Multiple Kernel Driven Clustering with Locally Consistent and Selfish Graph in Industrial IoT

***Abstract*—In the cognitive computing of intelligent Industrial Internet of Things (IIoT), clustering is a fundamental machine learning problem to exploit the latent data relationships. To overcome the challenge of kernel choice for non-linear clustering tasks, multiple kernel clustering (MKC) has attracted intensive attention. However, existing graph-based MKC methods mainly aim to learn a consensus kernel as well as an affinity graph from multiple candidate kernels, which cannot fully exploit the latent graph information. In this paper, we propose a novel pure graph-based MKC method. Specifically, a new graph model is proposed to preserve the local manifold structure of the data in kernel space so as to learn multiple candidate graphs. Afterwards, the latent consistency and selfishness of these candidate graphs are fully considered. Furthermore, a graph connectivity constraint is introduced to avoid requiring any post-processing clustering step. Comprehensive experimental results demonstrate the superiority of our method.**

***Index Terms*—Cognitive computing, Industrial Internet-of-Things, graph learning, clustering, multiple kernel clustering.**

I. Introduction

CLUSTERING is used ubiquitously across the smart factories, intelligent machines, networked processes and big data [1], as a fundamental procedure in the analysis of scientific data [2], [3] and cognitive computing [4]. Its goal is to partition unlabeled data points into their own clusters. With the developing of Industry 4.0 or the Industrial Internet-of-Things (IIoT) [5], [6], [7], the unlabeled and non-linear data are getting more and more, so clustering has emerged to be an important learning paradigm to exploit the latent data relationships. Despite remarkable progress in a number of learning methods, how to effectively handle non-linear data is still a challenging problem. The traditional single kernel methods can alleviate this challenge to a certain degree, nevertheless, these methods require the user to select and tune a single pre-defined kernel, therefore have been facing with *the curse of kernel choice*: (1) the most suitable kernel for a specific task is usually challenging to decide; and (2) it is impractical and time-consuming to exhaustively search a suitable kernel from multiple candidate kernels. In this paper, we seamlessly integrate graph-based clustering (GBC) [8], [9] and multiple kernel learning (MKL) [10], [11] to tackle this challenge.

Due to the effectiveness of capturing the complex structure hidden in data, GBC methods have been widely investigated [8], [12], which consist of first constructing an affinity graph based on graphical representations of the relationships among data points, and then applying spectral algorithm (*e.g.*, spectral clustering) or graphtheoretic algorithm (*e.g.*, normalized cut and ratio cut) to accomplish clustering. Obviously, it is crucial to construct a high-quality affinity graph that could accurately capture the intrinsic sample relations. Overall, the mainstream technologies can be typically divided into four main prototypes. The first one is to construct a predefined similarity graph as affinity graph, relying on binary similarity, cosine similarity, or Gaussian kernel similarity [13]. The second one is adaptive neighbors graph learning [8], [12], which builds a graph by assigning a probability for each sample as the neighborhood of another sample. Accordingly, the homogeneous samples have high affinity values, while those heterogeneous samples have low affinity values, hence, the resulting probability is deemed as the affinity between two samples. The third one is based on the data self-expressiveness [14], which reconstructs every data point by a linear combination of all other data points and produces a coefficient matrix that is used to construct an affinity graph. The last one learns a new representation of original data by non-negative matrix factorization (NMF) or concept factorization (CF) [15], and then constructs an affinity graph relying on the above ways. Generally, the graph-based methods are superior to the k-means-based ones [16], [17].

On the other hand, MKL [11] not only can effectively handle non-linear data but also alleviate the curse of kernel choice. Usually, it aims to learn a consensus kernel by weighting multiple candidate kernels in a kernel pool, meanwhile, it has the great potential to fully exploit complementary information between these kernels. Overall, three weight paradigms are widely used: (1) using equally weighted combination of base kernels, *i.e.*, each kernel has the same weight value [18]; (2) using the linearly or non-linearly combination of base kernels [11], [19]; and (3) using the idea of adaptive neighbor to learn a self-weighted consensus kernel [16], [17], *i.e.*, the important kernel should be assigned a large weight, and vice versa.

Based on both GBC and MKL, although the existing multiple kernel clustering (MKC) methods has gained promising results, the existing MKC methods still suffer from the following drawbacks: (1) they always pay more attention to the learning of consensus kernel rather than affinity graph, this violates the fact that the affinity graph is the crucial role of graph-based clustering; significantly, some important graph information of each candidate kernel may be lost, thus impair the final clustering performance greatly; and (2) they require an additional clustering step to produce the final clusters.

To tackle these drawbacks, a novel MKC method, termed *Locally Consistent and Selfish Graph* (LCSG), is proposed in this paper. In summary, its main contributions are three-fold:

- Unlike existing MKC methods, which distractingly learn a consensus kernel and an affinity graph, LCSG concentrates intently on graph learning. Notably, it has three main highlights: (1) a new kernel graph learning model is proposed to preserve local manifold structure of data in kernel space; (2) the objective function considers both

the consistency and selfishness of multiple new graphs, the former exploits the underlying consistent clustering structure between these graphs, and the latter motivates the selfishness of each graph to learn a consensus affinity graph; and (3) theoretically, it is much faster than existing competitors, as without performing matrix inversion.
- LCSG does not need to run an additional clustering algorithm to produce the final cluster labels, since a graph connectivity constraint is imposed to partition the data points naturally into the required number of clusters.
- To the best of our knowledge, the highest clustering performance on nine widely used benchmark datasets is obtained to date reported.

The rest of article is organized as follows. Section II introduces related works. In Section III, we propose the LCSG method. The solver, computational complexity, and convergence of the optimum problem are provided in Section IV. In subsequent Section V, adequate experimentation and analysis are presented. The conclusion is founded in Section VI.

II. Related work

In recent years, MKC has rapidly developed and produced several state-of-the-art methods [18], [19], [20], [21], [17], which typically work as follows: (1) predefining multiple kernel matrices over the given kernel pool, (2) learning both a consensus kernel and an affinity graph, (3) performing spectral clustering on the affinity graph, and (4) producing the discrete clustering results by some postprocessings like k-means. For instance, affinity aggregation for spectral clustering (AASC) [22], multiple kernel k-means (MKKM) [23], robust multiple kernel k-means (RMKKM) [18], spectral clustering with multiple kernels (SCMK) [24], and neighbor-kernel-based MKL (NKBM) [19] seek for the optimal (convex) linear combination of the given multiple kernels to build an integrated kernel. Based on MKKM, multiview clustering via late fusion alignment maximization (MVCLFA) [25] proposes to maximally align the consensus partition with the weighted base partitions, which can significantly reduce the computational complexity. Unlike the above methods, self-weighted multiple kernel learning (SMKL) [16], low-rank kernel learning graph-based clustering (LKGr) [20], sparse kernel learning graph-based clustering (LKGs) [20], local structural graph and low-rank consensus MKL (LLMKL) [17], and robust multiple kernel subspace clustering (JMKSC) [21] use a self-weighted strategy to learn an optimal consensus kernel, based on the assumption that the consensus kernel is a neighbor of all candidate kernels and the important kernels should receive relatively large weights, and vice versa. Amongst them, MKKM, RMKKM [18] and NKBM are k-means-based methods, which usually focus on how to reduce redundancy and enhance the diversity between selected kernels to learn a linear weighted kernel, and then perform k-means to obtain clusters; while others are graph-based methods, which usually aim to learn a consensus kernel as well as an affinity graph resorting to the extra prior knowledge, and then perform graph clustering to obtain clusters.

III. Methodology

A. Notations

Throughout the paper, matrices and vectors are denoted as boldface capital letters and boldface lowercase letters, respectively. For an arbitrary matrix $\boldsymbol{Q}$, q_{ij} denotes its (i,j)-th entry, and $\boldsymbol{q}_i$ denotes its i-th column. Moreover, $\mathrm{Tr}(\boldsymbol{G})$, $\mathrm{rank}(\boldsymbol{G})$, $\|\boldsymbol{G}\|_F^2$, and $\|\boldsymbol{G}\|_*$ denote the trace operator, rank function, Frobenius-norm, and nuclear-norm of matrix $\boldsymbol{G}$, respectively; $\mathbf{1}$ is vector of all ones with compatible size. $\boldsymbol{I}$ indicates identity matrix with compatible size. The scalars n, c, and m are the numbers of samples, clusters, and candidate kernels, respectively.

B. Locally Manifold Kernel Graph (LMKG)

Recent studies on spectral graph theory [26], [13] and manifold learning theory [27] have demonstrated that the local manifold structure can be effectively captured over a Euclidean distance based nearest neighbor graph. It is generally formulated as follows:

$$\min_{\boldsymbol{G}} \sum_{i,j=1}^{n} \left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 g_{ij} + \alpha g_{ij}^2\right) \; s.t. \; \boldsymbol{g}_i^T \mathbf{1} = 1, \boldsymbol{g}_i \geq 0 \quad (1)$$

where α is a tradeoff parameter, g_{ij} characterizes the similarity between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and the constraints, $\boldsymbol{g}_i^T \mathbf{1} = 1$, $\boldsymbol{g}_i \geq 0$, are used to guarantee the probability property of $\boldsymbol{g}_i$.

However, problem (1) cannot effectively handle non-linear data. To preserve the local manifold structure in kernel space, one may think of using $\|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\|_2^2$ instead of $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$ intuitively, where ϕ is a mapping from the input space to the reproducing kernel Hilbert space; nevertheless, it is difficult to solve that. Based on kernel trick, we propose a new model to learn a locally manifold kernel graph (LMKC) as follows:

$$\min_{\boldsymbol{G}} \sum_{i,j=1}^{n} \left(-ker(\boldsymbol{x}_i, \boldsymbol{x}_j) g_{ij} + \alpha g_{ij}^2\right) \; s.t. \; \boldsymbol{g}_i^T \mathbf{1} = 1, \boldsymbol{g}_i \geq 0 \quad (2)$$

where $ker : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a kernel function. Based on the fact that if $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_j$ in kernel space, the term $ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$ will has a higher value, and the extra minus will lead to a smaller value. Therefore, $-ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$ can be used to measure the similarity between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in Hilbert space. Mathematically, the (i,j)-th entry of the kernel Gram matrix $\boldsymbol{K}, k_{ij}$, is defined as $ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$, so problem (2) can be transformed into

$$\min_{\boldsymbol{G}} -\mathrm{Tr}(\boldsymbol{K}\boldsymbol{G}) + \alpha \|\boldsymbol{G}\|_F^2 \quad s.t. \; \boldsymbol{G} \geq 0, \boldsymbol{G}^T \mathbf{1} = \mathbf{1} \quad (3)$$

where $\mathrm{Tr}(*)$ is the trace operation. Note that $\alpha \geq 0$ can tune the graph structure according to the following Proposition 1.

Proposition 1. *By tuning parameter α, a trade-off between two extreme graph structures can be obtained:*

(1) A sparse graph that one vertex is linked with only one other vertex.

(2) A complete graph that all vertices are linked with each other vertices by the same edge weight $\frac{1}{n}$.

