This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

# BGAE: Auto-encoding Multi-view Bipartite Graph Clustering

Liang Li, Yuangang Pan, Jie Liu, Yue Liu, Xinwang Liu, *Senior Member, IEEE,* Kenli Li, *Senior Member, IEEE,* Ivor W. Tsang, *Fellow, IEEE,* and Keqin Li, *Fellow, IEEE*

**Abstract**—Unsupervised multi-view bipartite graph clustering (MVBGC) is a fast-growing research, due to promising scalability in large-scale tasks. Although many variants are proposed by various strategies, a common design is to construct the bipartite graph directly from the input data, i.e. only consider the unidirectional "encoding" process. However, "encoding-decoding" mechanism is a popular design for deep learning, the most representative one is auto-encoder (AE). Enlightened by this, this paper rethinks existing MVBGC paradigms and transfers the "encoding-decoding" design into graph machine learning, and proposes a novel framework termed auto-encoding multi-view bipartite graph clustering (BGAE), which integrates encoding, bipartite graph construction, and decoding modules in a self-supervised learning manner. The encoding module extracts a latent joint representation from the input data, the bipartite graph construction module learns a bipartite graph with connectivity constraint in latent semantic space, and the decoding module recreates the input data via the bipartite graph. Therefore, our novel BGAE combines representation learning, bipartite graph learning, reconstruction learning, and label inference into a unified framework. All the modules are seamlessly integrated and mutually reinforcing for clustering-friendly purposes. Extensive experiments verify the superiority of our novel design and the significance of "decoding" process. To the best of our knowledge, this is the first attempt to explore "encoding-decoding" design in traditional MVBGC. The code is provided at https://github.com/liliangnudt/BGAE.

**Index Terms**—Encoding-decoding, Bipartite graph learning, Graph machine learning, Multi-view clustering.

## 1 Introduction

With the rapid growth of data in the real world, human annotations induce expensive costs, developing unsupervised or self-supervised learning is a trend to explore hidden patterns without human intervention [1]. Clustering is a central topic in unsupervised learning to find intrinsic data groupings and latent structures [2]. Owing to the flexibility and powerful capacity of graph to represent complex data structures [3], [4], graph clustering is a fast-growing research [5], [6]. To process multi-view or multimodal data (e.g., image features can be characterized by LBP, PHOG, and GIST) [7], [8], [9], multi-view clustering (MVC) has gained extensive research and achieved superior embeddings or partitions than single-view clustering by fusing consistency and complementarity [10], [11]. Multi-view graph clustering (MVGC) [12], [13] is an active branch, widely applied in data mining, natural language processing, and computer vision [14], [15].

Since traditional MVGC paradigms require building fully connected graphs with cubic time complexity and quadratic space complexity respecting sample number [16], [17], greatly limiting scalability in large-scale applications, multi-view bipartite graph clustering (MVBGC) [18], [19] is developed by building correlations of representative anchors/landmarks and all instances, i.e., bipartite graph. In

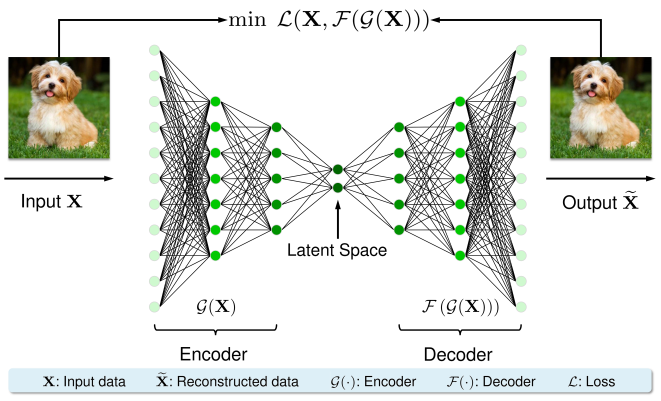

$\min\ \mathcal{L}(\mathbf{X}, \mathcal{F}(\mathcal{G}(\mathbf{X})))$
Input $\mathbf{X}$
Output $\widetilde{\mathbf{X}}$
Latent Space
$\mathcal{G}(\mathbf{X})$
$\mathcal{F}(\mathcal{G}(\mathbf{X}))$
Encoder
Decoder
$\mathbf{X}$: Input data
$\widetilde{\mathbf{X}}$: Reconstructed data
$\mathcal{G}(\cdot)$: Encoder
$\mathcal{F}(\cdot)$: Decoder
$\mathcal{L}$: Loss

Fig. 1: Sketch of auto-encoder.

- Liang Li, Yue Liu, and Xinwang Liu are with School of Computer, National University of Defense Technology, Changsha 410073, China. E-mail: liliang1037@gmail.com; yueliu19990731@163.com; xinwangliu@nudt.edu.cn
- Jie Liu is with the Science and Technology and Parallel and Distributed Processing Laboratory, National University of Defense Technology, Changsha 410073, China, and also with the Laboratory of Software Engineering for Complex Systems, National University of Defense Technology, Changsha 410073, China. E-mail: liujie@nudt.edu.cn
- Yuangang Pan and Ivor W. Tsang are with the Center for Frontier AI Research, Agency for Science, Technology and Research (A*STAR), Singapore 138632. E-mail: yuangang.pan@gmail.com; ivor.tsang@gmail.com
- Kenli Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410073, China, and also with the Supercomputing and Cloud Computing Institute, Hunan University, Changsha 410073, China. E-mail: lkl@hnu.edu.cn
- Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA. E-mail: lik@newpaltz.edu

*This work was supported in part by the National Science Fund for Distinguished Young Scholars of China under Grant 62325604, and in part by National Key Research and Development Program of China under Grant 2021YFB0300101 and in part by the National Natural Science Foundation of China under Project 62276271, and in part by A*STAR C22281201, and in part by China Scholarship Council No. 202306110001. (Corresponding author: Xinwang Liu.)*

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

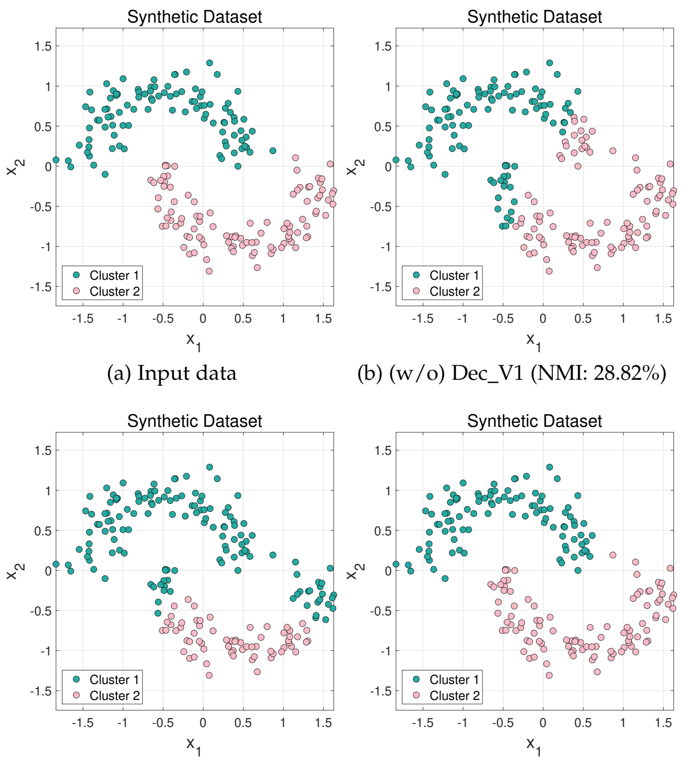
This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

this way, complexity can be reduced to linear magnitude, greatly expanding scalability.

Many MVBGC models have been proposed by various strategies to construct a "nice" bipartite graph, such as using sampling [20], [21] or optimizing manners [22], [23] to select representative anchors, introducing different regularizations [24] or constraints [25] to refine intrinsic structures, or concatenating multi-scale bipartite graphs [26], [27] across multiple views to achieve ensemble clustering.

By carefully reviewing existing BGC models in graph machine learning, we find that a common design is constructing the bipartite graph directly from the input data, i.e., only focusing on an unidirectional "encoding" process. However, "encoding-decoding" is a prevalent design in deep learning. The most representative model is auto-encoder [28], Fig. 1 shows the sketch. AE is composed of an encoder and a decoder. Encoder plays the role of information extractor to extract discriminative embeddings by multilayer neural networks. The decoder acts as a data reconstructor to recreate the input data from the learned intermediate representation. AE has gained great success, and many popular models are derived, such as denoising AE [29], VAE [30], GAE [31], MAE [32], widely applied in dimensionality reduction, image processing, and information retrieval [33]. It has also been extended to multi-view learning and shows promising performance, e.g., it integrates with generative models to fuse consistency and complementarity, serving to predict missing instances. [34].

Enlightened by AE, we transfer the "encoding-decoding" design into traditional graph machine learning and propose a novel "auto-encoding" MVBGC framework, called BGAE. Fig. 3 plots the framework. Firstly, we design an "encoding" module. Considering directly constructing the bipartite graph from input data may be unreliable, as real-world data may involve noise, outliers, or redundancy. Analogously to the encoder, we extract a consistent latent representation across multiple views, and set it to the input of the bipartite graph construction module. Then, we use the graph manifold learning paradigm to learn a bipartite graph, and further enforce it to hold connectivity constraint to output discrete labels directly. Finally, we introduce a "decoding" module. Instead of almost perfectly duplicating the input data from the intermediate representation, we recreate the input data via the bipartite graph. Such a setting not only provides feedback on the learning process, enabling the learned representation not far from the input data to retain the initial manifold, but also accords with the idea of undercomplete AE that avoids "close to perfect" duplication [28]. Fig. 2 visualizes the "benefits" of decoding learning. Compared to the baselines that mistaken partitions with poor performance (NMI: 28.82% and 45.48%), our proposed BGAE well captures the initial manifold with promising performance (NMI: 95.96%).



(a) Input data (b) (w/o) Dec_V1 (NMI: 28.82%)

(c) (w/o) Dec_V2 (NMI: 45.48%) (d) Proposed BGAE (NMI: 95.96%)

Fig. 2: Visualization of "benefits" from the decoding module on a synthetic dataset. "Decoding" learning provides feedback on optimization, enabling the learned representation not far from input data, thereby retaining the initial manifold. Detailed experimental settings are available in Table 5.

As a result, our end-to-end "auto-encoding" BGAE integrates representation learning, bipartite graph learning, reconstruction learning, and label inference into a unified framework. The contributions are outlined as follows:

1) Enlightened by the popular AE, we rethinking the traditional MVBGC paradigms in graph machine learning, and find that existing models adopt a common design that constructs the bipartite graph directly from the input data, that is, only consider an unidirectional "encoding" process, but lack the corresponding "decoding" learning.
2) We take the first step towards transferring the "encoding-decoding" design into graph machine learning and propose a novel "auto-encoding" MVBGC model, termed BGAE, integrating representation learning, bipartite graph learning, reconstruction learning, and label inference into a unified framework. All modules are mutually promoted and seamlessly integrated.
3) We design an efficient ADMM solver with linear complexity respecting instances, making it can scale to large-scale tasks. Extensive experiments empirically verify the superiority of our novel design and the significance of the "decoding" process.

## 2 Related Work

### 2.1 Non-negative Matrix Factorization

Given the input data $\mathbf{X} \in \mathbb{R}^{\widetilde{d} \times n}$ drawn from $k$ clusters, matrix factorization methods [35], [36] can decompose it into a base matrix $\mathbf{U}$ and a coefficient matrix $\mathbf{V}$. The most representative method is non-negative matrix factorization (NMF) [37] that holds both $\mathbf{U}$ and $\mathbf{V}$ to be non-negative, i.e.,

$$\min_{\mathbf{U},\mathbf{V}} \widetilde{\mathcal{L}}(\mathbf{X}, \mathbf{UV}), \text{ s.t. } \mathbf{U} \geq 0, \mathbf{V} \geq 0, \tag{1}$$

where $\widetilde{\mathcal{L}}(\cdot)$ is the loss function commonly formulated in Frobenius-norm or Kullback-Leibler divergence.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

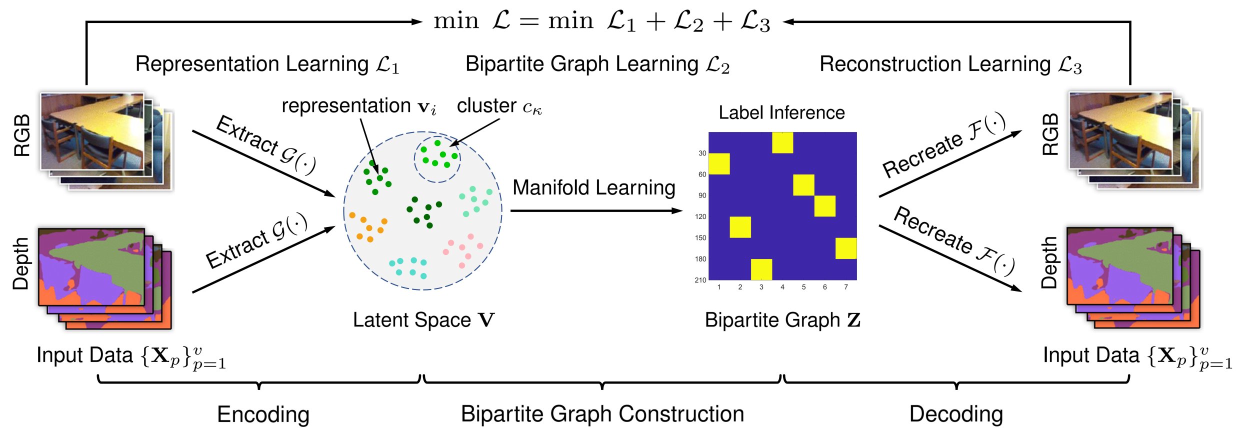

$\min\ \mathcal{L} = \min\ \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3$
Representation Learning $\mathcal{L}_1$
Bipartite Graph Learning $\mathcal{L}_2$
Reconstruction Learning $\mathcal{L}_3$
RGB
Depth
Extract $\mathcal{G}(\cdot)$
Extract $\mathcal{G}(\cdot)$
representation $\mathbf{v}_i$
cluster $c_\kappa$
Latent Space $\mathbf{V}$
Manifold Learning
Label Inference
Bipartite Graph $\mathbf{Z}$
Recreate $\mathcal{F}(\cdot)$
Recreate $\mathcal{F}(\cdot)$
RGB
Depth
Input Data $\{\mathbf{X}_p\}_{p=1}^{v}$
Input Data $\{\mathbf{X}_p\}_{p=1}^{v}$
Encoding
Bipartite Graph Construction
Decoding

Fig. 3: Framework of BGAE. For simplicity, consider a two-view image dataset (RGB and Depth). The input data $\{\mathbf{X}_p\}_{p=1}^{v}$ are first encoded into a unified latent space by $\mathcal{G}(\cdot)$. Then, we construct a bipartite graph $\mathbf{Z}$ in the latent semantic space $\mathbf{V}$. Finally, the bipartite graph is decoded by $\mathcal{F}(\cdot)$ to recreate the input data. Therefore, our BGAE integrates "encoding", "bipartite graph construction", and "decoding" modules, i.e. jointly minimize representation learning loss $\mathcal{L}_1$, bipartite graph learning loss $\mathcal{L}_2$, and reconstruction learning loss $\mathcal{L}_3$, into a unified framework in graph machine learning settings.

