Architecture Alternative Deep Multi-View Clustering

Ke Ping, Shuxiao Li, Chuhan Wu, and Zhenwen Ren

Abstract—**Due to the strong non-linear fitting ability of deep neural networks, deep multi-view clustering has become a popular topic in the fields of signal processing and machine learning. Multi-view clustering based on deep auto-encoder can effectively capture the nonlinear features of high-dimensional data. However, the existing methods still have the following problems: 1) most autoencoder-based deep multi-view clustering methods ignore the differences between cross-view data and lose view-diversity features by using the same encoder structure for different views, and 2) many current deep multi-view techniques rely on single-lane neural networks for extracting feature data from each view. The current approach has limitations in its capability to accurately analyze comprehensive complementary information and multilevel features. To address these issues, we introduce a new clustering method Architecture Alternative Deep Multi-view Clustering (AADMC). Specifically, AADMC proposes a dynamic encoder network to adapt the encoder structure of each view according to the diversities of different views. Subsequently, AADMC proposes utilizing the Hilbert-Schmidt Independent Criterion (HSIC) to analyze the diversity of information between each encoder output. Moreover, AADMC integrates high-order and low-order information of the data into a shared connection matrix. To be more specific, the low-rank constraint is employed in order to effectively investigate and utilize the consensus information derived from all available views. The effectiveness and superiority of AADMC are demonstrated through experimental results conducted on various public datasets.**

Index Terms—**Alternative architecture, auto-encoders, clustering structure, low-rank constraint, multi-view clustering, self-expression.**

I. Introduction

With the rise of deep learning, deep multi-view clustering has become an active research area. Compared to traditional multi-view clustering methods [1], [2], [3], [4], [5], deep multi-view clustering benefits from the excellent extraction ability of deep neural networks for complex features, making it better than traditional multi-view clustering in dealing with high-dimensional and nonlinear data. To provide a structured overview of the existing deep multi-view methods, we categorize them into three groups based on their underlying theoretical foundations: deep embedded clustering-based (DEC), deep subspace clustering-based (DSC), and graph neural networks-based (GNN) approaches.

Manuscript received 20 September 2023; revised 17 October 2023; accepted 17 October 2023. Date of publication 25 October 2023; date of current version 2 November 2023. This work was supported in part by the Project of Guangxi Key Laboratory of Machine Vision and Intelligent Control under Grant 2022B07, in part by the Miaozi Project of Sichuan Province under Grant MZGC20230072, and in part by the Natural Science Foundation of Sichuan Province under Grant 2023NSFSC1373. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yipeng Liu. *(Corresponding author: Zhenwen Ren.)*

Ke Ping, Shuxiao Li, and Chuhan Wu are with the School of National Defence Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China (e-mail: pingke@mails.swust.edu.cn; shuxiao_li@outlook.com; wu_chuhan_hxqyc@163.com).

Zhenwen Ren is with the School of National Defence Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China, and also with the Guangxi Key Laboratory of Machine Vision and Intelligent Control, Wuzhou University, Wuzhou 543002, China (e-mail: rzw@njust.edu.cn).

Digital Object Identifier 10.1109/LSP.2023.3327652

(a) (b)

Fig. 1. FRGC dataset visualization. (a) Sample image appears in different views; (b) corresponding edge maps.

DEC methods [6], [7], [8], [9] primarily focus on achieving effective clustering results. They employ auto-encoders to learn low-dimensional representations in a latent space. Optimization of these representations involves minimizing the Kullback-Leibler divergence between the feature representations of a student's t-distribution and a desired supplementary distribution.

Deep multi-view subspace clustering methods aim to explore the consistency and complementarity of multiple views and integrate them meaningfully. For instance, the deep multi-view sparse subspace clustering method (DMVSSC) [10] combines a self-expressive module based on canonical correlation analysis (CCA) [11], [12] and convolutional autoencoders (CAEs). Recent studies have also demonstrated improved results using deep subspace-based approaches [13], [14], [15], [16], [17], [18], [19], [20].