Proof. First, we have the following problem when $\alpha \to 0$.

$$\max_{\boldsymbol{g}_i} \boldsymbol{k}_i^T \boldsymbol{g}_i \quad s.t. \; \boldsymbol{g}_i \geq 0, \boldsymbol{g}_i^T \mathbf{1} = 1 \quad (4)$$

which returns a maximum value $g_{ij} = max(\boldsymbol{k}_i)$, hence the j-th entry of $\boldsymbol{g}_i$ is assigned to one and others are zeros, *i.e.*, in sparse graph $\boldsymbol{G}$, the j-th vertex is only linked to only one other the i-th vertex with the edge weight of $g^*_{ij} = 1$. Second, we have the following problem when $\alpha \to \infty$.

$$\min_{\boldsymbol{g}_i} \boldsymbol{g}_i^T \boldsymbol{g}_i \quad s.t.\ \boldsymbol{g}_i \geq 0, \boldsymbol{g}_i^T \mathbf{1} = 1 \tag{5}$$

whose solution is $g^*_{ij} = \frac{1}{n}$, *i.e.*, in complete graph $\boldsymbol{G}$, the j-th vertex is linked with all other vertices with the edge weights of $\frac{1}{n}$. Thus, α can tune the graph structure of graph $\boldsymbol{G}$. □

C. Multiple Kernel Clustering Using Locally Consistent and Selfish Graph

In a multiple kernel clustering setting, a kernel pool with multiple kernels, $\{\boldsymbol{K}_i\}_{i=1}^m$, is predefined. Consequently, m LMKGs, $\{\boldsymbol{G}_i\}_{i=1}^m$, can then be achieved according to Eq. (3). In this paper, we design a pure graph learning paradigm to intently learn an affinity graph based on the following two intuitive assumptions. (1) **Consistency**: any pair of LMKGs trust each other and admit the same underlying consistent clustering structure; (2) **Selfishness:** the optimal consensus affinity graph can be elected by all LMKGs, a group of meaningful reward values to measure the efficiency of each LMKG. Formally, the proposed objective function is as below:

$$\begin{aligned} &\min_{\boldsymbol{G}_i, \boldsymbol{A}, \boldsymbol{w}} \underbrace{\sum_{i=1}^m -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2}_{\text{Locally manifold kernel graph learning}} \\ &+ \underbrace{\gamma \sum_{i=1}^m \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2}_{\text{Consistency term}} + \underbrace{\beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2}_{\text{Selfishness term}} \\ &s.t.\ \boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}, \boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}, \\ &\quad rank(\boldsymbol{L}_A) = n - c, 0 \leq w_i \leq 1, \boldsymbol{w}^T \mathbf{1} = 1 \end{aligned} \tag{6}$$

where α, β, and γ are tradeoff parameters, the i-th entry of $\boldsymbol{w} = \{w_1, \cdots, w_m\}$ is the reward value of the i-th LMKG according to its efficiency, $\boldsymbol{A}$ is the expected consensus affinity graph, $\boldsymbol{L}_A = \boldsymbol{D}_A - 0.5(\boldsymbol{A}^T + \boldsymbol{A})$ and $\boldsymbol{D}_A$ (with the i-th diagonal entry $d_{ii} = \sum_j^n 0.5(a_{ij} + a_{ji})$) are the Laplacian matrix and degree matrix of matrix $\boldsymbol{A}$, respectively.

In problem (6), the fist term is the LMKG learning term, which learns m LMKGs from multiple candidate kernels and captures the underlying locally manifold structure of each candidate kernel. The second term is the consistency term, which enforces to exploit the underlying consistent clustering structure between all the LMKGs. The third term is the selfishness term, which encourages each LMKG to selfishly obtain different reward according to its efficiency, so as to learn a consensus affinity graph used for spectral clustering. The nonnegative affine constraint, $\boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}$, akin to $\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}$, is used to guarantee the probability property of $\boldsymbol{A}$. According to graph theory, if the graph connectivity constraint, $rank(\boldsymbol{L}_A) = n - c$, is satisfied, the graph $\boldsymbol{A}$ has exact c strongly connected subgraphs [8], by which way, the ideal neighbors assignment with clear clustering structure can be achieved directly. $0 \leq w_i \leq 1, \boldsymbol{w}^T \mathbf{1} = 1$ is used to control the scale of $\boldsymbol{w}$.

As a result, the three terms in problem (6) jointly tackle the first drawback (*i.e.*, distractible graph learning), meanwhile, the connectivity constraint tackles the second drawback (*i.e.*, post-processing clustering burden).

IV. Optimization

A. Solver of LCSG

The solver iteratively updates one variable at a time by fixing the others. The solutions of the subproblems are as follows:

(1) $\boldsymbol{G}_i$-subproblem: With other variables fixed, $\boldsymbol{G}_i$ could be solved by the following problem:

$$\begin{aligned} &\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} \sum_{i=1}^m -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &+ \gamma \sum_{i=1}^m \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \end{aligned} \tag{7}$$

Note that problem (7) is independent for different i, so we can solve the following problem separately for each i, namely

$$\begin{aligned} &\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &+ \gamma \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \end{aligned} \tag{8}$$

Solving the problem above without constraints yields

$$\boldsymbol{G}_i^* = \frac{\boldsymbol{K}_i + 2\beta w_i \boldsymbol{A} + \gamma \sum_{j=1, j\neq i}^m \boldsymbol{G}_j}{2\alpha + 2\beta w_i + \gamma(m-1)} \tag{9}$$

Afterwards, analogously to [28] in virtue of a two-step fast approximation strategy, the problem *w.r.t.* $\boldsymbol{G}_i$ can then be approximate to

$$\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} \|\boldsymbol{G}_i - \boldsymbol{G}_i^*\|_F^2 \tag{10}$$

which needs to compute the Euclidean projection of a point onto the capped simplex, it can be effectively solved by a valid iterative algorithm proposed in [8].

(2) $\boldsymbol{A}$-subproblem: Since $\boldsymbol{L}_A$ is positive semidefinite, its p-th smallest eigenvalue is denoted as $\sigma_p(\boldsymbol{L}_A)$ and satisfied $\sigma_p(\boldsymbol{L}_A) \geq 0$. Theoretically, $rank(\boldsymbol{L}_A) = n - c$ indicates $\sum_{p=1}^c \sigma_p(\boldsymbol{L}_A) = 0$. According to Ky Fan's theory, this graph connectivity constraint can be rewritten as

$$\min_{\boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I}} \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) = \sum_{k,l=1}^n \|\boldsymbol{h}_k - \boldsymbol{h}_l\|_2^2 a_{kl} \tag{11}$$

where $\boldsymbol{H} = \{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_c\} \in \mathbb{R}^{n \times c}$ is the embedding matrix. By dropping other irrelevant variables and introducing a large enough value of λ, $\boldsymbol{A}$ and $\boldsymbol{H}$ are involved into

$$\begin{aligned} &\min_{\boldsymbol{A}, \boldsymbol{H}} \beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \\ &s.t.\ \boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}, \boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I} \end{aligned} \tag{12}$$

This problem can be solved by updating $\boldsymbol{A}$ and $\boldsymbol{H}$ alternately.

(i) Solving $\boldsymbol{A}$ when $\boldsymbol{H}$ is fixed, letting $s_{kl} = \|\boldsymbol{h}_k - \boldsymbol{h}_l\|_2^2$ and $g_{kl}^i = (\boldsymbol{G}_i)_{kl}$ for ease of notation, Eq. (12) turns to be

$$\begin{aligned} \min_{\boldsymbol{A}} \sum_{i=1}^{m} \sum_{k,l=1}^{n} w_i (a_{kl} - g_{kl}^i)^2 + \frac{\lambda}{\beta} \sum_{k,l=1}^{n} s_{kl} a_{kl} \\ s.t.\ \forall k, \boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1 \end{aligned} \tag{13}$$

Problem (13) can be separated into a set of smaller independent problems for each k, *i.e.*,

$$\min_{\boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1} \sum_{i=1}^{m} w_i \left\| \boldsymbol{a}_k - \boldsymbol{g}_k^i \right\|_2^2 + \frac{\lambda}{\beta} \boldsymbol{s}_k^T \boldsymbol{a}_k \tag{14}$$

This problem is equivalent to solve the following problem:

$$\min_{\boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1} \left\| \boldsymbol{a}_k - \frac{1}{m} \sum_{i=1}^{m} \left(\boldsymbol{g}_k^i - \frac{\lambda}{2m\beta w_i} \boldsymbol{s}_k \right) \right\|_2^2 \tag{15}$$

which can be solved just like problem (10).

(ii) Solving $\boldsymbol{H}$ when $\boldsymbol{A}$ is fixed, Eq. (12) degrades into

$$\min_{\boldsymbol{H} \in \mathbb{R}^{n \times c}, \boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I}} \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \tag{16}$$

whose solution is formed by the c eigenvectors of $\boldsymbol{L}_A$ corresponding to its c smallest eigenvalues.

(3) $\boldsymbol{w}$-subproblem: $\boldsymbol{w}$ is updated according to Proposition 2.

Proposition 2. *The reward of the i-th LMKG is determined by normalized $w_i = \frac{1}{2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F + \zeta}$, where ζ is infinitely close to zero.*

Proof. Motivated by [29], we define an auxiliary problem without $\boldsymbol{w}$ as follows:

$$\begin{aligned} \min_{\boldsymbol{A}} \sum_{i=1}^{m} \sqrt{\|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2} + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \\ s.t.\ a_{ij} \geq 0, \boldsymbol{a}_i^T \mathbf{1} = 1 \end{aligned} \tag{17}$$

whose Lagrange function is $\sum_{i=1}^{m} \sqrt{\|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2} + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) + \Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$, where $\boldsymbol{\Lambda}$ is Lagrange multiplier, and $\Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$ indicates the indicator function of $\boldsymbol{A}$ from the constraints. Taking the derivative of the Lagrange function *w.r.t.* $\boldsymbol{A}$ and setting the derivative to zero, we have

$$\sum_{i=1}^{m} \widehat{w}_i \frac{\partial \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2}{\partial \boldsymbol{A}} + \frac{\partial \Omega(\boldsymbol{A})}{\partial \boldsymbol{A}} = 0 \tag{18}$$

where $\Omega(\boldsymbol{A}) = \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) + \Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$ and $\widehat{w}_i = 1/(2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F)$. Obviously, Eq. (18) is the same as the derivation of the Lagrange function of problem (12). Thus, $\widehat{w}_i$ can be considered as the w_i in (12). To avoid dividing by zero in theory, $\widehat{w}_i$ can be transformed into

$$w_i = \frac{1}{2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F + \zeta} \tag{19}$$

where ζ is infinitely close to zero. □

Note that the convergence criterion is $rank(\boldsymbol{L}_A) = n - c$, thus parameter λ should be automatically increased or decreased when the number of connected subgraphs of graph $\boldsymbol{A}$ is smaller or greater than c during the iteration. The pseudocode of our LCSG is depicted in Algorithm 1.

Algorithm 1 The algorithm of LCSG

Input: Multiple kernels $\{\boldsymbol{K}_i\}_{i=1}^m$, parameters α, β, and γ.

1: Initialize $w_i = 1/m$ for each graph, and $\lambda = 10^{-5}$;
2: **repeat**
3: Update each LMKG $\boldsymbol{G}_i$ by problem (10);
4: Update the consensus graph $\boldsymbol{A}$ by problem (15);
5: Update the embedding matrix $\boldsymbol{H}$ by problem (16);
6: Update the weight vector $\boldsymbol{w}$ by problem (19);
7: **until** $rank(\boldsymbol{L}_A) = n - c$ is satisfied;
8: Use graphconncomp function to find the strongly connected components of graph $\boldsymbol{A}$.

Output: Clustering results.

B. Computational Complexity Analysis

In Algorithm 1, the computational complexity of updating $\{\boldsymbol{G}_i\}_{i=1}^m$, $\boldsymbol{A}$, $\boldsymbol{H}$ and $\boldsymbol{w}$ are $\mathcal{O}(mn^2)$, $\mathcal{O}(n^2)$, $\mathcal{O}(cn^2)$ and $\mathcal{O}(mn^2)$, respectively. Hence, the computational complexity of our LCSG is only $\mathcal{O}(n^2)$ in each iteration. while that of other graph-based MKC methods are at least $\mathcal{O}(n^3)$. The main reason is that the existing graph-based MKC methods always involve the matrix reverse operator, where the computational complexity of the matrix reverse operator is $\mathcal{O}(n^3)$.

C. Convergence Analysis

Objective function (6) is convex *w.r.t.* one variable while fixing the others. For each subproblem, it is convex minimization problem and has optimal solution. Thus, by solving these subproblems alternatively, our algorithm will reduce the objective function monotonically. Moreover, we prove that the whole function is lower bounded in virtue of Proposition 3. Thus, the convergence of our algorithm can be guaranteed.