TABLE 1: Basic notations

| Notation | Definition |
|---|---|
| $n, v, k, m$ | Number of samples, views, clusters, anchors |
| $d_p$ | Feature dimension for the $p$-th view |
| $d$ | Latent feature dimension |
| $\boldsymbol{\alpha} \in \mathbb{R}^{v\times 1}$ | View weights in "encoding" process |
| $\boldsymbol{\gamma} \in \mathbb{R}^{v\times 1}$ | View weights in "decoding" process |
| $\mathbf{X}_p \in \mathbb{R}^{d_p\times n}$ | Input data for the $p$-th view |
| $\mathbf{U}_p \in \mathbb{R}^{d_p\times d}$ | Base matrix for input data $\mathbf{X}_p$ |
| $\mathbf{V} \in \mathbb{R}^{d\times n}$ | Consistent latent representation |
| $\mathbf{W}_p \in \mathbb{R}^{d_p\times m}$ | Projection matrix |
| $\mathbf{A} \in \mathbb{R}^{d\times m}$ | Anchor matrix |
| $\mathbf{Z} \in \mathbb{R}^{n\times m}$ | Bipartite graph matrix |
| $\mathbf{S} \in \mathbb{R}^{(n+m)\times(n+m)}$ | Augmented graph of $\mathbf{Z}$ |
| $\widetilde{\mathbf{L}} \in \mathbb{R}^{(n+m)\times(n+m)}$ | Normalized Laplacian matrix of $\mathbf{Z}$ |
| $\mathbf{E}_p \in \mathbb{R}^{d_p\times n}$ | Auxiliary variables in ADMM |
| $\boldsymbol{\Lambda}_p \in \mathbb{R}^{d_p\times n}$ | ALM multipliers |

Many variants [38], [39] have been derived based on the NMF backbone, widely used for extracting low-rank representation. Ding et al. [40] proposed semi-NMF by removing the constraint on base matrix $\mathbf{U}$, tackling the data with mixed-signs, i.e.,

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X}-\mathbf{U}\mathbf{V}\|_F^2 = \min_{\mathbf{U},\mathbf{V}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{U}\mathbf{v}_i\|_2^2, \text{s.t. } \mathbf{V} \geq 0. \quad (2)$$

Furthermore, Ding et al. [41] proposed a one-sided $\mathbf{V}$-orthogonal version to enlarge the diversity of the representation and hold the uniqueness of the solution, i.e.,

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X}-\mathbf{U}\mathbf{V}\|_F^2, \text{ s.t. } \mathbf{U} \geq 0, \mathbf{V} \geq 0, \mathbf{V}\mathbf{V}^\top = \mathbf{I}_d, \quad (3)$$

where $d$ denotes the feature dimension of $\mathbf{V}$.

### 2.2 Bipartite Graph Construction

A bipartite graph describes the correlation between two separate sets of vertices, i.e., anchors/landmarks and instances. Given an instance set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ and an anchor set $\mathcal{A} = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m\}$, we can construct an undirected bipartite graph $\mathcal{Z} = (\mathcal{X}, \mathcal{A}, \mathcal{E}, \mathbf{Z})$ by building their edges $\mathcal{E}$, $\mathbf{Z}$ is the affinity matrix weighing the connections in $\mathcal{Z}$. Typically, anchors are selected in the original space with the intention of recovering the complete point cloud.

Based on this, various BGC methods are proposed [23], [25]. Particularly, the locality-preserving paradigm is popular with researchers, supposing that the original high-dimensional feature space actually lies in a low-dimensional manifold [42]. For the $i$-th sample, $j$-th anchor is connected as a neighbor with probability $z_{ij}$. Intuitively, the anchor-node pair with a shorter distance $\|\mathbf{x}_i - \mathbf{a}_j\|_2^2$ corresponds to a larger probability $z_{ij}$, which is expressed by

$$\begin{aligned} &\min_{\mathbf{Z}} \sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{x}_i - \mathbf{a}_j\|_2^2\, z_{ij} + \zeta z_{ij}^2, \\ &\text{s.t. } \mathbf{Z}\mathbf{1}_m = \mathbf{1}_n,\ \mathbf{Z} \geq 0, \end{aligned} \quad (4)$$

where $\zeta$ is a penalized parameter that can be tuned by grid search or pre-determined following the technique [43]. Typically, anchors $\mathbf{A}$ are selected by $k$-means [26] or heuristic sampling methods [21], [25].

Along this framework, Li et al. [25] proposed a structural bipartite graph fusion model coupled with Laplacian rank constraint. Nie et al. [44], and Chen et al. [24] introduced feature selection and re-weighting mechanisms to select valuable features. Lu et al. [45] designed a fusion scheme to refine the representation. Yan et al. [46] incorporated feature learning and pseudo-labels generated by the fused bipartite graph to seek project direction and refined the graph by manifold regularization.

## 3 METHODOLOGY

### 3.1 Motivation

Although various graph machine learning based MVBGC models are proposed to pursue a "nice" bipartite graph, they adopt a common design that constructs the bipartite graph directly from the input data via Eq. (4), which means that optimization merely involves an unidirectional "encoding" process. However, "encoding-decoding" is a popular

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

manner in unsupervised deep learning, and the most typical one is AE [28]. Enlightened by this, this section carefully transfers the insight from AE into graph machine learning, and presents a novel "auto-encoding" BGAE framework.

### 3.2 The Proposed BGAE Framework

Firstly, we build an "encoding" module. In AE, the encoder plays a role in extracting discriminative embedding $\mathcal{G}(\cdot)$ by neutral networks [28], i.e. mapping the original high-dimensional data $\mathbf{X}$ into a low-dimensional semantic space $\mathcal{G}(\mathbf{X})$. Analogously, our encoding module is designed in a similar manner. Recall that NMF is widely applied for dimensionality reduction, which can be used to learn new representations [37]. However, an apparent deficiency of the standard NMF with $F$-norm is that it is sensitive to outliers. Concretely, the residual of each sample is measured in the squared form $\|\mathbf{x}^i - \mathbf{U}\mathbf{v}_i\|_2^2$. Therefore, several outliers with huge errors will easily prevail the objective [38]. Instead, we utilize $\ell_{2,1}$-norm [47] based on the orthogonal backbone. It is verified that $\ell_{2,1}$-norm is robust to noise or outliers, and can hold the row rotation invariance property [47]. Moreover, to maximally explore and fuse the discriminative information across multiple views, we learn the latent joint representation. Therefore, our encoding module (representation learning loss $\mathcal{L}_1$) is formulated as

$$\begin{aligned}\min\ \mathcal{L}_1 :\ &\min_{\boldsymbol{\alpha},\mathbf{U}_p,\mathbf{V}} \sum_{p=1}^{v} \alpha_p^2 \|\mathbf{X}_p - \mathbf{U}_p\mathbf{V}\|_{2,1},\\ =&\min_{\boldsymbol{\alpha},\mathbf{U}_p,\mathbf{V}} \sum_{p=1}^{v}\sum_{i=1}^{n} \alpha_p^2 \left\|\mathbf{x}_p^i - \mathbf{U}_p\mathbf{v}_i\right\|_2,\\ &\text{s.t.} \begin{cases}\boldsymbol{\alpha}^\top\mathbf{1} = 1,\ \alpha_p \geq 0,\\ \mathbf{V}\mathbf{V}^\top = \mathbf{I}_d.\end{cases}\end{aligned} \tag{5}$$

where $\mathbf{V}$ is the latent consistent representation, $\mathbf{U}_p$ is the view-specific base matrix, and $\alpha_p$ measures the contribution of different views. Actually, Eq. (5) is to impose $\ell_2$-norm within a sample and $\ell_1$-norm among all instances across multiple views. Compared to $F$-norm, $\ell_{2,1}$-norm measures residuals by non-squared $\left\|\mathbf{x}_p^i - \mathbf{U}_p\mathbf{v}_i\right\|_2$, reducing the impact of outliers. Note that we relax the non-negative constraints, which enlarge the feasible region to fully explore the intrinsic structures of input data with mixed-signs.

Then, we build a bipartite graph construction module. Considering that the latent representation integrates the discriminative information of the input data, we construct a bipartite graph with locality-preserving property in the latent semantic space instead of the original space as existing methods do. Such a module (bipartite graph construction loss $\mathcal{L}_2$) is formulated as

$$\begin{aligned}\min\ \mathcal{L}_2 :\ &\min_{\mathbf{A},\mathbf{Z}} \sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{v}_i - \mathbf{a}_j\|_2^2 z_{ij},\\ &\text{s.t. } \mathbf{Z}\mathbf{1}_m = \mathbf{1}_n,\ \mathbf{Z} \geq 0,\end{aligned} \tag{6}$$

where $\mathbf{Z}$ is the bipartite graph and $\mathbf{A}$ is the anchors learned by constraint-free optimization.

Finally, we elaborate on how to build the "decoding" module. In AE, the decoder recreates the input data from the encoded representation via neural networks, i.e., $\mathcal{F}(\mathcal{G}(\mathbf{X}))$. However, in traditional settings, *the first core question is which variable should be used to recreate the input data, the latent representation* $\mathbf{V}$ *or the bipartite graph* $\mathbf{Z}$*?* The former choice is likely to be a "close to perfect" duplication of $\mathbf{X}$, since $\mathbf{V}$ is just extracted from $\mathbf{X}$ in Eq. (5), duplication is meaningless for bipartite graph optimization. In deep learning, standard AE also notices this problem and derives undercomplete AE for message compression and dimensionality reduction [28]. Enlightened by this, we should design an undercomplete "auto-encoding" framework instead of perfectly recreating the input data, so that the latter choice is more reasonable and practical. Furthermore, *the second core question is how to recreate the input data by the bipartite graph?* The inconsistent sizes of $\mathbf{Z}$ and $\{\mathbf{X}_p\}_{p=1}^{v}$ make it difficult to build their correlation. For simplicity, we introduce orthogonal projection to hold their consistent feature dimension, thus the "decoding" module (reconstruction loss $\mathcal{L}_3$) is formulated as

$$\begin{aligned}\min\ \mathcal{L}_3 :\ &\min_{\boldsymbol{\gamma},\mathbf{Z},\mathbf{W}_p} \sum_{p=1}^{v} \gamma_p^2 \left\|\mathbf{W}_p\mathbf{Z}^\top - \mathbf{X}_p\right\|_F^2,\\ &\text{s.t.} \begin{cases}\boldsymbol{\gamma}^\top\mathbf{1} = 1,\ \gamma_p \geq 0,\\ \mathbf{Z}\mathbf{1}_m = \mathbf{1}_n,\ \mathbf{Z} \geq 0,\\ \mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_m,\end{cases}\end{aligned} \tag{7}$$

where $\mathbf{W}_p$ is the projection matrices, $\gamma_p$ measures the capacity of bipartite graph $\mathbf{Z}$ to recreate input data.

So far, we have carefully presented our motivation and technical route. Note that loss of AE is to measure the discrepancy between the input data and the reconstructed representation via neural networks, i.e., $\mathcal{L}(\mathbf{X}, \mathcal{F}(\mathcal{G}(\mathbf{X})))$. However, in traditional machine learning scenarios, it is difficult to exactly follow such a setting. For simplicity, we directly combine the three losses $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3$. In addition, we introduce Laplacian rank constraint to enforce the bipartite graph holds clear $k$-connected components, so that it can naturally infer discrete labels without any post-processing. Our novel end-to-end "auto-encoding" BGAE framework is as follows,

$$\begin{aligned}&\min\ \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3 :\\ &\min_{\substack{\boldsymbol{\alpha},\boldsymbol{\gamma},\mathbf{U}_p,\mathbf{V},\\ \mathbf{A},\mathbf{Z},\mathbf{W}_p}} \underbrace{\sum_{p=1}^{v} \alpha_p^2 \|\mathbf{X}_p - \mathbf{U}_p\mathbf{V}\|_{2,1}}_{Encoding} + \underbrace{\sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{v}_i - \mathbf{a}_j\|_2^2 z_{ij}}_{Bipartite\ Graph\ Construction}\\ &\quad + \underbrace{\sum_{p=1}^{v} \gamma_p^2 \left\|\mathbf{W}_p\mathbf{Z}^\top - \mathbf{X}_p\right\|_F^2}_{Decoding},\\ &\text{s.t.} \begin{cases}\boldsymbol{\alpha}^\top\mathbf{1} = 1,\ \alpha_p \geq 0,\\ \boldsymbol{\gamma}^\top\mathbf{1} = 1,\ \gamma_p \geq 0,\\ \mathbf{V}\mathbf{V}^\top = \mathbf{I}_d,\\ \mathbf{Z}\mathbf{1}_m = \mathbf{1}_n,\ \mathbf{Z} \geq 0,\\ \mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_m,\\ rank(\widetilde{\mathbf{L}}) = n + m - k,\end{cases}\end{aligned} \tag{8}$$

where $\widetilde{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$ denotes the normalized Laplacian matrix of $\mathbf{S} \in \mathbb{R}^{(n+m)\times(n+m)}$, $\mathbf{S}$ and $\mathbf{D}$ are the aug-

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

mented graph and diagonal matrix of $\mathbf{Z}$ defined by

$$\mathbf{S}=\begin{bmatrix}\mathbf{0} & \mathbf{Z}\\ \mathbf{Z}^\top & \mathbf{0}\end{bmatrix},\ \mathbf{D}=\begin{bmatrix}\mathbf{D}_n & \mathbf{0}\\ \mathbf{0} & \mathbf{D}_m\end{bmatrix}, \tag{9}$$

where $\mathbf{D}_n = diag(\mathbf{Z1}) = \mathbf{I}_n$ and $\mathbf{D}_m = diag(\mathbf{Z}^\top\mathbf{1}) \in \mathbb{R}^{m\times m}$. As noted in [25], Lemma 1 and Remark 1 illustrate that such a connectivity constraint can guarantee clear $k$-connected components of $\mathbf{S}$ and $\mathbf{Z}$, and each component naturally corresponds to a disjoint cluster.

**Lemma 1.** *The multiplicity of eigenvalue zeros of the normalized Laplacian matrix $\widetilde{\mathbf{L}}$ equals the number of connected components in the graph associated with $\mathbf{S}$.*

**Remark 1.** *The augmented graph $\mathbf{S}$ consists of a bipartite graph matrix $\mathbf{Z}$ and its transposed form $\mathbf{Z}^\top$. $\mathbf{S}$ and $\mathbf{Z}$ share the same number of connected components.*

**Remark 2.** *Note that Eq. (6) does not adhere to the standard bipartite graph learning paradigm in Eq. (4) as we remove the regularizer $\zeta\sum_{i=1}^{n}\sum_{j=1}^{m}z_{ij}^2=\zeta\mathrm{Tr}\left(\mathbf{ZZ}^\top\right)$, where $\zeta$ is a hyper-parameter requiring fine-tuning or heuristic method [43]. The reason is that Eq. (7) inherently incorporates $\gamma_p^2\mathrm{Tr}\left(\mathbf{ZZ}^\top\right)$, which naturally plays a role of avoiding the sparse trivial solution. Furthermore, $\gamma_p$ can be optimized instead of the time-consuming parameter-tuning process induced by $\eta$.*

In summary, this section proposes a novel MVBGC design, termed "auto-encoding" BGAE. The encoding module extracts a latent representation across multiple views. The bipartite graph construction module introduces manifold graph learning to explore intrinsic geometrical structures. The decoding module recreates the input data via bipartite graph, which provides feedback to learning process that allows bipartite graph not far from the input multi-view data, enforcing it contains complementary information. Therefore, by jointly minimizing $\mathcal{L}=\mathcal{L}_1+\mathcal{L}_2+\mathcal{L}_3$, the bipartite graph balances consistency and complementary properties across all views, which is a vital pursuit in multi-view learning [34], [48]. Compared to most existing models that only consider encoding input data into bipartite graphs by Eq. (4), our novel design achieves end-to-end "encoding-decoding" bipartite graph machine learning.

### 3.3 Optimization Algorithm

Since we impose $\ell_{2,1}$-norm, orthogonal constraint, and Laplacian rank constraint, it is difficult to solve the model directly. This section designs an ADMM solver.