Multi-view graphs [21], [22], [23], [24], [25], [26], [27], [28] have proven effective in representing the complexity of real-world graph data. Graph convolutional networks (GCNs) exhibit great potential for processing graph-structured data. Multi-GCN, a graph-based convolutional network, has been developed with a specific focus on analyzing multi-view data [29].

These methods strive to uncover the shared information across different views, they often neglect to account for the diverse information between views, which is equally essential. The challenges faced by current deep multi-view clustering methods can be generalized into two categories. The first challenge is the insufficient ability to capture comprehensive complementary information and multilevel features. The second challenge arises from the uniform encoder structure for each view, which may not effectively extract view-specific information. Let's consider the FRGC dataset as an example, illustrated in Fig. 1. It's evident that various views within this dataset exhibit distinct texture features. It goes against scientific intuition to employ a uniform structural encoder for extracting the underlying features of these differing structures.

To overcome these issues, we propose a novel deep multi-view clustering method, namely AADMC. The network's encoder structure is customized to capture view-specific details by assessing data characteristics, such as the size of the view's input data. This assessment helps select the most suitable encoder structure from a set of pre-defined options, while simultaneously integrating features from multiple layers. Hence, to gather the shared information among views, a low-rank constraint can be enforced on the shared connection matrix. After that, to alleviate the computational burden of the network, we also employ a latent low-rank constraint on the self-expressive layer, thereby diminishing the computational complexity. In comparison to prior methods, AADMC introduces the following contributions:

1) To effectively utilize the diverse information from multiple views, an alternative encoder network is proposed that can alternatively modify the encoder structure of each view according to the diversities of different views.
2) We propose a centralization constraint that considers the consistency across the divergent views to explore the consistent information among the views. In this way, AADMC can effectively capture the underlying relationships among different views, thereby enhancing the model's accuracy and robustness.
3) In order to reduce the computational load during gradient backpropagation, we impose a latent low-rank constraint on the self-expression layer, thereby greatly reducing the number of parameters and the computational burden.
4) Numerous experimental results on four multi-view datasets demonstrate the superiority of AADMC compared with several state-of-the-art methods.

II. Methodology

Given multi-view data $\{X^{(v)} \in \mathbb{R}^{d_v \times n}\}_{v=1}^{V}$, with n samples and V views, indicating the latent subspace and the dimensionality of the v-th view by $Z^{(v)}$ and d_v. The comprehensive objective function comprises the following components: reconstruction error loss L_r, hierarchical self-expression loss L_s, diversity loss L_d, and the universality regularizer L_u, as outlined below:

$$\mathcal{L} = \mathcal{L}_{\mathrm{r}} + \lambda_1 \mathcal{L}_{\mathrm{s}} + \lambda_2 \mathcal{L}_d + \mathcal{L}_u \tag{1}$$

We will expand (1) in detail in the rest of this section.

A. Network Architecture

AADMC consists of five components: view-specific auto-encoder, universality regularizer, hierarchical self-expression module, reconstruction loss, and a latent network constraint.

The architecture of AADMC is illustrated in Fig. 2.

1) Reconstruction Loss: An end-to-end training is utilized in the model to train the network, resulting in a reconstruction loss between the original view $X^{(v)}$ and the corresponding reconstructed view $\overline{X}^{(v)}$, which serves as a training objective. The architecture of the alternative multi-lane decoder utilizes symmetric hyperparameter settings that are identical to the ones employed in the encoder, with the exception that the convolutional layer is substituted by a deconvolution layer. The reconstruction loss during the end-to-end training of AADMC can be expressed as follows

$$L_{\mathrm{r}} = \sum_{v=1}^{V} \left\| X^{(v)} - \overline{X}^{(v)} \right\|_F^2 . \tag{2}$$

