Ensemble Clustering With Attentional Representation

Zhezheng Hao, Zhoumin Lu, Guoxu Li, Feiping Nie, *Senior Member, IEEE*, Rong Wang, *Member, IEEE*, and Xuelong Li, *Fellow, IEEE*

***Abstract*—Ensemble clustering has emerged as a powerful framework for analyzing heterogeneous and complex data. Despite the abundance of existing schemes, co-association matrix-based methods remain the mainstream approach. However, focusing solely on pairwise correlations falls short of fully capturing the intricate cluster relationships. Moreover, despite its potential, ensemble clustering has yet to effectively leverage the powerful representation capabilities of neural networks. To address these limitations, we propose a deep ensemble clustering method called Ensemble Clustering with Attentional Representation (ECAR). Our method considers the results of base partitions as groups with related information to explore higher-order fusion information. ECAR captures the importance of each sample's association with its related group by employing an attentional network, and encodes this information into a low-dimensional representation. The attentional network is trained by jointly optimizing the clustering loss from soft assignments learned from the embeddings and the reconstruction loss from the weighted graph generated from ensemble clustering. During training, the weights of base partitions are adaptively refined to promote diversity and consistency while reducing the impact of low-quality and redundant base partitions. Extensive experimental results on real-world datasets demonstrate the substantial improvement of our method over existing baseline ensemble clustering methods and deep clustering methods.**

***Index Terms*—Deep clustering, ensemble clustering, graph autoencoder, self-attentional, unsupervised representation learning.**

I. Introduction

THE proliferation of heterogeneous data from diverse sources has posed unprecedented challenges to the analysis and modeling of Big Data. Among the many data analysis and pattern recognition methods, clustering is a long-standing and intricate research component [1]. Different clustering techniques tend to extract data based on distinct structures, such as cluster size and shape [2], [3]. While a well-designed clustering model may yield satisfactory results on one dataset, it may not generalize to others. Moreover, some clustering algorithms exhibit high sensitivity to initialization and hyperparameters, leading to unstable clustering outcomes [4], [5]. In the absence of prior knowledge about the structural properties of a dataset, identifying an optimal or even reasonable clustering model becomes a formidable task.

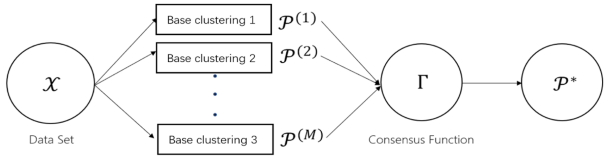

Fig. 1. Traditional paradigm of ensemble clustering.

To address the above issues, researchers have proposed ensemble clustering as a promising scheme [6], [7], [8], [9]. Ensemble clustering involves integrating different clustering models to achieve an improved final result. By combining the partition results obtained from different types of base partition, ensemble clustering can uncover the structural information of the dataset to a greater extent, capturing the structural elements of the dataset from multiple perspectives. Thus, ensemble clustering is expected to overcome the limitations of unique algorithms that rely on specific assumptions about the data structure. Moreover, ensemble clustering is also considered a viable approach for enhancing the robustness and stability of clustering tasks. Due to its sound clustering results, ensemble clustering methods have been widely used in computer vision [10], [11], [12], text mining [13], [14], multi-source fusion [15], [16], [17] and other data-mining areas [18], [19], [20], [21], [22].

Over the years, ensemble clustering has emerged as a burgeoning area of research, garnering increasing attention [23]. In the existing methods, after the base clusters are generated through various means or parameters, the typical process is to fuse their results using a consensus function to obtain the final clustering results, as illustrated in Fig. 1. It has been recognized as an important alternative to the traditional cluster analysis [24] Specifically, much of the research on ensemble clustering has centered on the advancement of consensus functions.

Despite enhancing stability over individual partitions [25], the existing ensemble clustering methods often fail to substantially enhance clustering performance. In an effort to promote the ensemble clustering performance, a novel deep ensemble clustering method is proposed in this article, called **E**nsemble **C**lustering with **A**ttentional **R**epresentation (**ECAR**). Compared

Manuscript received 15 April 2023; revised 25 June 2023; accepted 2 July 2023. Date of publication 5 July 2023; date of current version 11 January 2024. This work was supported by the National Science Foundation of China under Grants 62276212 and 62176212. Recommended for acceptance by Y. Tong. *(Corresponding author: Feiping Nie.)*

Zhezheng Hao and Guoxu Li are with the School of Artificial Intelligence, Optics and Electronics (iOPEN), School of Cybersecurity, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, P.R. China (e-mail: hzz@mail.nwpu.edu.cn; lgx2683109120@gmail.com).

Zhoumin Lu and Feiping Nie are with the School of Computer Science, School of Artificial Intelligence, Optics and Electronics (iOPEN), Key Laboratory of Intelligent Interaction and Applications, Ministry of Industry and Information Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, P.R. China (e-mail: walker.zhoumin.lu@gmail.com; feipingnie@gmail.com).

Rong Wang and Xuelong Li are with the School of Artificial Intelligence, Optics and Electronics (iOPEN), Key Laboratory of Intelligent Interaction and Applications, Ministry of Industry and Information Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, P.R. China (e-mail: wangrong07@tsinghua.org.cn; li@nwpu.edu.cn).

Digital Object Identifier 10.1109/TKDE.2023.3292573

to existing ensemble clustering methods, ECAR boasts the following four distinctive characteristics:

- The commonly used co-association matrix merely captures the pairwise associations between data points, which may fail to uncover valuable clustering information. In contrast, ECAR consider each class derived from the base partition as a higher-order group that reveals the underlying relationships among samples.
- Most state-of-the-art ensemble clustering methods employ the base partition results as new representations of the samples, which heavily relies on the clustering outcomes and neglects the original features. ECAR leverages the base partition results as higher-order connections to guide the clustering process on original features.
- By harnessing the representation capabilities of neural networks, particularly self-supervised mechanisms, ECAR learns a low-dimensional embedding that is customized for clustering via an attentional encoder.
- To mitigate the impact of low-quality and redundant base partitions, ECAR adaptively assigns weight to each base partition based on their diversity and consistency, which effectively enhances the discriminative power of the clustering.

In general, the contributions of this article can be summarized as follows:

- Firstly, we introduce deep self-supervised methods into the ensemble clustering paradigm, thereby opening up the potential to introduce other deep clustering approaches. Specifically, we propose a novel autoencoder to achieve clustering of features by the guidance of base partition results, which is the first to our knowledge.
- Our proposed ECAR method distinguishes itself from traditional ensemble clustering methods by incorporating the aforementioned characteristics, which provides the possibility of achieving reliable clustering results on more challenging data.
- Extensive experiments are conducted both on synthetic and realistic datasets, and the results demonstrate considerable improvement compared to existing ensemble clustering algorithms. Besides the visualization of training and parameter study disclose the feasibility and effectiveness of our proposed method.

The remainder of this article is organized as follows. In Section II, we revisit the previous ensemble clustering methods and deep clustering methods. The proposed method for deep ensemble clustering, along with its corresponding optimization and discussions, are presented in Section III. Extensive experiments conducted on synthetic and real-world datasets are shown in Section IV. Section V concludes the entire article.

II. Related Work

In this section, we revisit related works on ensemble clustering and deep clustering separately, with a view to establishing their respective strengths and weaknesses.

A. Ensemble Clustering Methods

Ensemble clustering comprises two fundamental research aspects: constructing base partitions and designing a consensus function to generate the final partitioning. The primary strategies of the former include homogeneous ensembles [2], [26], [27], heterogeneous ensembles [28], [29], [30], variable numbers of clusters [2], [25], [31], and random sampling [32], [33], [34]. After obtaining base partitions, a series of effective ensemble clustering methods have been proposed recently, which fuse the base partition results from different perspectives. Current ensemble clustering methods can be broadly categorized into three classes based on the fusion mode: [8], [35], co-association matrix-based methods [2], [36] and graph-based methods [6], [37].

The voting-based methods utilize the voting strategy derived from ensemble classification [38]. Unlike in ensemble classification, where true labels are available, these methods require mapping the labels of individual base partitions to a unified label. This process essentially corresponds to solving a weighted bipartite graph matching problem, which is typically accomplished using the Hungarian algorithm [27], [39]. Although these methods inherit the concept of majority voting, their efficacy is compromised due to inadequate consideration of the base partition fusion methodology and their inability to accurately capture the inherent local structure of the data leads to suboptimal performance.

The methods based on co-association (CA) matrix makes use of pairwise-similarity relationships to aggregate multiple partitions. Fred and Jain [2] presented a general framework for such methods, where the base partition results are represented as the similarity between point pairs, and subsequent clustering methods (e.g., spectral clustering [40], hierarchical clustering [41]) can be used. This class of methods applies many various consensus functions to such CA matrix to generate the final clustering results. Li et al. [42] introduced a weighted consensus clustering method, where each input clustering is weighted and the weights are determined in such a way that the final consensus clustering provides a higher-quality solution. Subsequently, Yang et al. [19] proposed a novel weighted consensus function guided by clustering validation criteria to reconcile initial partitions to candidate consensus partitions from different perspectives. Lam-On et al. [43] achieved a link-based approach using the similarity between clusters that are estimated from a link network model of the ensemble. To alleviate the higher time and space complexity, Liu et al. [44], [45] employed spectral clustering on the CA matrix and developed an efficient solution. It has theoretical equivalence to weighted K-means clustering and results in vastly reduced algorithmic complexity. To accomplish the same purpose, U-SENC [46] achieves nearly linear time complexity with the construction of a sparse affinity sub-matrix. Huang et al. [47] considered the weight assignment of each cluster obtained as an individual and proposed an ensemble-driven cluster uncertainty estimation with local weighting strategy. Jia et al. [48] developed a low-rank tensor approximation based ensemble method to address the issue of the co-association matrix being dominated by inferior base

partitions. The CA matrix-based methods have shown favorable outcomes when applied to benchmark data. However, its efficacy deteriorates when dealing with datasets encompassing intricate global information. We attribute this reason to the fact that focusing exclusively similarities among individual data points does not provide a comprehensive portrayal of the inter-cluster associations.