Firstly, we derive the matrix form of Eq. (6), i.e.,

$$\begin{aligned}&\sum_{i=1}^{n}\sum_{j=1}^{m}\|\mathbf{v}_i-\mathbf{a}_j\|_2^2 z_{ij}\\ =&\sum_{i=1}^{n}\sum_{j=1}^{m}(\mathbf{v}_i-\mathbf{a}_j)^\top(\mathbf{v}_i-\mathbf{a}_j)z_{ij}\\ =&\sum_{i=1}^{n}\mathbf{v}_i^\top\left(\sum_{j=1}^{m}z_{ij}\right)\mathbf{v}_i-2\sum_{i=1}^{n}\sum_{j=1}^{m}\mathbf{v}_i^\top z_{ij}\mathbf{a}_j\\ &+\sum_{j=1}^{m}\mathbf{a}_j{}^\top\left(\sum_{i=1}^{n}z_{ij}\right)\mathbf{a}_j\\ =&\mathrm{Tr}\left(\mathbf{VD}_n\mathbf{V}^\top-2\mathbf{VZA}^\top+\mathbf{AD}_m\mathbf{A}^\top\right)\end{aligned} \tag{10}$$

Then, considering that the non-convex Laplacian rank constraint is difficult to deal with, we solve it with a relaxed solution. Denoting $\sigma_i(\widetilde{\mathbf{L}})$ is the $i$-th smallest eigenvalue of $\widetilde{\mathbf{L}}$. Note that $\widetilde{\mathbf{L}}$ satisfies semi-definite property, i.e., $\sigma_i(\widetilde{\mathbf{L}}) \ge 0$. Once rank-$k$ smallest $\sigma_i(\widetilde{\mathbf{L}})$ equals zero, the rank constraint will be achieved, and $\mathbf{S}$ will be an ideal graph preserving clear $k$-connected components structures. According to Ky Fan's Theorem [49], we have $\sum_{i=1}^{k}\sigma_i(\widetilde{\mathbf{L}}) = \min_{\mathbf{F}^\top\mathbf{F}=\mathbf{I}_k}\mathrm{Tr}\left(\mathbf{F}^\top\widetilde{\mathbf{L}}\mathbf{F}\right)$, where $\mathbf{F}\in\mathbb{R}^{(n+m)\times k}$ denotes the graph embedding.

Finally, by introducing $v$ auxiliary variables $\{\mathbf{E}_p = \mathbf{X}_p-\mathbf{U}_p\mathbf{V}\}_{p=1}^{v}$ to separate constraints and hold equivalence during optimization, our model is transformed into the following Augmented Lagrangian Multiplier (ALM) problem

$$\begin{aligned}\min_{\substack{\boldsymbol{\alpha},\boldsymbol{\gamma},\mathbf{U}_p,\mathbf{V},\mathbf{A},\\ \mathbf{Z},\mathbf{F},\mathbf{W}_p,\mathbf{E}_p,\boldsymbol{\Lambda}_p}}&\sum_{p=1}^{v}\alpha_p^2\|\mathbf{E}_p\|_{2,1}+\\ &\mathrm{Tr}\left(\mathbf{VD}_n\mathbf{V}^\top-2\mathbf{VZA}^\top+\mathbf{AD}_m\mathbf{A}^\top\right)+\\ &\sum_{p=1}^{v}\gamma_p^2\left\|\mathbf{W}_p\mathbf{Z}^\top-\mathbf{X}_p\right\|_F^2+\mu\mathrm{Tr}\left(\mathbf{F}^\top\widetilde{\mathbf{L}}\mathbf{F}\right)+\\ &\frac{\beta}{2}\sum_{p=1}^{v}\left\|\mathbf{X}_p-\mathbf{U}_p\mathbf{V}-\mathbf{E}_p+\frac{1}{\beta}\boldsymbol{\Lambda}_p\right\|_F^2,\\ \text{s.t.}&\begin{cases}\boldsymbol{\alpha}^\top\mathbf{1}=1,\ \alpha_p\ge 0,\\ \boldsymbol{\gamma}^\top\mathbf{1}=1,\ \gamma_p\ge 0,\\ \mathbf{VV}^\top=\mathbf{I}_d,\\ \mathbf{Z1}_m=\mathbf{1}_n,\ \mathbf{Z}\ge 0,\\ \mathbf{W}_p^\top\mathbf{W}_p=\mathbf{I}_m,\\ \mathbf{F}^\top\mathbf{F}=\mathbf{I}_k,\end{cases}\end{aligned} \tag{11}$$

where $\{\boldsymbol{\Lambda}_p\}_{p=1}^{v}$ are ALM multipliers to penalize the discrepancy between the original objective and the introduced auxiliary variable, $\mu$ is a penalized parameter that should be large enough to hold the rank-$k$ smallest $\sigma_i(\widetilde{\mathbf{L}})$ infinitely close to zero, and $\beta$ is the ALM parameter.

We optimize Eq. (11) by block-coordinate descent strategy that alternately updates each variable with others being fixed. Algorithm 1 summarizes the overall workflow.

#### 3.3.1 *Update* $\mathbf{U}_p$

With others being fixed, each $\mathbf{U}_p$ is solved by

$$\min_{\mathbf{U}_p}\frac{\beta}{2}\sum_{p=1}^{v}\left\|\mathbf{X}_p-\mathbf{U}_p\mathbf{V}-\mathbf{E}_p+\frac{1}{\beta}\boldsymbol{\Lambda}_p\right\|_F^2. \tag{12}$$

Since no constraint is imposed on $\mathbf{U}_p$ and $\mathbf{VV}^\top=\mathbf{I}_d$, each $\mathbf{U}_p$ can be updated independently by

$$\min_{\mathbf{U}_p}\left\|\mathbf{U}_p-\left(\mathbf{X}_p-\mathbf{E}_p+\frac{1}{\beta}\boldsymbol{\Lambda}_p\right)\mathbf{V}^\top\right\|_F^2. \tag{13}$$

Clearly, we have

$$\mathbf{U}_p=\left(\mathbf{X}_p-\mathbf{E}_p+\frac{1}{\beta}\boldsymbol{\Lambda}_p\right)\mathbf{V}^\top. \tag{14}$$

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

### 3.3.2 Update $\mathbf{V}$

With others being fixed, $\mathbf{V}$ can be solved by

$$\max_{\mathbf{V}} \operatorname{Tr}(\mathbf{VM}), \text{ s.t. } \mathbf{VV}^\top = \mathbf{I}_d, \tag{15}$$

where $\mathbf{M} = 2\mathbf{ZA}^\top + \beta \sum_{p=1}^{v} \mathbf{Q}_p^\top \mathbf{U}_p$ and $\mathbf{Q}_p = \mathbf{X}_p - \mathbf{E}_p + \frac{1}{\beta}\mathbf{\Lambda}_p$. The analytical solution can be achieved by singular value decomposition (SVD) [22].

### 3.3.3 Update $\mathbf{Z}$

With others being fixed, $\mathbf{Z}$ is solved by

$$\begin{aligned}\min_{\mathbf{Z},\mathbf{F}} \operatorname{Tr}\Bigg(&\sum_{p=1}^{v}\gamma_p^2\mathbf{Z}^\top\mathbf{Z} + \mathbf{1}_n\left(diag\left(\mathbf{A}^\top\mathbf{A}\right)^\top\right)\mathbf{Z}^\top - \\ &2\mathbf{V}^\top\mathbf{AZ}^\top - 2\sum_{p=1}^{v}\gamma_p^2\mathbf{X}_p^\top\mathbf{W}_p\mathbf{Z}^\top\Bigg) + \mu\operatorname{Tr}\left(\mathbf{F}^\top\widetilde{\mathbf{L}}\mathbf{F}\right), \\ \text{s.t. } &\mathbf{Z1}_m = \mathbf{1}_n,\ \mathbf{Z}\geq 0,\ \mathbf{F}^\top\mathbf{F} = \mathbf{I}_k.\end{aligned} \tag{16}$$

Since Eq. (16) involves two variables, we use the block-coordinate descent method to update $\mathbf{Z}$ and $\mathbf{F}$ alternatively. With $\mathbf{Z}$ being fixed, Eq. (16) is simplified to,

$$\min_{\mathbf{F}} \operatorname{Tr}\left(\mathbf{F}^\top\widetilde{\mathbf{L}}\mathbf{F}\right), \text{s.t. } \mathbf{F}^\top\mathbf{F} = \mathbf{I}_k. \tag{17}$$

To efficiently solve Eq. (17), we solve the singular values of $\mathbf{Z}$ rather than the eigenvalues of $\mathbf{S}$. By decomposing $\mathbf{F} = \begin{bmatrix}\mathbf{F}_n \\ \mathbf{F}_m\end{bmatrix}$, Eq. (17) can be rewritten as,

$$\begin{aligned}&\max_{\mathbf{F}_n,\mathbf{F}_m} \operatorname{Tr}\left(\mathbf{F}_n^\top\mathbf{D}_n^{-\frac{1}{2}}\mathbf{Z}\mathbf{D}_m^{-\frac{1}{2}}\mathbf{F}_m\right), \\ &\quad \text{s.t. } \mathbf{F}_n^\top\mathbf{F}_n + \mathbf{F}_m^\top\mathbf{F}_m = \mathbf{I}_k.\end{aligned} \tag{18}$$

Theorem 1 provides the analytical solution of Eq. (18).

**Theorem 1.** *Supposing* $\mathbf{P}\in\mathbb{R}^{n\times k}, \mathbf{O}\in\mathbb{R}^{n\times m}, \mathbf{R}\in\mathbb{R}^{m\times k}$, *we have*

$$\begin{aligned}&\max_{\mathbf{P},\mathbf{R}} \operatorname{Tr}\left(\mathbf{P}^\top\mathbf{OR}\right), \\ &\quad s.t.\ \mathbf{P}^\top\mathbf{P} + \mathbf{R}^\top\mathbf{R} = \mathbf{I}_k.\end{aligned} \tag{19}$$

*The optimal solutions are* $\mathbf{P} = \frac{\sqrt{2}}{2}\mathbf{U_o}$ *and* $\mathbf{R} = \frac{\sqrt{2}}{2}\mathbf{V_o}$, *where* $\mathbf{U_o}$ *and* $\mathbf{V_o}$ *are the rank-k left and right singular vectors of* $\mathbf{O}$.

Detailed proof is provided in supplementary material.

After optimizing $\mathbf{F}$, we turn to optimize $\mathbf{Z}$. We have

$$\operatorname{Tr}\left(\mathbf{F}^\top\widetilde{\mathbf{L}}\mathbf{F}\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} t_{ij}z_{ij} = \operatorname{Tr}\left(\mathbf{TZ}^\top\right), \tag{20}$$

where $t_{ij} = \left\|\frac{\mathbf{f}_n^i}{\sqrt{\mathbf{D}_{n[i,i]}}} - \frac{\mathbf{f}_m^j}{\sqrt{\mathbf{D}_{m[j,j]}}}\right\|_2^2$.

Eq. (16) can be rewritten as $n$ row independent problem w.r.t. $\mathbf{Z}$, i.e.,

$$\begin{aligned}&\min_{\mathbf{Z}_{[i,:]}} \frac{1}{2}\left\|\mathbf{Z}_{[i,:]} - \widetilde{\mathbf{Z}}_{[i,:]}\right\|_2^2, \\ &\text{s.t. } \mathbf{Z}_{[i,:]}\mathbf{1} = 1,\ \mathbf{Z}_{[i,:]}\geq 0,\end{aligned} \tag{21}$$

where $\widetilde{\mathbf{Z}}_{[i,:]} = -\frac{\mathbf{g}^\top}{2\sum_{p=1}^{v}\gamma_p^2}$, $\mathbf{g}^\top = diag\left(\mathbf{A}^\top\mathbf{A}\right)^\top - \left(2\mathbf{V}^\top\mathbf{A} + 2\sum_{p=1}^{v}\gamma_p^2\mathbf{X}_p^\top\mathbf{W}_p - \mu\mathbf{T}\right)_{[i,:]}$.

Theorem 2 gives the analytical solution of Eq. (21).

**Theorem 2.** *The analytical solution of Eq. (21) is*

$$\mathbf{Z}_{[i,:]} = \left(\widetilde{\mathbf{Z}}_{[i,:]} + \epsilon_i\mathbf{1}_m^\top\right)_+, \tag{22}$$

*where* $\epsilon_i$ *can be solved by Newton's method efficiently.*

Detailed proof is provided in [22].

### 3.3.4 Update $\mathbf{E}_p$

With others being fixed, each $\mathbf{E}_p$ is independently solved by

$$\min_{\mathbf{E}_p} \alpha_p^2\left\|\mathbf{E}_p\right\|_{2,1} + \frac{\beta}{2}\|\mathbf{X}_p - \mathbf{U}_p\mathbf{V} - \mathbf{E}_p + \frac{1}{\beta}\mathbf{\Lambda}_p\|_F^2, \tag{23}$$

which can be further rewritten as the following compact formulation

$$\min_{\mathbf{E}_p} \frac{\alpha_p^2}{\beta}\left\|\mathbf{E}_p\right\|_{2,1} + \frac{1}{2}\|\mathbf{E}_p - \mathbf{H}_p\|_F^2, \tag{24}$$

where $\mathbf{H}_p = \mathbf{X}_p - \mathbf{U}_p\mathbf{V} + \frac{1}{\beta}\mathbf{\Lambda}_p$. According to [39], the solution is

$$\mathbf{e}_p^i = \begin{cases}\left(1 - \frac{\alpha_p}{\beta\|\mathbf{h}_p^i\|_2}\right)\mathbf{h}_p^i, & if\ \left\|\mathbf{h}_p^i\right\|_2 > \frac{\alpha_p}{\beta}, \\ 0, & otherwise.\end{cases} \tag{25}$$

### 3.3.5 Update $\mathbf{A}$

With others being fixed, $\mathbf{A}$ is solved by

$$\min_{\mathbf{A}} \operatorname{Tr}\left(-2\mathbf{VZA}^\top + \mathbf{AD}_m\mathbf{A}^\top\right). \tag{26}$$

Taking the partial derivative on $\mathbf{A}$, we have

$$\begin{aligned}&\frac{\partial}{\partial\mathbf{A}}\operatorname{Tr}\left(-2\mathbf{VZA}^\top + \mathbf{AD}_m\mathbf{A}^\top\right) \\ &= -2\mathbf{VZ} + 2\mathbf{AD}_m.\end{aligned} \tag{27}$$

By enforcing the partial derivative equals to 0, we have

$$\mathbf{VZ} = \mathbf{AD}_m. \tag{28}$$

Supposing $(\mathbf{D}_m)^{-1}$ exists, we have

$$\mathbf{A} = \mathbf{VZ}(\mathbf{D}_m)^{-1}. \tag{29}$$

**Remark 3.** $(\mathbf{D}_m)^{-1}$ *exists means that the column sum of the bipartite graph* $\mathbf{Z}\in\mathbb{R}^{n\times m}$ *are always greater than 0. However, such ideal cases do not always hold, and there is still a minimal probability that the j-th column sum of* $\mathbf{Z}$ *is 0 in experiments. That is,* $\mathbf{a}_j$ *is an isolated anchor without building membership with other instances. In experiments, we find that such undesirable cases may occur during inchoate iterations, and the existence of isolated* $\mathbf{a}_j$ *has no impact on the objective value, so it will not affect exploring the final graph representation. Therefore, we remove the isolated anchors directly.*

### 3.3.6 Update $\mathbf{W}_p$

With others being fixed, each $\mathbf{W}_p$ is solved by

$$\max_{\mathbf{W}_p} \operatorname{Tr}\left(\mathbf{W}_p^\top\mathbf{B}_p\right), \text{ s.t. } \mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_m, \tag{30}$$

where $\mathbf{B}_p = \mathbf{X}_p\mathbf{Z}$, Eq. (30) can be efficiently solved by SVD.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

**Algorithm 1** BGAE

```
1: Input: Input data $\{\mathbf{X}_p\}_{p=1}^{v}$, cluster number $k$, anchor number $m$, latent dimension $d$, maximal iteration $\Gamma$.
2: Initialize $\{\mathbf{U}_p\}_{p=1}^{v}$, $\mathbf{A}$, $\{\mathbf{W}_p\}_{p=1}^{v}$, $\mathbf{Z}$, $\mathbf{V}$, $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$, $\{\boldsymbol{\Lambda}_p\}_{p=1}^{v}$.
3: while not converged and iteration less than $\Gamma$ do
4:     Optimize $\{\mathbf{E}_p\}_{p=1}^{v}$ by updating Eq. (23).
5:     Optimize $\{\mathbf{U}_p\}_{p=1}^{v}$ by updating Eq. (12).
6:     Optimize $\mathbf{A}$ by updating Eq. (26).
7:     Optimize $\{\mathbf{W}_p\}_{p=1}^{v}$ by updating Eq. (30).
8:     Optimize $\mathbf{Z}$ by updating Eq. (16).
9:     Optimize $\mathbf{V}$ by updating Eq. (15).
10:    Optimize $\boldsymbol{\alpha}$ by updating Eq. (31).
11:    Optimize $\boldsymbol{\gamma}$ by updating Eq. (33).
12:    Optimize $\{\boldsymbol{\Lambda}_p\}_{p=1}^{v}$ and $\beta$ by updating Eq. (35).
13: end while
14: Output: The predicted clustering labels $\widetilde{\mathbf{Y}}$.
```

#### 3.3.7 Update $\boldsymbol{\alpha}$

With other variables being fixed, each $\alpha_p$ is independently optimized by

$$\min_{\alpha_p} \sum_{p=1}^{v} \alpha_p^2 \tau_p, \text{ s.t. } \boldsymbol{\alpha} \geq 0, \boldsymbol{\alpha}^\top \mathbf{1} = 1, \tag{31}$$

where $\tau_p = \|\mathbf{X}_p - \mathbf{U}_p\mathbf{V}\|_{2,1}$. The solution of $\alpha_p$ can be straightly computed by Cauchy-Schwarz inequality, i.e.,

$$\alpha_p = \frac{1/\tau_p}{\sum_{p=1}^{v} 1/\tau_p}. \tag{32}$$

#### 3.3.8 Update $\gamma$

With other variables being fixed, each $\gamma_p$ is independently optimized by

$$\min_{\gamma_p} \sum_{p=1}^{v} \gamma_p^2 \delta_p, \text{ s.t. } \boldsymbol{\gamma} \geq 0, \boldsymbol{\gamma}^\top \mathbf{1} = 1, \tag{33}$$

where $\delta_p = \left\|\mathbf{W}_p\mathbf{Z}^\top - \mathbf{X}_p\right\|_F^2$. The solution of $\gamma_p$ can be straightly computed by Cauchy-Schwarz inequality, i.e.,

$$\gamma_p = \frac{1/\delta_p}{\sum_{p=1}^{v} 1/\delta_p}. \tag{34}$$

#### 3.3.9 Update $\boldsymbol{\Lambda}_p$ and $\beta$

Each Lagrangian multiplier $\boldsymbol{\Lambda}_p$ and $\beta$ can be updated by

$$\begin{aligned} \boldsymbol{\Lambda}_p &= \boldsymbol{\Lambda}_p + \beta\left(\mathbf{X}_p - \mathbf{U}_p\mathbf{V} - \mathbf{E}_p\right), \\ \beta &= \rho\beta, \end{aligned} \tag{35}$$

where $\rho$ controls the convergence speed and we empirically set $\rho = 2$ in experiments.