Proposition 3. *Objective function* (6) *is lower bounded.*

Proof. Objective function (6) can be divided into two parts (*i.e.*, $\boldsymbol{\Theta}_1$ and $\boldsymbol{\Theta}_2$). First, the lower bound of $\boldsymbol{\Theta}_1$ is given by

$$\begin{aligned} \boldsymbol{\Theta}_1 &= -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &= \alpha \|\boldsymbol{G}_i\|_F^2 - \langle \boldsymbol{K}_i^T, \boldsymbol{G}_i \rangle + \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 - \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \\ &= \|\sqrt{\alpha} \boldsymbol{G}_i - \frac{1}{2\sqrt{\alpha}} \boldsymbol{K}_i\|_F^2 - \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \geq -\frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \end{aligned} \tag{20}$$

Second, the lower bound of $\boldsymbol{\Theta}_2$ is given by

$$\boldsymbol{\Theta}_2 = \gamma \sum_{i=1}^{m} \sum_{j=1, j \neq i}^{m} \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta \sum_{i=1}^{m} w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \geq 0 \tag{21}$$

Hence, the whole function, $\boldsymbol{\Theta} = \sum \boldsymbol{\Theta}_1 + \boldsymbol{\Theta}_2$, is lower bounded as $\boldsymbol{\Theta} \geq -\frac{1}{4\alpha} \sum_{i=1}^{m} \|\boldsymbol{K}_i\|_F^2$. □

V. Experiments

We demonstrate the effectiveness of our LCSG by conducting several experiments on nine public benchmark datasets.

A. Datasets and Kernel Pool

Following [18], [21], we employ nine widely used benchmark datasets, including six image datasets (*i.e.*, Yale, Jaffe, ORL, AR, COIL20, and BA) and three text corporas (*i.e.*, TR11, TR41, and TR45). These datasets can stand for the complex IIoT non-linear data for evaluating the performance of the proposed method. The statistics of these datasets are briefly summarized in Table I.

TABLE I: Statistics of the nine benchmark datasets

Dataset	# Classes	# Samples	# Features
Yale	15	165	1024
Jaffe	10	213	676
AR	120	840	768
ORL	40	400	1024
COIL-20	20	1440	1024
BA	36	1404	320
TR11	9	414	6429
TR41	10	878	7454
TR45	10	690	8261

In the same way as [18], a kernel pool is built in advance, which consists of 12 candidate kernels (*i.e.*, $m = 12$): a cosine kernel $k_{ij} = (\boldsymbol{x}_i^T \boldsymbol{x}_j)/(\|\boldsymbol{x}_i\|_2^2 \cdot \|\boldsymbol{x}_j\|_2^2)$; four polynomial kernels $k_{ij} = (u + \boldsymbol{x}_i^T \boldsymbol{x}_j)^v$ where u varies from $\{0, 1\}$ and v varies from $\{2, 4\}$; and seven radial basis function (RBF) kernels $k_{ij} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2/(2\theta\tau^2))$, where θ varies from the set of $\{0.01, 0.05, 0.1, 1, 10, 50, 100\}$ and τ is the maximum distance between any two samples. All the kernels are normalized to $[0, 1]$ by $k_{ij} = k_{ij}/\sqrt{k_{ii}k_{jj}}$.

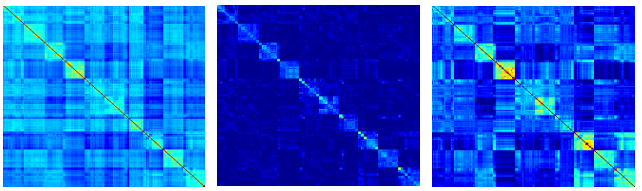
B. Competitors and Evaluating Metric

We compare the proposed LCSG method with the following state-of-the-art competitors: MKKM [23], RMKKM [18], AASC [22], SCMK [24], LKGr [20], SMKL [16], JMKSC [21], and MVCLFA [25]. Amongst these methods, MKKM, RMKKM, and MVCLFA are k-means-based methods, while others are graph-based methods. For MVCLFA, we take the kernels as views and fed into it. For fair comparison, the involved parameters of these competitors have been carefully tuned as recommended by their respective authors. To quantitatively investigate the clustering performance, three widely used metrics, clustering accuracy (ACC), normalized mutual information (NMI), and purity, are applied here. For the three metrics, the higher values indicate the better performance. Meanwhile, to alleviate the instability caused by k-means in spectral clustering, we independently repeat each experiment 20 times.

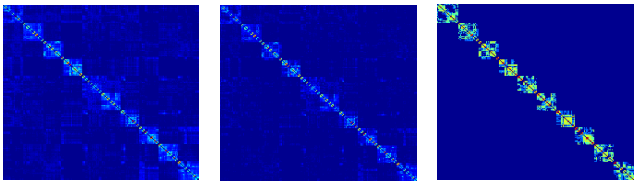
C. Performance Evaluation

The clustering results are presented in Tables II, III and IV. It can clearly be seen that our LCSG consistently obtains the best performance, and the improvements are significant in most case. Surprisingly, our LCSG improves by 8.0%, 5.4%, and 6.0%, respectively, compared to JMKSC (the best competitor) in terms of ACC, NMI, and purity. Note here that owing to the introduced graph connectivity constraint (*i.e.*, $\text{rank}(\boldsymbol{L}_A) = n - c$), our LCSG yields a standard deviation of zero in every case. These results indicate the higher effectiveness of our pure graph learning than the existing non-graph learning and distractible graph learning for MKC tasks.

Furthermore, to evaluate the quality of the learned consensus affinity matrix (also known as affinity graph) $\boldsymbol{A}$, we illustrate $\boldsymbol{A}$ produced by the comparison methods on the Jaffe dataset by using a visual assessment similar to [30]. The results are shown in Fig. 1. Obviously, the matrix $\boldsymbol{A}$ of our LCSG has better block diagonal property and inter-cluster separability than the competitors. Thanks to the introduced graph connectivity constraint (*i.e.*, $\text{rank}(\boldsymbol{L}_A) = n - c$), the learned graph $\boldsymbol{A}$ can be exactly partitioned into c strongly connected subgraphs by automatically tuning λ. What's more, the phenomenon that all the standard deviations of our LCSG (presented in Tables II, III and IV) are zeros is consistent with the above graph theory.



(a) AASC [22] (b) SCMK [24] (c) LKGr [20]



(d) SMKL [16] (e) JMKSC [21] (f) Our LCSG

Fig. 1: Visualization of the learned affinity graph $\boldsymbol{A}$ on the Jaffe dataset. The Jaffe dataset consists of 10 clusters. Note that the darker the blue color, the value is closer to zero. (Zoom in for best view).

D. Parameter Sensitivity

In the proposed LCSG method, there are three parameters, α, γ and β, needed to be tuned. Take the Yale and ORL datasets, for example. By fixing $\alpha = 1$ and using a grid search strategy, the searching regions of β and γ are selected from $\{10^{-4}, \cdots, 10^{1}\}$ and $\{10^{-3}, \cdots, 10^{2}\}$, respectively. We then show the parameter sensitivities *w.r.t.* β and γ in Fig. 2. Subsequently, by fixing $\beta = 0.1$, $\gamma = 10$ and $\beta = 0.01$, $\gamma = 100$ for the Yale and ORL datasets, respectively, α is tuned from the range of $\{10^{-8}, \cdots, 10^{8}\}$. We then show the parameter sensitivities *w.r.t.* α in Fig. 3. Overall, satisfactory performance is obtained over a large range of parameter values for all datasets. For simplicity, we fix $\alpha = 1$ in all experiments, one can tune it for better performance.

E. Convergence

Theoretically, the convergence of our LCSG can be guaranteed (see Section IV-C). Experimentally, we evaluate the convergence of our LCSG on the Yale and ORL datasets. Note here that the convergence criterion is $\text{rank}(\boldsymbol{L}_A) = n - c$,

TABLE II: Clustering performance comparison (average ± standard deviation) in term of ACC.

Dataset	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.457±0.041	0.521±0.034	0.406±0.027	0.582±0.025	0.540±0.030	0.582±0.017	0.630±0.006	0.618±0.011	**0.661±0.000**
ORL	0.475±0.023	0.556±0.024	0.272±0.009	0.656±0.015	0.616±0.016	0.573±0.032	0.725±0.014	0.692±0.005	**0.810±0.000**
Jaffe	0.746±0.069	0.871±0.053	0.304±0.008	0.869±0.022	0.861±0.052	0.967±0.000	0.967±0.007	0.981±0.005	**1.000±0.000**
AR	0.286±0.014	0.344±0.012	0.332±0.006	0.544±0.024	0.314±0.015	0.263±0.009	0.609±0.007	0.667±0.008	**0.779±0.000**
BA	0.405±0.019	0.434±0.018	0.271±0.003	0.384±0.014	0.444±0.018	0.246±0.012	0.484±0.015	0.413±0.005	**0.523±0.000**
COIL	0.548±0.058	0.667±0.028	0.349±0.050	0.591±0.028	0.618±0.051	0.487±0.031	0.696±0.016	0.664±0.013	**0.863±0.000**
TR11	0.501±0.048	0.577±0.094	0.472±0.008	0.549±0.015	0.607±0.043	0.708±0.033	0.737±0.002	0.572±0.026	**0.756±0.000**
TR41	0.561±0.068	0.627±0.073	0.459±0.001	0.650±0.068	0.595±0.020	0.671±0.002	0.689±0.004	0.594±0.005	**0.788±0.000**
TR45	0.585±0.066	0.640±0.071	0.526±0.008	0.634±0.058	0.663±0.042	0.671±0.004	0.687±0.036	0.721±0.002	**0.778±0.000**

TABLE III: Clustering performance comparison (average ± standard deviation) in term of NMI.

Dataset	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.501±0.036	0.556±0.025	0.468±0.028	0.576±0.012	0.566±0.025	0.614±0.015	0.631±0.006	0.609±0.009	**0.643±0.000**
ORL	0.689±0.016	0.748±0.018	0.438±0.007	0.808±0.008	0.794±0.008	0.733±0.027	0.852±0.012	0.836±0.003	**0.889±0.000**
Jaffe	0.798±0.058	0.893±0.041	0.272±0.006	0.868±0.021	0.869±0.031	0.951±0.000	0.952±0.010	0.970±0.008	**1.000±0.000**
AR	0.592±0.014	0.655±0.015	0.651±0.005	0.775±0.009	0.648±0.007	0.568±0.014	0.820±0.002	0.844±0.002	**0.894±0.000**
BA	0.569±0.008	0.585±0.011	0.423±0.004	0.544±0.012	0.604±0.009	0.486±0.011	0.621±0.007	0.556±0.002	**0.666±0.000**
COIL	0.707±0.033	0.773±0.017	0.419±0.027	0.726±0.011	0.766±0.023	0.628±0.018	0.818±0.007	0.782±0.005	**0.928±0.000**
TR11	0.446±0.046	0.561±0.118	0.394±0.003	0.371±0.018	0.597±0.031	0.557±0.068	0.673±0.002	0.582±0.012	**0.683±0.000**
TR41	0.578±0.042	0.635±0.092	0.431±0.000	0.492±0.017	0.604±0.023	0.625±0.004	0.660±0.003	0.575±0.006	**0.729±0.000**
TR45	0.562±0.056	0.627±0.092	0.420±0.014	0.584±0.051	0.671±0.020	0.622±0.007	0.690±0.022	0.681±0.001	**0.772±0.000**

TABLE IV: Clustering performance comparison (average ± standard deviation) in term of Purity.