Graph-based ensemble clustering methods were initially proposed by Strehl et al. [6]. In their work, CA matrix is treated as an undirected weighted graph and partitioned into C clusters of similar size using a multi-level C-way graph partitioning method called METIS [49] in the CSPA. Similarly, HGPA treats the result of base partition as a hypergraph and applies HMETIS [50] to partition it. The idea of MCLA is to group and collapse the relevant hyperedges and assign each object to the collapsed hyperedge in which it is most strongly involved. Fern and Brodley [7] proposed a reduction method, HBGF, that constructs a bipartite graph by a given base partitions. The resulting graph models both instances and clusters of the ensemble simultaneously as vertices in the graph. Yu et al. [51] developed an ensemble clustering method based on double nearest neighbor propagation, which used multiple distance functions and pruning of noisy attributes, followed by a normalized cut algorithm to obtain the final result. Huang et al. [52] proposed a ensemble clustering method based on sparse graph representation with elite neighbor selection strategy and probability trajectory analysis. Zhou et al. [53] learned a structured bipartite graph by self-paced learning, where the reliability of each edge is automatically decided and involved in graph learning in order of their reliability. In addition to these, there exist several effective methods that aim to address the issue of base partition fusion from a range of different perspectives, as detailed in the literature [54], [55], [56], [57], [58], [59], [60]. Despite the considerable progress that has been made in the field of ensemble clustering, the approach has yet to be integrated with deep learning techniques, which could potentially yield advancements in the characterization of complex data.

B. Deep Clustering Methods

Deep clustering has gained favor in recent years due to its expansive parameter space, which has been shown to effectively handle increasingly complex and high-dimensional data. Traditional clustering methods such as those proposed in [61], [62], [63] may lack the necessary power to accurately characterize such data as its complexity and dimension grow. The majority of current deep clustering techniques rely on self-supervised strategies, which can be broadly categorized into two distinct groups: contrastive-based methods [64] and reconstruction-based methods [65]. Contrastive-based methods provide supervisory signals for clustering tasks by pulling positive samples closer together and pushing negative samples further apart, aiming to learn embeddings with discriminative properties. In this context, the classic work contrastive clustering [66], [67], [68] introduced cluster-level contrastive loss and jointly optimized it with instance-level objectives. Subsequent works [69], [70], [71] aimed to enhance the uniformity of embeddings by maximizing the distances between prototype representations.

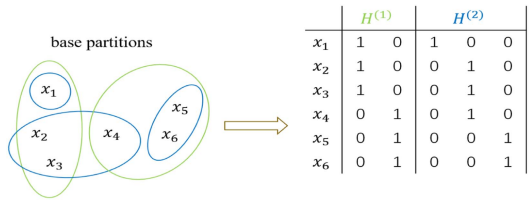

Fig. 2. Illustrative ensemble clustering problem.

The reconstruction of data in deep clustering is typically accomplished through the utilization of autoencoders [72]. Many approaches encode and cluster data by incorporating specific network modules and applying diverse clustering losses. For example, DEC [73] and IDEC [74] are trained using the KL divergence loss joint reconstruction loss. SDNE [75] exploits the first-order and second-order proximity jointly to preserve the network structure. Ji et al. [76] introduced a self-expressive layer between the encoder and the decoder to mimic the self-expressiveness property. Huang et al. [77] proposed DeepCluE, which integrates instance-level contrast, cluster-level contrast, and multilayer collaborative ensemble clustering into a unified framework. Although DeepCluE attempts to achieve ensemble clustering through deep learning, it is limited to image clustering due to its utilization of convolutional neural networks as the backbone. In general, there is still a gap that needs to be bridged between deep learning and ensemble clustering.

III. Proposed Method

A. Formulation of Ensemble Clustering

Notations: Throughout the article, the vectors and matrices are denoted by bold lowercase and bold uppercase letters, respectively. The transpose, the j-th column, the (i, j)-th entry, the trace and the Frobenius norm of matrix $\mathbf{A}$ are denoted by $\mathbf{A}^T$, $\mathbf{a}_j$, $\mathbf{A}_{ij}$, $Tr(\mathbf{A})$, $\|\mathbf{A}\|_F^2$, respectively.

Matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times N}$ is the numerical representation of the feature of dataset $\mathcal{X}$. C denotes the number of gound-truth classes for clustering and M denotes the number of base partitions. For the m-th base partition, the number of clusters is set to be k_m, typically larger than C. The m-th partition is denoted as $\mathcal{P}^{(m)} = \{\mathcal{P}_1^{(m)}, \mathcal{P}_2^{(m)}, \ldots, \mathcal{P}_{k_m}^{(m)}\}$. In this article, we only discuss hard clusters, which means clusters do not overlap each other, i. e., $\mathcal{P}_i^{(m)} \cap \mathcal{P}_j^{(m)} = \varnothing, i \neq j$ and $\cup_{i=1}^{k_m} \mathcal{P}_i^{(m)} = \mathcal{X}$.

B. Ensemble Clustering With Attentional Encoder

The binary matrix $\mathbf{H}^{(m)} \in \mathbb{R}^{N\times k_m}$ can be constructed as the indicator matrix of the partition $\mathcal{P}^{(m)}$:

$$\mathbf{H}_{ij}^{(m)} = \begin{cases} 1, & x_i \in \mathcal{P}_j^{(m)}, \\ 0, & x_i \notin \mathcal{P}_j^{(m)}. \end{cases} \tag{1}$$

Fig. 2 shows an illustrative ensemble clustering problem with two base partitions as an example.

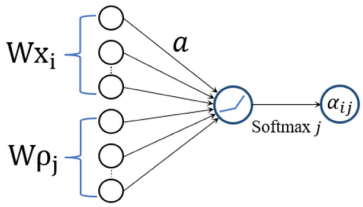

Fig. 3. Attention mechanism employed in our model.

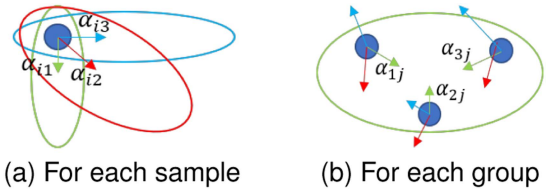

Fig. 4. Movement of sample points with attention factor.

The overall base clustering results can be obtained by the concatenation operation $||$ as $\mathbf{H} = [\mathbf{H}^{(1)}||\cdots||\mathbf{H}^{(M)}] \in \mathbb{R}^{N\times K}$, where $K = \sum_{m=1}^{M} k_m$. From this, the CA matrix can be constructed as

$$\mathbf{S}_0 = \frac{1}{M}\mathbf{H}\mathbf{H}^T. \tag{2}$$

Each column of $\mathbf{H}$ is a cluster in the base partitions, which we consider as a tiny "group" with tight association. The core of our proposed method is to combine feature and topological information to parametrically learn the importance of each sample with respect to the group of interest, thereby generating a new representation of each sample. To achieve this, we adopt a straightforward approach whereby we represent the group by the centroid of all samples in that group:

$$\boldsymbol{\rho}_j = \frac{1}{\|\mathbf{h}_j\|_1}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{H}_{ij}. \tag{3}$$

Thus, the importance of correlation between a sample and its associated group can be learned in feature-level. A learnable linear transformation is required for each layer to adequate sufficient representational power that enables the production of high-level features from input features. For this purpose, a shared linear transformation parameterized by the matrix $\mathbf{W} \in \mathbb{R}^{d'\times d}$ is applied to each node. Besides, the existing ensemble clustering methods fail to effectively portray the relationship between sample points and the corresponding groups. Drawing inspiration from the shared self-attention mechanism on nodes [78], we parameterize a vector $\mathbf{a} \in \mathbb{R}^{d'}$ in each layer to characterize the degree of affiliation of each node to its groups of interest:

$$e_{ij} = \delta\left(\mathbf{a}^T[\mathbf{W}\mathbf{x}_i||\mathbf{W}\boldsymbol{\rho}_j]\right) \tag{4}$$

where δ is the nonlinear activation function LeakyReLU with a slope of 0.2. To facilitate the generation of new representations, we apply the softmax function for normalization:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k=1}^{K}\exp(e_{ik})\mathbf{H}_{ik}}. \tag{5}$$

Fig. 3 depicts a visualization of the self-attention mechanism. It should be noted that each layer shares the same attention factor $\mathbf{a}$ to maintain consistency across the entire data.

With the normalized attention coefficients obtained, the output features of this layer for sample $\mathbf{x}_i$ can be calculated through a linear combination in the following form:

$$\begin{aligned}\mathbf{x}_i^{(l+1)} &= layer^{(l)}(\mathbf{x}_i^{(l)}) \\ &= \theta\sum_{j=1}^{K}\alpha_{ij}^{(l)}\mathbf{H}_{ij}\mathbf{W}^{(l)}\boldsymbol{\rho}_j^{(l)} + (1-\theta)\mathbf{x}_i^{(l)}.\end{aligned} \tag{6}$$

Different from [78], an identity transformation is implemented to prevent excessive shifts in the resulting representation. This is due to the possibility that the centroids of the groups and samples may be located far apart. The parameter θ is consistently set to be 0.5 in all our experiments.

Fig. 4(a) and (b) provide a visual representation of how the sample points move with the attention coefficient. The blue point represent a sample and the oval circles represent the groups associated with it. Metaphorically speaking, various groups - denoting varying topological connections - are persistently exerting force on the sample points through the attention mechanism, ultimately guiding the sample points towards their appropriate large clusters.