### 3.4 Complexity Analysis

In graph machine learning, constructing similarity graphs is necessary. Given a desktop with 64 GB RAM, a double precision floating point format requires 8 bytes, the largest matrix that can be stored is $92,682 \times 92,682$, and larger sizes will incur out-of-memory error. This section carefully analyses the complexities. For simplicity, we set $g_1 = \sum_{p=1}^{v} d_p$ and $g_2 = \sum_{p=1}^{v} d_p^2$. Commonly, $n \gg d$ and $n \gg m$.

TABLE 2: MVC datasets statistics

| Dataset | Samples | Views | Clusters | Feature Dims |
|---|---|---|---|---|
| Yale | 165 | 3 | 15 | 4,096/3,304/6,750 |
| 3sources | 169 | 3 | 6 | 3,560/3,631/3,068 |
| MSRCV1 | 210 | 6 | 7 | 1,302/48/512/100/256/210 |
| Dermatology | 358 | 2 | 6 | 12/22 |
| ORL_3views | 400 | 3 | 40 | 4,096/3,304/6,750 |
| ORL_4views | 400 | 4 | 40 | 256/256/256/256 |
| SUN RGB-D | 10,335 | 2 | 45 | 4,096/4,096 |
| YouTubeFace20 | 63,896 | 4 | 20 | 944/576/512/640 |
| YouTubeFace50 | 126,054 | 4 | 50 | 944/576/512/640 |
| YouTubeFace100 | 195,537 | 4 | 100 | 944/576/512/640 |

TABLE 3: Comparison of algorithm complexity

| Method | Space Complexity | Time Complexity |
|---|---|---|
| RMKM [50] | $\mathcal{O}(n^2 + n(g_1 + k) + g_1k)$ | $\mathcal{O}(n^2)$ |
| AMGL [51] | $\mathcal{O}(n^2v + n(g_1 + k))$ | $\mathcal{O}(n^3)$ |
| FMR [52] | $\mathcal{O}(n^2 + n(g_1 + k))$ | $\mathcal{O}(n^3)$ |
| PMSC [53] | $\mathcal{O}(n^2v + n(g_1 + kv))$ | $\mathcal{O}(n^3)$ |
| BMVC [54] | $\mathcal{O}((n + d_{mean})l)$ | $\mathcal{O}(n)$ |
| LMVSC [26] | $\mathcal{O}(n(g_1 + mv) + g_1mv)$ | $\mathcal{O}(n)$ |
| SMVSC [18] | $\mathcal{O}(nm + (g_1 + m)k)$ | $\mathcal{O}(n)$ |
| SFMC [25] | $\mathcal{O}(n(g_1 + mv))$ | $\mathcal{O}(n)$ |
| FMCNOF [55] | $\mathcal{O}(n(g_1 + mv + k) + mk)$ | $\mathcal{O}(n)$ |
| FPMVS [56] | $\mathcal{O}(nk + (g_1 + k)k)$ | $\mathcal{O}(n)$ |
| SDAFG [45] | $\mathcal{O}(n(g_1 + mv))$ | $\mathcal{O}(n)$ |
| UDBGL [57] | $\mathcal{O}(n(g_1 + mv) + g_1m)$ | $\mathcal{O}(n)$ |
| FastMICE [21] | $\mathcal{O}(n(g_1 + k + k_{neibor} + m + v))$ | $\mathcal{O}(n)$ |
| Proposed | $\mathcal{O}(n(g_1 + d + m) + g_1(d + m) + dm)$ | $\mathcal{O}(n)$ |

*Time Complexity*: The time complexity consists of nine parts: (1) updating $\{\mathbf{E}_p\}_{p=1}^{v}$ requires $\mathcal{O}(ng_1d)$ complexity, (2) updating $\{\mathbf{U}_p\}_{p=1}^{v}$ requires $\mathcal{O}(ng_1d)$ complexity, (3) updating $\mathbf{A}$ requires $\mathcal{O}(ndm)$ complexity, (4) updating $\{\mathbf{W}_p\}_{p=1}^{v}$ requires $\mathcal{O}(n(g_1m + g_2))$ complexity, (5) updating $\mathbf{Z}$ requires $\mathcal{O}\left(n(g_1m + dm + m^2)\right)$ complexity, (6) updating $\mathbf{V}$ requires $\mathcal{O}\left(n(g_1m + dm + d^2)\right)$ complexity, (7) updating $\boldsymbol{\alpha}$ requires $\mathcal{O}(ng_1d)$ complexity, (8) updating $\boldsymbol{\gamma}$ requires $\mathcal{O}(n(g_1m + g_2))$ complexity, (9) updating $\{\boldsymbol{\Lambda}_p\}_{p=1}^{v}$ requires $\mathcal{O}(ng_1d)$ complexity. So, the total time complexity for each iteration is $\mathcal{O}\left(n(g_1d + g_1m + g_2 + dm + d^2 + m^2)\right)$.

*Space Complexity*: The space complexity comes from storing huge matrices, i.e., $\{\mathbf{E}_p\}_{p=1}^{v}$, $\{\mathbf{X}_p\}_{p=1}^{v}$, $\{\mathbf{U}_p\}_{p=1}^{v}$, $\mathbf{A}$, $\{\mathbf{W}_p\}_{p=1}^{v}$, $\mathbf{Z}$, $\mathbf{V}$, and $\{\boldsymbol{\Lambda}_p\}_{p=1}^{v}$. The total space complexity is $\mathcal{O}(n(g_1 + d + m) + g_1(d + m) + dm)$.

Therefore, the complexities are linear with $n$, enabling it can scale to large datasets with $n \geq 100,000$.

## 4 Experiment

### 4.1 Experimental Settings

#### 4.1.1 Synthetic Datasets

To visualize the "benefits" from the "decoding" learning in retaining the initial manifold structures, we design a synthetic two-moon data from two clusters, shown in Fig. 2 (a). Each moon consists of 100 samples, one moon is colored with green dots and the other with pink dots.

#### 4.1.2 Realistic Datasets

Table 2 lists ten public MVC datasets. Yale[1] involves 165 grayscale images of 15 individuals. 3sources[2] is a text

1. http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html
2. http://mlg.ucd.ie/datasets/3sources.html

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

TABLE 4: Summary of clustering metrics. The best results are symbolized in bold, the runner-up ones are underlined and italic, "OOM" denotes out-of-memory errors, and "-" denotes time-out errors.

| Datasets | RMKM[‡] | AMGL | FMR | PMSC | BMVC[‡] | LMVSC | SMVSC | SFMC[‡] | FMCNOF[‡] | FPMVS | SDAFG[‡] | Proposed[‡] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ACC (%) | | | | | | | | | | | |
| Yale | 61.21±0.00 | 64.52±4.27 | *68.81±5.95* | 58.80±4.43 | 41.21±0.00 | 61.47±3.47 | 66.06±0.00 | 47.27±0.00 | 33.94±0.00 | 67.27±0.00 | 65.45±0.00 | **72.12±0.00** |
| 3sources | 49.70±0.00 | 20.67±0.34 | 59.23±5.24 | *73.09±4.09* | 47.34±0.00 | 44.26±1.90 | 63.63±1.29 | 34.91±0.00 | 65.09±0.00 | 71.01±0.00 | 38.46±0.00 | **77.51±0.00** |
| MSRCV1 | 71.43±0.00 | 76.44±6.30 | 77.48±6.40 | 47.45±4.23 | 26.67±0.00 | *83.73±7.20* | 70.51±4.98 | 60.48±0.00 | 47.14±0.00 | 71.95±5.36 | 70.95±0.00 | **85.71±0.00** |
| Dermatology | 74.86±0.00 | 22.57±0.59 | 81.72±5.66 | 80.75±4.46 | 63.97±0.00 | 79.02±6.63 | 78.64±5.41 | 49.44±0.00 | 62.01±0.00 | *82.96±7.44* | 56.70±0.00 | **89.11±0.00** |
| ORL_3views | 55.50±0.00 | 71.15±2.81 | 65.79±3.26 | 63.47±3.09 | 48.00±0.00 | 65.65±2.93 | 65.16±1.10 | 50.00±0.00 | 27.50±0.00 | 66.75±0.00 | *73.50±0.00* | **76.00±0.00** |
| ORL_4views | 47.00±0.00 | 59.73±2.78 | 25.21±1.10 | 21.48±1.02 | 43.25±0.00 | *61.50±2.96* | 47.76±2.36 | 37.00±0.00 | 21.50±0.00 | 54.63±1.49 | 57.75±0.00 | **65.00±0.00** |
| SUN RGB-D | 19.51±0.00 | 9.81±0.37 | OOM | OOM | 21.02±0.00 | 17.87±0.39 | *23.34±0.38* | 11.02±0.00 | 19.67±0.00 | 23.26±0.50 | 16.85±0.00 | **24.78±0.00** |
| YouTubeFace20 | *71.10±0.00* | OOM | OOM | OOM | 57.39±0.00 | 67.26±3.53 | 67.13±4.20 | - | 38.61±0.00 | 63.08±3.79 | 61.88±0.00 | **75.55±0.00** |
| YouTubeFace50 | OOM | OOM | OOM | OOM | 66.00±0.00 | 68.32±2.45 | *69.65±2.46* | - | 21.66±0.00 | 64.24±2.97 | 62.44±0.00 | **77.39±0.00** |
| YouTubeFace100 | OOM | OOM | OOM | OOM | *70.14±0.00* | 63.52±2.12 | 60.63±1.91 | - | 12.10±0.00 | 55.23±2.48 | 45.51±0.00 | **72.94±0.00** |
| Average Rank | 6.50 | 7.14 | 4.83 | 7.50 | 7.50 | 4.70 | 4.80 | 9.86 | 8.60 | 4.00 | 6.40 | 1.00 |
| | NMI (%) | | | | | | | | | | | |
| Yale | 64.71±0.00 | 67.73±1.86 | **74.72±3.38** | 63.74±2.98 | 45.58±0.00 | 65.43±1.92 | 69.83±0.00 | 54.27±0.00 | 39.50±0.00 | 71.06±0.00 | 69.20±0.00 | *73.11±0.00* |
| 3sources | 35.02±0.00 | 8.07±0.43 | 45.96±2.69 | **67.06±3.08** | 48.86±0.00 | 33.13±1.89 | 54.35±1.62 | 6.28±0.00 | 51.96±0.00 | 61.72±0.00 | 10.37±0.00 | *66.09±0.00* |
| MSRCV1 | 63.03±0.00 | 77.65±3.23 | 69.48±3.31 | 34.29±2.81 | 8.29±0.00 | *78.93±4.60* | 62.01±2.61 | 60.23±0.00 | 38.42±0.00 | 65.69±3.27 | 76.23±0.00 | **80.50±0.00** |
| Dermatology | 71.10±0.00 | 3.20±0.57 | *79.97±3.67* | **85.11±1.91** | 60.79±0.00 | 70.17±3.94 | 66.62±2.66 | 38.68±0.00 | 54.24±0.00 | 71.90±5.05 | 51.61±0.00 | 77.29±0.00 |
| ORL_3views | 76.28±0.00 | 87.64±1.07 | 81.20±1.38 | 80.93±1.39 | 69.15±0.00 | 83.35±1.13 | 84.85±0.29 | 81.58±0.00 | 49.23±0.00 | 86.26±0.00 | **88.80±0.00** | *87.96±0.00* |
| ORL_4views | 71.83±0.00 | *80.28±1.37* | 48.33±0.81 | 43.87±0.84 | 65.32±0.00 | 79.39±1.15 | 72.73±1.13 | 76.30±0.00 | 43.32±0.00 | 77.41±0.54 | 76.89±0.00 | **81.13±0.00** |
| SUN RGB-D | **27.81±0.00** | 18.46±0.66 | OOM | OOM | 12.98±0.00 | 24.50±0.37 | 22.71±0.41 | 2.30±0.00 | 15.66±0.00 | 22.84±0.82 | 11.37±0.00 | *26.89±0.00* |
| YouTubeFace20 | 78.34±0.00 | OOM | OOM | OOM | 70.65±0.00 | 76.78±1.34 | *78.36±2.39* | - | 45.45±0.00 | 74.30±1.95 | 73.18±0.00 | **81.17±0.00** |
| YouTubeFace50 | OOM | OOM | OOM | OOM | 81.90±0.00 | 82.43±0.78 | *83.63±0.85* | - | 43.03±0.00 | 82.08±1.07 | 77.18±0.00 | **86.43±0.00** |
| YouTubeFace100 | OOM | OOM | OOM | OOM | *82.23±0.00* | 81.38±0.60 | 79.90±0.77 | - | 29.96±0.00 | 77.06±1.53 | 61.43±0.00 | **84.37±0.00** |
| Average Rank | 6.25 | 6.14 | 5.50 | 7.00 | 7.90 | 4.60 | 4.80 | 9.29 | 8.90 | 4.20 | 6.20 | 1.60 |
| | Purity (%) | | | | | | | | | | | |
| Yale | 62.42±0.00 | 66.64±3.14 | *70.08±5.39* | 60.47±3.92 | 43.03±0.00 | 62.40±3.27 | 66.06±0.00 | 48.48±0.00 | 35.15±0.00 | 67.27±0.00 | 66.06±0.00 | **72.12±0.00** |
| 3sources | 59.17±0.00 | 21.21±0.32 | 67.74±2.56 | 78.31±2.52 | 67.46±0.00 | 60.52±1.34 | 75.73±1.83 | 35.50±0.00 | 66.86±0.00 | *78.70±0.00* | 39.05±0.00 | **82.25±0.00** |
| MSRCV1 | 74.76±0.00 | 80.45±4.29 | 79.01±4.16 | 49.91±3.78 | 27.14±0.00 | *85.25±5.56* | 71.51±4.02 | 62.86±0.00 | 50.48±0.00 | 72.33±5.01 | 70.95±0.00 | **85.71±0.00** |
| Dermatology | 75.70±0.00 | 23.12±0.50 | 84.79±2.87 | *85.37±1.89* | 65.08±0.00 | 80.97±4.29 | 80.35±3.73 | 50.00±0.00 | 62.85±0.00 | 83.55±6.84 | 66.48±0.00 | **89.11±0.00** |
| ORL_3views | 61.25±0.00 | 76.47±2.02 | 69.10±2.70 | 67.21±2.73 | 51.00±0.00 | 69.18±2.19 | 72.17±1.08 | **79.25±0.00** | 28.25±0.00 | 73.75±0.00 | *79.00±0.00* | 78.75±0.00 |
| ORL_4views | 53.00±0.00 | 66.92±2.07 | 26.48±1.14 | 23.90±1.08 | 47.50±0.00 | 65.68±2.53 | 51.91±2.16 | **78.00±0.00** | 21.75±0.00 | 58.97±1.39 | 63.00±0.00 | *69.50±0.00* |
| SUN RGB-D | **38.55±0.00** | 10.74±0.37 | OOM | OOM | 21.13±0.00 | 37.42±0.53 | 32.64±0.65 | 11.47±0.00 | 25.15±0.00 | 32.77±1.15 | 17.65±0.00 | *37.52±0.00* |
| YouTubeFace20 | *77.24±0.00* | OOM | OOM | OOM | 62.76±0.00 | 73.40±2.75 | 72.40±3.96 | - | 40.34±0.00 | 64.92±3.83 | 68.31±0.00 | **79.45±0.00** |
| YouTubeFace50 | OOM | OOM | OOM | OOM | *73.64±0.00* | 73.21±2.18 | 72.72±2.61 | - | 22.83±0.00 | 66.84±3.02 | 67.83±0.00 | **81.81±0.00** |
| YouTubeFace100 | OOM | OOM | OOM | OOM | *74.75±0.00* | 70.03±1.65 | 64.18±2.07 | - | 12.17±0.00 | 57.89±2.68 | 49.99±0.00 | **75.86±0.00** |
| Average Rank | 6.00 | 6.86 | 5.33 | 7.50 | 7.60 | 4.60 | 5.30 | 7.43 | 9.10 | 4.70 | 6.30 | 1.40 |
| | F-score (%) | | | | | | | | | | | |
| Yale | 42.68±0.00 | 41.47±3.53 | *56.30±4.88* | 43.23±4.15 | 21.15±0.00 | 46.42±2.74 | 52.60±0.00 | 31.28±0.00 | 17.97±0.00 | 54.19±0.00 | 45.91±0.00 | **58.12±0.00** |
| 3sources | 44.16±0.00 | 26.80±0.30 | 56.03±4.84 | **67.76±4.40** | 45.93±0.00 | 43.64±1.98 | 56.95±1.14 | 37.77±0.00 | 59.00±0.00 | 62.81±0.00 | 36.74±0.00 | *66.88±0.00* |
| MSRCV1 | 59.98±0.00 | 70.28±4.42 | 66.76±4.50 | 34.05±2.34 | 16.01±0.00 | **77.43±6.43** | 59.31±2.82 | 52.43±0.00 | 33.85±0.00 | 61.55±3.54 | 63.58±0.00 | *76.42±0.00* |
| Dermatology | 74.82±0.00 | 18.46±0.78 | 77.59±5.15 | **83.50±4.24** | 56.62±0.00 | 70.50±4.17 | 70.06±3.60 | 42.90±0.00 | 57.89±0.00 | 77.37±6.23 | 53.89±0.00 | *82.79±0.00* |
| ORL_3views | 41.34±0.00 | 53.73±6.36 | 54.00±3.15 | 52.57±3.06 | 31.60±0.00 | 56.52±3.38 | 55.98±0.76 | 32.35±0.00 | 13.80±0.00 | *58.73±0.00* | 47.51±0.00 | **63.35±0.00** |
| ORL_4views | 33.66±0.00 | 35.12±4.54 | 9.27±0.83 | 6.48±0.54 | 24.84±0.00 | **50.10±2.86** | 32.37±1.71 | 23.74±0.00 | 12.09±0.00 | 42.87±1.47 | 23.11±0.00 | *45.96±0.00* |
| SUN RGB-D | 13.15±0.00 | 6.46±0.22 | OOM | OOM | 15.16±0.00 | 11.41±0.23 | 14.99±0.14 | 12.17±0.00 | 14.08±0.00 | *15.23±0.39* | 13.80±0.00 | **16.25±0.00** |
| YouTubeFace20 | *65.59±0.00* | OOM | OOM | OOM | 49.04±0.00 | 62.43±2.91 | 61.68±5.99 | - | 25.84±0.00 | 57.81±4.00 | 42.07±0.00 | **70.40±0.00** |
| YouTubeFace50 | OOM | OOM | OOM | OOM | 57.09±0.00 | 62.49±2.45 | *63.52±2.56* | - | 15.67±0.00 | 56.89±3.18 | 24.29±0.00 | **71.52±0.00** |
| YouTubeFace100 | OOM | OOM | OOM | OOM | **59.77±0.00** | 56.47±2.41 | 49.79±2.75 | - | 6.33±0.00 | 37.44±5.64 | 5.57±0.00 | *59.58±0.00* |
| Average Rank | 6.38 | 8.00 | 5.17 | 6.33 | 7.10 | 4.30 | 4.80 | 9.43 | 8.30 | 3.80 | 7.50 | 1.50 |