Data	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.475±0.037	0.536±0.031	0.423±0.026	0.610±0.014	0.554±0.029	0.667±0.014	0.673±0.007	0.624±0.010	**0.703±0.000**
ORL	0.514±0.021	0.602±0.024	0.316±0.007	0.699±0.015	0.658±0.017	0.648±0.017	0.753±0.012	0.732±0.004	**0.830±0.000**
Jaffe	0.768±0.062	0.889±0.045	0.331±0.008	0.882±0.023	0.859±0.038	0.967±0.000	0.967±0.007	0.981±0.005	**1.000±0.000**
AR	0.305±0.012	0.368±0.010	0.350±0.006	0.642±0.014	0.330±0.014	0.530±0.014	0.656±0.010	0.685±0.003	**0.805±0.000**
BA	0.435±0.014	0.463±0.015	0.303±0.004	0.606±0.009	0.479±0.017	0.623±0.011	0.563±0.018	0.438±0.006	**0.646±0.000**
COIL	0.590±0.053	0.699±0.022	0.391±0.044	0.635±0.013	0.650±0.039	0.683±0.004	0.806±0.010	0.690±0.013	**0.913±0.000**
TR11	0.655±0.044	0.729±0.096	0.547±0.000	0.783±0.011	0.776±0.030	**0.835±0.048**	0.819±0.001	0.768±0.009	0.787±0.000
TR41	0.728±0.042	0.776±0.065	0.621±0.001	0.758±0.034	0.759±0.031	0.761±0.003	0.799±0.003	0.757±0.008	**0.833±0.000**
TR45	0.691±0.058	0.752±0.074	0.575±0.011	0.728±0.048	0.800±0.026	0.816±0.004	0.822±0.031	0.806±0.001	**0.883±0.000**

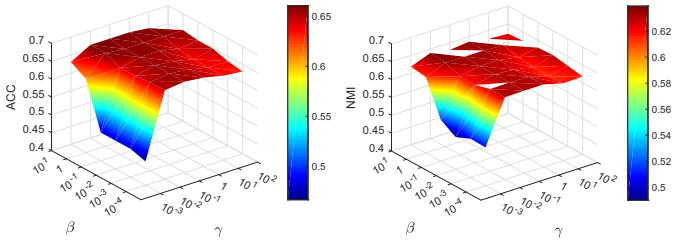
i.e., the connected subgraphs of the learned graph $\boldsymbol{A}$ is equal to c, so we need to continuously self-tune parameter λ until the desired graph is obtained. The results presented in Fig. 4 suggest that the objective value is monotonically decreased, and the clustering performance is gradually improving. Notably, although the algorithm seems to converge after only 5 iterations for the ORL dataset, the additional iterations are also need to meet the convergence criterion, *i.e.*, $\text{rank}(\boldsymbol{L}_A) = n-c$. Usually, LCSG converges in less than 20 iterations for all evaluated datasets.

F. Running Time

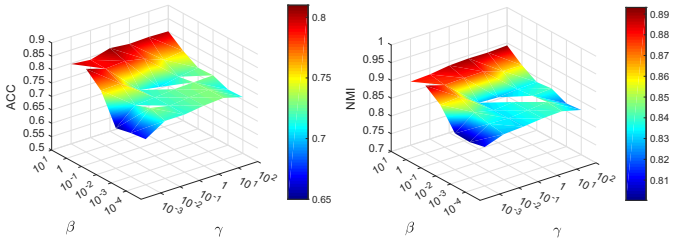
We compare the running time (in seconds) of all competitors on the Yale, ORL, TR11, and TR45 datasets. All codes are implemented in MATLAB 2016b and run on a Mac PC with a 3.2 GHz Intel Core i7 processor, 16-GB RAM, and macOS Mojave operating system. The mean and standard deviations of 20 trials are reported in Table V, the proposed LCSG has a competitive superiority on running time. Although the running time of MKKM and MVCLFA is lower than that of our LCSG, their clustering performance is worse than that of ours.

VI. Conclusion

In this paper, we have proposed a pure graph-based MKC method to address the changeling non-linear clustering issues for cognitive computing of intelligent IIoT. Specifically, a new graph model, termed as LMKG, that can preserve the local manifold structure of data in kernel space is introduced to learn

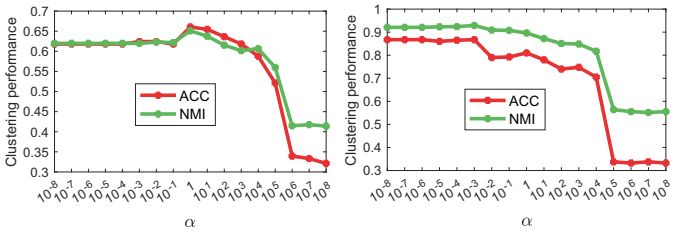


(a) ACC (b) NMI



(c) ACC (d) NMI

Fig. 2: ACC and NMI of our LCSG *w.r.t.* β and γ on the Yale (the fist row) and ORL (the second row) datasets. α is fixed to 1. (Zoom in for best view).



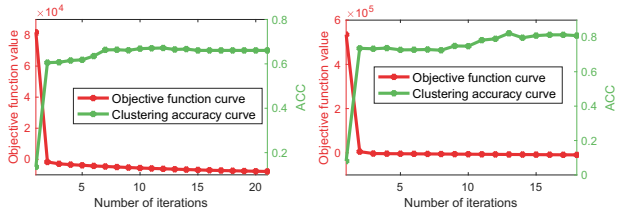
(a) Yale ($\beta = 0.1$, $\gamma = 10$) (b) ORL ($\beta = 0.01$, $\gamma = 100$)

Fig. 3: ACC and NMI of our LCSG *w.r.t.* α on the Yale and ORL datasets.

multiple LMKGs from multiple candidate kernels. By considering both the consistency and selfishness of these LMKGs, the quality of affinity graph achieves significant improvement. Further, the graph connectivity constraint avoids requiring any post-processing step such that the clustering results can be immediately obtained. Comprehensive experimental results clearly demonstrates the superiority of our method. Therefore, our LCSG method can be used to effectively handle the non-

TABLE V: Computational time (in seconds) comparison.

Method	Yale	ORL	TR11	TR45
MKKM [23]	0.015±0.001	0.128±0.003	0.059 ±0.002	0.162±0.005
RMKKM [18]	0.870±0.011	3.622±0.085	4.144±0.117	6.908±0.121
AASC [22]	0.221±0.005	0.910±0.007	0.977±0.014	1.686±0.019
SCMK [24]	5.492±0.144	42.040±1.782	51.454±3.102	218.715±5.384
LKGr [20]	1.422±0.015	7.425±0.227	13.718±0.340	80.558±3.211
SMKL [16]	1.439±0.022	12.836±0.565	9.863±0.144	154.683±5.101
JMKSC [21]	1.219±0.015	2.462±0.108	3.974±0.125	8.765±0.183
MVCLFA [25]	0.251±0.004	0.893±0.011	1.116±0.420	2.243±0.047
Our LCSG	0.620±0.023	1.882±0.054	2.121±0.113	5.921±0.121



(a) Yale (b) ORL

Fig. 4: Convergence curve of our LCSG on the Yale and ORL datasets.

linear data from intelligent IIoT and other industrial sensor networks.

In our future work, it is potentially interesting to extend the proposed method to handle large-scale non-linear data for cognitive computing.

REFERENCES

[1] P. Jia, X. Wang, and K. Zheng, "Distributed clock synchronization based on intelligent clustering in local area industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3697–3707, 2019.

[2] J. Xiao, Y. Tian, L. Xie, X. Jiang, and J. Huang, "A hybrid classification framework based on clustering," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2177–2188, 2020.

[3] H. Hassan, A. K. Bashir, M. Ahmad, V. G. Menon, I. U. Afridi, R. Nawaz, and B. Luo, "Real-time image dehazing by superpixels segmentation and guidance filter," *Journal of Real-Time Image Processing*, pp. 1–21, 2020.

[4] X. Liu and X. Zhang, "Noma-based resource allocation for cluster-based cognitive industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2019.

[5] S. Jacob, V. G. Menon, and S. Joseph, "Depth information enhancement using block matching and image pyramiding stereo vision enabled rgb-d sensor," *IEEE Sensors Journal*, vol. 20, no. 10, pp. 5406–5414, 2020.

[6] T. Wang, H. Luo, W. Jia, A. Liu, and M. Xie, "Mtes: An intelligent trust evaluation scheme in sensor-cloud-enabled industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2054–2062, 2020.

[7] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, "Massive internet of things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 28–33, 2017.

[8] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained laplacian rank algorithm for graph-based clustering," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1969–1976.

[9] K. Zhan, C. Zhang, J. Guan, and J. Wang, "Graph learning for multiview clustering," *IEEE transactions on cybernetics*, vol. 48, no. 10, pp. 2887–2895, 2017.

[10] Z. Y. Ren, S. X., Q. Sun, and T. Wang, "Consensus affinity graph learning for multiple kernel clustering," *IEEE Transactions on Cybernetics*, p. 10.1109/TCYB.2020.3000947, 2020.

[11] X. Liu, X. Zhu, M. Li, L. Wang, E. Zhu, T. Liu, M. Kloft, D. Shen, J. Yin, and W. Gao, "Multiple kernel k-means with incomplete kernels," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 5, pp. 1191–1204, 2020.

[12] Z. Kang, H. Pan, S. C. Hoi, and Z. Xu, "Robust graph learning from noisy data," *IEEE transactions on cybernetics*, vol. 28, no. 4, pp. 1007–1021, 2020.

[13] H. Wang, Y. Yang, B. Liu, and H. Fujita, "A study of graph-based system for multi-view clustering," *Knowledge-Based Systems*, vol. 163, pp. 1009–1019, 2019.

[14] C. Lu, J. Feng, Z. Lin, T. Mei, and S. Yan, "Subspace clustering by block diagonal representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 487–501, 2019.

[15] H. Li, J. Zhang, J. Hu, C. Zhang, and J. Liu, "Graph-based discriminative concept factorization for data representation," *Knowledge-Based Systems*, vol. 118, pp. 70–79, 2017.

[16] Z. Kang, X. Lu, J. Yi, and Z. Xu, "Self-weighted multiple kernel learning for graph-based clustering and semi-supervised classification," *IJCAI*, pp. 2312–2318, 2018.

[17] Z. Ren, H. Li, C. Yang, and Q. Sun, "Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning," *Knowledge-Based Systems*, p. 105040, 2019.

[18] L. Du, P. Zhou, L. Shi, H. Wang, M. Fan, W. Wang, and Y.-D. Shen, "Robust multiple kernel k-means using l21-norm," pp. 3476–3482, 2015.

[19] S. Zhou, X. Liu, M. Li, E. Zhu, L. Liu, C. Zhang, and J. Yin, "Multiple kernel clustering with neighbor-kernel subspace segmentation," *IEEE Transactions on Neural Networks*, pp. 1–12, 2019.

[20] Z. Kang, L. Wen, W. Chen, and Z. Xu, "Low-rank kernel learning for graph-based clustering," *Knowledge-Based Systems*, vol. 163, pp. 510–517, 2019.

[21] C. Yang, Z. Ren, Q. Sun, M. Wu, M. Yin, and Y. Sun, "Joint correntropy metric weighting and block diagonal regularizer for robust multiple kernel subspace clustering," *Information Sciences*, vol. 500, pp. 48–66, 2019.

[22] H.-C. Huang, Y.-Y. Chuang, and C.-S. Chen, "Affinity aggregation for spectral clustering," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 773–780.

[23] H. Huang, Y. Chuang, and C. Chen, "Multiple kernel fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 1, pp. 120–134, 2012.

[24] Z. Kang, C. Peng, Q. Cheng, and Z. Xu, "Unified spectral clustering with optimal graph," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 3366–3373.

[25] S. Wang, X. Liu, E. Zhu, C. Tang, J. Liu, J. Hu, J. Xia, and J. Yin, "Multi-view clustering via late fusion alignment maximization," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 3778–3784.

[26] Z. Lin, F. Wen, Y. Ding, and Y. Xue, "Data-driven coherency identification for generators based on spectral clustering," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1275–1285, 2018.

[27] Z. Ren and Q. Sun, "Simultaneous global and local graph structure preserving for multiple kernel clustering," *IEEE Transactions on Neural Networks and Learning Systems*, p. 10.1109/TNNLS.2020.2991366, 2020.

[28] M. Iliadis, H. Wang, R. Molina, and A. K. Katsaggelos, "Robust and low-rank representation for fast face identification with occlusions," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2203–2218, 2017.

[29] F. Nie, J. Li, X. Li *et al.*, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification." in *IJCAI*, 2016, pp. 1881–1887.

[30] J. C. Bezdek and R. J. Hathaway, "Vat: a tool for visual assessment of (cluster) tendency," vol. 3, pp. 2225–2230, 2002.