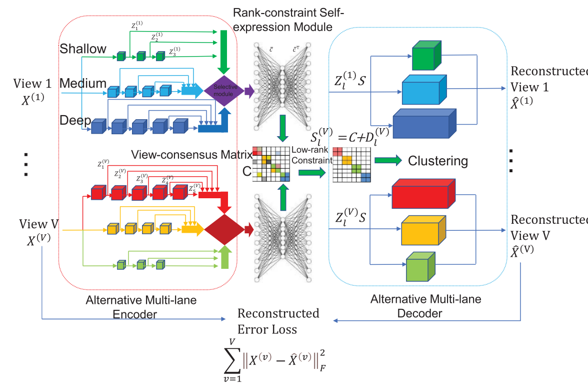


Fig. 2. AADMC architecture consists of a structured multi-lane auto-encoder, a selection module adapting the encoder structure based on view input size, a symmetric decoder structure mirroring the encoder, and a hierarchical self-expression module where each layer's encoder output can be directly directed to the self-expression module, along with a module for low-rank subspace learning to enhance the consistency of connection matrix C for clustering.

2) Hierarchical Self-Expression Module: After implementing the structured multi-lane encoder, equipped with L convolutional layers, for the v-th view, the obtained result yields L latent subspaces, $\{Z_l^{(v)}, l = 1, 2 \ldots L\}$, where $Z_l^{(v)}$ represents the acquired embedded space from the l-th layer ConvNet in the v-th view. Subsequently, the self-expression matrix $S_l^{(v)}$ is learned using the multilayer self-expression module as $Z_l^{(v)} = Z_l^{(v)} S_l^{(v)}$.

Similar to [30], to balance both consistency and diversity, we partition the self-representation matrix $S_l^{(v)}$ into two components: a shared matrix C, which holds common information across all views, and a view-specific matrix $D_l^{(v)}$ that is associated with each convolutional layer in each view. We apply constraints to preserve these characteristics during learning. Additionally, we introduce an implicit low-rank constraint on C, as discussed later. To compute the objective function for self-expression, the following definition is used:

$$L_s = \sum_{l=1}^{L} \sum_{v=1}^{V} \left\| Z_l^{(v)} - Z_l^{(v)} S_l^{(v)} \right\|_F^2$$
$$\text{s.t. } S_l^{(v)} = C + D_l^{(v)}, \quad \operatorname{diag}(C) = 0 \tag{3}$$

where C and $D_l^{(v)}$ denote the view consensus matrix and the specific self-representation matrix, which encodes unique information for the L layer features of the V view.

3) View-Specific Auto-Encoders: We design the view-specific auto-encoders to show the diversity of each view. Specifically, we utilize the v-th alternative multi-lane encoder to select the appropriate number of convolutional layers and kernel sizes, thereby effectively extracting view-private features. HSIC is commonly applied in multi-view subspace clustering due to its exceptional capability to capture high-order and nonlinear relationships.

Given N independent observations derived from p_{xy}, denoted as $\mathcal{Z} := (\mathbf{x}1, \mathbf{y}1), \ldots, (\mathbf{x}N, \mathbf{y}N) \subseteq \mathfrak{X} \times \mathfrak{Y}$, we can define an estimator for HSIC, denoted as $\mathrm{HSIC}(\mathfrak{Z}, \mathfrak{F}, \mathfrak{G})$, as follows:

$$\mathrm{HSIC}(\mathfrak{Z}, \mathfrak{F}, \mathfrak{G}) = (N-1)^{-2} tr(\mathbf{G}_1 \mathbf{H} \mathbf{G}_2 \mathbf{H}), \tag{4}$$

Here, $tr(\cdot)$ represents the trace of a square matrix. $\mathbf{G}1$ and $\mathbf{G}2$ are the Gram matrices, where $g_{1,ij} = g_1(\mathbf{x}i, \mathbf{x}j)$ and