[‡] denotes stable algorithm.

dataset. MSRCV1 [58] contains 210 images from 7 clusters. ORL_3views and ORL_4views[3] are face datasets containing 400 images from 40 categories but with different views. SUN RGB-D[4] contains 10,335 real RGB-D images of room scenes. YouTubeFace20, YouTubeFace50, and YouTubeFace100 are face video datasets extracted from YouTube [21] with different number of clusters.

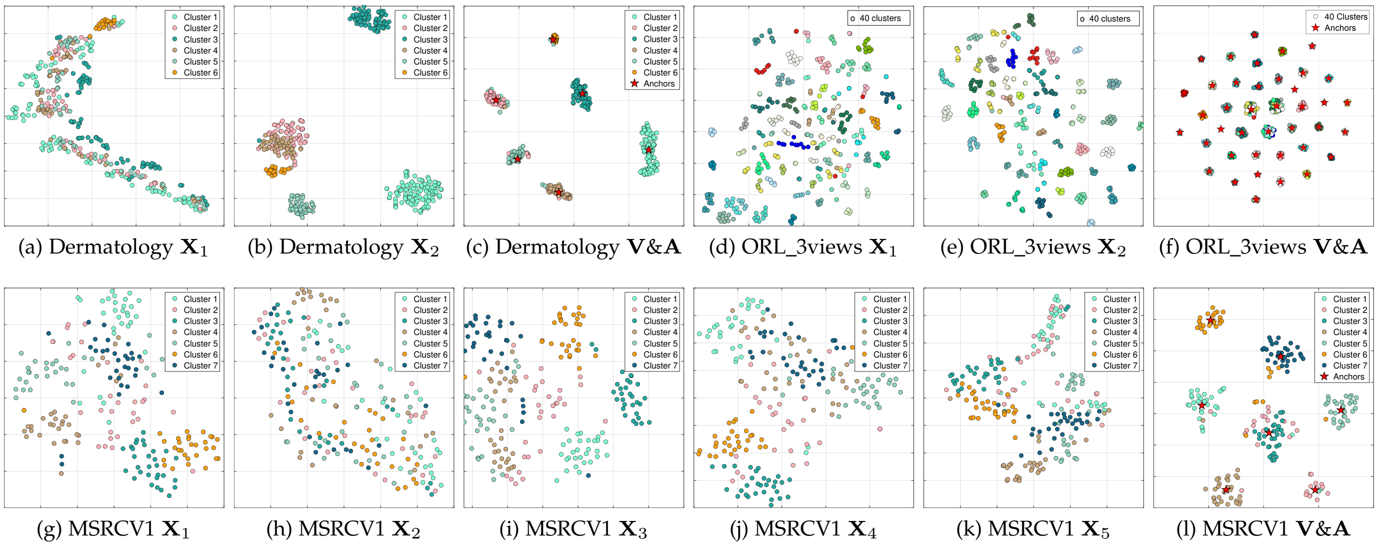
### 4.1.3 Compared Baselines

To evaluate the effectiveness of the proposed BGAE framework, thirteen state-of-the-art baselines are collected. 1) RMKM [50] proposes a robust MVC method using $\ell_{2,1}$-norm. 2) AMGL [51] designs a multi-graph clustering model with an auto-weighted strategy. 3) FMR [52] introduces kernel dependence measure to extract latent representation with nonlinear and high-order structures. 4) PMSC [53] fuses multiple views in the partition level. 5) BMVC [54] incorporates collaborative binary coding and binary cluster structure learning. 6) LMVSC [26] proposes sampling anchors by $k$-means and concatenating multiple anchor graphs to exploit complementary information. 7) SMVSC [18] proposes to project multiple views into a latent space and learn unified anchors through optimization. 8) SFMC [25] restricts generating the same anchors across multiple views and fuses a unified bipartite graph with Laplacian rank constraint. 9) FMCNOF [55] designs an orthogonal NMF variant and fuses a unified indicator matrix to predict labels directly. 10) FPMVS [56] is a parameter-free extension of SMVSC. 11) SDAFG [45] exploits structural diversity by merging diverse anchor graphs into a large target graph with connectivity constraints. 12) UDBGL [57] fuses view-wise and view-consensus information to learn a unified anchor graph and coupled with connectivity constraints to hold discrete cluster structures. 13) FastMICE [21] introduces the concept of random view groups to capture multi-view relationships and devises a hybrid fusion method to combine diversities of features, anchors, and neighbors, achieving ensemble clustering.

3. https://cam-orl.co.uk/facedatabase.html
4. https://rgbd.cs.princeton.edu/

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217



(a) Dermatology $\mathbf{X}_1$ (b) Dermatology $\mathbf{X}_2$ (c) Dermatology $\mathbf{V}$&$\mathbf{A}$ (d) ORL_3views $\mathbf{X}_1$ (e) ORL_3views $\mathbf{X}_2$ (f) ORL_3views $\mathbf{V}$&$\mathbf{A}$

(g) MSRCV1 $\mathbf{X}_1$ (h) MSRCV1 $\mathbf{X}_2$ (i) MSRCV1 $\mathbf{X}_3$ (j) MSRCV1 $\mathbf{X}_4$ (k) MSRCV1 $\mathbf{X}_5$ (l) MSRCV1 $\mathbf{V}$&$\mathbf{A}$

Fig. 4: t-SNE visualization of the input multi-view data $\{\mathbf{X}_p\}_{p=1}^v$ and the extracted latent representation $\mathbf{V}$.

#### *4.1.4 Technical Details*

The codes of compared baselines are collected directly from the authors' homepage or GitHub without corrections, hyper-parameters are carefully tuned following the authors' suggestions, and we report the best metrics. For baselines involving $k$-means, the results mean $\pm$ std are reported by repeating 50 times to alleviate randomness. The cluster number $k$ is assumed pre-known following most experimental settings [59], [60], [61].

For our BGAE, the anchor number $m$ is fixed at $k$, the latent dimension $d$ of representation $\mathbf{V}$ varies in $[k, 2k, \ldots, 9k]$ and $d \leq \min_p \{d_p\}_{p=1}^v$ should be satisfied. Following [25], $\mu$ is heuristically updated by measuring the gap of $k$ and $\iota$, where $\iota$ denotes the multiplicity of eigenvalue zeros. Specifically, $\mu$ is initially set to 1 and iteratively updated by $\mu = 2 \times \mu$ if $\iota \leq k$ or $\mu = \frac{\mu}{2}$ if $\iota > k + 1$.

Performance is measured by accuracy (ACC), normalized mutual information (NMI), purity, and F-score [62], [63]. Experiments were performed on a desktop with Intel(R) i9 9900K CPUs @3.6GHZ, 64 GB RAM, and Matlab 2020b.

### 4.2 Effectiveness of BGAE Compared to Baselines

Table 4 reports clustering metrics, and we observe that:

1) Our novel BGAE achieves competitive performance and ranks first in most cases. Compared to the runner-up ones, ours achieves 3.31%, 4.43%, 1.98%, 6.15%, 2.50%, 3.50%, 1.44%, 4.45%, 7.74%, and 2.80% improvement of ACC on ten datasets, respectively. In particular, our superiority is evident on large-scale datasets ($n \geq 38,654$), demonstrating the effectiveness. Moreover, our end-to-end model does not require post-processing, avoiding the randomness of $k$-means.
2) RMKM, AMGL, FMR, and PMSC are MVC models with space complexity $\mathcal{O}(n^2)$, requiring to construct fully connected graphs, they suffer "OOM" on large-scale datasets ($n \geq 63,896$) and exhibit limited scalability, while our BGAE can still handle such challenging tasks with promising performance.
3) LMVSC and FPMVS are the two strongest MVBGC competitors. However, the separation of clustering and post-processing results in unstable performance and suboptimal solutions. Mostly, all BGC baselines only consider the "encoding" learning process but omit the "decoding" process in bipartite graph learning, their performance is inferior to ours.

### 4.3 Significance of the "Encoding" Module

To reveal the discrimination of the encoded representation intuitively, Fig. 4 t-SNE visualizes the original data distribution $\{\mathbf{X}_p\}_{p=1}^v$ and the latent representation $\mathbf{V}$ on MRSCV1 (7 clusters), Dermatology (6 clusters), and ORL_3views (40 clusters). We observe that:

1) Input data stacks together and shows complex curved or folded manifold structures. The separability for disjoint clusters is inconspicuous. By contrast, by extracting the latent representation, the data show much clearer separability, even for ORL_3views that includes 40 clusters, the decision boundaries are still distinct. Although the encoded latent representation may mistake the correlation of instances caused by complex manifolds, subsequent bipartite graph learning can further exploit intrinsic structures and correct several mistaken memberships, as shown in Fig. 6.
2) An interesting phenomenon is that the learned anchors almost lie in the centroids of clusters, exhibiting discriminative property, and conform to our intuitive and ideal pursuit in bipartite graph learning. A reasonable explanation is that we enforce the connectivity constraint on the bipartite graph to ensure the $k$-connected components, making the bipartite graph $\mathbf{Z}$ sparse enough to satisfy such a constraint. Recalling anchors are optimized without constraint, and our experimental setting $m = k$ contributes to approaching the intuitive result that each anchor lies in the corresponding centroid of the cluster. The results verify the reasonability of our unconstrained optimization strategy on anchors $\mathbf{A}$.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

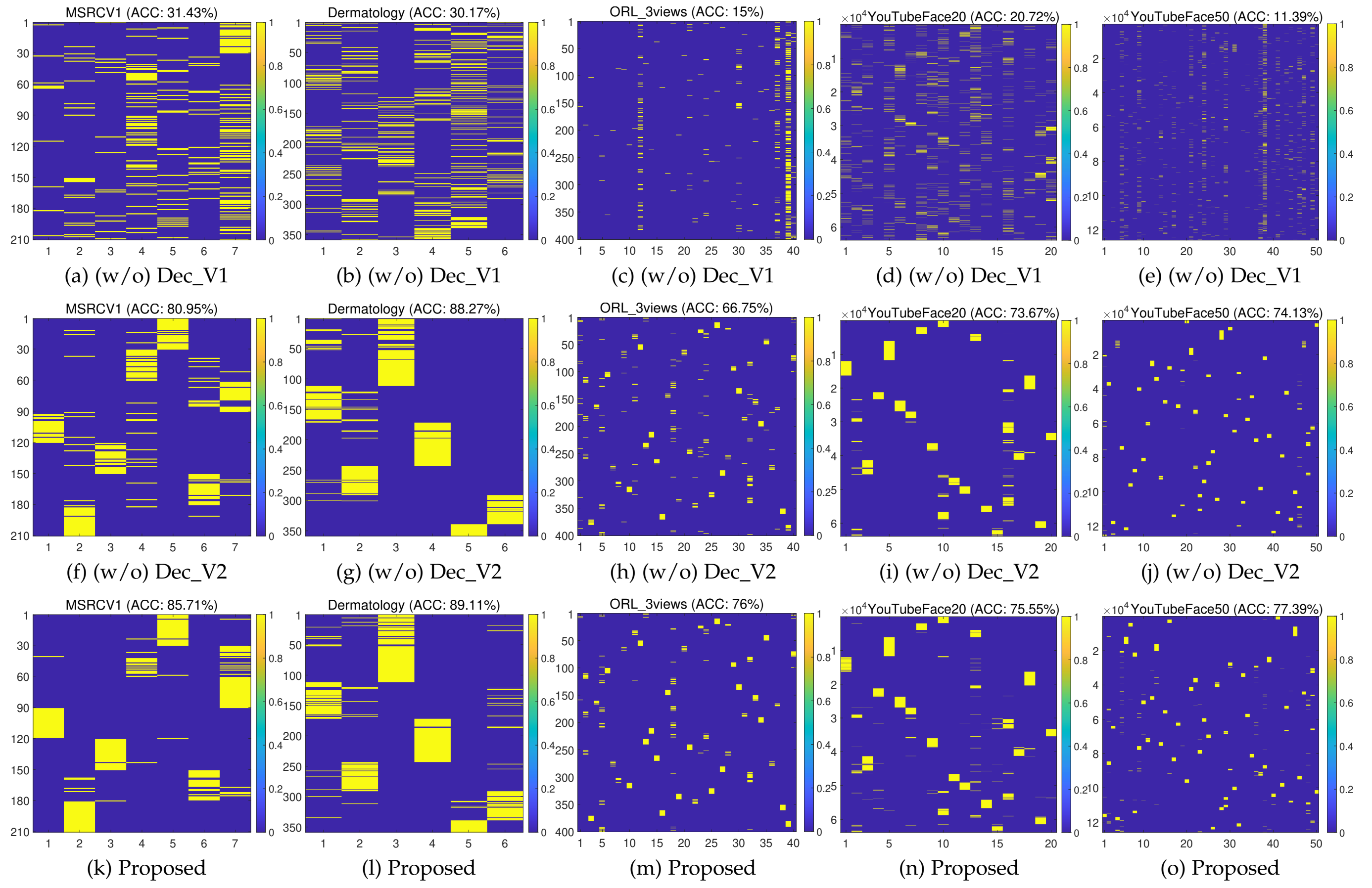


Fig. 5: Ablation analysis of the "decoding" module by visualizing the bipartite graph representation.