Multiple Kernel Driven Clustering with Locally Consistent and Selfish Graph in Industrial IoT

***Abstract*—In the cognitive computing of intelligent Industrial Internet of Things (IIoT), clustering is a fundamental machine learning problem to exploit the latent data relationships. To overcome the challenge of kernel choice for non-linear clustering tasks, multiple kernel clustering (MKC) has attracted intensive attention. However, existing graph-based MKC methods mainly aim to learn a consensus kernel as well as an affinity graph from multiple candidate kernels, which cannot fully exploit the latent graph information. In this paper, we propose a novel pure graph-based MKC method. Specifically, a new graph model is proposed to preserve the local manifold structure of the data in kernel space so as to learn multiple candidate graphs. Afterwards, the latent consistency and selfishness of these candidate graphs are fully considered. Furthermore, a graph connectivity constraint is introduced to avoid requiring any post-processing clustering step. Comprehensive experimental results demonstrate the superiority of our method.**

***Index Terms*—Cognitive computing, Industrial Internet-of-Things, graph learning, clustering, multiple kernel clustering.**

I. Introduction

CLUSTERING is used ubiquitously across the smart factories, intelligent machines, networked processes and big data [1], as a fundamental procedure in the analysis of scientific data [2], [3] and cognitive computing [4]. Its goal is to partition unlabeled data points into their own clusters. With the developing of Industry 4.0 or the Industrial Internet-of-Things (IIoT) [5], [6], [7], the unlabeled and non-linear data are getting more and more, so clustering has emerged to be an important learning paradigm to exploit the latent data relationships. Despite remarkable progress in a number of learning methods, how to effectively handle non-linear data is still a challenging problem. The traditional single kernel methods can alleviate this challenge to a certain degree, nevertheless, these methods require the user to select and tune a single pre-defined kernel, therefore have been facing with *the curse of kernel choice*: (1) the most suitable kernel for a specific task is usually challenging to decide; and (2) it is impractical and time-consuming to exhaustively search a suitable kernel from multiple candidate kernels. In this paper, we seamlessly integrate graph-based clustering (GBC) [8], [9] and multiple kernel learning (MKL) [10], [11] to tackle this challenge.

Due to the effectiveness of capturing the complex structure hidden in data, GBC methods have been widely investigated [8], [12], which consist of first constructing an affinity graph based on graphical representations of the relationships among data points, and then applying spectral algorithm (*e.g.*, spectral clustering) or graphtheoretic algorithm (*e.g.*, normalized cut and ratio cut) to accomplish clustering. Obviously, it is crucial to construct a high-quality affinity graph that could accurately capture the intrinsic sample relations. Overall, the mainstream technologies can be typically divided into four main prototypes. The first one is to construct a predefined similarity graph as affinity graph, relying on binary similarity, cosine similarity, or Gaussian kernel similarity [13]. The second one is adaptive neighbors graph learning [8], [12], which builds a graph by assigning a probability for each sample as the neighborhood of another sample. Accordingly, the homogeneous samples have high affinity values, while those heterogeneous samples have low affinity values, hence, the resulting probability is deemed as the affinity between two samples. The third one is based on the data self-expressiveness [14], which reconstructs every data point by a linear combination of all other data points and produces a coefficient matrix that is used to construct an affinity graph. The last one learns a new representation of original data by non-negative matrix factorization (NMF) or concept factorization (CF) [15], and then constructs an affinity graph relying on the above ways. Generally, the graph-based methods are superior to the k-means-based ones [16], [17].

On the other hand, MKL [11] not only can effectively handle non-linear data but also alleviate the curse of kernel choice. Usually, it aims to learn a consensus kernel by weighting multiple candidate kernels in a kernel pool, meanwhile, it has the great potential to fully exploit complementary information between these kernels. Overall, three weight paradigms are widely used: (1) using equally weighted combination of base kernels, *i.e.*, each kernel has the same weight value [18]; (2) using the linearly or non-linearly combination of base kernels [11], [19]; and (3) using the idea of adaptive neighbor to learn a self-weighted consensus kernel [16], [17], *i.e.*, the important kernel should be assigned a large weight, and vice versa.

Based on both GBC and MKL, although the existing multiple kernel clustering (MKC) methods has gained promising results, the existing MKC methods still suffer from the following drawbacks: (1) they always pay more attention to the learning of consensus kernel rather than affinity graph, this violates the fact that the affinity graph is the crucial role of graph-based clustering; significantly, some important graph information of each candidate kernel may be lost, thus impair the final clustering performance greatly; and (2) they require an additional clustering step to produce the final clusters.

To tackle these drawbacks, a novel MKC method, termed *Locally Consistent and Selfish Graph* (LCSG), is proposed in this paper. In summary, its main contributions are three-fold:

- Unlike existing MKC methods, which distractingly learn a consensus kernel and an affinity graph, LCSG concentrates intently on graph learning. Notably, it has three main highlights: (1) a new kernel graph learning model is proposed to preserve local manifold structure of data in kernel space; (2) the objective function considers both

the consistency and selfishness of multiple new graphs, the former exploits the underlying consistent clustering structure between these graphs, and the latter motivates the selfishness of each graph to learn a consensus affinity graph; and (3) theoretically, it is much faster than existing competitors, as without performing matrix inversion.
- LCSG does not need to run an additional clustering algorithm to produce the final cluster labels, since a graph connectivity constraint is imposed to partition the data points naturally into the required number of clusters.
- To the best of our knowledge, the highest clustering performance on nine widely used benchmark datasets is obtained to date reported.

The rest of article is organized as follows. Section II introduces related works. In Section III, we propose the LCSG method. The solver, computational complexity, and convergence of the optimum problem are provided in Section IV. In subsequent Section V, adequate experimentation and analysis are presented. The conclusion is founded in Section VI.

II. Related work

In recent years, MKC has rapidly developed and produced several state-of-the-art methods [18], [19], [20], [21], [17], which typically work as follows: (1) predefining multiple kernel matrices over the given kernel pool, (2) learning both a consensus kernel and an affinity graph, (3) performing spectral clustering on the affinity graph, and (4) producing the discrete clustering results by some postprocessings like k-means. For instance, affinity aggregation for spectral clustering (AASC) [22], multiple kernel k-means (MKKM) [23], robust multiple kernel k-means (RMKKM) [18], spectral clustering with multiple kernels (SCMK) [24], and neighbor-kernel-based MKL (NKBM) [19] seek for the optimal (convex) linear combination of the given multiple kernels to build an integrated kernel. Based on MKKM, multiview clustering via late fusion alignment maximization (MVCLFA) [25] proposes to maximally align the consensus partition with the weighted base partitions, which can significantly reduce the computational complexity. Unlike the above methods, self-weighted multiple kernel learning (SMKL) [16], low-rank kernel learning graph-based clustering (LKGr) [20], sparse kernel learning graph-based clustering (LKGs) [20], local structural graph and low-rank consensus MKL (LLMKL) [17], and robust multiple kernel subspace clustering (JMKSC) [21] use a self-weighted strategy to learn an optimal consensus kernel, based on the assumption that the consensus kernel is a neighbor of all candidate kernels and the important kernels should receive relatively large weights, and vice versa. Amongst them, MKKM, RMKKM [18] and NKBM are k-means-based methods, which usually focus on how to reduce redundancy and enhance the diversity between selected kernels to learn a linear weighted kernel, and then perform k-means to obtain clusters; while others are graph-based methods, which usually aim to learn a consensus kernel as well as an affinity graph resorting to the extra prior knowledge, and then perform graph clustering to obtain clusters.

III. Methodology

A. Notations

Throughout the paper, matrices and vectors are denoted as boldface capital letters and boldface lowercase letters, respectively. For an arbitrary matrix $\boldsymbol{Q}$, q_{ij} denotes its (i,j)-th entry, and $\boldsymbol{q}_i$ denotes its i-th column. Moreover, $\mathrm{Tr}(\boldsymbol{G})$, $\mathrm{rank}(\boldsymbol{G})$, $\|\boldsymbol{G}\|_F^2$, and $\|\boldsymbol{G}\|_*$ denote the trace operator, rank function, Frobenius-norm, and nuclear-norm of matrix $\boldsymbol{G}$, respectively; $\mathbf{1}$ is vector of all ones with compatible size. $\boldsymbol{I}$ indicates identity matrix with compatible size. The scalars n, c, and m are the numbers of samples, clusters, and candidate kernels, respectively.

B. Locally Manifold Kernel Graph (LMKG)

Recent studies on spectral graph theory [26], [13] and manifold learning theory [27] have demonstrated that the local manifold structure can be effectively captured over a Euclidean distance based nearest neighbor graph. It is generally formulated as follows:

$$\min_{\boldsymbol{G}} \sum_{i,j=1}^{n} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 g_{ij} + \alpha g_{ij}^2) \ s.t.\ \boldsymbol{g}_i^T \mathbf{1} = 1, \boldsymbol{g}_i \geq 0 \quad (1)$$

where α is a tradeoff parameter, g_{ij} characterizes the similarity between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and the constraints, $\boldsymbol{g}_i^T \mathbf{1} = 1$, $\boldsymbol{g}_i \geq 0$, are used to guarantee the probability property of $\boldsymbol{g}_i$.

However, problem (1) cannot effectively handle non-linear data. To preserve the local manifold structure in kernel space, one may think of using $\|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\|_2^2$ instead of $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$ intuitively, where ϕ is a mapping from the input space to the reproducing kernel Hilbert space; nevertheless, it is difficult to solve that. Based on kernel trick, we propose a new model to learn a locally manifold kernel graph (LMKC) as follows:

$$\min_{\boldsymbol{G}} \sum_{i,j=1}^{n} (-ker(\boldsymbol{x}_i, \boldsymbol{x}_j) g_{ij} + \alpha g_{ij}^2) \ s.t.\ \boldsymbol{g}_i^T \mathbf{1} = 1, \boldsymbol{g}_i \geq 0 \quad (2)$$

where $ker : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a kernel function. Based on the fact that if $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_j$ in kernel space, the term $ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$ will has a higher value, and the extra minus will lead to a smaller value. Therefore, $-ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$ can be used to measure the similarity between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in Hilbert space. Mathematically, the (i,j)-th entry of the kernel Gram matrix $\boldsymbol{K}, k_{ij}$, is defined as $ker(\boldsymbol{x}_i, \boldsymbol{x}_j)$, so problem (2) can be transformed into

$$\min_{\boldsymbol{G}} -\mathrm{Tr}(\boldsymbol{K}\boldsymbol{G}) + \alpha \|\boldsymbol{G}\|_F^2 \quad s.t.\ \boldsymbol{G} \geq 0, \boldsymbol{G}^T \mathbf{1} = \mathbf{1} \quad (3)$$

where $\mathrm{Tr}(*)$ is the trace operation. Note that $\alpha \geq 0$ can tune the graph structure according to the following Proposition 1.

Proposition 1. *By tuning parameter α, a trade-off between two extreme graph structures can be obtained:*

(1) A sparse graph that one vertex is linked with only one other vertex.

(2) A complete graph that all vertices are linked with each other vertices by the same edge weight $\frac{1}{n}$.

Proof. First, we have the following problem when $\alpha \to 0$.

$$\max_{\boldsymbol{g}_i} \boldsymbol{k}_i^T \boldsymbol{g}_i \quad s.t.\ \boldsymbol{g}_i \geq 0, \boldsymbol{g}_i^T \mathbf{1} = 1 \quad (4)$$

which returns a maximum value $g_{ij} = max(\boldsymbol{k}_i)$, hence the j-th entry of $\boldsymbol{g}_i$ is assigned to one and others are zeros, *i.e.*, in sparse graph $\boldsymbol{G}$, the j-th vertex is only linked to only one other the i-th vertex with the edge weight of $g^*_{ij} = 1$. Second, we have the following problem when $\alpha \to \infty$.

$$\min_{\boldsymbol{g}_i} \boldsymbol{g}_i^T \boldsymbol{g}_i \quad s.t.\ \boldsymbol{g}_i \geq 0, \boldsymbol{g}_i^T \mathbf{1} = 1 \tag{5}$$

whose solution is $g^*_{ij} = \frac{1}{n}$, *i.e.*, in complete graph $\boldsymbol{G}$, the j-th vertex is linked with all other vertices with the edge weights of $\frac{1}{n}$. Thus, α can tune the graph structure of graph $\boldsymbol{G}$. □

C. Multiple Kernel Clustering Using Locally Consistent and Selfish Graph

In a multiple kernel clustering setting, a kernel pool with multiple kernels, $\{\boldsymbol{K}_i\}_{i=1}^m$, is predefined. Consequently, m LMKGs, $\{\boldsymbol{G}_i\}_{i=1}^m$, can then be achieved according to Eq. (3). In this paper, we design a pure graph learning paradigm to intently learn an affinity graph based on the following two intuitive assumptions. (1) **Consistency**: any pair of LMKGs trust each other and admit the same underlying consistent clustering structure; (2) **Selfishness:** the optimal consensus affinity graph can be elected by all LMKGs, a group of meaningful reward values to measure the efficiency of each LMKG. Formally, the proposed objective function is as below:

$$\begin{aligned} &\min_{\boldsymbol{G}_i, \boldsymbol{A}, \boldsymbol{w}} \underbrace{\sum_{i=1}^m -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2}_{\text{Locally manifold kernel graph learning}} \\ &+ \underbrace{\gamma \sum_{i=1}^m \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2}_{\text{Consistency term}} + \underbrace{\beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2}_{\text{Selfishness term}} \\ &s.t.\ \boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}, \boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}, \\ &\quad rank(\boldsymbol{L}_A) = n - c, 0 \leq w_i \leq 1, \boldsymbol{w}^T \mathbf{1} = 1 \end{aligned} \tag{6}$$

where α, β, and γ are tradeoff parameters, the i-th entry of $\boldsymbol{w} = \{w_1, \cdots, w_m\}$ is the reward value of the i-th LMKG according to its efficiency, $\boldsymbol{A}$ is the expected consensus affinity graph, $\boldsymbol{L}_A = \boldsymbol{D}_A - 0.5(\boldsymbol{A}^T + \boldsymbol{A})$ and $\boldsymbol{D}_A$ (with the i-th diagonal entry $d_{ii} = \sum_j^n 0.5(a_{ij} + a_{ji})$) are the Laplacian matrix and degree matrix of matrix $\boldsymbol{A}$, respectively.