Further, the latent embedding $\mathbf{Z} \in \mathbb{R}^{\tilde{d}\times N}$ of the samples can be obtained by stacking two attentional layers:

$$\begin{aligned}\mathbf{z}_i' &= layer^{(1)}(\tilde{\mathbf{x}}_i), \\ \mathbf{z}_i &= layer^{(2)}(\mathbf{z}_i').\end{aligned} \tag{7}$$

where the input $\tilde{\mathbf{x}}_i$ is the normalization of the original features $\mathbf{x}_i$. Despite comprising only two layers, our encoder module exhibits a parsimonious but efficient design, without compromising its functionality.

C. Loss Function

After the latent embedding $\mathbf{Z}$ obtained, there are various decoders available. We adopt the following simple but effective inner product decoder like [79]:

$$\tilde{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^T), \tag{8}$$

where σ is the sigmoid function that maps the inner product of $\mathbf{Z}$ to the range of 0 and 1. This further derives the reconstruction loss we require:

$$L_R = \|\tilde{\mathbf{A}} - \sum_{m=1}^{M}\gamma_m\mathbf{S}_m\|_F^2 \tag{9}$$

where $\mathbf{S}_m = \mathbf{H}^{(m)}\mathbf{H}^{(m)^T}$ represents the adjacency matrix of each base partition. $\boldsymbol{\gamma}$ is the weight of base partitions. It is worth noting that during the pre-training stage, only the auto-encoder is working. At this time $\boldsymbol{\gamma}$ is fixed to $\frac{1}{M}$, which is equivalent

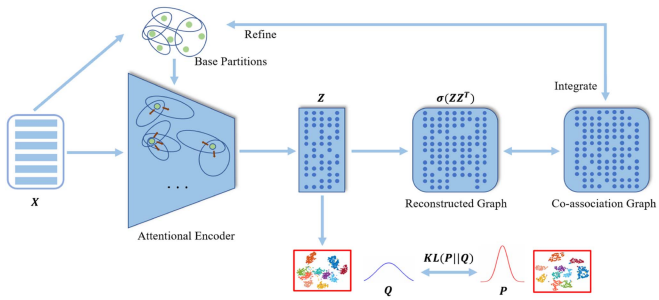

Fig. 5. The architecture for the proposed ECAR.

to reconstructing the original CA matrix in (2). The loss in the pre-training stage can be simplified as a projection problem:

$$L_{pre} = \|\tilde{\mathbf{A}} - \mathbf{S}_0\|_F^2 \tag{10}$$

In the training stage, the weight of the base partitions is expected to be refined by joint clustering losses, at which point $\boldsymbol{\gamma}$ is learnable.

Apart from minimizing the reconstruction loss, the guidance for clustering loss is also indispensable. Following DEC [73], we adopt the Student's t-distribution as a kernel to measure the similarity between embedding $\mathbf{z}_i$ and centroid $\boldsymbol{\mu}_j$:

$$\mathbf{Q}_{ij} = \frac{(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_j\|/v)^{-\frac{v+1}{2}}}{\sum_{k=1}^{c}(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|/v)^{-\frac{v+1}{2}}}, \tag{11}$$

where v is the degree of freedom of the Student's t-distribution and the value of v is typically taken as 1. The resulting $\mathbf{Q}$ represents the soft clustering of each sample to class. We iteratively strengthen the clustering predictions by learning from their high confidence assignments with the assistance of the following target distribution:

$$\mathbf{P}_{ij} = \frac{\mathbf{Q}_{ij}^2 / \sum_{l=1}^{n} \mathbf{Q}_{lj}}{\sum_{k=1}^{c}(\mathbf{Q}_{ik}^2 / \sum_{l=1}^{n} \mathbf{Q}_{lk})}. \tag{12}$$

The target distribution $\mathbf{P}$ can be regarded as a more 'confident' clustering assignment to sharpen the initial assignment $\mathbf{Q}$. Then a KL divergence between the two distribution can be employed as clustering loss:

$$L_C = KL(\mathbf{P}||\mathbf{Q}) = \sum_{i=1}^{N}\sum_{j=1}^{C} \mathbf{P}_{ij} log \frac{\mathbf{P}_{ij}}{\mathbf{Q}_{ij}} \tag{13}$$

Our proposed method is jointly trained through the above two losses, and the total objective loss function is defined as:

$$Loss = L_R + \lambda L_C. \tag{14}$$

λ is the trade-off parameter to control the balance between reconstruction loss and clustering loss. The inherent significance of this joint loss lies in its persistent guidance of the current embeddings, enabling them to acquire discriminative features within an expansive feasible space that adheres to the constraints of reconstruction error. The architecture for ECAR is shown in Fig. 5.

D. Optimization and Discussions

The loss function involves several variables, including the weight of base partitions $\boldsymbol{\gamma}$, the learned embedding $\mathbf{z}$, its centroid $\boldsymbol{\mu}$ the soft assignment distribution $\mathbf{Q}$ and the target distribution $\mathbf{P}$. This may appear difficult to optimize, but in fact the update of $\boldsymbol{\mu}$ depends only on $\mathbf{z}$ by k-means, the update of $\mathbf{Q}$ depends only on $\boldsymbol{\mu}$ and $\mathbf{z}$ by (11), and the update of $\mathbf{P}$ depends only on $\mathbf{Q}$ by (12). Therefore, the optimization process actually involves only two variables: $\boldsymbol{\gamma}$ and $\mathbf{z}$. We adopt the alternating optimization strategy to optimize the $Loss(\mathbf{z}, \boldsymbol{\gamma})$ in (14).

Update $\boldsymbol{\gamma}$ with $\mathbf{z}$ fixed. To avoid trivial solutions and concentration of weights on few base partitions, the regular term of the weight $\boldsymbol{\gamma}$ is imposed. From the reconstruction loss in (9), the subproblem of $\boldsymbol{\gamma}$ can be written as the following constrainted optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\gamma}} \|\tilde{\mathbf{A}} - \sum_{m=1}^{M} \gamma_m \mathbf{S}_m\|_F^2 + \beta\|\boldsymbol{\gamma}\|_2^2, \\ &\text{s.t. } \boldsymbol{\gamma}^T\mathbf{1} = 1, \boldsymbol{\gamma} \geq 0. \end{aligned} \tag{15}$$

Let $\mathbf{S}$ denote $\sum_{m=1}^{M} \gamma_m \mathbf{S}_m$. Expanding through the matrix trace, (15) is equivalent to

$$\begin{aligned} &\min_{\boldsymbol{\gamma}} Tr(\mathbf{S}^T\mathbf{S}) - 2Tr(\mathbf{S}^T\tilde{\mathbf{A}}) + \beta\|\boldsymbol{\gamma}\|_2^2, \\ &\text{s.t. } \boldsymbol{\gamma}^T\mathbf{1} = 1, \boldsymbol{\gamma} \geq 0. \end{aligned} \tag{16}$$

The first part of the objective function can further be rewritten in polynomial form:

$$\begin{aligned} &Tr(\mathbf{S}^T\mathbf{S}) - 2Tr(\mathbf{S}^T\tilde{\mathbf{A}}) \\ &= \sum_{m=1}^{M}\sum_{l=1}^{M} Tr(\mathbf{S}_m\mathbf{S}_l)\gamma_m\gamma_l - 2\sum_{m=1}^{M} Tr(\mathbf{S}_m^T\tilde{\mathbf{A}})\gamma_m \end{aligned} \tag{17}$$

Therefore (15) can be converted into the following quadratic optimization problem

$$\begin{aligned} &\min_{\boldsymbol{\gamma}} \boldsymbol{\gamma}^T(\boldsymbol{\Phi} + \beta\mathbf{I})\boldsymbol{\gamma} - 2\boldsymbol{\eta}^T\boldsymbol{\gamma} \\ &\text{s.t. } \boldsymbol{\gamma}^T\mathbf{1} = 1, \boldsymbol{\gamma} \geq 0, \end{aligned} \tag{18}$$

where $\boldsymbol{\Phi}_{ij} = Tr(\mathbf{S}_i\mathbf{S}_j)$ and $\boldsymbol{\eta}_i = Tr(\mathbf{S}_i^T\tilde{\mathbf{A}})$. And $\boldsymbol{\Phi}$ positive semidefinite in accordance with (17). The above quadratic programming problem with constraints can be solved by active set mehtod [80], sequential quadratic programming [81], interior-point [82] and so on.

Then we adopt Adam optimization strategy [83] to train the network with $\boldsymbol{\gamma}$ fixed. It should be noted that the target distribution $\mathbf{P}$ is treated as the "current sharper assignment" to strengthen the prediction, but frequent updates of $\mathbf{P}$ will cause the soft distribution $\mathbf{Q}$ to be pulled back and forth, making it difficult to update. Therefore, the update of $\mathbf{P}$ needs to be done at intervals of several epochs (generally less than 5). The update of the weight γ is performed similarly.

The final clustering result can be obtained directly by selecting the category with the highest confidence level from the soft

assignment $\mathbf{Q}$ after convergence:

$$y_i = \arg\max_j \mathbf{Q}_{ij}. \tag{19}$$

Furthermore, from the update of the weight of the base partitions in (18), two properties can be summarized from the above derivation:

Diversity: On the one hand, minimizing the term $\boldsymbol{\gamma}^T \boldsymbol{\Phi} \boldsymbol{\gamma}$ in (18) embodies the diversity of weight assignments. By defining operator $sum(\mathbf{X}) = \sum_i \sum_j \mathbf{X}_{ij}$ on matrix, there is

$$Tr(\mathbf{S}_i \mathbf{S}_j) = Tr(\mathbf{S}_i^T \mathbf{S}_j) = sum(\mathbf{S}_i \circ \mathbf{S}_j) \tag{20}$$

where $\circ$ is hadamard product. According to (17), the larger the value of $sum(\mathbf{S}_i \circ \mathbf{S}_j)$, the smaller the value of $\gamma_m \gamma_l$ is expected to be. This is in line with the original intention of consensus clustering, i.e., providing clustering results from multiple perspectives to uncover various types of structural information.

