC algorithm. Due to space restrictions we only show convergence plots for NMI in the two-view setting.

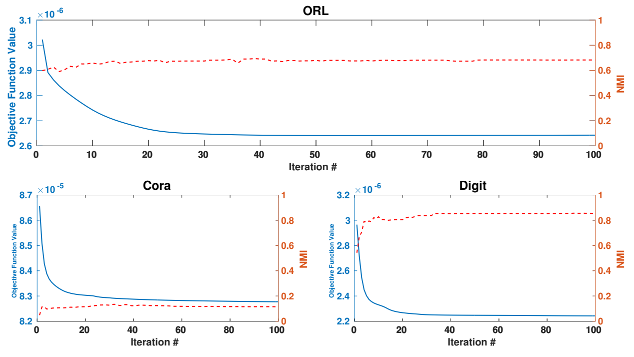

Fig. 4. NMI AND OBJECTIVE FUNCTION VALUE V.S. ITERATIONS OF GPMVC ALGORITHM @PER=50%

VI. CONCLUSION AND FUTURE WORK

In this paper we presented the **GPMVC** (**G**raph Regularized **P**artial **M**ulti-**V**iew **C**lustering) method, which is an extension

[6]The GitHub URL shows additional plots

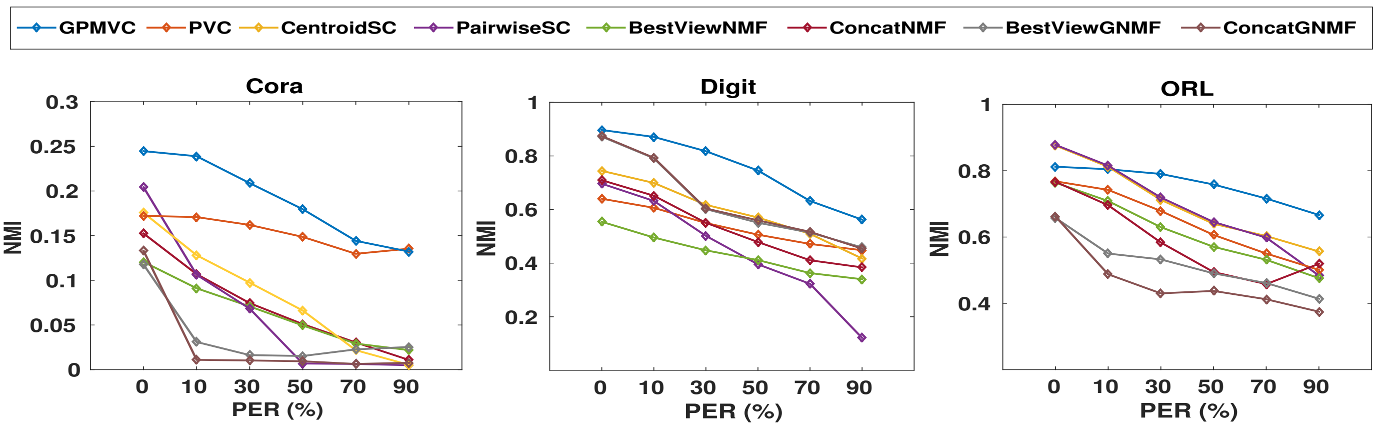

Fig. 2. NMI values vs. PER for Two View datasets (best viewed in color)