In problem (6), the fist term is the LMKG learning term, which learns m LMKGs from multiple candidate kernels and captures the underlying locally manifold structure of each candidate kernel. The second term is the consistency term, which enforces to exploit the underlying consistent clustering structure between all the LMKGs. The third term is the selfishness term, which encourages each LMKG to selfishly obtain different reward according to its efficiency, so as to learn a consensus affinity graph used for spectral clustering. The nonnegative affine constraint, $\boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}$, akin to $\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}$, is used to guarantee the probability property of $\boldsymbol{A}$. According to graph theory, if the graph connectivity constraint, $rank(\boldsymbol{L}_A) = n - c$, is satisfied, the graph $\boldsymbol{A}$ has exact c strongly connected subgraphs [8], by which way, the ideal neighbors assignment with clear clustering structure can be achieved directly. $0 \leq w_i \leq 1, \boldsymbol{w}^T \mathbf{1} = 1$ is used to control the scale of $\boldsymbol{w}$.

As a result, the three terms in problem (6) jointly tackle the first drawback (*i.e.*, distractible graph learning), meanwhile, the connectivity constraint tackles the second drawback (*i.e.*, post-processing clustering burden).

IV. Optimization

A. Solver of LCSG

The solver iteratively updates one variable at a time by fixing the others. The solutions of the subproblems are as follows:

(1) $\boldsymbol{G}_i$-subproblem: With other variables fixed, $\boldsymbol{G}_i$ could be solved by the following problem:

$$\begin{aligned} &\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} \sum_{i=1}^m -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &+ \gamma \sum_{i=1}^m \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \end{aligned} \tag{7}$$

Note that problem (7) is independent for different i, so we can solve the following problem separately for each i, namely

$$\begin{aligned} &\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &+ \gamma \sum_{j=1, j\neq i}^m \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \end{aligned} \tag{8}$$

Solving the problem above without constraints yields

$$\boldsymbol{G}_i^* = \frac{\boldsymbol{K}_i + 2\beta w_i \boldsymbol{A} + \gamma \sum_{j=1, j\neq i}^m \boldsymbol{G}_j}{2\alpha + 2\beta w_i + \gamma(m-1)} \tag{9}$$

Afterwards, analogously to [28] in virtue of a two-step fast approximation strategy, the problem *w.r.t.* $\boldsymbol{G}_i$ can then be approximate to

$$\min_{\boldsymbol{G}_i \geq 0, \boldsymbol{G}_i^T \mathbf{1} = \mathbf{1}} \|\boldsymbol{G}_i - \boldsymbol{G}_i^*\|_F^2 \tag{10}$$

which needs to compute the Euclidean projection of a point onto the capped simplex, it can be effectively solved by a valid iterative algorithm proposed in [8].

(2) $\boldsymbol{A}$-subproblem: Since $\boldsymbol{L}_A$ is positive semidefinite, its p-th smallest eigenvalue is denoted as $\sigma_p(\boldsymbol{L}_A)$ and satisfied $\sigma_p(\boldsymbol{L}_A) \geq 0$. Theoretically, $rank(\boldsymbol{L}_A) = n - c$ indicates $\sum_{p=1}^c \sigma_p(\boldsymbol{L}_A) = 0$. According to Ky Fan's theory, this graph connectivity constraint can be rewritten as

$$\min_{\boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I}} \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) = \sum_{k,l=1}^n \|\boldsymbol{h}_k - \boldsymbol{h}_l\|_2^2 a_{kl} \tag{11}$$

where $\boldsymbol{H} = \{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_c\} \in \mathbb{R}^{n \times c}$ is the embedding matrix. By dropping other irrelevant variables and introducing a large enough value of λ, $\boldsymbol{A}$ and $\boldsymbol{H}$ are involved into

$$\begin{aligned} &\min_{\boldsymbol{A}, \boldsymbol{H}} \beta \sum_{i=1}^m w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \\ &s.t.\ \boldsymbol{A} \geq 0, \boldsymbol{A}^T \mathbf{1} = \mathbf{1}, \boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I} \end{aligned} \tag{12}$$

This problem can be solved by updating $\boldsymbol{A}$ and $\boldsymbol{H}$ alternately.

(i) Solving $\boldsymbol{A}$ when $\boldsymbol{H}$ is fixed, letting $s_{kl} = \|\boldsymbol{h}_k - \boldsymbol{h}_l\|_2^2$ and $g_{kl}^i = (\boldsymbol{G}_i)_{kl}$ for ease of notation, Eq. (12) turns to be

$$\begin{aligned} \min_{\boldsymbol{A}} \sum_{i=1}^{m} \sum_{k,l=1}^{n} w_i (a_{kl} - g_{kl}^i)^2 + \frac{\lambda}{\beta} \sum_{k,l=1}^{n} s_{kl} a_{kl} \\ s.t.\ \forall k, \boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1 \end{aligned} \tag{13}$$

Problem (13) can be separated into a set of smaller independent problems for each k, *i.e.*,

$$\min_{\boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1} \sum_{i=1}^{m} w_i \left\| \boldsymbol{a}_k - \boldsymbol{g}_k^i \right\|_2^2 + \frac{\lambda}{\beta} \boldsymbol{s}_k^T \boldsymbol{a}_k \tag{14}$$

This problem is equivalent to solve the following problem:

$$\min_{\boldsymbol{a}_k \geq 0, \boldsymbol{a}_k^T \mathbf{1} = 1} \left\| \boldsymbol{a}_k - \frac{1}{m} \sum_{i=1}^{m} \left(\boldsymbol{g}_k^i - \frac{\lambda}{2m\beta w_i} \boldsymbol{s}_k \right) \right\|_2^2 \tag{15}$$

which can be solved just like problem (10).

(ii) Solving $\boldsymbol{H}$ when $\boldsymbol{A}$ is fixed, Eq. (12) degrades into

$$\min_{\boldsymbol{H} \in \mathbb{R}^{n \times c}, \boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I}} \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \tag{16}$$

whose solution is formed by the c eigenvectors of $\boldsymbol{L}_A$ corresponding to its c smallest eigenvalues.

(3) $\boldsymbol{w}$-subproblem: $\boldsymbol{w}$ is updated according to Proposition 2.

Proposition 2. *The reward of the i-th LMKG is determined by normalized $w_i = \frac{1}{2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F + \zeta}$, where ζ is infinitely close to zero.*

Proof. Motivated by [29], we define an auxiliary problem without $\boldsymbol{w}$ as follows:

$$\begin{aligned} \min_{\boldsymbol{A}} \sum_{i=1}^{m} \sqrt{\|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2} + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) \\ s.t.\ a_{ij} \geq 0, \boldsymbol{a}_i^T \mathbf{1} = 1 \end{aligned} \tag{17}$$

whose Lagrange function is $\sum_{i=1}^{m} \sqrt{\|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2} + \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) + \Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$, where $\boldsymbol{\Lambda}$ is Lagrange multiplier, and $\Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$ indicates the indicator function of $\boldsymbol{A}$ from the constraints. Taking the derivative of the Lagrange function *w.r.t.* $\boldsymbol{A}$ and setting the derivative to zero, we have

$$\sum_{i=1}^{m} \widehat{w}_i \frac{\partial \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2}{\partial \boldsymbol{A}} + \frac{\partial \Omega(\boldsymbol{A})}{\partial \boldsymbol{A}} = 0 \tag{18}$$

where $\Omega(\boldsymbol{A}) = \lambda \operatorname{Tr}(\boldsymbol{H}^T \boldsymbol{L}_A \boldsymbol{H}) + \Phi(\boldsymbol{\Lambda}, \boldsymbol{A})$ and $\widehat{w}_i = 1/(2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F)$. Obviously, Eq. (18) is the same as the derivation of the Lagrange function of problem (12). Thus, $\widehat{w}_i$ can be considered as the w_i in (12). To avoid dividing by zero in theory, $\widehat{w}_i$ can be transformed into

$$w_i = \frac{1}{2\|\boldsymbol{A} - \boldsymbol{G}_i\|_F + \zeta} \tag{19}$$

where ζ is infinitely close to zero. □

Note that the convergence criterion is $rank(\boldsymbol{L}_A) = n - c$, thus parameter λ should be automatically increased or decreased when the number of connected subgraphs of graph $\boldsymbol{A}$ is smaller or greater than c during the iteration. The pseudocode of our LCSG is depicted in Algorithm 1.

Algorithm 1 The algorithm of LCSG

Input: Multiple kernels $\{\boldsymbol{K}_i\}_{i=1}^m$, parameters α, β, and γ.

1: Initialize $w_i = 1/m$ for each graph, and $\lambda = 10^{-5}$;
2: **repeat**
3: Update each LMKG $\boldsymbol{G}_i$ by problem (10);
4: Update the consensus graph $\boldsymbol{A}$ by problem (15);
5: Update the embedding matrix $\boldsymbol{H}$ by problem (16);
6: Update the weight vector $\boldsymbol{w}$ by problem (19);
7: **until** $rank(\boldsymbol{L}_A) = n - c$ is satisfied;
8: Use graphconncomp function to find the strongly connected components of graph $\boldsymbol{A}$.

Output: Clustering results.

B. Computational Complexity Analysis

In Algorithm 1, the computational complexity of updating $\{\boldsymbol{G}_i\}_{i=1}^m$, $\boldsymbol{A}$, $\boldsymbol{H}$ and $\boldsymbol{w}$ are $\mathcal{O}(mn^2)$, $\mathcal{O}(n^2)$, $\mathcal{O}(cn^2)$ and $\mathcal{O}(mn^2)$, respectively. Hence, the computational complexity of our LCSG is only $\mathcal{O}(n^2)$ in each iteration. while that of other graph-based MKC methods are at least $\mathcal{O}(n^3)$. The main reason is that the existing graph-based MKC methods always involve the matrix reverse operator, where the computational complexity of the matrix reverse operator is $\mathcal{O}(n^3)$.

C. Convergence Analysis

Objective function (6) is convex *w.r.t.* one variable while fixing the others. For each subproblem, it is convex minimization problem and has optimal solution. Thus, by solving these subproblems alternatively, our algorithm will reduce the objective function monotonically. Moreover, we prove that the whole function is lower bounded in virtue of Proposition 3. Thus, the convergence of our algorithm can be guaranteed.