$g_{2,ij} = g_2(\mathbf{y}i, \mathbf{y}j)$. Additionally, $h_{ij} = \delta_{ij} - 1/N$ centers the Gram matrix, ensuring it has a zero mean in the feature space. For more details about HSIC, please refer to [31]. Then, AADMC adopts HSIC to ensure that each view contains unique information and incorporates features from all dimensions, relying on smaller and more diffuse weight vectors to promote diversity. Hereto, the diversity function can be written as

$$L_d = \sum_l \sum_{ij} HSIC\left(D_l^{(i)}, D_l^{(j)}\right). \tag{5}$$

4) Universality Regularizer: Because the fundamental structure of each view's self-expressive matrix remains constant, it's crucial to create a common relational graph to assess consistency. In this regard, we propose a centralization regularization that is defined by the (6)

$$L_u = \sum_{l=1}^{L} \sum_{v=1}^{V} \left\| C - D_l^{(v)} \right\|_F^2. \tag{6}$$

Here, C represents the common self-representation matrix shared by all views, while $D_l^{(v)}$ denotes the specific matrix corresponding to each layer of an individual view's encoder.

5) Optimization: Inspired by the advantages of low-rank representation (LRR) in subspace clustering, our aim is to apply deep learning frameworks for profound LRR of data. We intend to impose a rank restriction on the common self-representation matrix C for implicit self-expressive LRR. The common approach to promoting low rankness in matrix C is by using nuclear norm regularization in the loss function, but this can be computationally expensive for deep models due to gradient computation issues during back-propagation. Our proposal involves replacing the common self-representation matrix $\mathbf{C} \in \mathbb{R}^{n\times n}$ with a fully-connected linear layer $\overline{c} \in \mathbb{R}^{n\times m}$ and its transpose, where $m(m \ll n)$ is a hyperparameter limiting the maximum rank of matrix C. Creating a symmetric matrix ($C \triangleq \overline{c}\overline{c}^\top$) reduces redundancy and enhances model symmetry. To realize this matrix, we learn a weight matrix $\overline{c} \in \mathbb{R}^{n\times m}$. The common self-representation matrix C ensures $\mathrm{rank}(\overline{c}) \le \min(m, n)$, enforcing a rank constraint on C by varying m. Consequently, our architecture intrinsically enforces a rank constraint on C, which is shared among (3) and (6) and can be replaced with $\overline{c}\overline{c}^\top$:

$$L_u = \sum_{l=1}^{L} \sum_{v=1}^{V} \left\| \overline{c}\overline{c}^\top - D_l^{(v)} \right\|_F^2 \tag{7}$$

$$L_s = \sum_{l=1}^{L} \sum_{v=1}^{V} \left\| Z_l^{(v)} - Z_l^{(v)} S_l^{(v)} \right\|_F^2$$

$$\text{s.t. } S_l^{(v)} = \overline{c}\overline{c}^\top + D_l^{(v)}, \quad \mathrm{diag}(\overline{c}\overline{c}^\top) = 0. \tag{8}$$

The evaluation of the nuclear norm for an m-dimensional matrix is computationally less intensive than that for an n-dimensional matrix. This is primarily due to the considerable difference in size between m and n. Thus, computing the nuclear norm of an m-dimensional matrix serves as an effective way to mitigate computation complexity.

B. Implementation

Just as mentioned before, AADMC reconstructs the data as vectors and adapts the number of convolution layers and kernel size according to the specific characteristics of each input view. To achieve this, utilizing different combinations of convolution kernel sizes and numbers of layers depending on the size of the input data. For views with a size larger than 64 pixels, we apply an encoder structure consisting of three convolutional layers: the first layer utilizes a kernel size of 5, stride of 2, and padding of 1, the second layer has a kernel size of 3, stride of 1, and padding of 1, and the third layer has a kernel size of 5, stride of 2, and padding of 1. Conversely, for smaller view data, we implement an encoder structure consisting of a first convolutional layer with a kernel size of 4, stride of 2, and padding of 1, a second convolutional layer with a kernel size of 3, stride of 1, and padding of 1, and a third and final convolutional layer with a kernel size of 4, stride of 2, and padding of 1. The optimization process employs the Adam Optimizer [32], while the implementation of AADMC utilizes PyTorch [33] as the underlying framework. Experiments are conducted on an Ubuntu 22.04 system with a NVIDIA 3090 GPU and 32 GB RAM.