Consistency: On the other hand, maximizing the term $\boldsymbol{\eta}^T \boldsymbol{\gamma}$ in (18) embodies the consistency of weight assignments. Similar to the above, there is

$$Tr(\mathbf{S}_i^T \tilde{\mathbf{A}}) = sum(\mathbf{S}_i \circ \tilde{\mathbf{A}}) \tag{21}$$

According to (17), the smaller the value of $sum(\mathbf{S}_i \circ \tilde{\mathbf{A}})$, the smaller the value of γ_m is expected to be. This implies to reduce the weights of base partitions that are of low quality or differ significantly from the embedding. These two properties filter the base partitions from the initial perspective of ensemble.

The procedure for our proposed ECAR is outlined in Algorithm 1.

Algorithm 1: Procedure for ECAR.

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{d \times N}$, ensemble size M, trade-off parameters λ and β, learning rate lr, interval of update t_1, t_2, maximum epoch number T.
Initialization: Run the clustering algorithm to get base partitions $\mathbf{H}$ and the adjacency matrix $\mathbf{S}_m$ of each.
Pre-train the autoencoder by minimizing loss in (10).
for $epoch = 1, 2, \ldots, T$ **do**:
  if $epoch \% t_1 == 0$ **then**
    Update the weight $\boldsymbol{\gamma}$ by solving (18).
  end
  Obtain the contriod $\boldsymbol{\mu}$ by k-means on $\mathbf{Z}$.
  Calculate the soft assignment $\mathbf{Q}$ by (11).
  if $epoch \% t_2 == 0$ **then**
    Calculate the target distribution $\mathbf{P}$ by (12).
  end
  Backpropagate the loss in (14) and update $\mathbf{W}$, $\boldsymbol{\alpha}$.
end
Output: Clustering result by (19).

Besides, the time overhead of ECAR inevitably reaches a quadratic complexity due to the loss of reconstruction involving the weighted graph. Self-attentional operations for each sample and its associated cluster can be performed in parallel. The time overhead for the computation of each attention layer is $\mathcal{O}(Kd' + Ndd')$, where d is the input dimension and d' is the output dimension. The time overhead for the computation of the loss function in (14) is $\mathcal{O}(d'N^2)$. The time overhead of solving the quadratic programming to update the weight is approximately $\mathcal{O}(K^3)$. In total, the time complexity of ECAR is $\mathcal{O}((d'N^2 + K^3)T)$, where T is the number of epochs.

IV. Experiments

In this section, we empirically evaluate the effectiveness of our methods on real-world datasets and analyze the experimental results.

A. Experimental Settings

To ensure the fairness and simplicity of the experiments, each experiment utilizes the results of 20 times k-means as the base partitions. We perform base partition using a number of clusters greater than C, i.e., variable numbers of clusters, to explore the local information contained within the data. Each of the base partitions is produced by the k-means algorithm with the number of clusters k_m randomly selected from the range of $[2C, \sqrt{N}]$, where C is the known number of ground-truth clusters and N is the total number of samples. The number of pretraining and training epochs in each experiment is set to 50 respectively. The interval of update t_1 and t_2 is set by grid $[1, 3, 5]$. To rule out the chance findings and increase reliability, each experiment is repeatedly run five times and its mean value is reported.

To evaluate the effectiveness of the clustering, the study used three metrics, namely accuracy (ACC), normalized mutual information (NMI), and adjusted Rand index (ARI), all of which have a range of values from 0 to 1, except for ARI which ranges from -1 to 1. The labels obtained through the clustering process are compared with the labels provided by the data set using these metrics. A higher value for each of the metrics is considered indicative of a better performance for the clustering algorithm.

B. Illustrative Experiment

A rudimentary toy data was generated to explicate the concept of variable numbers of clusters. The dataset is composed of two dimensions and encompasses 2200 samples, which can be categorized into two linear classes and four spherical classes. The linear classes each consist of 500 samples, while each spherical class is made up of 300 samples. The dataset with reference labels is visualized in Fig. 6(a)

We perform k-means on the toy data, commencing with k being set to the true number of class and gradually increasing it. The clustering results for $k = 6$ and $k = 20$ are depicted in Fig. 6(b) and (c), respectively. The obtained results from employing k-means clustering with $k = 6$ on the rudimentary data exhibit unsatisfactory performance, as evidenced by the relatively low values of ACC and NMI measures, which amount to 0.673 and 0.632, respectively. The suboptimal performance of k-means on this particular dataset can be attributed to its distance-based property, which is suited for spherical data. In contrast, the partition mode of the toy dataset relies on density connectivity, rendering the k-means approach less effective. This phenomenon is more prominent in high-dimensional datasets

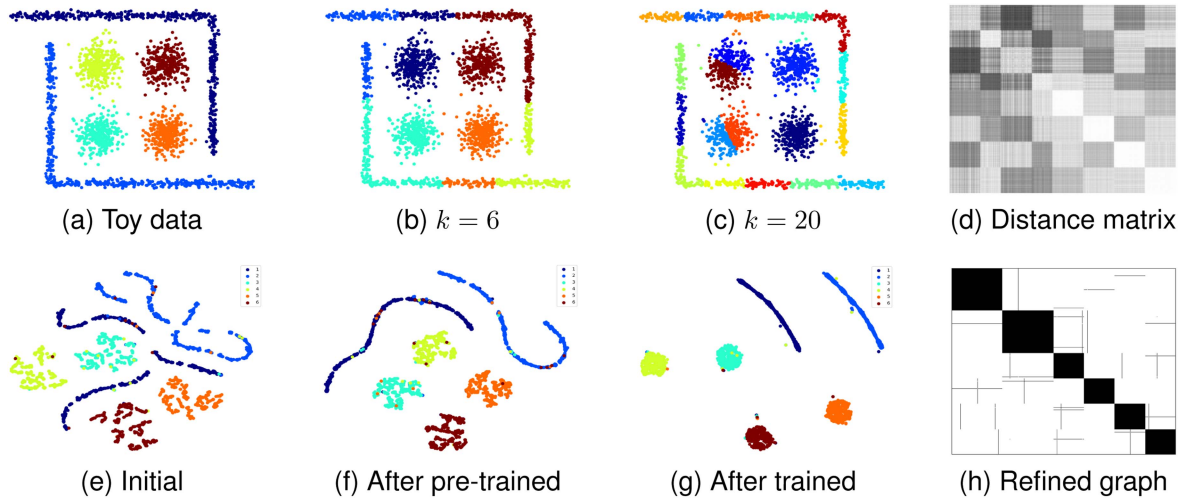

Fig. 6. Experiments on toy data.

due to the nature that data of the same class tend to be distributed within the same subspace. However, through Fig. 6(c), it can be found that, as the parameter k increases, the subdivision of the overall division practically contain the desired results, i.e., the convincing partition can be restored by merging mini-clusters. This indicates that the variable number of cluster-based methods are capable of more comprehensively mining the intrinsic structure of the data.

Utilizing the outcomes of base partitions as the input to ECAR, ACC and NMI of the final result is 0.954 and 0.937, which far exceed those obtained from a singular k-means. During the training process, the dimension reduction results generated by t-SNE [84] at $epoch = 0$, after pre-trained and after trained are depicted in Fig. 6(e), (f), and (g), respectively. The visual representation demonstrates that the toy data has been gradually clustered into six distinguishable clusters. The distance matrix of toy data and the graph refined by updating the weight are presented as grayscale plots in Fig. 6(d) and (h), respectively. It is evident that despite the infeasibility of distance-dependent clustering techniques in the present scenario, it can be rectified through the amalgamation of several base partitions.

C. Datesets and Compared Methods

We perform clustering experiments on 12 realistic datasets and validate the performance of our proposed algorithm by comparing it with a base clusterman and seven advanced and representative ensemble clustering algorithms.

The datasets we choosed are widely-used and their details are shown in Table I, where MSRA can be found at[1], ORL, Yale32, USPS, MNIST, ISOLET, TOX can be found at[2], ARCENE, Letter, Segment, Vote, Vehicle can be found at[3]. They belong to different types and come from different areas.