Proposition 3. *Objective function* (6) *is lower bounded.*

Proof. Objective function (6) can be divided into two parts (*i.e.*, $\boldsymbol{\Theta}_1$ and $\boldsymbol{\Theta}_2$). First, the lower bound of $\boldsymbol{\Theta}_1$ is given by

$$\begin{aligned} \boldsymbol{\Theta}_1 &= -\operatorname{Tr}(\boldsymbol{K}_i \boldsymbol{G}_i) + \alpha \|\boldsymbol{G}_i\|_F^2 \\ &= \alpha \|\boldsymbol{G}_i\|_F^2 - \langle \boldsymbol{K}_i^T, \boldsymbol{G}_i \rangle + \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 - \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \\ &= \|\sqrt{\alpha} \boldsymbol{G}_i - \frac{1}{2\sqrt{\alpha}} \boldsymbol{K}_i\|_F^2 - \frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \geq -\frac{1}{4\alpha} \|\boldsymbol{K}_i\|_F^2 \end{aligned} \tag{20}$$

Second, the lower bound of $\boldsymbol{\Theta}_2$ is given by

$$\boldsymbol{\Theta}_2 = \gamma \sum_{i=1}^{m} \sum_{j=1, j \neq i}^{m} \|\boldsymbol{G}_i - \boldsymbol{G}_j\|_F^2 + \beta \sum_{i=1}^{m} w_i \|\boldsymbol{A} - \boldsymbol{G}_i\|_F^2 \geq 0 \tag{21}$$

Hence, the whole function, $\boldsymbol{\Theta} = \sum \boldsymbol{\Theta}_1 + \boldsymbol{\Theta}_2$, is lower bounded as $\boldsymbol{\Theta} \geq -\frac{1}{4\alpha} \sum_{i=1}^{m} \|\boldsymbol{K}_i\|_F^2$. □

V. Experiments

We demonstrate the effectiveness of our LCSG by conducting several experiments on nine public benchmark datasets.

A. Datasets and Kernel Pool

Following [18], [21], we employ nine widely used benchmark datasets, including six image datasets (*i.e.*, Yale, Jaffe, ORL, AR, COIL20, and BA) and three text corporas (*i.e.*, TR11, TR41, and TR45). These datasets can stand for the complex IIoT non-linear data for evaluating the performance of the proposed method. The statistics of these datasets are briefly summarized in Table I.

TABLE I: Statistics of the nine benchmark datasets

Dataset	# Classes	# Samples	# Features
Yale	15	165	1024
Jaffe	10	213	676
AR	120	840	768
ORL	40	400	1024
COIL-20	20	1440	1024
BA	36	1404	320
TR11	9	414	6429
TR41	10	878	7454
TR45	10	690	8261

In the same way as [18], a kernel pool is built in advance, which consists of 12 candidate kernels (*i.e.*, $m = 12$): a cosine kernel $k_{ij} = (\boldsymbol{x}_i^T \boldsymbol{x}_j)/(\|\boldsymbol{x}_i\|_2^2 \cdot \|\boldsymbol{x}_j\|_2^2)$; four polynomial kernels $k_{ij} = (u + \boldsymbol{x}_i^T \boldsymbol{x}_j)^v$ where u varies from $\{0, 1\}$ and v varies from $\{2, 4\}$; and seven radial basis function (RBF) kernels $k_{ij} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2/(2\theta\tau^2))$, where θ varies from the set of $\{0.01, 0.05, 0.1, 1, 10, 50, 100\}$ and τ is the maximum distance between any two samples. All the kernels are normalized to $[0, 1]$ by $k_{ij} = k_{ij}/\sqrt{k_{ii}k_{jj}}$.

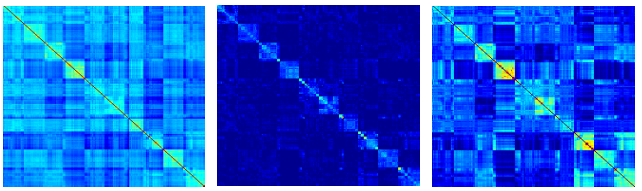
B. Competitors and Evaluating Metric

We compare the proposed LCSG method with the following state-of-the-art competitors: MKKM [23], RMKKM [18], AASC [22], SCMK [24], LKGr [20], SMKL [16], JMKSC [21], and MVCLFA [25]. Amongst these methods, MKKM, RMKKM, and MVCLFA are k-means-based methods, while others are graph-based methods. For MVCLFA, we take the kernels as views and fed into it. For fair comparison, the involved parameters of these competitors have been carefully tuned as recommended by their respective authors. To quantitatively investigate the clustering performance, three widely used metrics, clustering accuracy (ACC), normalized mutual information (NMI), and purity, are applied here. For the three metrics, the higher values indicate the better performance. Meanwhile, to alleviate the instability caused by k-means in spectral clustering, we independently repeat each experiment 20 times.

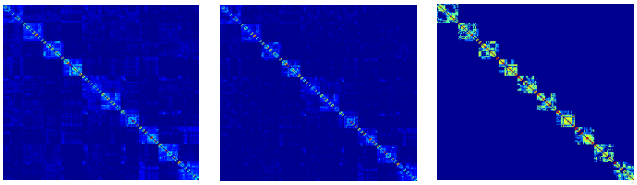
C. Performance Evaluation

The clustering results are presented in Tables II, III and IV. It can clearly be seen that our LCSG consistently obtains the best performance, and the improvements are significant in most case. Surprisingly, our LCSG improves by 8.0%, 5.4%, and 6.0%, respectively, compared to JMKSC (the best competitor) in terms of ACC, NMI, and purity. Note here that owing to the introduced graph connectivity constraint (*i.e.*, $\mathrm{rank}(\boldsymbol{L}_A) = n - c$), our LCSG yields a standard deviation of zero in every case. These results indicate the higher effectiveness of our pure graph learning than the existing non-graph learning and distractible graph learning for MKC tasks.

Furthermore, to evaluate the quality of the learned consensus affinity matrix (also known as affinity graph) $\boldsymbol{A}$, we illustrate $\boldsymbol{A}$ produced by the comparison methods on the Jaffe dataset by using a visual assessment similar to [30]. The results are shown in Fig. 1. Obviously, the matrix $\boldsymbol{A}$ of our LCSG has better block diagonal property and inter-cluster separability than the competitors. Thanks to the introduced graph connectivity constraint (*i.e.*, $\mathrm{rank}(\boldsymbol{L}_A) = n - c$), the learned graph $\boldsymbol{A}$ can be exactly partitioned into c strongly connected subgraphs by automatically tuning λ. What's more, the phenomenon that all the standard deviations of our LCSG (presented in Tables II, III and IV) are zeros is consistent with the above graph theory.



(a) AASC [22] (b) SCMK [24] (c) LKGr [20]



(d) SMKL [16] (e) JMKSC [21] (f) Our LCSG

Fig. 1: Visualization of the learned affinity graph $\boldsymbol{A}$ on the Jaffe dataset. The Jaffe dataset consists of 10 clusters. Note that the darker the blue color, the value is closer to zero. (Zoom in for best view).

D. Parameter Sensitivity

In the proposed LCSG method, there are three parameters, α, γ and β, needed to be tuned. Take the Yale and ORL datasets, for example. By fixing $\alpha = 1$ and using a grid search strategy, the searching regions of β and γ are selected from $\{10^{-4}, \cdots, 10^{1}\}$ and $\{10^{-3}, \cdots, 10^{2}\}$, respectively. We then show the parameter sensitivities *w.r.t.* β and γ in Fig. 2. Subsequently, by fixing $\beta = 0.1$, $\gamma = 10$ and $\beta = 0.01$, $\gamma = 100$ for the Yale and ORL datasets, respectively, α is tuned from the range of $\{10^{-8}, \cdots, 10^{8}\}$. We then show the parameter sensitivities *w.r.t.* α in Fig. 3. Overall, satisfactory performance is obtained over a large range of parameter values for all datasets. For simplicity, we fix $\alpha = 1$ in all experiments, one can tune it for better performance.

E. Convergence

Theoretically, the convergence of our LCSG can be guaranteed (see Section IV-C). Experimentally, we evaluate the convergence of our LCSG on the Yale and ORL datasets. Note here that the convergence criterion is $\mathrm{rank}(\boldsymbol{L}_A) = n - c$,

TABLE II: Clustering performance comparison (average ± standard deviation) in term of ACC.

Dataset	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.457±0.041	0.521±0.034	0.406±0.027	0.582±0.025	0.540±0.030	0.582±0.017	0.630±0.006	0.618±0.011	**0.661±0.000**
ORL	0.475±0.023	0.556±0.024	0.272±0.009	0.656±0.015	0.616±0.016	0.573±0.032	0.725±0.014	0.692±0.005	**0.810±0.000**
Jaffe	0.746±0.069	0.871±0.053	0.304±0.008	0.869±0.022	0.861±0.052	0.967±0.000	0.967±0.007	0.981±0.005	**1.000±0.000**
AR	0.286±0.014	0.344±0.012	0.332±0.006	0.544±0.024	0.314±0.015	0.263±0.009	0.609±0.007	0.667±0.008	**0.779±0.000**
BA	0.405±0.019	0.434±0.018	0.271±0.003	0.384±0.014	0.444±0.018	0.246±0.012	0.484±0.015	0.413±0.005	**0.523±0.000**
COIL	0.548±0.058	0.667±0.028	0.349±0.050	0.591±0.028	0.618±0.051	0.487±0.031	0.696±0.016	0.664±0.013	**0.863±0.000**
TR11	0.501±0.048	0.577±0.094	0.472±0.008	0.549±0.015	0.607±0.043	0.708±0.033	0.737±0.002	0.572±0.026	**0.756±0.000**
TR41	0.561±0.068	0.627±0.073	0.459±0.001	0.650±0.068	0.595±0.020	0.671±0.002	0.689±0.004	0.594±0.005	**0.788±0.000**
TR45	0.585±0.066	0.640±0.071	0.526±0.008	0.634±0.058	0.663±0.042	0.671±0.004	0.687±0.036	0.721±0.002	**0.778±0.000**

TABLE III: Clustering performance comparison (average ± standard deviation) in term of NMI.

Dataset	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.501±0.036	0.556±0.025	0.468±0.028	0.576±0.012	0.566±0.025	0.614±0.015	0.631±0.006	0.609±0.009	**0.643±0.000**
ORL	0.689±0.016	0.748±0.018	0.438±0.007	0.808±0.008	0.794±0.008	0.733±0.027	0.852±0.012	0.836±0.003	**0.889±0.000**
Jaffe	0.798±0.058	0.893±0.041	0.272±0.006	0.868±0.021	0.869±0.031	0.951±0.000	0.952±0.010	0.970±0.008	**1.000±0.000**
AR	0.592±0.014	0.655±0.015	0.651±0.005	0.775±0.009	0.648±0.007	0.568±0.014	0.820±0.002	0.844±0.002	**0.894±0.000**
BA	0.569±0.008	0.585±0.011	0.423±0.004	0.544±0.012	0.604±0.009	0.486±0.011	0.621±0.007	0.556±0.002	**0.666±0.000**
COIL	0.707±0.033	0.773±0.017	0.419±0.027	0.726±0.011	0.766±0.023	0.628±0.018	0.818±0.007	0.782±0.005	**0.928±0.000**
TR11	0.446±0.046	0.561±0.118	0.394±0.003	0.371±0.018	0.597±0.031	0.557±0.068	0.673±0.002	0.582±0.012	**0.683±0.000**
TR41	0.578±0.042	0.635±0.092	0.431±0.000	0.492±0.017	0.604±0.023	0.625±0.004	0.660±0.003	0.575±0.006	**0.729±0.000**
TR45	0.562±0.056	0.627±0.092	0.420±0.014	0.584±0.051	0.671±0.020	0.622±0.007	0.690±0.022	0.681±0.001	**0.772±0.000**

TABLE IV: Clustering performance comparison (average ± standard deviation) in term of Purity.