TABLE I
STATISTICS DATASETS

Datasets	View number	Class number	Sample number
COIL-20	2	20	1400
YTF	3	41	10000
FRGC	3	20	2462
Fashion-MNIST	2	10	70000

TABLE II
AVERAGE CLUSTERING RESULTS ON THE FOUR DATASETS

Datasets	COIL-20		YTF		FRGC		Fashion-MNIST	
Metrics	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
K-means	0.575	0.732	0.560	0.752	0.236	0.271	0.513	0.499
DCCA	0.558	0.649	0.452	0.604	0.229	0.248	0.527	0.538
RMKMC	0.610	0.749	0.572	0.746	0.235	0.259	0.533	0.529
DEC	0.680	0.803	0.371	0.446	0.378	0.505	0.518	0.546
DGCCA	0.540	0.624	0.473	0.614	0.238	0.245	0.563	0.570
DMSC	0.741	0.868	0.628	0.802	<u>0.728</u>	**0.810**	0.596	<u>0.651</u>
GMC	0.742	0.825	0.554	0.742	0.561	0.642	0.567	0.629
LALMVC	0.743	0.838	0.556	0.715	0.526	0.657	0.522	0.599
CMSC-DCCA	<u>0.826</u>	<u>0.915</u>	<u>0.662</u>	<u>0.827</u>	0.708	0.786	**0.629**	**0.683**
FPMVSC	0.497	0.731	0.241	0.243	0.531	0.554	0.505	0.528
AADMC	**0.839**	**0.918**	**0.687**	**0.843**	**0.753**	<u>0.808</u>	<u>0.614</u>	0.643

III. EXPERIMENTS

The proposed AADMC's effectiveness is verified on four real-world multi-view datasets, as shown in Table I. The comparison methods include classical and state-of-the-art methods, *i.e.*, four traditional methods (RMKMC [34], FPMVSC [35], LALMVC [36], GMC [37], and K-means clustering [38]) and four deep methods (DCCA [11], DEC [39], CMSC-DCCA [40], DMSC [41], and DGCCA [42]).

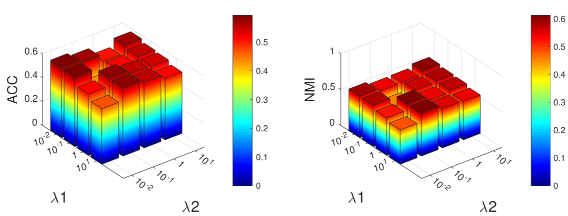
A. Result Analysis

Table II presents the outcomes of the experiments conducted on the Fashion-MNIST, COIL-20, FRGC, and YTF databases, displaying the ACC and NMI metrics. The most favorable outcome is highlighted in bold, and the next most favorable is underlined. As shown, deep multi-view methods mostly outperform traditional multi-view methods, which demonstrates the necessity of effective feature extraction capabilities for clustering. Our proposed method outperforms others on the COIL-20 and YTF datasets and remains competitive on the FRGC and Fashion-MNIST datasets. Specifically, on the YTF dataset, it achieves ACC and NMI scores that are 2.5% and 1.6% higher, respectively, than the second-best method. Furthermore, our

TABLE III
ABLATION STUDY OF SETTING DIFFERENT ENCODER STRUCTURES FOR DIFFERENT VIEWS

	1st layer	2nd layer	3rd layer	4th layer	ACC	NMI
Convolutional	✓○		✓○			
Linear		✓○			0.4545	0.5641
None				✓○		
Convolutional	✓○	✓	✓○			
Linear		○			0.5276	0.6552
None				✓○		
Convolutional	✓○	○	✓○			
Linear		✓			0.4041	0.4796
None				✓○		
Convolutional	✓○	✓○				
Linear					0.5678	0.7490
None			✓○	✓○		
Convolutional	✓○	✓○	✓			
Linear					0.6084	0.7624
None			○	✓○		
Convolutional	✓○	✓○	○			
Linear					0.7273	0.7674
None			✓	✓○		
Convolutional	✓○	✓○	✓○	✓		
Linear					0.6418	0.7581
None				○		
Convolutional	✓○	✓○	✓○	○		
Linear					0.6596	0.7785
None				✓		

[1] "✓" represents the settings of the first view.
[2] "○" represents the settings of the second view.