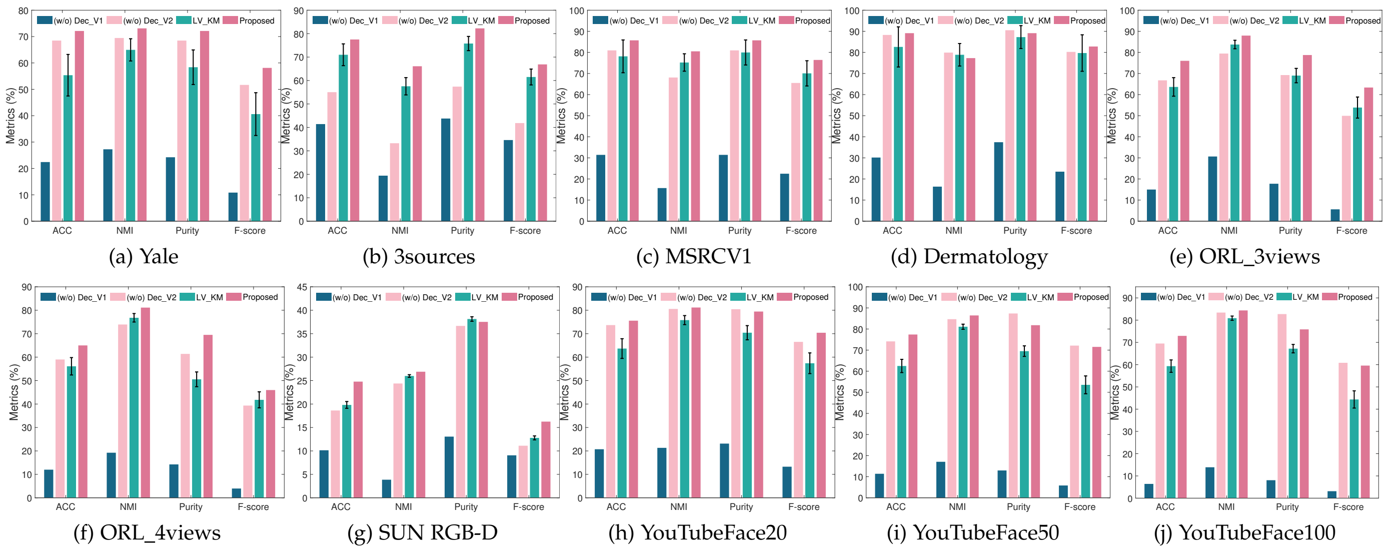


Fig. 6: Ablation analysis of the "decoding" module by quantifying the clustering metrics.

### 4.4 Ablation Analysis of the "Decoding" Module

To verify the significance of our novel "encoding-decoding" design, Table 5 lists ablation study settings. For simplicity, "(w/o) Dec_V1" denotes our model that removes the "decoding" module. "(w/o) Dec_V2" denotes "(w/o) Dec_V1" with a quadratic term of $\mathbf{Z}$, i.e. introducing an additional regularization term $\zeta z_{ij}^2$, where the balanced parameter $\zeta$ is heuristically pre-determined following [43]. As a reference, we also give the metrics of the latent representation $\mathbf{V}$ coupled $k$-means to output labels, called "LV_KM".

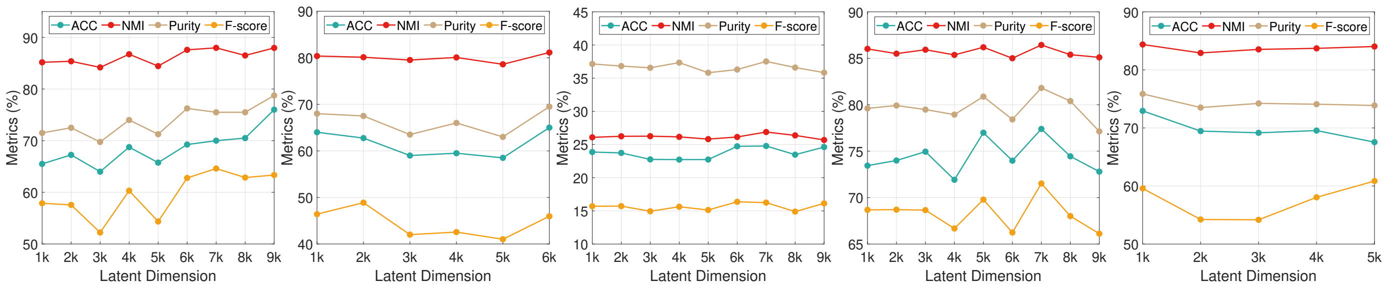
TABLE 5: Experimental settings of ablation analysis

| Model | $\mathcal{L}_1$ | $\mathcal{L}_2$ | $\mathcal{L}_3$ | Extra Regularizer |
|---|---|---|---|---|
| (w/o) Dec_V1 | ✓ | ✓ | – | – |
| (w/o) Dec_V2 | ✓ | ✓ | – | ✓ |
| Proposed | ✓ | ✓ | ✓ | – |

Firstly, Fig. 5 visualize bipartite graphs on MSRCV1, Dermatology, ORL_3views, YouTubeFace20, and YouTubeFace50 datasets. Since we impose connectivity constraint,

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217



(a) ORL_3views (b) ORL_4views (c) SUN RGB-D (d) YouTubeFace50 (e) YouTubeFace100

Fig. 7: Effect of latent feature dimension.

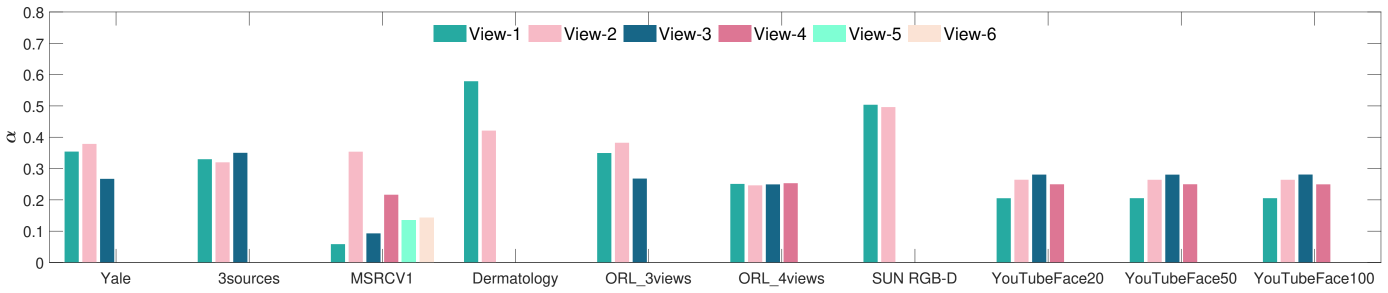


Fig. 8: View weights $\boldsymbol{\alpha}$ in representation learning process.

bipartite graphs show sparse representation. Caused by the undesirable trivial solution, "(w/o) Dec_V1" largely mistake the memberships of anchors and instances, immensely destroying graph representation. Introducing additional regularization can correct the mistaken memberships, as shown in "(w/o) Dec_V2". However, since "(w/o) Dec_V2" ignores the guidance of the "decoding" process, it still mistakes some memberships and degrades performance, as shown in MSRCV1 and ORL_3views. Mostly, the extra regularizer acts as a penalty term to avoid trivial solutions, which leads to an undesirable hyper-parameter $\zeta$ without practical interpretations, requiring additional parameter-tuning or heuristic solutions.

Furthermore, Fig. 6 quantifies clustering metrics. We observe that "(w/o) Dec_V1" outputs dramatically poor metrics. "(w/o) Dec_V2" apparently improves clustering metrics. "LV_KM" introduces unstable performance caused by the randomness of $k$-means. By contrast, our BGAE achieves the best metrics and outperforms baselines with large margins of 3.64%, 6.52%, 4.76%, 0.84%, 9.25%, 6.00%, 4.97%, 1.89%, 3.26%, and 3.44% of ACC, respectively. In particular, our BGAE shows significant improvement over "LV_KM" in most cases, indicating that the bipartite graph construction module can further explore and refine the latent representation encoded from "encoding" module.

In summary, this is convincing evidence corroborating the effectiveness of our novel MVBGC design and the improvement brought by the "decoding" module.

### 4.5 Effect of Latent Feature Dimension

Considering pre-determining the optimal latent feature dimension $d$ is still a challenging problem in unsupervised learning, Fig. 7 reports clustering metrics with varying latent feature dimensions in the range $[k, 2k, \dots, 9k]$ on five datasets. As pointed in Section 4.1.4, $d \leq \min\limits_{p}\{d_p\}_{p=1}^{v}$ should be satisfied, the maximum latent dimension available for ORL_4views and YouTubeFace100 is $d = 6k$ and $d = 5k$, respectively. We find that dimension-metric curves show dataset-related results and do not increase monotonously but fluctuate. The results coincide with the knowledge that higher dimensions can enrich the volume of information, but may also induce redundancy or noise. How to pre-determine the optimal latent feature dimension in unsupervised learning is still an open question, which deserves future research.

### 4.6 View Weight Distribution

Fig. 8 plots the view contribution $\boldsymbol{\alpha}$ in "encoding" module. We observe that the distribution exhibits dataset-related results. Due to potential noise or redundancy within the input data, different views provide different contributions to extracting latent representation, as shown on MRSCV1, Dermatology, and ORL_3views. The results demonstrate that designing flexible and adaptive fusion mechanisms is important in multi-view learning. The $\boldsymbol{\gamma}$ distribution in the "decoding" module is available in supplementary material.

### 4.7 Efficiency

Fig. 9 plots time consumption, we observe that:

1) Although our model requires comparative even more execution time compared to full graph baselines on small-scale datasets, such as Yale, 3sources, MSRCV1, and ORL_3views, which is mainly caused by complex optimization, unacceptable "OOM" on large-scale datasets will not occur for our BGAE, demonstrating the superiority of our promising scalability with linear complexity.
2) Although BGC baselines and ours share similar linear complexity, our BGAE costs comparative or more running time due to ADMM solver. However, these baselines omit the "decoding" process with degraded unstable performance. Generally, we believe that the extra computation is worthwhile for competitive performance.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

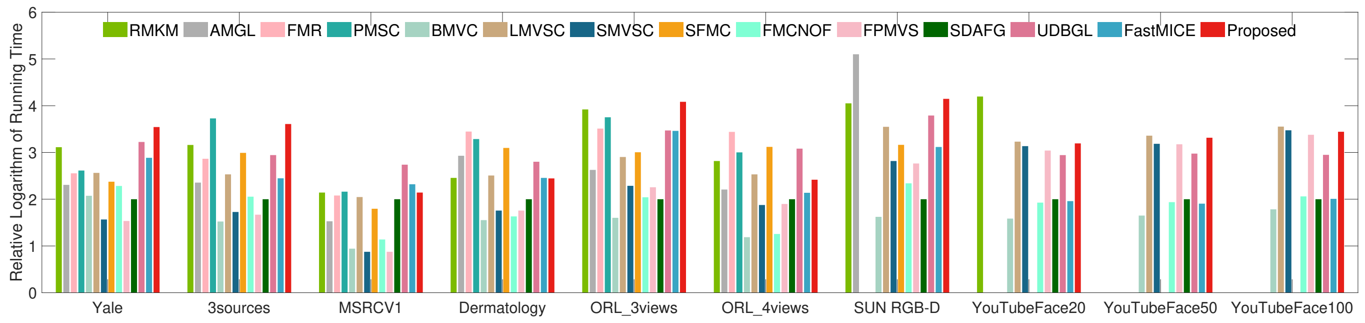
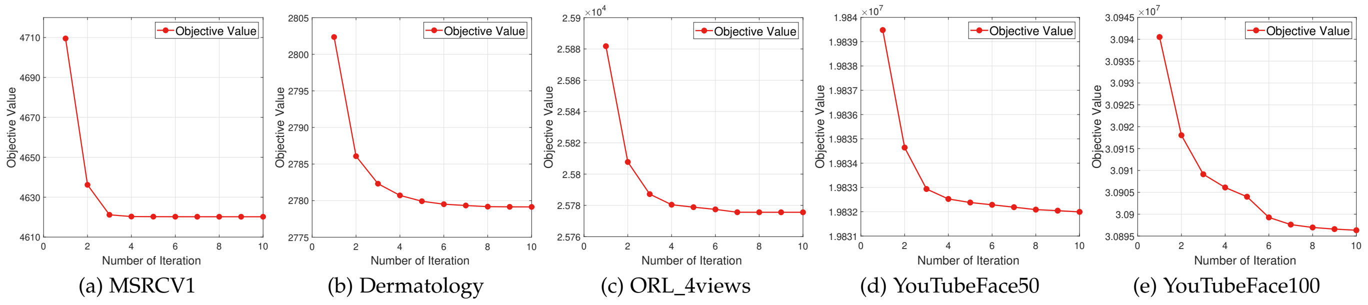


Fig. 9: Comparison of the relative logarithm running time. The compared SDAFG is the baseline.



(a) MSRCV1 (b) Dermatology (c) ORL_4views (d) YouTubeFace50 (e) YouTubeFace100

Fig. 10: Empirical validation of the convergence.

## 4.8 Convergence

Our solver uses a block-coordinate descent method. The original objective in Eq. (8) is separated into eight subproblems, and each one has a closed-form solution. Although the ALM parameter $\beta$ increases iteratively, it controls the convergence speed and generally has little impact on the final results. Ideally, as $\beta$ increases, the last term of Eq. (11) will be close to 0, and the ALM objective converges asymptotically to the original function bounded by 0. According to previous research on ALM framework [38], [39], the original function decreases monotonically with iterations and thus converges to a local optimal solution.

Fig. 10 empirically validates the convergence of the original function, further confirming the convergence on all benchmark datasets. Our model typically converges within 20 iterations, demonstrating its efficiency. More experimental results are provided in supplementary material.

## 5 Conclusion

This paper revisits existing MVBGC paradigms and finds that existing models adopt a common design that encodes input data directly into bipartite graphs. Enlightened by the popular AE in deep learning, we transfer the "auto-encoding" design into traditional graph machine learning, and propose a novel BGAE model, which consists of encoding, bipartite graph construction, and decoding modules. The encoding module extracts a latent representation from the input data in a robust manner, the bipartite graph construction module learns a discriminative bipartite graph, and the decoding module recreates the input data. All these modules are seamlessly integrated and mutually enhanced. We design an ADMM solver with linear complexity respecting instances. Empirical experiments on a synthetic dataset visualize the "benefit" of decoding learning to retain the initial manifold, and ablation analysis further verifies the effectiveness. This paper investigates how to build "auto-encoding" design in graph machine learning, we believe these novel insights will promote more variants proposed based on our novel design. This paper introduces $\ell_{2,1}$-norm to hold robustness to noise or outliers. Recent uncertainty-aware learning [64], [65] provides another solution that can measure the evidence for predictions, so developing trusted MVBGC that reduces uncertainty is meaningful. In addition, this paper assumes that the input data is complete. However, incomplete data within multi-view data is more common and challenging in real-world scenarios. So, another of our future work is to extend "auto-encoding" design to incomplete scenarios.