Data	MKKM [23]	RMKKM [18]	AASC [22]	SCMK [24]	LKGr [20]	SMKL [16]	JMKSC [21]	MVCLFA [25]	Our LCSG
Yale	0.475±0.037	0.536±0.031	0.423±0.026	0.610±0.014	0.554±0.029	0.667±0.014	0.673±0.007	0.624±0.010	**0.703±0.000**
ORL	0.514±0.021	0.602±0.024	0.316±0.007	0.699±0.015	0.658±0.017	0.648±0.017	0.753±0.012	0.732±0.004	**0.830±0.000**
Jaffe	0.768±0.062	0.889±0.045	0.331±0.008	0.882±0.023	0.859±0.038	0.967±0.000	0.967±0.007	0.981±0.005	**1.000±0.000**
AR	0.305±0.012	0.368±0.010	0.350±0.006	0.642±0.014	0.330±0.014	0.530±0.014	0.656±0.010	0.685±0.003	**0.805±0.000**
BA	0.435±0.014	0.463±0.015	0.303±0.004	0.606±0.009	0.479±0.017	0.623±0.011	0.563±0.018	0.438±0.006	**0.646±0.000**
COIL	0.590±0.053	0.699±0.022	0.391±0.044	0.635±0.013	0.650±0.039	0.683±0.004	0.806±0.010	0.690±0.013	**0.913±0.000**
TR11	0.655±0.044	0.729±0.096	0.547±0.000	0.783±0.011	0.776±0.030	**0.835±0.048**	0.819±0.001	0.768±0.009	0.787±0.000
TR41	0.728±0.042	0.776±0.065	0.621±0.001	0.758±0.034	0.759±0.031	0.761±0.003	0.799±0.003	0.757±0.008	**0.833±0.000**
TR45	0.691±0.058	0.752±0.074	0.575±0.011	0.728±0.048	0.800±0.026	0.816±0.004	0.822±0.031	0.806±0.001	**0.883±0.000**

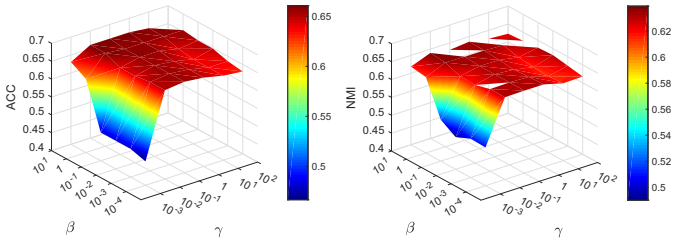
i.e., the connected subgraphs of the learned graph $\boldsymbol{A}$ is equal to c, so we need to continuously self-tune parameter λ until the desired graph is obtained. The results presented in Fig. 4 suggest that the objective value is monotonically decreased, and the clustering performance is gradually improving. Notably, although the algorithm seems to converge after only 5 iterations for the ORL dataset, the additional iterations are also need to meet the convergence criterion, *i.e.*, $\mathrm{rank}(\boldsymbol{L}_A) = n-c$. Usually, LCSG converges in less than 20 iterations for all evaluated datasets.

F. Running Time

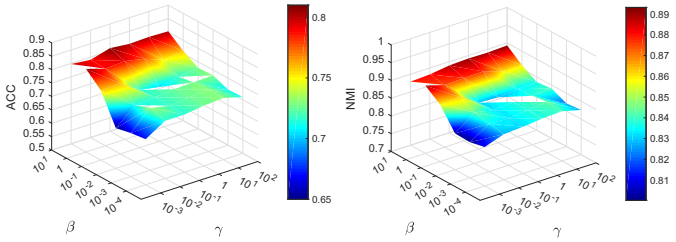
We compare the running time (in seconds) of all competitors on the Yale, ORL, TR11, and TR45 datasets. All codes are implemented in MATLAB 2016b and run on a Mac PC with a 3.2 GHz Intel Core i7 processor, 16-GB RAM, and macOS Mojave operating system. The mean and standard deviations of 20 trials are reported in Table V, the proposed LCSG has a competitive superiority on running time. Although the running time of MKKM and MVCLFA is lower than that of our LCSG, their clustering performance is worse than that of ours.

VI. Conclusion

In this paper, we have proposed a pure graph-based MKC method to address the changeling non-linear clustering issues for cognitive computing of intelligent IIoT. Specifically, a new graph model, termed as LMKG, that can preserve the local manifold structure of data in kernel space is introduced to learn

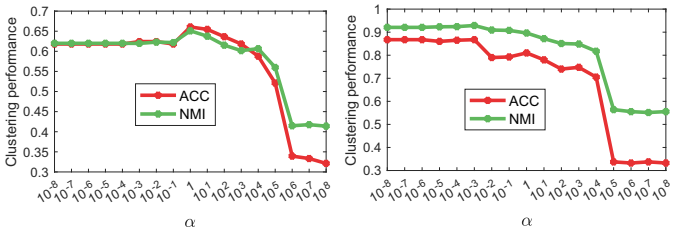


(a) ACC (b) NMI



(c) ACC (d) NMI

Fig. 2: ACC and NMI of our LCSG *w.r.t.* β and γ on the Yale (the fist row) and ORL (the second row) datasets. α is fixed to 1. (Zoom in for best view).



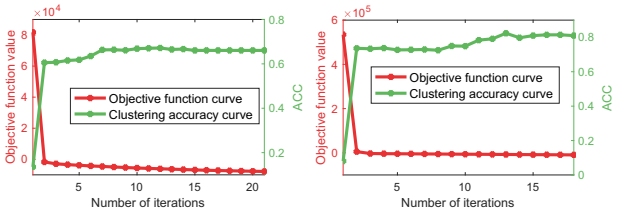
(a) Yale ($\beta = 0.1$, $\gamma = 10$) (b) ORL ($\beta = 0.01$, $\gamma = 100$)

Fig. 3: ACC and NMI of our LCSG *w.r.t.* α on the Yale and ORL datasets.

multiple LMKGs from multiple candidate kernels. By considering both the consistency and selfishness of these LMKGs, the quality of affinity graph achieves significant improvement. Further, the graph connectivity constraint avoids requiring any post-processing step such that the clustering results can be immediately obtained. Comprehensive experimental results clearly demonstrates the superiority of our method. Therefore, our LCSG method can be used to effectively handle the non-

TABLE V: Computational time (in seconds) comparison.

Method	Yale	ORL	TR11	TR45
MKKM [23]	0.015±0.001	0.128±0.003	0.059 ±0.002	0.162±0.005
RMKKM [18]	0.870±0.011	3.622±0.085	4.144±0.117	6.908±0.121
AASC [22]	0.221±0.005	0.910±0.007	0.977±0.014	1.686±0.019
SCMK [24]	5.492±0.144	42.040±1.782	51.454±3.102	218.715±5.384
LKGr [20]	1.422±0.015	7.425±0.227	13.718±0.340	80.558±3.211
SMKL [16]	1.439±0.022	12.836±0.565	9.863±0.144	154.683±5.101
JMKSC [21]	1.219±0.015	2.462±0.108	3.974±0.125	8.765±0.183
MVCLFA [25]	0.251±0.004	0.893±0.011	1.116±0.420	2.243±0.047
Our LCSG	0.620±0.023	1.882±0.054	2.121±0.113	5.921±0.121



(a) Yale (b) ORL

Fig. 4: Convergence curve of our LCSG on the Yale and ORL datasets.

linear data from intelligent IIoT and other industrial sensor networks.

In our future work, it is potentially interesting to extend the proposed method to handle large-scale non-linear data for cognitive computing.

REFERENCES

[1] P. Jia, X. Wang, and K. Zheng, "Distributed clock synchronization based on intelligent clustering in local area industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3697–3707, 2019.

[2] J. Xiao, Y. Tian, L. Xie, X. Jiang, and J. Huang, "A hybrid classification framework based on clustering," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2177–2188, 2020.

[3] H. Hassan, A. K. Bashir, M. Ahmad, V. G. Menon, I. U. Afridi, R. Nawaz, and B. Luo, "Real-time image dehazing by superpixels segmentation and guidance filter," *Journal of Real-Time Image Processing*, pp. 1–21, 2020.

[4] X. Liu and X. Zhang, "Noma-based resource allocation for cluster-based cognitive industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2019.

[5] S. Jacob, V. G. Menon, and S. Joseph, "Depth information enhancement using block matching and image pyramiding stereo vision enabled rgb-d sensor," *IEEE Sensors Journal*, vol. 20, no. 10, pp. 5406–5414, 2020.

[6] T. Wang, H. Luo, W. Jia, A. Liu, and M. Xie, "Mtes: An intelligent trust evaluation scheme in sensor-cloud-enabled industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2054–2062, 2020.

[7] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, "Massive internet of things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 28–33, 2017.

[8] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained laplacian rank algorithm for graph-based clustering," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1969–1976.

[9] K. Zhan, C. Zhang, J. Guan, and J. Wang, "Graph learning for multiview clustering," *IEEE transactions on cybernetics*, vol. 48, no. 10, pp. 2887–2895, 2017.

[10] Z. Y. Ren, S. X., Q. Sun, and T. Wang, "Consensus affinity graph learning for multiple kernel clustering," *IEEE Transactions on Cybernetics*, p. 10.1109/TCYB.2020.3000947, 2020.

[11] X. Liu, X. Zhu, M. Li, L. Wang, E. Zhu, T. Liu, M. Kloft, D. Shen, J. Yin, and W. Gao, "Multiple kernel k-means with incomplete kernels," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 5, pp. 1191–1204, 2020.

[12] Z. Kang, H. Pan, S. C. Hoi, and Z. Xu, "Robust graph learning from noisy data," *IEEE transactions on cybernetics*, vol. 28, no. 4, pp. 1007–1021, 2020.

[13] H. Wang, Y. Yang, B. Liu, and H. Fujita, "A study of graph-based system for multi-view clustering," *Knowledge-Based Systems*, vol. 163, pp. 1009–1019, 2019.

[14] C. Lu, J. Feng, Z. Lin, T. Mei, and S. Yan, "Subspace clustering by block diagonal representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 487–501, 2019.

[15] H. Li, J. Zhang, J. Hu, C. Zhang, and J. Liu, "Graph-based discriminative concept factorization for data representation," *Knowledge-Based Systems*, vol. 118, pp. 70–79, 2017.

[16] Z. Kang, X. Lu, J. Yi, and Z. Xu, "Self-weighted multiple kernel learning for graph-based clustering and semi-supervised classification," *IJCAI*, pp. 2312–2318, 2018.

[17] Z. Ren, H. Li, C. Yang, and Q. Sun, "Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning," *Knowledge-Based Systems*, p. 105040, 2019.

[18] L. Du, P. Zhou, L. Shi, H. Wang, M. Fan, W. Wang, and Y.-D. Shen, "Robust multiple kernel k-means using l21-norm," pp. 3476–3482, 2015.

[19] S. Zhou, X. Liu, M. Li, E. Zhu, L. Liu, C. Zhang, and J. Yin, "Multiple kernel clustering with neighbor-kernel subspace segmentation," *IEEE Transactions on Neural Networks*, pp. 1–12, 2019.

[20] Z. Kang, L. Wen, W. Chen, and Z. Xu, "Low-rank kernel learning for graph-based clustering," *Knowledge-Based Systems*, vol. 163, pp. 510–517, 2019.

[21] C. Yang, Z. Ren, Q. Sun, M. Wu, M. Yin, and Y. Sun, "Joint correntropy metric weighting and block diagonal regularizer for robust multiple kernel subspace clustering," *Information Sciences*, vol. 500, pp. 48–66, 2019.

[22] H.-C. Huang, Y.-Y. Chuang, and C.-S. Chen, "Affinity aggregation for spectral clustering," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 773–780.

[23] H. Huang, Y. Chuang, and C. Chen, "Multiple kernel fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 1, pp. 120–134, 2012.

[24] Z. Kang, C. Peng, Q. Cheng, and Z. Xu, "Unified spectral clustering with optimal graph," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 3366–3373.

[25] S. Wang, X. Liu, E. Zhu, C. Tang, J. Liu, J. Hu, J. Xia, and J. Yin, "Multi-view clustering via late fusion alignment maximization," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 3778–3784.

[26] Z. Lin, F. Wen, Y. Ding, and Y. Xue, "Data-driven coherency identification for generators based on spectral clustering," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1275–1285, 2018.

[27] Z. Ren and Q. Sun, "Simultaneous global and local graph structure preserving for multiple kernel clustering," *IEEE Transactions on Neural Networks and Learning Systems*, p. 10.1109/TNNLS.2020.2991366, 2020.

[28] M. Iliadis, H. Wang, R. Molina, and A. K. Katsaggelos, "Robust and low-rank representation for fast face identification with occlusions," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2203–2218, 2017.

[29] F. Nie, J. Li, X. Li *et al.*, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification." in *IJCAI*, 2016, pp. 1881–1887.

[30] J. C. Bezdek and R. J. Hathaway, "Vat: a tool for visual assessment of (cluster) tendency," vol. 3, pp. 2225–2230, 2002.