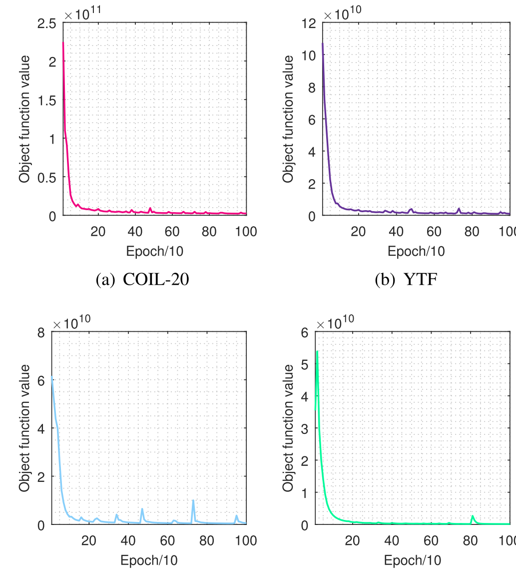
(a) ACC (b) NMI

Fig. 3. Influence of parameters on the clustering performance of MNIST. (a) ACC. (b) NMI.

method exhibits strong portability, implying that it can enhance the performance of other methods as well.

B. Parameters Analysis

In this study of AADMC, effective settings for two parameters, λ1 and λ2, are critical to the results. Thus, we evaluated the impact of λ1 and λ2 on the results. To investigate their impact, we kept the diversity regularizer parameter λ1 constant, while varying the values of the parameter λ2 for the universality regularizer. We performed a series of experiments with λ1 varied between 10, 1, 0.1, and 0.01. Fig. 3 illustrates the results, demonstrating that the highest ACC (accuracy results) are achieved for the MNIST dataset when using $\lambda 1 = 1$ and $\lambda_2 = 0.1$ which validates the stability of our model as changes to these values produce only minor variations in the clustering performance. To tailor our model more specifically to each dataset, we assigned different values of λ1 and λ2 for the Fashion-MNIST, COIL-20, FRGC, and YTF datasets. For instance, the values 1 and 0.01 were assigned to λ1 and λ2 for the Fashion-MNIST and COIL-20 datasets. For the FRGC dataset, we assigned 1 and 0.01 to λ1 and λ2, respectively. For the YTF dataset, λ1 was set to 0.1, whereas λ2 was set to 0.001. By customizing the values of λ1 and λ2, we aimed to optimize clustering performance for each dataset.



(a) COIL-20 (b) YTF (c) FRGC (d) Fashion-MNIST

Fig. 4. Convergence curves of our AADMC on four datasets.

C. Convergence Analysis

The optimization objectives of L_d, L_u, L_s, and L_r, represented by (2) to (6), exhibit convexity. Fig. 4 illustrates the progression of the objective function values as the number of epochs increases across the four databases. Our model demonstrates rapid convergence when the number of epochs falls below 500, ensuring the efficiency of the method. These results are corroborated by Fig. 4, where a decrease in loss values corresponds to an increase in clustering effectiveness. Therefore, the AADMC algorithm displays favorable convergence properties.

D. Ablation Study

We establish a separate encoder architecture for each view in which C and L denote convolutional and linear layers, respectively. For instance, we utilize a convolutional network with deeper layers for the second view in the FRGC dataset to extract more appropriate features for clustering. The parameter settings are shown in Table III.