## References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[2] Y. Yang, Z. Guan, Z. Wang, W. Zhao, C. Xu, W. Lu, and J. Huang, "Self-supervised heterogeneous graph pre-training based on structural clustering," in *Proc. Adv. Neural Inf. Process. Syst., New Orleans, LA, USA*, 2022.
[3] L. Zou, F. Zhang, Y. Lin, and Y. Yu, "An efficient data structure for dynamic graph on gpus," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 1–15, 2023.
[4] K. Liang, Y. Liu, S. Zhou, W. Tu, Y. Wen, X. Yang, X. Dong, and X. Liu, "Knowledge graph contrastive learning based on relation-symmetrical structure," *IEEE Trans. Knowl. Data Eng.*, pp. 1–12, 2023.
[5] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. of the 23-th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., Halifax, Canada*, 2017, pp. 555–564.
[6] H. Wang, Y. Yang, B. Liu, and H. Fujita, "A study of graph-based system for multi-view clustering," *Knowl. Based Syst.*, vol. 163, pp. 1009–1019, 2019.
[7] W. Zhao, C. Xu, Z. Guan, X. Wu, W. Zhao, Q. Miao, X. He, and Q. Wang, "Telecomnet: Tag-based weakly-supervised modally cooperative hashing network for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7940–7954, 2022.
[8] S. Sun, J. Fei, J. Zhao, and L. Mao, "Multi-view collaborative gaussian process dynamical systems," *J. Mach. Learn. Res.*, vol. 24, no. 258, pp. 1–32, 2023.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

[9] K. Liang, L. Meng, M. Liu, Y. Liu, W. Tu, S. Wang, S. Zhou, X. Liu, and F. Sun, "Reasoning over different types of knowledge graphs: Static, temporal and multi-modal," *arXiv preprint arXiv:2212.05767*, 2022.

[10] U. Fang, M. Li, J. Li, L. Gao, T. Jia, and Y. Zhang, "A comprehensive survey on multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12 350–12 368, 2023.

[11] S. Sun, L. Mao, Z. Dong, and L. Wu, *Multiview machine learning*. Springer, 2019.

[12] S. Huang, I. W. Tsang, Z. Xu, and J. Lv, "Latent representation guided multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7082–7087, 2022.

[13] H. Wang, Y. Yang, and B. Liu, "GMC: graph-based multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 6, pp. 1116–1129, 2020.

[14] H. Xiao, Y. Chen, and X. Shi, "Knowledge graph embedding based on multi-view clustering framework," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 2, pp. 585–596, 2021.

[15] H. Wang, Z. Cheng, J. Sun, X. Yang, X. Wu, H. Chen, and Y. Yang, "Debunking free fusion myth: Online multi-view anomaly detection with disentangled product-of-experts modeling," in *Proc. of the 31-th ACM Int. Conf. Multimedia, Ottawa, ON, Canada*, 2023, pp. 3277–3286.

[16] F. Nie, W. Chang, Z. Hu, and X. Li, "Robust subspace clustering with low-rank structure constraint," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 3, pp. 1404–1415, 2022.

[17] L. Li, S. Wang, X. Liu, E. Zhu, L. Shen, K. Li, and K. Li, "Local sample-weighted multiple kernel clustering with consensus discriminative graph," *IEEE Trans. Neural Networks*, pp. 1–14, 2022.

[18] M. Sun, P. Zhang, S. Wang, S. Zhou, W. Tu, X. Liu, E. Zhu, and C. Wang, "Scalable multi-view subspace clustering with unified anchors," in *Proc. of the 29-th ACM Int. Conf. Multimedia, Virtual Event, China*, 2021, pp. 3528–3536.

[19] N. Zhang, X. Zhang, and S. Sun, "Efficient multiview representation learning with correntropy and anchor graph," *IEEE Trans. Knowl. Data Eng.*, pp. 1–14, 2023.

[20] H. Zhang, F. Nie, and X. Li, "Large-scale clustering with structured optimal bipartite graph," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 1–14, 2023.

[21] D. Huang, C. Wang, and J. Lai, "Fast multi-view clustering via ensembles: Towards scalability, superiority, and simplicity," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11 388–11 402, 2023.

[22] L. Li, J. Zhang, S. Wang, X. Liu, K. Li, and K. Li, "Multi-view bipartite graph clustering with coupled noisy feature filter," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 1–13, 2023.

[23] L. Li and H. He, "Bipartite graph based multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 7, pp. 3111–3125, 2022.

[24] H. Chen, F. Nie, R. Wang, and X. Li, "Fast unsupervised feature selection with bipartite graph and $\ell_{2,0}$-norm constraint," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4781–4793, 2023.

[25] X. Li, H. Zhang, R. Wang, and F. Nie, "Multiview clustering: A scalable and parameter-free bipartite graph fusion method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 330–344, 2022.

[26] Z. Kang, W. Zhou, Z. Zhao, J. Shao, M. Han, and Z. Xu, "Large-scale multi-view subspace clustering in linear time," in *Proc. of the 34-th Conf. Artif. Intell., New York, NY, USA*, 2020, pp. 4412–4419.

[27] P. Zhang, S. Wang, L. Li, C. Zhang, X. Liu, E. Zhu, Z. Liu, L. Zhou, and L. Luo, "Let the data choose: Flexible and diverse anchor graph fusion for scalable multi-view clustering," in *Proc. of the 37-th AAAI Conf. Artif. Intell., Washington, DC, USA*, 2023, pp. 11 262–11 269.

[28] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[29] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. of the 25-th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[30] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[31] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[32] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proc. of the IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 16 000–16 009.

[33] C. Hong, J. Yu, J. Wan, D. Tao, and M. Wang, "Multimodal deep autoencoder for human pose recovery," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5659–5670, 2015.

[34] C. Xu, Z. Guan, W. Zhao, H. Wu, Y. Niu, and B. Ling, "Adversarial incomplete multi-view clustering," in *Proc. of the 28-th IJCAI Int. Jt. Conf. Artif. Intell., Macao, China*, 2019, pp. 3933–3939.

[35] S. Lin, W. Yang, H. Wang, Q. Tsai, and K. Li, "Stm-multifrontal QR: streaming task mapping multifrontal QR factorization empowered by GCN," in *Int. Conf. High Perform. Comput. Netw. Storage Anal., St. Louis, Missouri, USA*, 2021, p. 71.

[36] S. Lin, W. Yang, Y. Hu, Q. Cai, M. Dai, H. Wang, and K. Li, "Hps cholesky: Hierarchical parallelized supernodal cholesky with adaptive parameters," *ACM Trans. Parallel Comput.*, 2023.

[37] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[38] D. Kong, C. H. Q. Ding, and H. Huang, "Robust nonnegative matrix factorization using $\ell_{2,1}$-norm," in *Proc. of the 20th ACM Int. Conf. Inf. Knowl. Manag., Glasgow, United Kingdom*, 2011, pp. 673–682.

[39] J. Huang, F. Nie, H. Huang, and C. Ding, "Robust manifold non-negative matrix factorization," *ACM Trans. Knowl. Discov. Data.*, vol. 8, no. 3, pp. 1–21, 2014.

[40] C. H. Q. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 45–55, 2010.

[41] C. H. Q. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix t-factorizations for clustering," in *Proc. of the 12-th ACM SIGKDD Int. Conf. on Knowl. Discov. Data. Min., Philadelphia, PA, USA*, 2006, pp. 126–135.

[42] L. K. Saul and S. T. Roweis, "Think globally, fit locally: unsupervised learning of low dimensional manifolds," *J. Mach. Learn. Res.*, vol. 4, no. Jun, pp. 119–155, 2003.

[43] Z. Wang, L. Zhang, R. Wang, F. Nie, and X. Li, "Semi-supervised learning via bipartite graph construction with adaptive neighbors," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 5257–5268, 2023.

[44] F. Nie, X. Dong, L. Tian, R. Wang, and X. Li, "Unsupervised feature selection with constrained $\ell_{2,0}$-norm and optimized graph," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 4, pp. 1702–1713, 2020.

[45] X. Lu and S. Feng, "Structure diversity-induced anchor graph fusion for multi-view clustering," *ACM Trans. Knowl. Discov. Data*, vol. 17, no. 2, pp. 17:1–17:18, 2023.

[46] W. Yan, J. Xu, J. Liu, G. Yue, and C. Tang, "Bipartite graph-based discriminative feature learning for multi-view clustering, lisboa, portugal," in *Proc. of the 30-th ACM Int. Conf. Multimedia, Lisboa, Portugal*, 2022, pp. 3403–3411.

[47] C. H. Q. Ding, D. Zhou, X. He, and H. Zha, "$R_1$-pca: rotational invariant $\ell_1$-norm principal component analysis for robust subspace factorization," in *Proc. of the 23-th Int. Conf. Mach. Learn., Pittsburgh, Pennsylvania, USA*, vol. 148, 2006, pp. 281–288.

[48] C. Xu, W. Zhao, J. Zhao, Z. Guan, Y. Yang, L. Chen, and X. Song, "Progressive deep multi-view comprehensive representation learning," in *Proc. of the 37-th AAAI Conf. Artif. Intell., Washington, DC, USA*, 2023, pp. 10 557–10 565.

[49] K. Fan, "On a theorem of weyl concerning eigenvalues of linear transformations i," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 35, no. 11, pp. 652–655, 1949.

[50] X. Cai, F. Nie, and H. Huang, "Multi-view k-means clustering on big data," in *Proc. of the 23-th IJCAI Int. Jt. Conf. Artif. Intell., Beijing, China*, 2013, pp. 2598–2604.

[51] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification," in *Proc. of the 25-th IJCAI Int. Jt. Conf. Artif. Intell., New York, NY, USA*, 2016, pp. 1881–1887.

[52] R. Li, C. Zhang, Q. Hu, P. Zhu, and Z. Wang, "Flexible multi-view representation learning for subspace clustering," in *Proc. of the 28-th IJCAI Int. Jt. Conf. Artif. Intell., Macao, China*, 2019, pp. 2916–2922.

[53] Z. Kang, X. Zhao, C. Peng, H. Zhu, J. T. Zhou, X. Peng, W. Chen, and Z. Xu, "Partition level multiview subspace clustering," *Neural Netw.*, vol. 122, pp. 279–288, 2020.

[54] Z. Zhang, L. Liu, F. Shen, H. T. Shen, and L. Shao, "Binary multi-view clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1774–1782, 2019.

[55] B. Yang, X. Zhang, F. Nie, F. Wang, W. Yu, and R. Wang, "Fast multi-view clustering via nonnegative and orthogonal factorization," *IEEE Trans. Image Process.*, vol. 30, pp. 2575–2586, 2020.

[56] S. Wang, X. Liu, X. Zhu, P. Zhang, Y. Zhang, F. Gao, and E. Zhu, "Fast parameter-free multi-view subspace clustering with consen-

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2024.3363217

sus anchor guidance," *IEEE Trans. Image Process.*, vol. 31, pp. 556–568, 2022.
[57] S. Fang, D. Huang, X. Cai, C. Wang, C. He, and Y. Tang, "Efficient multi-view clustering via unified and discrete bipartite graph learning," *IEEE Trans. Neural Networks*, pp. 1–12, 2023.
[58] J. M. Winn and N. Jojic, "LOCUS: learning object classes with unsupervised segmentation," in *Proc. of the 10-th IEEE Int. Conf. Comput. Vis., Beijing, China*, 2005, pp. 756–763.
[59] X. Li, Y. Sun, Q. Sun, and Z. Ren, "Consensus cluster center guided latent multi-kernel clustering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 6, pp. 2864–2876, 2023.
[60] X. Zhang, Y. Yang, D. Zhai, T. Li, J. Chu, and H. Wang, "Local2global: Unsupervised multi-view deep graph representation learning with nearest neighbor constraint," *Knowl. Based Syst.*, vol. 231, p. 107439, 2021.
[61] Y. Liu, K. Liang, J. Xia, S. Zhou, X. Yang, X. Liu, and S. Z. Li, "Dink-net: Neural clustering on large graphs," in *Proc. of the 40-th Int. Conf. Mach. Learn., Honolulu, Hawaii, USA*, vol. 202, 2023, pp. 21 794–21 812.
[62] S. Sun, W. Dong, and Q. Liu, "Multi-view representation learning with deep gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4453–4468, 2021.
[63] X. Li, Q. Sun, Z. Ren, and Y. Sun, "Dynamic incomplete multi-view imputing and clustering," in *Proc. of the 30-th ACM Int. Conf. Multimedia, Lisboa, Portugal*, 2022, pp. 3412–3420.
[64] M. Sensoy, L. M. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Proc. Adv. Neural Inf. Process. Syst., Montréal, Canada*, 2018, pp. 3183–3193.
[65] C. Xu, W. Zhao, J. Zhao, Z. Guan, X. Song, and J. Li, "Uncertainty-aware multiview deep learning for internet of things applications," *IEEE Trans. Industr Inform*, vol. 19, no. 2, pp. 1456–1466, 2023.

**Liang Li** received the bachelor's degree from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018, and the master's degree from National University of Defense Technology (NUDT), Changsha, China, in 2020, where he is currently pursuing the Ph.D. degree since 2021. Now, he is a visiting Ph.D. student at A*STAR Centre for Frontier AI Research, Singapore. His research interests include graph learning and self-supervised learning.

**Yuangang Pan** received the Ph.D. degree in computer science from the University of Technology Sydney (UTS), Ultimo, NSW, Australia, in 2020. He is working as a Research Scientist at the A*STAR Centre for Frontier AI Research, Singapore. He has authored or coauthored articles in various top journals, such as JMLR, TPAMI, TNNLS, TKDE, and TOIS. His research interests include deep clustering, deep generative learning, and robust ranking aggregation.

**Jie Liu** received the Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), Changsha, China. He is a professor with the College of Computer, National University of Defense Technology. His research interests include high performance computing and machine learning.

**Yue Liu** received the bachelor's degree from the Northeastern University, Qinhuangdao, China, in 2021. He is pursuing the master degree with the National University of Defense Technology, Changsha, China. His current research interests include graph neural networks, deep clustering and self-supervised learning.

**Xinwang Liu** (Senior Member, IEEE) received the Ph.D. degree from the National University of Defense Technology (NUDT), Changsha, China, in 2013. He is currently a Full Professor with the School of Computer, NUDT. His current research interests include kernel learning and unsupervised feature learning. He has published over 100 peer-reviewed papers, such as TPAMI, TKDE, TIP, TNNLS, TMM, TIFS, ICML, NeurIPS, ICCV, CVPR, AAAI, and IJCAI. He serves as an Associated Editor of the TNNLS and TCYB.

**Kenli Li** (Senior Member, IEEE) received the Ph.D. degree from Huazhong University of Science and Technology (HUST), China, in 2003. He is currently a full professor of computer science and technology at Hunan University and director of the National Supercomputing Center in Changsha. His research interests include parallel and distributed computing, high-performance computing, AI, cloud computing, and big data. He has published over 600 research papers, such as TC, TPDS, TCC, TKDE, DAC, AAAI, ICPP, etc. He serves as the editorial board of TC.

**Ivor W. Tsang** (Fellow, IEEE) is the Director of A*STAR Centre for Frontier AI Research, Singapore. He is a Professor of artificial intelligence with the University of Technology Sydney (UTS), Australia, and the Research Director of the Australian Artificial Intelligence Institute (AAII). His research interests include transfer learning, deep generative models, learning with weakly supervision, Big Data analytics for data with extremely high dimensions in features, samples and labels.

Dr. Tsang was the recipient of the ARC Future Fellowship for his outstanding research on Big Data analytics and large-scale machine learning, in 2013. In 2019, his JMLR article toward ultrahigh dimensional feature selection for Big Data was the recipient of the International Consortium of Chinese Mathematicians Best Paper Award. In 2020, he was recognized as the AI 2000 AAAI/IJCAI Most Influential Scholar in Australia for his outstanding contributions to the field between 2009 and 2019. He serves as the Editorial Board for JMLR, MLJ, JAIR, TPAMI, TAI, TBD, and TETCI. He serves/served as an AC or Senior AC for NeurIPS, ICML, AAAI, and IJCAI, and the steering committee of ACML.