[1]http://123.57.240.48/forum.php?mod=viewthread&tid=1327
[2]https://github.com/jundongl/scikit-feature/tree/master/skfeature/data
[3]https://archive.ics.uci.edu/ml/datasets.php

TABLE I
DETAILS OF THE DATASETS

Dataset	Instances	Features	Classes	Data Type
MSRA	1799	256	12	Object Image
ORL	400	1024	40	Face Image
Yale32	2414	1024	38	Face Image
USPS	9298	256	10	Handwritten Image
MNIST	70000	784	10	Handwritten Image
ISOLET	1560	617	26	Speech
TOX	171	5748	4	Microarray
ARCENE	200	10000	2	Spectrometry
Letter	20000	16	26	Attribute
Segment	2310	19	7	Attribute
Vote	435	16	2	Attribute
Vehicle	946	18	4	Attribute

The competitive algorithms include: 1) k-means average (**KM-avg**), which is the average result of the base clusterings. 2) k-means best (**KM-best**), which is the best result of the base clusterings. 3) Cluster-based Similarity Partitioning Algorithm (**CSPA**) [6], which treats CA matrix as an undirected weighted graph and partitioned into C clusters. 4) HyperGraph Partitioning Algorithm (**HGPA**) [6], which treats the result of base clustering as a hypergraph and applies multi-level C-way graph partitioning method. 5) Probability Trajectory Accumulation (**PTA**) [52], which bases on hierarchical agglomerative clustering and probability trajectory analysis. 6) Spectral Ensemble Clustering (**SEC**) [45], which performs spectral clustering on CA matrix. 7) Locally Weighted Evidence Accumulation (**LWEA**) [47], which performs hierarchical agglomerative clustering on locally weighted CA matrix. 8) Autoencoder + k-means (**AE**) [72], which consists of a three-layer encoder and decoder, and employs k-means clustering on the learned embeddings. 9) Improved Deep Embedded Clustering (**IDEC**) [74], which manipulates the feature space to scatter data points using a clustering loss as guidance. 10) Deep Attentional Embedded Graph Clustering (**DAEGC**) [85], which encodes the topological structure into a compact representation by capturing the importance

TABLE II
AVERAGE PERFORMANCE COMPARISON OF DIFFERENT CLUSTERING METHODS

Datesets	Metrics	KM-avg	KM-best	CSPA	HGPA	PTA	SEC	LWEA	AE	IDEC	DAEGC	SDCN	**ECAR**
MSRA	ACC	0.5292	0.5590	0.4986	0.5244	0.5407	0.4902	0.5612	0.5492	0.5631	0.5308	0.4479	**0.9216**
	NMI	0.6323	0.6519	0.6306	0.5733	0.6344	0.5895	0.6529	0.6171	0.6174	0.5881	0.5393	**0.9298**
	ARI	0.3858	0.3999	0.2808	0.4261	0.3691	0.3670	0.4404	0.3755	0.3918	0.3634	0.3090	**0.8608**
ORL	ACC	0.5725	0.6313	0.6231	0.6346	0.6403	0.5483	0.6070	0.5827	0.6874	0.7150	0.7050	**0.7644**
	NMI	0.7551	0.7804	0.7549	0.7852	0.7640	0.7274	0.7722	0.7254	0.7106	0.7828	0.7806	**0.8784**
	ARI	0.4519	0.4644	0.4157	0.4779	0.4892	0.3427	0.3893	0.3808	0.5091	0.5457	0.5219	**0.6793**
Yale32	ACC	0.1023	0.1219	0.1486	0.1548	0.1517	0.1390	0.1289	0.1371	0.1288	0.1446	0.1275	**0.2730**
	NMI	0.1377	0.1492	0.1192	0.1382	0.1553	0.1044	0.1099	0.1633	0.2216	0.1681	0.1583	**0.3823**
	ARI	0.0167	0.0444	0.0492	0.0994	0.0957	0.0877	0.0718	0.0218	0.0400	0.0833	0.0784	**0.1381**
USPS	ACC	0.5858	0.6113	0.4967	0.4580	0.6025	0.5052	0.5916	0.6499	0.7597	0.6770	0.7722	**0.8160**
	NMI	0.5706	0.5906	0.4726	0.3955	0.6132	0.4594	0.5837	0.5930	0.7704	0.6492	**0.7712**	0.7472
	ARI	0.4716	0.4843	0.3332	0.2059	0.4828	0.3008	0.4496	0.5093	0.6911	0.5826	0.7024	**0.7328**
MNIST	ACC	0.4892	0.5303	0.4370	0.4227	0.6379	0.4923	0.6409	0.5618	0.7917	0.6308	0.7842	**0.8308**
	NMI	0.4299	0.4607	0.4283	0.3917	0.5933	0.3618	0.6282	0.4954	0.7750	0.5112	0.7745	**0.7996**
	ARI	0.3187	0.3381	0.3278	0.3003	0.5601	0.3198	0.6152	0.3931	0.7353	0.5092	0.7250	**0.7738**
ISOLET	ACC	0.5497	0.5722	0.5929	0.5407	0.6351	0.6117	0.6336	0.5466	0.6436	0.5125	0.4526	**0.7173**
	NMI	0.7398	0.7633	0.6995	0.6127	0.7194	0.6888	0.7285	0.7031	0.7712	0.6648	0.5962	**0.8024**
	ARI	0.5060	0.5299	0.4608	0.3680	0.4592	0.4377	0.5424	0.4427	0.5666	0.4787	0.3112	**0.5970**
TOX	ACC	0.4266	0.4703	0.4425	0.4151	0.4300	0.4133	0.4229	0.4743	0.3801	0.4292	0.3626	**0.4971**
	NMI	0.1910	0.2332	0.1626	0.1237	0.1351	0.1276	0.1251	0.2667	0.0929	0.1251	0.0746	**0.2812**
	ARI	0.1255	0.1858	0.1247	0.1003	0.0990	0.096	0.1083	0.1712	0.0504	0.1115	0.0456	**0.2343**
ARCENE	ACC	0.6425	0.6550	0.6286	0.5230	0.6390	0.5750	0.6085	0.6500	0.6600	0.6200	0.6300	**0.6925**
	NMI	0.0829	0.0945	0.0563	0.0017	0.0878	0.0684	0.0834	0.0900	0.0908	0.0714	0.0738	**0.1278**
	ARI	0.0852	0.1097	0.0893	-0.0507	0.1045	-0.0771	0.0938	0.0850	0.0977	0.0833	0.0623	**0.1737**
Letter	ACC	0.2596	0.2628	0.3381	0.3469	0.3003	0.2428	0.2976	0.2705	0.2872	0.2399	0.3297	**0.3483**
	NMI	0.3552	0.3864	0.3594	0.3580	0.3621	0.3361	0.3647	0.3635	0.3784	0.3618	0.4299	**0.4180**
	ARI	0.1321	0.1466	0.1136	0.1225	0.1574	0.1302	0.1494	0.1404	0.1477	0.1397	0.1932	**0.2000**
Segment	ACC	0.5766	0.5923	0.2831	0.4269	0.5684	0.3512	0.5730	0.7047	0.5987	0.5602	0.5964	**0.7310**
	NMI	0.4990	0.5322	0.3012	0.4077	0.5540	0.4659	0.6028	**0.6449**	0.5967	0.5898	0.5830	0.6286
	ARI	0.4460	0.4729	0.0913	0.2358	0.3561	0.3824	0.4969	0.5436	0.4525	0.4762	0.4708	**0.5641**
Vote	ACC	0.8590	0.8763	0.8478	0.7456	0.8965	0.8411	0.8706	0.8368	0.8391	0.8719	0.8799	**0.9257**
	NMI	0.4495	0.4629	0.4234	0.2758	0.4925	0.4489	0.4696	0.3734	0.3845	0.4261	0.4574	**0.5574**
	ARI	0.5433	0.5752	0.5482	0.3695	0.6147	0.5282	0.5746	0.4524	0.4586	0.5306	0.5540	**0.6677**
Vehicle	ACC	0.3472	0.3588	0.3926	0.3862	0.3610	0.3294	0.3881	0.4314	0.4326	0.4872	0.4921	**0.5396**
	NMI	0.1100	0.1292	0.1923	0.1605	0.1448	0.0981	0.1920	0.1667	0.1940	0.1945	0.2212	**0.2609**
	ARI	0.0731	0.0912	0.0962	0.0842	0.0934	0.0793	0.1009	0.1083	0.1131	0.1174	0.1718	**0.2170**

of neighboring nodes. 11) Structural Deep Clustering Network (**SDCN**) [86], which transfers the representations learned by AE to a GCN layer, and employs a dual self-supervised mechanism to unify the two neural architectures. We retain all the default parameter settings from the authors' original implementation, while employing grid search to achieve a fair and comprehensive comparison.

D. Experimental Results

The results of the clustering experiments conducted on twelve chosen datasets, compared with eleven algorithms, are presented in Table II. With the exception of **KM-avg** and **KM-best**, the best results are bolded and the second best results are underlined. As can be seen from the table, ECAR achieves clear improvement over other ensemble clustering algorithms for the three metrics, especially on MSRA, ORL and other datasets. As is evident from the data, ECAR demonstrates significant improvement over other ensemble clustering methods across all three metrics, with particular promote observed on the MSRA, ORL, USPS, etc. It is apparent that the traditional ensemble clustering method falls short in comparison to **KM-best** in the majority of cases, whereas ECAR outperforms **KM-best** on all datasets. Moreover, the implementation of ECAR does not necessitate a complex training process, but rather involves only two layers of attentional networks. From the comparative experiments, it is evident that ECAR consistently outperforms the selected traditional and deep algorithms across all datasets.

To visually demonstrate the impact and training methodology of ECAR, we present several visualizations in Fig. 7, where "Initial" represents the t-SNE for the original data, while "After pre-trained" and "After trained" represent the t-SNE for the pre-trained and the final embedding, respectively. It is evident that the raw dimensionality-reduced data points are widely dispersed, and without the aid of color-coded class labels, the underlying distribution of classes may not be readily discernible. As shown in Fig. 7(c), (f), (i), and (j), ECAR eventually congregates the scattered data points belonging to the same class

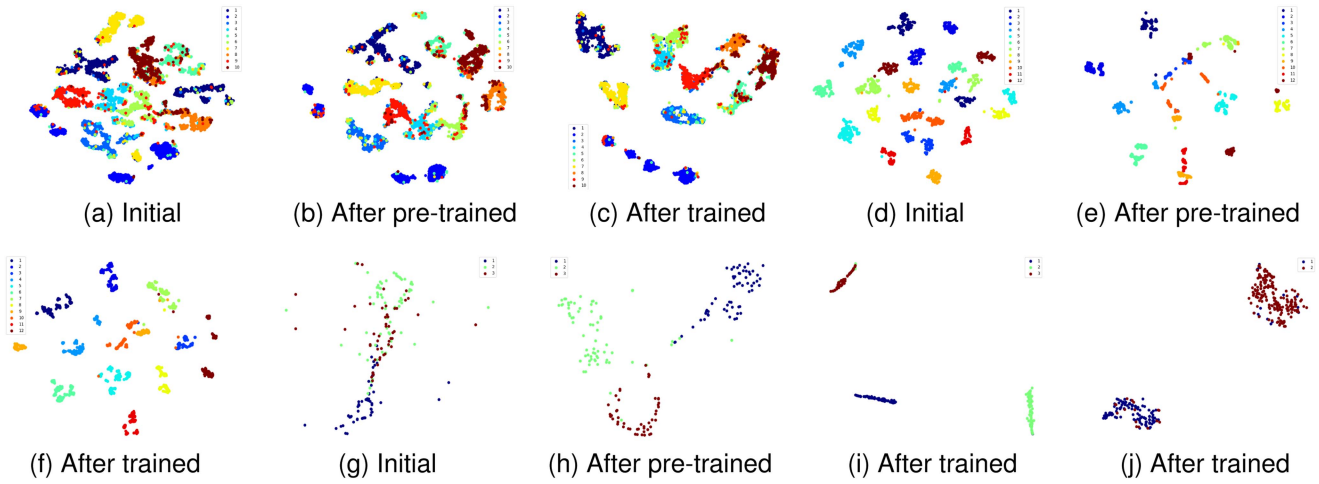

Fig. 7. Visualization on realistic datasets. USPS: (a)–(c), MSRA: (d)–(f), Wine: (g)–(i), Vote: (j).