IV. CONCLUSION

To address the problem of inadequate feature extraction for view-specific features, we propose an Architecture Alternative Deep Multi-view Clustering (AADMC) method. Specifically, our proposed AADMC uses an alternative view attribute encoder to obtain multi-level features of each view. Through diversity constraints, each view endows its specific information. Then, a low-rank constraint is applied to the shared matrix of views to mine the common information among the views and reduce noise interference. In addition, we also designed a network constraint to reduce the computational complexity of the network. A plentiful of experimental results on multiple public datasets show the superiority of our AADMC compared with ten state-of-the-art methods.

REFERENCES

[1] Y. Sun, D. Peng, H. Huang, and Z. Ren, "Feature and semantic views consensus hashing for image set classification," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 2097–2105.

[2] X. Wu, Z. Ren, and F. R. Yu, "Parameter-free shifted Laplacian reconstruction for multiple kernel clustering," *IEEE/CAA J. Automatica Sinica*, early access, Aug. 01, 2023, doi: 10.1109/JAS.2023.123600.

[3] Y. Lu, Y. Liu, Z. Long, Z. Chen, and C. Zhu, "O-minus decomposition for multi-view tensor subspace clustering," *IEEE Trans. Artif. Intell.*, vol. 1, no. 5, pp. 1–14, 2023, doi: 10.1109/TAI.2023.3293479.

[4] Z. Long, C. Zhu, P. Comon, and Y. Liu, "Feature space recovery for incomplete multi-view clustering," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2023, pp. 1–5.

[5] Z. Long, C. Zhu, J. Chen, Z. Li, Y. Ren, and Y. Liu, "Multi-view MERA subspace clustering," *IEEE Trans. Multimedia*, to be published, doi: 10.1109/TMM.2023.3307239.

[6] M. Kheirandishfard, F. Zohrizadeh, and F. Kamangar, "Deep low-rank subspace clustering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 864–865.

[7] H. Lu, C. Chen, H. Wei, Z. Ma, K. Jiang, and Y. Wang, "Improved deep convolutional embedded clustering with re-selectable sample training," *Pattern Recognit.*, vol. 127, 2022, Art. no. 108611.

[8] J. Cai, J. Fan, W. Guo, S. Wang, Y. Zhang, and Z. Zhang, "Efficient deep embedded subspace clustering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1–10.

[9] Y. Ding et al., "Unsupervised self-correlated learning smoothy enhanced locality preserving graph convolution embedding clustering for hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5536716.

[10] X. Tang, X. Tang, W. Wang, L. Fang, and X. Wei, "Deep multi-view sparse subspace clustering," in *Proc. VII Int. Conf. Netw. Commun. Comput.*, 2018, pp. 115–119.

[11] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1247–1255.

[12] W. Wang, R. Arora, K. Livescu, and J. Bilmes, "On deep multi-view representation learning," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1083–1092.

[13] Y. Qin, H. Wu, X. Zhang, and G. Feng, "Semi-supervised structured subspace learning for multi-view clustering," *IEEE Trans. Image Process.*, vol. 31, pp. 1–14, 2022.

[14] Z. Zhou et al., "Sequential order-aware coding-based robust subspace clustering for human action recognition in untrimmed videos," *IEEE Trans. Image Process.*, vol. 32, pp. 13–28, 2023.

[15] M. Brbić and I. Kopriva, "Multi-view low-rank sparse subspace clustering," *Pattern Recognit.*, vol. 73, pp. 247–258, 2018.

[16] J. Wang, B. Wu, Z. Ren, and Y. Zhou, "Multi-scale deep subspace clustering with discriminative learning," *IEEE Access*, vol. 10, pp. 91283–91293, 2022.

[17] X. Si, Q. Yin, X. Zhao, and L. Yao, "Consistent and diverse multi-view subspace clustering with structure constraint," *Pattern Recognit.*, vol. 121, 2022, Art. no. 108196.

[18] Q. Wang, Z. Tao, Q. Gao, and L. Jiao, "Multi-view subspace clustering via structured multi-pathway network," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 28, 2022, doi: 10.1109/TNNLS.2022.3213374.