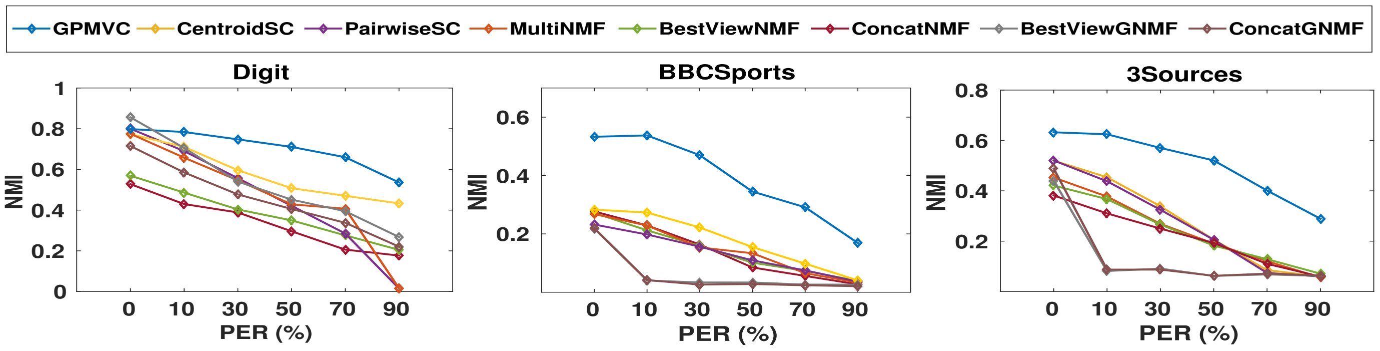

Fig. 3. NMI values vs. PER for Multi-View datasets (best viewed in color)

to the PVC method for partial view datasets. We extend the PVC method to support multi-views and view-specific graph laplacian regularization. This enables our method to exploit the intrinsic geometry of the data distribution in each view. In our experiments we compared our method against 7 baselines on 5 publicly available datasets. In almost all settings our method outperformed existing competitive baselines. As future work we would like to investigate ways by which we can scale up GPMVC method to large datasets.

References

[1] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1548–1560, 2011.

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

[3] Jing Gao, Jiawei Han, Jialu Liu, and Chi Wang. Multi-view clustering via joint nonnegative matrix factorization. In *SDM*, pages 252–260. SIAM, 2013.

[4] Derek Greene and Padraig Cunningham. A matrix factorization approach for integrating multiple data views. In Wray L. Buntine, Marko Grobelnik, Dunja Mladenic, and John Shawe-Taylor, editors, *ECML/PKDD (1)*, volume 5781 of *Lecture Notes in Computer Science*, pages 423–438. Springer, 2009.

[5] Yuhong Guo. Convex subspace representation learning from multi-view data. In Marie desJardins and Michael L. Littman, editors, *AAAI*. AAAI Press, 2013.

[6] Abhishek Kumar and Hal Daum III. A co-training approach for multi-view spectral clustering. In Lise Getoor and Tobias Scheffer, editors, *ICML*, pages 393–400. Omnipress, 2011.

[7] Abhishek Kumar, Piyush Rai, and Hal Daum III. Co-regularized multi-view spectral clustering. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 1413–1421, 2011.

[8] Shao-Yuan Li, Yuan Jiang, and Zhi-Hua Zhou. Partial multi-view clustering. In Carla E. Brodley and Peter Stone, editors, *AAAI*, pages 1968–1974. AAAI Press, 2014.

[9] Yeqing Li, Feiping Nie, Heng Huang, and Junzhou Huang. Large-scale multi-view spectral clustering via bipartite graph. In Blai Bonet and Sven Koenig, editors, *AAAI*, pages 2750–2756. AAAI Press, 2015.

[10] Chih-Jen Lin. On the convergence of multiplicative update algorithms for nonnegative matrix factorization. *IEEE Transactions on Neural Networks*, 18(6):1589–1596, 2007.

[11] Zhouchen Lin, Minming Chen, Leqin Wu, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, 2010.

[12] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. Robust multi-view spectral clustering via low-rank and sparse decomposition. In Carla E. Brodley and Peter Stone, editors, *AAAI*, pages 2149–2155. AAAI Press, 2014.

[13] Xianchao Zhang, Linlin Zong, Xinyue Liu, and Hong Yu. Constrained nmf-based multi-view clustering on unmapped data. In Blai Bonet and Sven Koenig, editors, *AAAI*, pages 3174–3180. AAAI Press, 2015.

[14] Miao Zheng, Jiajun Bu, Chun Chen, Can Wang, Lijun Zhang, Guang Qiu, and Deng Cai. Graph regularized sparse coding for image representation. *IEEE Transactions on Image Processing*, 20(5):1327–1336, 2011.

[15] Dengyong Zhou and Christopher J. C. Burges. Spectral clustering and transductive learning with multiple views. In Zoubin Ghahramani, editor, *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pages 1159–1166. ACM, 2007.