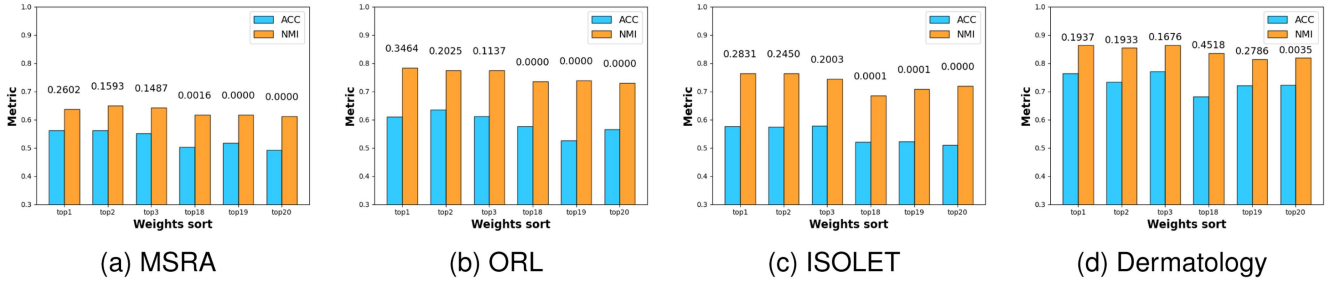

Fig. 8. The relationship between the base clustering quality and the learned weight.

into one cluster, forming a more distinct partition. Interestingly, for example in Fig. 7(c),we observe that the blue class in the lower left corner splits into four subclasses, possibly due to the separation of the samples by several tiny cluster's attentional pull. This indicates that the number of classes k in the base partitions should not be excessive. In other words, it is advisable to retain a few base clusterers with a relatively small number of classes to preserve global information.

To assess the efficacy of the graph correction approach in ECAR, we examined the correlation between the quality of the base clusterings and the learned weights. To make the quality measurable, we uniformly set m the number of categories per base clustering apparatus set to $k_m = C, m = 1, \ldots, M$ and use ACC and NMI as indicators. Although somewhat incomplete, these two measures provide an intuitive reflection of the clustering results in relation to the ground truth labels. After the training process converged, we arranged the weights of the 20 base clusterings in the final graph in descending order. In Fig. 8, the ACC and NMI of the top three and bottom three base clustering on four datasets are counted. The weight value of each base clustering is shown at the top of its corresponding two bars. From the figure, we can see that there is indeed a notable gap between the ACC and NMI of top three and bottom three (top 18 to 20), underscoring the rationality of the graph refinement strategy in ECAR.

Fig. 9 illustrates the loss values of the model as well as the variations in ACC and NMI of the learned embedding over the course of the training process. Notably, the first 50 epochs are devoted to pre-training, while the remaining 50 epochs are reserved for training process. It can be found that the pre-training results are somewhat unstable, but this is offset by a more stable and superior clustering outcome in training process with the integration of clustering loss and graph refinement. Throughout the training process, the clustering metric exhibits a gradual and consistent increase that correlates with a reduction in the loss function. This trend indicates both effectiveness and stable convergence behavior of the algorithm.

E. Parameter Analysis

In contrast to traditional methods, deep learning-based techniques are frequently hindered by a larger number of parameters that are challenging to fine-tune. Nevertheless, our proposed method has a relatively simple architecture, consisting of only two layers of network results, which makes it less complicated and more manageable. In this subsection, we study the parameters of the model. To keep things concise, we perform parameter sensitivity analysis on four data sets, MSRA, ORL, ISOLET and Dermatology.

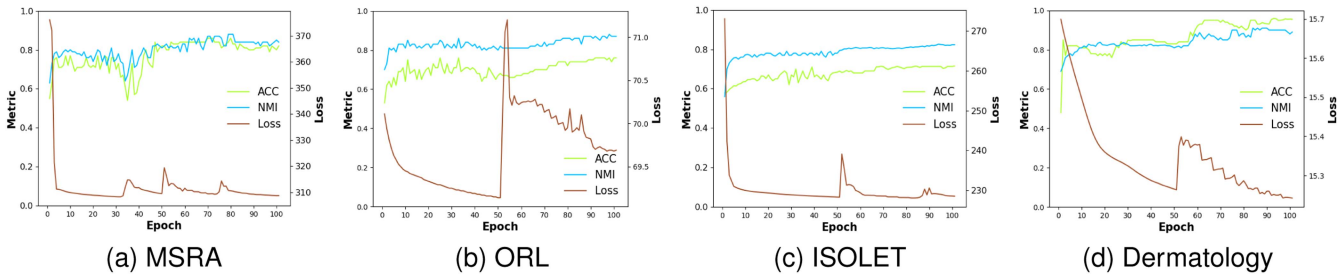

Fig. 9. Variation of loss and clustering results with epoch.

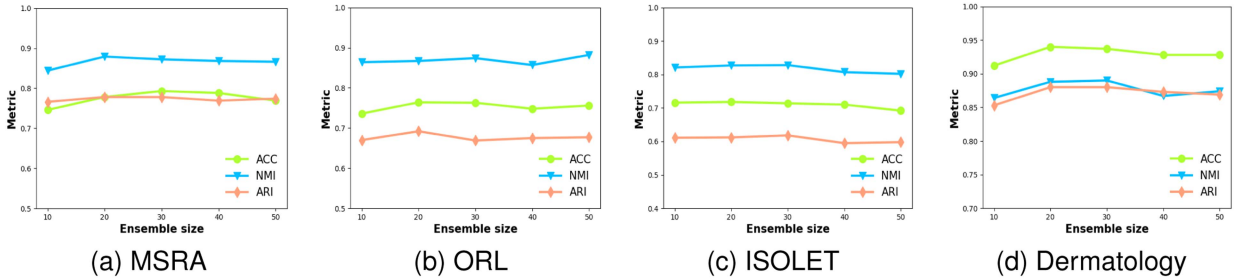

Fig. 10. Parameter study on ensemble size.

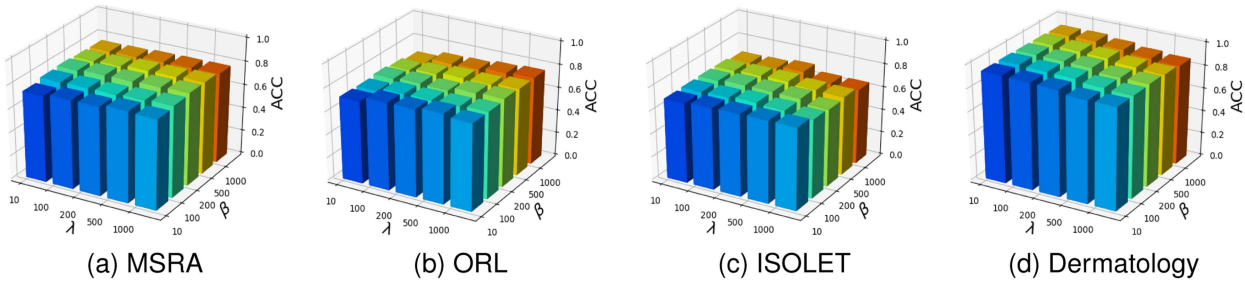

Fig. 11. Parameter study on two trade-off parameters.

Firstly, we explore the impact of varying the number of base clusterings M on the clustering performance. In the context of cluster analysis, the stability and representational power of the training results are influenced by the number of base clusterings utilized. Specifically, when a limited number of base clusterings are employed, the results obtained are prone to instability, and the representational capacity of the clustering model may be inadequate. On the other hand, a high number of base clusterings may lead to an abundance of redundant and low-quality clustering outcomes. Fig. 10 shows the performance of ECAR under three clustering metrics when M ranges from [10, 20, 30, 40, 50]. It can be seen that the performance is generally superior when M is set to 20. Meanwhile, the impact on clustering is insignificant as M fluctuates between 10 and 50, indicating that ECAR is robust to changes in the number of base clusterings.

The sensitivity analysis of the two trade-off parameters is depicted in Fig. 11, where λ is the parameter to balance the two terms in our loss function while β is the parameter related to the weight distribution. Specifically, we varies each parameter's value within the range of [10, 100, 200, 500, 1000] and observed the resulting effects. Our analysis indicated that the optimal values of lambda and beta vary for different datasets. For instance, the optimal value of λ for the ORL dataset is aound 1000, while the optimal value of λ for the ISOLET dataset is around 100. Fortunately, the effect of varying these two parameters within range [10, 1000] on the clustering performance is not significant. This indicates that ECAR is stable with respect to the two parameters and does not depend on precise parameter tuning to achieve satisfactory performance.