[19] Y. Xu, S. Chen, J. Li, C. Xu, and J. Yang, "Fast subspace clustering by learning projective block diagonal representation," *Pattern Recognit.*, vol. 135, 2023, Art. no. 109152.

[20] C. Chen, H. Lu, H. Wei, and X. Geng, "Deep subspace image clustering network with self-expression and self-supervision," *Appl. Intell.*, vol. 53, no. 4, pp. 4859–4873, 2023.

[21] J. Cheng, Q. Wang, Z. Tao, D. Xie, and Q. Gao, "Multi-view attribute graph convolution networks for clustering," in *Proc. 29th Int. Conf. Int. Joint Conf. Artif. Intell.*, 2021, pp. 2973–2979.

[22] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 1767–1776.

[23] S. Wang et al., "Highly-efficient incomplete large-scale multi-view clustering with consensus bipartite graph," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 9776–9785.

[24] W. Xia, S. Wang, M. Yang, Q. Gao, J. Han, and X. Gao, "Multi-view graph embedding clustering network: Joint self-supervision and block diagonal representation," *Neural Netw.*, vol. 145, pp. 1–9, 2022.

[25] W. Xia, Q. Gao, Q. Wang, X. Gao, C. Ding, and D. Tao, "Tensorized bipartite graph learning for multi-view clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 5187–5202, Apr. 2023.

[26] X. Ma, X. Yan, J. Liu, and G. Zhong, "Simultaneous multi-graph learning and clustering for multiview data," *Inf. Sci.*, vol. 593, pp. 472–487, 2022.

[27] C. Zhang, H. Li, W. Lv, Z. Huang, Y. Gao, and C. Chen, "Enhanced tensor low-rank and sparse representation recovery for incomplete multi-view clustering," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, pp. 11174–11182, 2023.

[28] Y. Huang et al., "C2IMUFS: Complementary and consensus learning-based incomplete multi-view unsupervised feature selection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10681–10694, Oct. 2023.

[29] M. R. Khan and J. E. Blumenstock, "Multi-GCN: Graph convolutional networks for multi-view networks, with applications to global poverty," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 606–613.

[30] S. Luo, C. Zhang, W. Zhang, and X. Cao, "Consistent and specific multi-view subspace clustering," in *Proc. 30nd AAAI Conf. Artif. Intell. 30th Innov. Appl. Artif. Intell. Conf. 8th AAAI Symp. Educ. Adv. Artif. Intell.*, 2018, pp. 3730–3737.

[31] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with Hilbert–Schmidt norms," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2005, pp. 63–77.

[32] P. D. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[33] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[34] X. Cai, F. Nie, and H. Huang, "Multi-view k-means clustering on Big Data," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 2598–2604.

[35] S. Wang et al., "Fast parameter-free multi-view subspace clustering with consensus anchor guidance," *IEEE Trans. Image Process.*, vol. 31, pp. 556–568, 2022.

[36] D. Xie, X. Zhang, Q. Gao, J. Han, S. Xiao, and X. Gao, "Multiview clustering by joint latent representation and similarity learning," *IEEE Trans. Cybern.*, vol. 50, no. 11, pp. 4848–4854, Nov. 2020.

[37] 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, Jun. 2020.

[38] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *J. Roy. Stat. Soc. Ser. C (Appl. Statist.)*, vol. 28, no. 1, pp. 100–108, 1979.

[39] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 478–487.

[40] Q. Gao, H. Lian, Q. Wang, and G. Sun, "Cross-modal subspace clustering via deep canonical correlation analysis," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3938–3945.

[41] M. Abavisani and V. M. Patel, "Deep multimodal subspace clustering networks," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 6, pp. 1601–1614, Dec. 2018.

[42] A. Benton, H. Khayrallah, B. Gujral, D. A. Reisinger, S. Zhang, and R. Arora, "Deep generalized canonical correlation analysis," in *Proc. 4th Workshop Representation Learn.*, 2019, pp. 1–6.