**Keqin Li** (Fellow, IEEE) received a B.S. degree in computer science from Tsinghua University in 1985 and a Ph.D. degree in computer science from the University of Houston in 1990. He is currently a SUNY Distinguished Professor with the State University of New York and a National Distinguished Professor with Hunan University (China). He has authored or co-authored more than 950 journal articles, book chapters, and refereed conference papers. He received several best paper awards from international conferences including PDPTA-1996, NAECON-1997, IPDPS-2000, ISPA-2016, NPC-2019, ISPA-2019, and CPSCom-2022. He holds nearly 70 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top five most influential scientists in parallel and distributed computing in terms of single-year and career-long impacts based on a composite indicator of the Scopus citation database. He was a 2017 recipient of the Albert Nelson Marquis Lifetime Achievement Award. He received the Distinguished Alumnus Award from the Computer Science Department at the University of Houston in 2018. He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He is a Member of the SUNY Distinguished Academy. He is an AAAS Fellow, an IEEE Fellow, and an AAIA Fellow. He is a Member of Academia Europaea (Academician of the Academy of Europe).

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on February 11,2024 at 02:41:21 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

# Supplementary materials of "BGAE: Auto-encoding Multi-view Bipartite Graph Clustering"

Liang Li, Yuangang Pan, Jie Liu, Yue Liu, Xinwang Liu, *Senior Member, IEEE,* Kenli Li, *Senior Member, IEEE,* Ivor W. Tsang, *Fellow, IEEE,* and Keqin Li, *Fellow, IEEE*

## 1 Proof of Theorem 1

**Theorem 1.** *Supposing* $\mathbf{P} \in \mathbb{R}^{n\times k}, \mathbf{O} \in \mathbb{R}^{n\times m}, \mathbf{R} \in \mathbb{R}^{m\times k}$, *we have*

$$\max_{\mathbf{P},\mathbf{R}} \operatorname{Tr}\left(\mathbf{P}^\top \mathbf{O}\mathbf{R}\right), \ \textit{s.t.}\ \mathbf{P}^\top\mathbf{P} + \mathbf{R}^\top\mathbf{R} = \mathbf{I}_k. \tag{1}$$

*The optimal solutions are* $\mathbf{P} = \frac{\sqrt{2}}{2}\mathbf{U_o}$ *and* $\mathbf{R} = \frac{\sqrt{2}}{2}\mathbf{V_o}$, *where* $\mathbf{U_o}$ *and* $\mathbf{V_o}$ *are the rank-*$k$ *left and right singular vectors of* $\mathbf{O}$.

*Proof.* Eq. (1) can be reorganized by

$$\max_{\mathbf{P},\mathbf{R}} \frac{1}{2}\operatorname{Tr}\left(\begin{bmatrix}\mathbf{P}^\top & \mathbf{R}^\top\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{O}\\ \mathbf{O}^\top & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{P}\\ \mathbf{R}\end{bmatrix}\right), \ \text{s.t.}\ \begin{bmatrix}\mathbf{P}^\top & \mathbf{R}^\top\end{bmatrix}\begin{bmatrix}\mathbf{P}\\ \mathbf{R}\end{bmatrix} = \mathbf{I}_k. \tag{2}$$

The optimum can be computed by

$$\frac{1}{2}\begin{bmatrix}\mathbf{0} & \mathbf{O}\\ \mathbf{O}^\top & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{P}\\ \mathbf{R}\end{bmatrix} = \begin{bmatrix}\mathbf{P}\\ \mathbf{R}\end{bmatrix}\mathbf{\Delta} \tag{3}$$

where $\mathbf{\Delta}$ is a diagonal matrix consist of eigenvalues of $\frac{1}{2}\begin{bmatrix}\mathbf{0} & \mathbf{O}\\ \mathbf{O}^\top & \mathbf{0}\end{bmatrix}$.

By expanding Eq. (3), we have

$$\begin{cases} \frac{1}{2}\mathbf{O}^\top\mathbf{P} = \mathbf{R}\mathbf{\Delta} \\ \frac{1}{2}\mathbf{O}\mathbf{R} = \mathbf{P}\mathbf{\Delta} \end{cases} \tag{4}$$

Then, we have

$$\begin{cases} \left(\frac{\sqrt{2}}{2}\mathbf{O}\right)\left(\frac{\sqrt{2}}{2}\mathbf{O}\right)^\top\mathbf{P} = \mathbf{P}\left(\sqrt{2}\mathbf{\Delta}\right)^2 \\ \left(\frac{\sqrt{2}}{2}\mathbf{O}\right)^\top\left(\frac{\sqrt{2}}{2}\mathbf{O}\right)\mathbf{R} = \mathbf{R}\left(\sqrt{2}\mathbf{\Delta}\right)^2 \end{cases} \tag{5}$$

We derive that $\mathbf{P} = \frac{\sqrt{2}}{2}\mathbf{U_o}$ and $\mathbf{R} = \frac{\sqrt{2}}{2}\mathbf{V_o}$, where $\mathbf{U_o}$ and $\mathbf{V_o}$ denote the rank-$k$ left and right singular vectors of $\mathbf{O}$. This completes the proof. □

## 2 Experiments

### 2.1 Comparison of performance with two recent baselines FastMICE and UDBGL

TABLE 1 compares the clustering metrics of UDBGL, FastMICE, and our BGAE. Since FastMICE and UDBGL can directly output discrete clustering labels, they are "stable" methods without standard deviation. For simplicity, we report the clustering metrics. Although our BGAE is inferior to FastMICE on ORL_3views and ORL_4views, ours performs better on other datasets, especially on the large-scale ones. Overall, the results validate the effectiveness of our novel BGAE design.

### 2.2 View Weight Distribution

Fig. 1 shows view contribution $\gamma$ in the "decoding" module, which shows similar distribution as $\alpha$ in the "encoding" process, but with different magnitudes. Their difference accords with the idea of undercomplete AE that avoids "close to perfect" duplication.

TABLE 1: Comparing clustering metrics of FastMICE, UDBGL, and our BGAE. The best results are symbolized in bold.

| Datasets | UDBGL‡ | FastMICE‡ | Proposed‡ | UDBGL‡ | FastMICE‡ | Proposed‡ | UDBGL‡ | FastMICE‡ | Proposed‡ | UDBGL‡ | FastMICE‡ | Proposed‡ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ACC (%) | | | NMI (%) | | | Purity (%) | | | Fscore (%) | | |
| Yale | 0.5515 | 0.6848 | **0.7212** | 0.6389 | 0.6972 | **0.7311** | 0.5515 | 0.6848 | **0.7212** | 0.4192 | 0.5195 | **0.5812** |
| 3sources | 0.4142 | 0.5621 | **0.7751** | 0.1803 | 0.5090 | **0.6609** | 0.5207 | 0.7219 | **0.8225** | 0.3410 | 0.5143 | **0.6688** |
| MSRCV1 | 0.8048 | 0.8286 | **0.8571** | 0.8344 | 0.7661 | **0.8050** | 0.8048 | 0.8286 | **0.8571** | 0.7656 | 0.7266 | **0.7642** |
| Dermatology | 0.8408 | 0.7374 | **0.8911** | 0.8741 | 0.7281 | **0.7729** | 0.8464 | 0.7933 | **0.8911** | 0.8425 | 0.7269 | **0.8279** |
| ORL_3views | 0.5650 | 0.7375 | **0.7600** | 0.7623 | 0.8781 | **0.8796** | 0.6200 | 0.7700 | **0.7875** | 0.4334 | **0.6404** | 0.6335 |
| ORL_4views | 0.5850 | **0.6675** | 0.6500 | 0.7797 | **0.8258** | 0.8113 | 0.6275 | **0.6950** | **0.6950** | 0.4568 | **0.5600** | 0.4596 |
| SUN RGB-D | 0.2343 | 0.1854 | **0.2478** | 0.1631 | 0.2517 | **0.2689** | 0.2386 | 0.3720 | **0.3752** | 0.1427 | 0.1209 | **0.1625** |
| YouTubeFace20 | 0.6962 | 0.6916 | **0.7555** | 0.8049 | 0.8039 | **0.8117** | 0.7676 | 0.7613 | **0.7945** | 0.4940 | 0.6482 | **0.7040** |
| YouTubeFace50 | 0.7160 | 0.7158 | **0.7739** | 0.8152 | 0.8366 | **0.8643** | 0.7582 | 0.7601 | **0.8181** | 0.3037 | 0.6125 | **0.7152** |
| YouTubeFace100 | 0.6150 | 0.6602 | **0.7294** | 0.7967 | 0.8285 | **0.8437** | 0.6855 | 0.7304 | **0.7586** | 0.3399 | 0.5807 | **0.5958** |

‡ denotes stable algorithm.

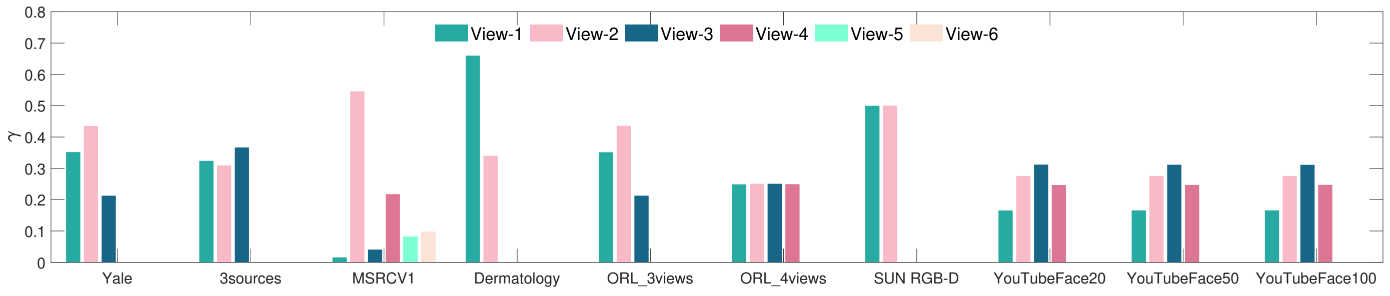


Fig. 1: View weights $\gamma$ in reconstruction learning process.

## 2.3 Variation of performance with respect to the number of views

For multi-view data, each individual view typically holds specific properties for a particular knowledge discovery task, and multiple views exhibit heterogeneous properties with potential connections [1], [2]. By fusing complementary and consistent information, MVC commonly achieves better embeddings or partitions than single-view clustering, which is beneficial for subsequent clustering tasks [3], [4]. In our BGAE, we transfer the "auto-encoding" design into graph machine learning to exploit the rich information across multiple views.

Fig. 2 experimentally investigates the variation of clustering metrics with respect to the number of views. We observe a general trend is that clustering metrics increase with the number of views, indicating that introducing multiple views helps to exploit more latent structures. Also, we observe that curves fluctuate when incorporating several views. A reasonable explanation is that potential noise or poor-quality data within the views degrade performance. Overall, the results indicate the importance of designing an effective fusion strategy in MVC.

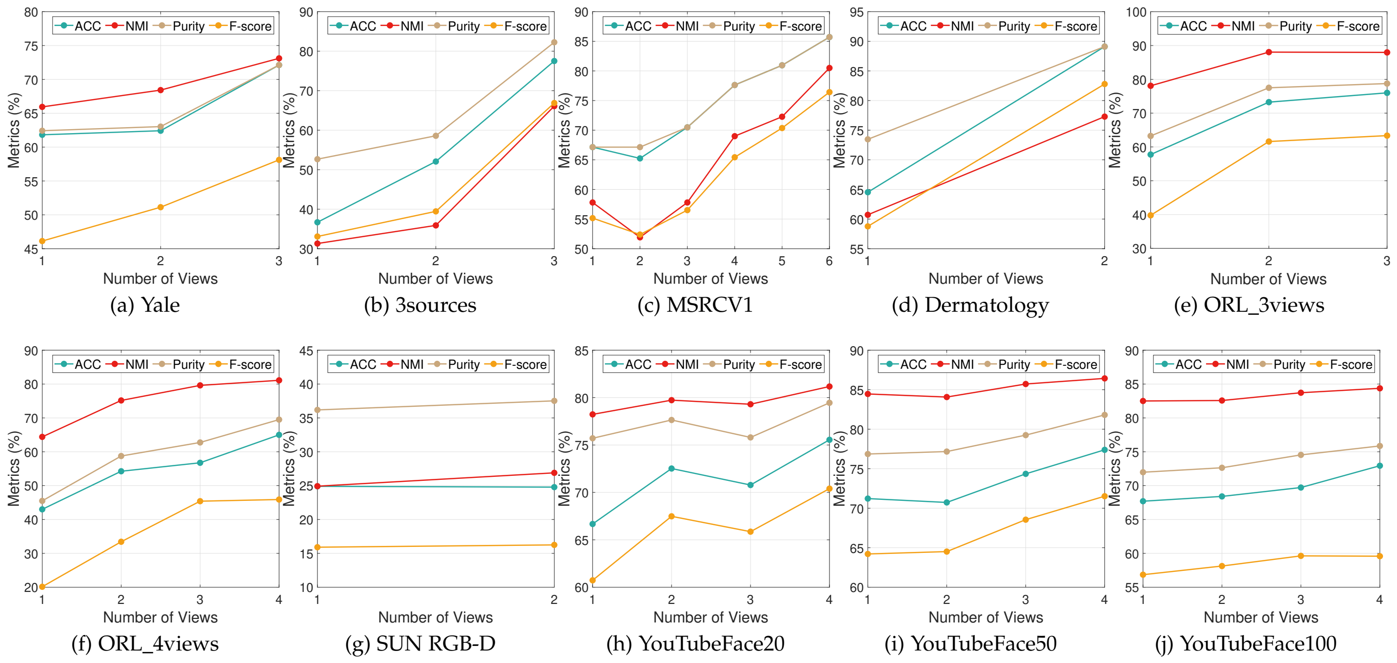
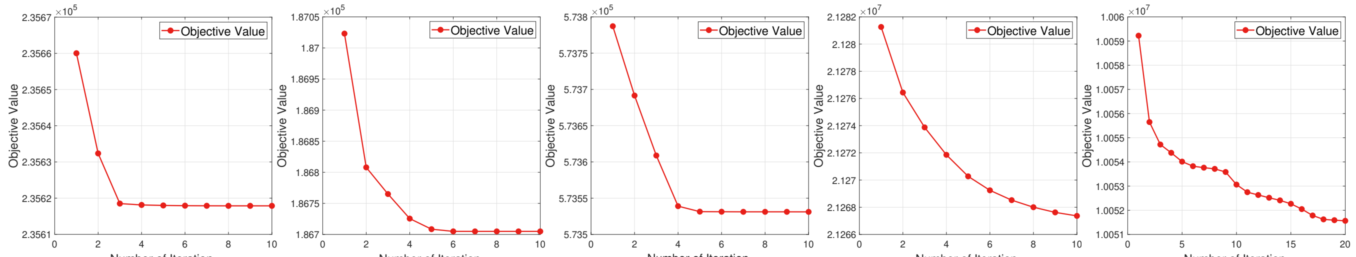


Fig. 2: Variation of performance with respect to the number of views.

## 2.4 Convergence

Fig. 3 empirically validates the convergence on Yale, 3sources, ORL_3views, SUN RGB-D, and YouTubeFace20 datasets.



(a) Yale (b) 3sources (c) ORL_3views (d) SUN RGB-D (e) YouTubeFace20

Fig. 3: Empirical validation of the convergence on Yale, 3sources, ORL_3views, SUN RGB-D, and YouTubeFace20.

## References

[1] S. Sun, L. Mao, Z. Dong, and L. Wu, *Multiview machine learning*. Springer, 2019.
[2] Y. Li, M. Yang, and Z. Zhang, "A survey of multi-view representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 1863–1883, 2019.
[3] G. Chao, S. Sun, and J. Bi, "A survey on multiview clustering," *Trans. Artif. Intell.*, vol. 2, no. 2, pp. 146–168, 2021.
[4] U. Fang, M. Li, J. Li, L. Gao, T. Jia, and Y. Zhang, "A comprehensive survey on multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12 350–12 368, 2023.