Regarding the network structure, we vary the dimension of the learned embedding, with different ranges of variation for different datasets, and the results are shown in Fig. 12. Upon initial inspection, it can be noticed that the performance decreases significantly when the dimension of the embedding is larger. Specifically, if the dimension of the embedding is not considerably smaller than the original dimension, the number of parameters to be learned increases, and it becomes challenging to ensure the validity of the features in the embedding. On the other hand, if the dimension of the embedding is too small, such as less than 10, it becomes relatively difficult to perform the reconstruction task, which further affects the graph correction, resulting in poorer results. In the case of datasets with a restricted number of feature dimensions, such as toy data and Glass, optimal results can be achieved by appropriately increasing the size

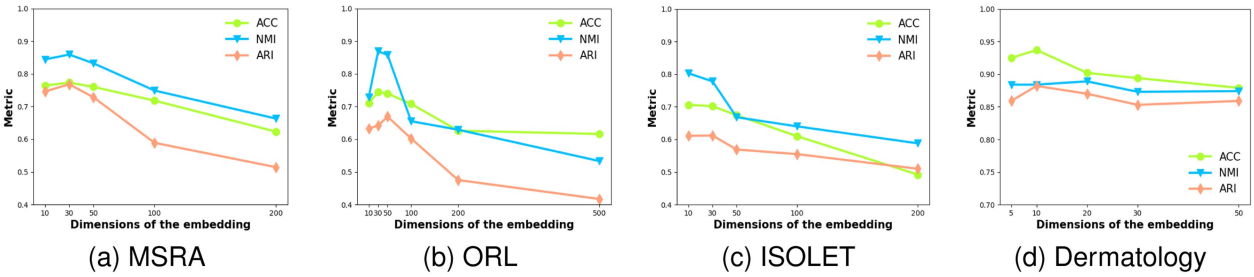

Fig. 12. Parameter study on dimension of the embedding.

TABLE III
TIME COST (SECONDS) OF COMPARISON METHODS

Datesets	Instances	Features	CSPA	PTA	SEC	LWEA	AE	IDEC	DAEGC	SDCN	**ECAR**
ARCENE	200	10000	1.7	3.0	1.0	1.3	13.6	14.8+10.3	3.4+2.4	40.6+14.9	3.7+2.5
Vehicle	946	18	1.9	2.9	1.6	1.3	4.3	4.3+4.4	2.2+2.3	51.8+18.3	2.2+2.9
MSRA	1799	256	2.7	9.0	4.4	3.5	9.6	9.4+12.0	6.8+6.8	5.4+7.3	9.5+7.3
USPS	9298	256	13.7	44.9	14.6	12.0	46.0	34.6+40.6	28.0+12.6	64.0+42.5	23.9+15.5
Letter	20000	16	36.8	167.3	22.9	25.0	174.1	125.5+284.0	69.2+29.8	410.8+261.1	83.7+47.9
MNIST	70000	784	762.8	2740.6	249.2	303.6	3429.7	3373.2+2147.2	2983.0+1332.2	7241.4+5360.5	3850.3+1697.2

of the hidden layer. Our findings suggest that the model achieves better performance when the dimension of the embedding is slightly less than the number of categories, which can be utilized as an experimental prior.

F. Efficiency

Due to the absence of network training procedures in traditional ensemble clustering algorithms, we place emphasis on the efficiency of ECAR during the experiments, as it significantly impacts the practical viability. Building on the previous experiments, we selected six datasets with increasing data sizes and examined the time overhead. We compared various traditional ensemble clustering methods and deep clustering methods, and the runtime of each algorithm is presented in Table III.

Deep clustering methods require pretraining to ensure stable results. In the table, the value on the left side of the "+" sign represents the pretraining time, while the value on the right side represents the training time. To ensure fairness, all deep methods are pretrained and trained for 50 epochs. Due to device memory limitations (DAEGC, SDCN, ECAR out of memory), all experiments on the MNIST dataset were conducted on a CPU. All experiments are conducted on the same machine with the Intel(R) Xeon(R) Gold 6248R 3.00 GHz CPU and NVIDIA TITAN Xp GPU with 10 GB RAM, running on the Ubuntu 20.04 operating system.

Apart from the MNIST (on CPU), the running time of ECAR slightly exceeds that of traditional ensemble clustering methods, but within an acceptable range. This is in line with our expectations: we trade off more parameters for improved clustering results. During actual execution, ECAR has fewer network layers and does not involve a decoding process, which mitigates the time cost associated with reconstruction error computation. As observed from the table, our proposed method and other deep learning methods have comparable runtime in high-dimensional and large-scale data scenarios, validating the feasibility of our method. From a holistic perspective, ECAR exhibits an enhancement in clustering performance within a permissible temporal range.

V. CONCLUSION

In conclusion, the proposed ECAR addresses the limitations of current mainstream ensemble clustering methods by incorporating attentional representation. By considering the results of base partition as groups with higher-order information, ECAR captures the importance of each sample's association with its related group in feature-level through the attentional network. The joint optimization of the clustering loss and reconstruction loss enhances the performance of ECAR. Moreover, the adaptive refinement of base partition weights during training ensures diversity and consistency, reducing the impact of low-quality and redundant base partitions. More importantly, our proposed method is a meaningful attempt to apply non-graph data to graph-based neural networks. The conducted experiments on parameter sensitivity and efficiency provide empirical evidence of the feasibility of our method. Extensive experimental results on real-world datasets demonstrate the substantial improvement of ECAR in comparison to existing baseline ensemble clustering methods and deep clustering methods, highlighting its potential for addressing complex data in the clustering domain.

REFERENCES

[1] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[2] A. L. Fred and A. K. Jain, "Combining multiple clusterings using evidence accumulation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 835–850, Jun. 2005.

[3] A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," *Data Knowl. Eng.*, vol. 63, no. 2, pp. 503–527, 2007.

[4] P. S. Bradley and U. M. Fayyad, "Refining initial points for k-means clustering," in *Proc. Int. Conf. Mach. Learn.*, 1998, pp. 91–99.

[5] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," *Proc. 17th Int. Conf. Neural Inf. Process. Syst.*, 2004, vol. 17, pp. 1601–1608.

[6] A. Strehl and J. Ghosh, "Cluster ensembles–A knowledge reuse framework for combining multiple partitions," *J. Mach. Learn. Res.*, vol. 3, pp. 583–617, 2002.

[7] X. Z. Fern and C. E. Brodley, "Solving cluster ensemble problems by bipartite graph partitioning," in *Proc. Int. Conf. Mach. Learn.*, 2004, pp. 36–43.

[8] A. P. Topchy, M. H. Law, A. K. Jain, and A. L. Fred, "Analysis of consensus partition in cluster ensemble," in *Proc. IEEE Int. Conf. Data Mining*, 2004, pp. 225–232.

[9] A. Topchy, A. K. Jain, and W. Punch, "Clustering ensembles: Models of consensus and weak partitions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 12, pp. 1866–1881, Dec. 2005.

[10] L. Franek, D. D. Abdala, S. Vega-Pons, and X. Jiang, "Image segmentation fusion using general ensemble clustering methods," in *Proc. Asian Conf. Comput. Vis.*, 2011, pp. 373–384.

[11] H. Kim, J. J. Thiagarajan, and P.-T. Bremer, "Image segmentation using consensus from hierarchical segmentation ensembles," in *Proc. IEEE Int. Conf. Image Process.*, 2014, pp. 3272–3276.

[12] H. Liu, M. Shao, S. Li, and Y. Fu, "Infinite ensemble for image clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1745–1754.

[13] C. Carpineto and G. Romano, "Consensus clustering based on a new probabilistic rand index with application to subtopic retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2315–2326, Dec. 2012.

[14] S. Affeldt, L. Labiod, and M. Nadif, "Ensemble block co-clustering: A unified framework for text data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 5–14.

[15] Z. Tao, H. Liu, S. Li, Z. Ding, and Y. Fu, "From ensemble clustering to multi-view clustering," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2843–2849.

[16] Z. Tao, H. Liu, S. Li, Z. Ding, and Y. Fu, "Marginalized multiview ensemble clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 2, pp. 600–611, Feb. 2020.

[17] Z. Yu, D. Wang, X.-B. Meng, and C. P. Chen, "Clustering ensemble based on hybrid multiview clustering," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 6518–6530, Jul. 2022.

[18] Y. Ye, T. Li, Y. Chen, and Q. Jiang, "Automatic malware categorization using cluster ensemble," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 95–104.

[19] Y. Yang and K. Chen, "Temporal data clustering via weighted clustering ensemble with different representations," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 2, pp. 307–320, 2010.

[20] W. Zhuang, Y. Ye, Y. Chen, and T. Li, "Ensemble clustering for internet security applications," *IEEE Trans. Cybern.*, vol. 42, no. 6, pp. 1784–1796, Nov. 2012.

[21] P. Blomstedt, J. Tang, J. Xiong, C. Granlund, and J. Corander, "A Bayesian predictive model for clustering data of mixed discrete and continuous type," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 489–498, Mar. 2014.

[22] Z. Yu et al., "Incremental semi-supervised clustering ensemble for high dimensional data clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 3, pp. 701–714, Mar. 2016.

[23] T. Boongoen and N. Iam-On, "Cluster ensembles: A survey of approaches with recent extensions and applications," *Comput. Sci. Rev.*, vol. 28, pp. 1–25, 2018.

[24] J. Wu, H. Liu, H. Xiong, J. Cao, and J. Chen, "K-means-based consensus clustering: A unified view," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 155–169, Jan. 2015.

[25] L. I. Kuncheva and D. P. Vetrov, "Evaluation of stability of k-means cluster ensembles with respect to random initialization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1798–1808, Nov. 2006.

[26] L. F. Ana and A. K. Jain, "Robust data clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, 2003, pp. II–II.

[27] A. Topchy, A. K. Jain, and W. Punch, "A mixture model for clustering ensembles," in *Proc. IEEE Int. Conf. Data Mininig*, 2004, pp. 379–390.

[28] M. H. Law, A. P. Topchy, and A. K. Jain, "Multiobjective data clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, 2004, pp. II–II.

[29] N. Nguyen and R. Caruana, "Consensus clusterings," in *Proc. IEEE Int. Conf. Data Mining*, 2007, pp. 607–612.

[30] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, pp. 4–es, 2007.

[31] N. Iam-On, T. Boongoen, and S. Garrett, "Refining pairwise similarity matrix for cluster ensemble problem with cluster relations," in *Discov. Sci. Int. Conf.*, 2008, pp. 222–233.

[32] X. Z. Fern and C. E. Brodley, "Random projection for high dimensional data clustering: A cluster ensemble approach," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 186–193.

[33] A. Topchy, A. K. Jain, and W. Punch, "Combining multiple weak clusterings," in *Proc. IEEE Int. Conf. Data Mining*, 2003, pp. 331–338.

[34] B. Fischer and J. M. Buhmann, "Bagging for path-based clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 11, pp. 1411–1415, Nov. 2003.

[35] S. Dudoit and J. Fridlyand, "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics*, vol. 19, no. 9, pp. 1090–1099, 2003.

[36] Y. Li, J. Yu, P. Hao, and Z. Li, "Clustering ensembles based on normalized edges," in *Proc. Adv. Knowl. Discov. Data Mining Pacific-Asia Conf.*, 2007, pp. 664–671.

[37] C. Domeniconi and M. Al-Razgan, "Weighted cluster ensembles: Methods and analysis," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 4, pp. 1–40, 2009.

[38] G. I. Webb and Z. Zheng, "Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 8, pp. 980–991, Aug. 2004.

[39] H. G. Ayad and M. S. Kamel, "Cumulative voting consensus method for partitions with variable number of clusters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 1, pp. 160–173, Jan. 2008.

[40] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. 14th Int. Conf. Neural Inf. Process. Syst.*, 2001, pp. 849–856.

[41] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, 1999.

[42] T. Li and C. Ding, "Weighted consensus clustering," in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 798–809.

[43] N. Iam-On, T. Boongoen, S. Garrett, and C. Price, "A link-based approach to the cluster ensemble problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2396–2409, Dec. 2011.

[44] H. Liu, T. Liu, J. Wu, D. Tao, and Y. Fu, "Spectral ensemble clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 715–724.

[45] H. Liu, J. Wu, T. Liu, D. Tao, and Y. Fu, "Spectral ensemble clustering via weighted k-means: Theoretical and practical evidence," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1129–1143, May 2017.

[46] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, and C.-K. Kwoh, "Ultra-scalable spectral clustering and ensemble clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 6, pp. 1212–1226, Jun. 2019.

[47] D. Huang, C.-D. Wang, and J.-H. Lai, "Locally weighted ensemble clustering," *IEEE Trans. Cybern.*, vol. 48, no. 5, pp. 1460–1473, May 2018.

[48] Y. Jia, H. Liu, J. Hou, and Q. Zhang, "Clustering ensemble meets low-rank tensor approximation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 7970–7978.

[49] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 359–392, 1998.

[50] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: Application in VLSI domain," in *Proc. 34th Annu. Des. Automat. Conf.*, 1997, pp. 526–529.

[51] Z. Yu, L. Li, J. Liu, J. Zhang, and G. Han, "Adaptive noise immune cluster ensemble using affinity propagation," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 12, pp. 3176–3189, Dec. 2015.

[52] D. Huang, J.-H. Lai, and C.-D. Wang, "Robust ensemble clustering using probability trajectories," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1312–1326, May 2016.

[53] P. Zhou, L. Du, and X. Li, "Self-paced consensus clustering with bipartite graph," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 2133–2139.

[54] L. Zheng, T. Li, and C. Ding, "Hierarchical ensemble clustering," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 1199–1204.

[55] O. Wu, W. Hu, S. J. Maybank, M. Zhu, and B. Li, "Efficient clustering aggregation based on data fragments," *IEEE Trans. Cybern.*, vol. 42, no. 3, pp. 913–926, Jun. 2012.

[56] J. Yi, T. Yang, R. Jin, A. K. Jain, and M. Mahdavi, "Robust ensemble clustering by matrix completion," in *Proc. IEEE Int. Conf. Data Mining*, 2012, pp. 1176–1181.

[57] J. Gao, M. Yamada, S. Kaski, H. Mamitsuka, S. Zhu, and S. Kambhampati, "A robust convex formulations for ensemble clustering," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1476–1482.
[58] Z. Tao, H. Liu, S. Li, and Y. Fu, "Robust spectral ensemble clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 367–376.
[59] D. Huang, C.-D. Wang, H. Peng, J. Lai, and C.-K. Kwoh, "Enhanced ensemble clustering via fast propagation of cluster-wise similarities," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 51, no. 1, pp. 508–520, Jan. 2021.
[60] P. Zhou, L. Du, Y.-D. Shen, and X. Li, "Tri-level robust clustering ensemble with multiple graph learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 11125–11133.
[61] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *J. Roy. Stat. Soc. Ser. C*, vol. 28, no. 1, pp. 100–108, 1979.
[62] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. IEEE Int. Conf. Data Mining*, 2005, pp. 606–610.
[63] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2765–2781, Nov. 2013.
[64] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.
[65] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.
[66] Y. Li, P. Hu, Z. Liu, D. Peng, J. T. Zhou, and X. Peng, "Contrastive clustering," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 8547–8555.
[67] Y. Li, S. Kan, and Z. He, "Unsupervised deep metric learning with transformed attention consistency and contrastive clustering loss," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 141–157.
[68] H. Zhong et al., "Graph contrastive clustering," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 9224–9233.
[69] J. Li, P. Zhou, C. Xiong, and S. Hoi, "Prototypical contrastive learning of unsupervised representations," in *Proc. Int. Conf. Learn. Representations*, 2021.
[70] Z. Huang, J. Chen, J. Zhang, and H. Shan, "Learning representation for clustering via prototype scattering and positive sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 7509–7524, Jun. 2023.
[71] S. Lin et al., "Prototypical graph contrastive learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 27 Jul. 2022, doi: 10.1109/TNNLS.2022.3191086.
[72] G. E. Hinton and R. Zemel, "Autoencoders, minimum description length and Helmholtz free energy," in *Proc. 6th Int. Conf. Neural Inf. Process. Syst.*, 1993, pp. 3–10.
[73] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 478–487.
[74] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 1753–1759.
[75] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1225–1234.
[76] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid, "Deep subspace clustering networks," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 23–32.
[77] D. Huang, D.-H. Chen, X. Chen, C.-D. Wang, and J.-H. Lai, "Deepclue: Enhanced image clustering via multi-layer ensembles in deep neural networks," 2022, *arXiv:2206.00359*.
[78] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[79] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[80] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Computation*, vol. 6, pp. 327–363, 2014.
[81] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta Numerica*, vol. 4, pp. 1–51, 1995.
[82] R. J. Vanderbei, "LOQO: An interior point code for quadratic programming," *Optim. Methods Softw.*, vol. 11, no. 1–4, pp. 451–484, 1999.
[83] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[84] L. V. d. Maaten and G. Hinton, "Visualizing data using T-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.
[85] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," 2019, *arXiv:1906.06532*.
[86] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proc. Web Conf.*, 2020, pp. 1400–1410.

Zhezheng Hao is currently working toward the graduation degree with the School of Artificial Intelligence, Optics and ElectroNics (iOPEN) and School of Cybersecurity, Northwestern Polytechnical University, Xi'an, China. His current research interests include machine learning and data mining.

Zhoumin Lu received the MS degree in computer technology from Fuzhou University, China, in 2021. He is currently working toward the PhD degree with the School of Computer Science, Northwestern Polytechnical University. His research interests include machine learning, deep learning and their applications, such as pattern recognition and data mining.

Guoxu Li is currently working toward the graduation degree with the School of Artificial Intelligence, OPtics and ElectroNics (iOPEN) and School of Cybersecurity, Northwestern Polytechnical University, Xi'an, China. His research interests include machine learning and deep learning.

Feiping Nie (Senior Member, IEEE) received the PhD degree in computer science from Tsinghua University, China in 2009. He is currently a full professor with Northwestern Polytechnical University, China. He has authored or coauthored more than 100 papers in the following journals and conferences: TPAMI, IJCV, TIP, TNNLS, TKDE, ICML, NIPS, KDD, IJCAI, AAAI, ICCV, CVPR, ACM MM. His research interests are machine learning and its applications, such as pattern recognition, data mining, computer vision, image processing and information retrieval. His papers have been cited more than 30000 times and the H-index is 99. He is currently an associate editor or PC member for several prestigious journals and conferences in the related fields.

Rong Wang (Member, IEEE) received the BS degree in information engineering, the MS degree in signal and information processing, and the PhD degree in computer science from the Xi'an Research Institute of Hi-Tech, Xi'an, China, in 2004, 2007 and 2013, respectively. During 2007 and 2013, he also studied with the Department of Automation, Tsinghua University, Beijing, China for his Ph.D. degree. He is currently an associate professor with the School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), Northwestern Polytechnical University, Xi'an, China. His research focuses on machine learning and its applications.

Xuelong Li (Fellow, IEEE) is currently a full professor with the School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), Northwestern Polytechnical University, Xi'an, China.